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., 2023; 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., 2024) 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., 2023). In this method, the LLM is prompted to answer whether a document is relevant to the query. Existing pointwise LLM rankers mostly ask the LLM to answer "Yes" or "No" and use their likelihood to derive a ranking score. Nevertheless, some documents cannot always be accurately classified into these two categories as they may not directly answer the query but still contain helpful information.

Studies on human subjects show that using binary options sometimes leads 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., 2020b,a). 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" and collect their likelihood scores from LLM predictions to derive the ranking score. The intuition is that the intermediate relevance labels in the prompt serve as a "cue" to the LLM to distinguish partially relevant documents from fully relevant or fully irrelevant ones.

Our evaluation on 8 BEIR (Thakur et al., 2021) datasets demonstrates that simply adding intermediate relevance labels significantly boosts LLM ranking performance across different datasets, regardless of the actual ground-truth label granularity. An in-depth analysis reveals that the proposed new prompt enables LLM rankers to distinguish documents previously indistinguishable with the binary-option prompt.

2 Related Work

Zero-shot LLM rankers. Shifted 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., 2023; Xian et al., 2023; Liu, 2009; Qin et al., 2021), there is an emerging thread of research exploring how to use general-purpose LLMs directly or indirectly (Jagerman et al., 2023; Li et al., 2024) for zero-shot text ranking.

Liang et al. (2023) and Sachan et al. (2022) adopt a pointwise approach which scores the relevance of one document at a time based on how likely the LLM would classify the document as relevant or how likely the LLM would generate the query from the document respectively. There are also explorations on pairwise (Qin et al., 2024) and listwise (Sun et al., 2023; Ma et al., 2023; Zhuang et al., 2024) LLM rankers which take multiple documents as input and return the ranking directly, but they are usually applied iteratively on smaller sets of documents. 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, where fine-grained relevance labels are also provided in the prompt. However, these methods do not use the likelihood scores of fine-grained relevance labels. The goal of LLM assessors is to provide a relevance label for every query-document pairs that aligns with the ground-truth relevance label, potentially created by human assessors. 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.

3 LLM Rankers

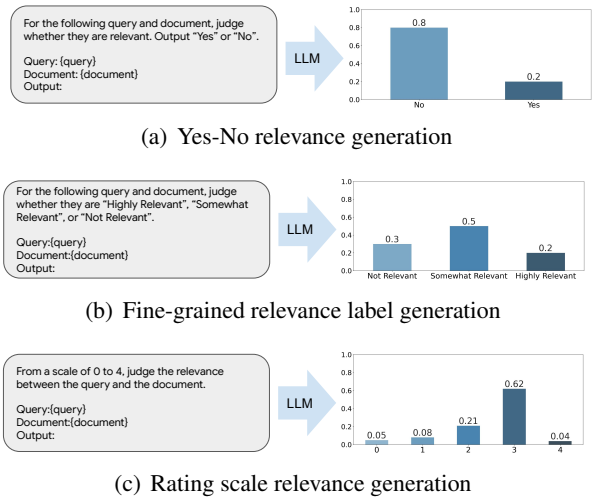
3.1 Preliminaries

Existing explorations using zero-shot LLMs as pointwise rankers can be broadly divided into two categories: relevance generation (Liang et al., 2023) and query generation (Sachan et al., 2022). We focus on relevance generation in this work.

Given a query q and a list of candidate documents $\mathbf{d} = (d_1, \ldots, d_m)$, an LLM ranker based on relevance generation takes each query-document pair (q, d_i) as input and prompts the LLM to answer whether the document is relevant to the query by "Yes" or "No" (see Figure 1(a)). Then a ranking score $f(q, d_i) \in \mathbb{R}$ for each document is calculated based on LLM's log-likelihood score $s_{i,1} = \text{LLM}(\text{Yes}|q, d_i)$ and $s_{i,0} = \text{LLM}(\text{No}|q, d_i)$ by using a softmax function (Nogueira et al., 2020):

$$f(q, d_i) = \frac{\exp(s_{i,1})}{\exp(s_{i,1}) + \exp(s_{i,0})}$$

The ranked list is obtained by sorting the documents based on their ranking scores.



(a) Yes-No relevance generation

(b) Fine-grained relevance label generation

(c) Rating scale relevance generation

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

3.2 Prompts

In many datasets, there exist documents that are only partially or marginally relevant to the query, which LLMs struggle to classify into two classes.

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_i), 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:

$$s_{i,k} = \text{LLM}(l_k|q, d_i) \quad (1)$$

This example is illustrated in Figure 1(b). The exact prompt can be found in Appendix G.

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_i on a scale from 0 to 4. We can then use the LLM to obtain the log-likelihood $[s_{i,0}, \ldots, s_{i,4}]$ 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. The exact prompt can be found in Appendix G.

3.3 Ranking Scores

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

Expected relevance values (ER). The most straightforward way is to calculate the expected relevance value. First, we need to assign a series of relevance values $[y_0, y_1, y_2]$ to all the relevance labels $[l_0, l_1, l_2]$, where $y_k \in \mathbb{R}$. Then we can calculate the expected relevance value by:

$$f(q, d_i) = \sum p_{i,k} \cdot y_k \tag{2}$$

$$\text{where } p_{i,k} = \frac{\exp(s_{i,k})}{\sum_{k'} \exp(s_{i,k'})}$$

The relevance values y_k can be provided by users or even tuned based on a training data set. We empirically find that naïvely assigning $y_k = k$ (with l_0 to l_k ordered from least to most relevant) already yields excellent performance. Therefore, we simply adopt $y_k = k$.

Peak relevance likelihood (PR). We can further simplify ranking score derivation by focusing on top-ranked items. We propose to only use the log-likelihood of the peak relevance label ("Highly Relevant" in this example). More formally, let l_{k^*} denote the relevance label with the highest relevance. We can simply rank the documents by:

$$f(q, d_i) = s_{i,k^*} \tag{3}$$

Note that s_{i,k^*} is the log-likelihood directly obtained from the LLM, instead of the marginal probability p_{i,k^*} in Equation (2). Hence, it is not necessary to score all 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.

Table 1: Relevance labels used in RG-kL. The relevance label l_{k^*} 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"**

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 the actual relevance granularity in each data set.

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 main experiments. Results of other LLMs can be found in Appendix D.

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 query likelihood from LLM given the document (Sachan et al., 2022).

2. Binary Relevance Generation (RG-YN). Prompting the LLM with a query-document pair and using "Yes/No" likelihood to calculate the ranking score (Liang et al., 2023).

3. k-Level Relevance Generation (RG-kL). Prompting the LLM to choose from k relevance labels for each query-document pair. The relevance labels 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. Note that for RG-S$(0, k)$, the LLM needs to score $(k + 1)$ labels.

By default, the ranking scores of our methods are derived using expected relevance (Equation (2)).

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***

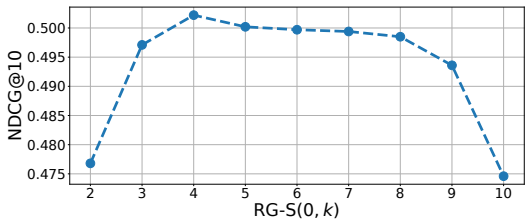


Figure 2: Average NDCG@10 on 8 BEIR data sets with different k in rating scale 0-to-k.

Table 3: Comparing ranking score derivation strategies measured by average NDCG@10 on BEIR data sets.

Prompts	Generated	Likelihood-ER	Likelihood-PR
RG-3L	0.3989	0.4992	0.5005
RG-4L	0.4259	0.4969	0.4934
RG-S$(0, 4)$	0.4445	0.5022	0.4988

5 Results

Overall performance. Table 2 summarizes the overall comparison results. It can be seen that prompting LLMs with fine-grained relevance labels achieves substantially higher performance than binary relevance labels (RG-YN, RG-2L). For example, 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.

There are a few potential explanations for the observed improvement. One explanation is that the estimated relevance becomes more accurate as we aggregate more log-likelihood scores of multiple relevance labels. Another is that the fine-grained relevance labels in the prompt help the LLMs to develop a more nuanced understanding of relevance. We conduct more experiments to further explore these explanations.

Number of relevance labels. We first explore the effect of using different number of relevance labels. Table 2 demonstrates that when using RG-kL, RG-4L performance is on par with RG-3L, suggesting that adding more relevance levels does not always improve the performance when using textual fine-grained relevance labels.

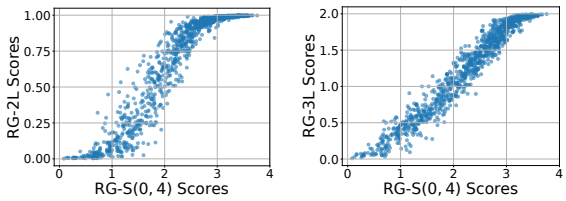
We also plot how the performance changes with regard to k for the rating scale prompting method RG-S$(0, k)$ in Figure 2. It shows that the performance from RG-S$(0, 4)$ to RG-S$(0, 8)$ remain similar. This again suggests that using more fine-grained relevance labels does not further improve the performance. Furthermore, performance declines for even larger k such as RG-S$(0, 9)$ and RG-S$(0, 10)$. This potentially indicates that LLMs struggle to understand prompts with excessive granularity (Thawani et al., 2021).

Notably, the performance trend in Figure 2 remains consistent across datasets regardless of varying granularity of ground-truth label (Appendix E). This illustrates that, in practice, the performance gains are robust to a wide range of k selections.

Ranking score derivation. We compare different strategies for deriving ranking scores.

Some existing work on LLM assessors (Faggioli et al., 2023; Thomas et al., 2023) directly use the generated labels or scores without using the likelihood. Technically, we can also rank documents directly based on the labels or scores parsed from the string outputs generated by LLMs. We include this method in our comparison, denoted as "Generated".

Additionally, we compare the two strategies proposed in Section 3.3: expected relevance values



(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.

(Likelihood-ER) and peak relevance likelihood (Likelihood-PR), both of which derive ranking scores from the predicted log-likelihood of LLMs.

The comparison results are presented in Table 3. It is clear that directly using the generated labels or scores results in lower ranking performance compared to deriving scores from the log-likelihood, as it tends to introduce ties between documents. On the other hand, peak relevance likelihood (Likelihood-PR) achieves very close performance to expected relevance values (Likelihood-ER) in most methods, despite only using the log-likelihood of one relevance label. This suggests that the improvement brought by scoring fine-grained relevance labels cannot be simply explained by improved accuracy of estimated relevance by *using more samples*. Instead, it is possible that including fine-grained relevance labels within the prompt may signal LLMs to attend to the subtle relevance differences.

Score distribution comparison. We compare the score distributions of different methods to gain deeper insight into how fine-grained relevance labels influence performance. Figure 3 presents a scatter plot of ranking scores (Likelihood-ER) from two methods for a random sample of query-document pairs in the Covid data set.

Figure 3(a) demonstrates that RG-2L's ranking scores are mostly positively correlated with RG-S$(0, 4)$'s (Figure 3(a)), but struggles to distinguish query-document pairs with higher scores from RG-S$(0, 4)$ and scores them almost equally with scores close to 1.0. This indicates that LLMs can differentiate better among higher-ranked relevant documents with fine-grained relevance labels. In contrast, the ranking scores from RG-3L and RG-S$(0, 4)$ (Figure 3(b)) exhibit strong correlation almost throughout the entire range. Correspondingly, RG-3L and RG-S$(0, 4)$ also achieve similar ranking performance on this data set.

6 Conclusion

We explore pointwise zero-shot LLM rankers which score fine-grained relevance labels (e.g., "Somewhat Relevant") instead of binary labels. 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 LLM likelihood scores of different relevance labels into ranking scores to rank the documents. Further experiments illustrate that the performance gains are not solely attributable to more precise relevance estimation by using more samples, as only using the log-likelihood of one relevance labels can also achieve similar performance gain. Instead, it is possible that the inclusion of fine-grained relevance labels in the prompt may guide LLMs to better differentiate documents, especially those ranked at the top.

We believe that this approach can be extended beyond information retrieval to many other applications (Liu et al., 2023). 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 proprietary LLMs where users can only call with specific APIs.

Our study is also limited to *ranking* performance of LLMs, without further evaluation or analysis on whether our prompts can also improve LLM assessors. Higher ranking performance does not always translate to higher relevance calibration performance (Cohen et al., 2021; Faggioli et al., 2023; Thomas et al., 2023), as the metrics have different emphasis. It is possible that one needs to apply an appropriate transformation on the derived ranking scores from LLM likelihoods to achieve the best relevance calibration performance, which can be non-trivial. We believe this is an intriguing research direction as it can further broaden the application (Bahri et al., 2020; Shtok et al., 2012) of the proposed methods.

References

Dara Bahri, Yi Tay, Che Zheng, Donald Metzler, and Andrew Tomkins. 2020. Choppy: Cut transformer for ranked list truncation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1513–1516.

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.

Daniel Cohen, Bhaskar Mitra, Oleg Lesota, Navid Rekabsaz, and Carsten Eickhoff. 2021. Not all relevance scores are equal: Efficient uncertainty and calibration modeling for deep retrieval models. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 654–664.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, and Daniel Campos. 2020a. Overview of the TREC 2020 deep learning track. In *Proceedings of the Twenty-Ninth Text REtrieval Conference, TREC 2020, Virtual Event [Gaithersburg, Maryland, USA], November 16-20, 2020*, volume 1266 of *NIST Special Publication*. National Institute of Standards and Technology (NIST).

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M. Voorhees. 2020b. Overview of the TREC 2019 deep learning track. *CoRR*, abs/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*.

Rolf Jagerman, Honglei Zhuang, Zhen Qin, Xuanhui Wang, and Michael Bendersky. 2023. Query expansion by prompting large language models. In *GenIR @ SIGIR 2023: The First Workshop on Generative Information Retrieval*.

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.

Minghan Li, Honglei Zhuang, Kai Hui, Zhen Qin, Jimmy Lin, Rolf Jagerman, Xuanhui Wang, and Michael Bendersky. 2024. Can query expansion improve generalization of strong cross-encoder rankers? In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, et al. 2023. Holistic evaluation of language models. *Transactions on Machine Learning Research*.

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*, pages 2356–2362.

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

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: Nlg evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522.

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 pre-trained 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. 2024. Large language models are effective text rankers with pairwise ranking prompting. In *Findings of the Association for Computational Linguistics: NAACL 2024*.

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 Sachan, Mike Lewis, Mandar Joshi, Armen Aghajanyan, Wen-tau Yih, Joelle Pineau, and Luke Zettlemoyer. 2022. Improving passage retrieval with zero-shot question generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3781–3797.

Anna Shtok, Oren Kurland, David Carmel, Fiana Raiber, and Gad Markovits. 2012. Predicting query performance by query-drift estimation. *ACM Transactions on Information Systems (TOIS)*, 30(2):1–35.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. 2023. Is ChatGPT good at search? investigating large language models as re-ranking agents. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14918–14937.

Yi Tay, Mostafa Dehghani, Vinh Q Tran, Xavier Garcia, Jason Wei, Xuezhi Wang, Hyung Won Chung, Dara Bahri, Tal Schuster, Steven Zheng, et al. 2022. UL2: Unifying language learning paradigms. In *The Eleventh International Conference on Learning Representations*.

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)*.

Avijit Thawani, Jay Pujara, Filip Ilievski, and Pedro Szekely. 2021. Representing numbers in NLP: a survey and a vision. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–656, Online. Association for Computational Linguistics.

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, Hui Xiong, and Enhong Chen. 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. 2023. Learning list-level domain-invariant representations for ranking. In *Advances in Neural Information Processing Systems*.

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. 2023. 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. 2024. A setwise approach for effective and highly efficient zero-shot ranking with large language models. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

A Alternative Relevance Labels

We replace the relevance labels 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 labels. Measured by average NDCG@10 on BEIR data sets.

Method	Relevance Labels	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.

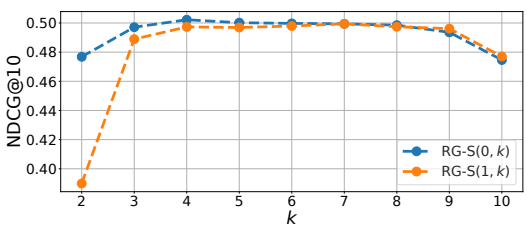


Figure 4: Comparing rating scale relevance generation with different prompts.

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. We then denote p_k as the random variable of the marginal probability $p_{i,k}$ derived for different query-document pairs (q, d_i). We the plot the estimated distribution of p_k for each 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 focuses on the top-ranked items, thus changing the relevance values of intermediate relevance labels may not significantly change the top-ranked items.

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

D Experiments on Other LLMs

To verify the generalizability of our proposed method, we also conduct experiments on two other LLMs. We use FLAN PaLM2 XS, which is a smaller alternative of FLAN PaLM2 S. We also use FLAN UL2 (Tay et al., 2022), which is an open-sourced LLM with 20B parameters. The results are presented in Table 6. We observe similar results where scoring fine-grained relevance labels (RG-3L, RG-S$(0, 4)$) can achieve better average performance than scoring binary labels (RG-2L). This shows that our method can generalize to different LLMs.

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., 2023). 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.
- Pairwise Ranking Prompts (PRP) (Qin et al., 2024). 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

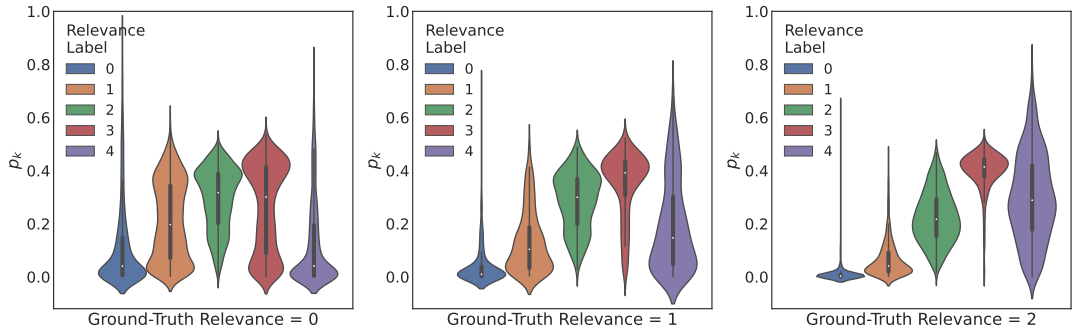


Figure 5: Distribution of marginal probability p_k of each relevance label in RG-S$(0, 4)$ for query-document pairs with different ground-truth labels on Covid data set

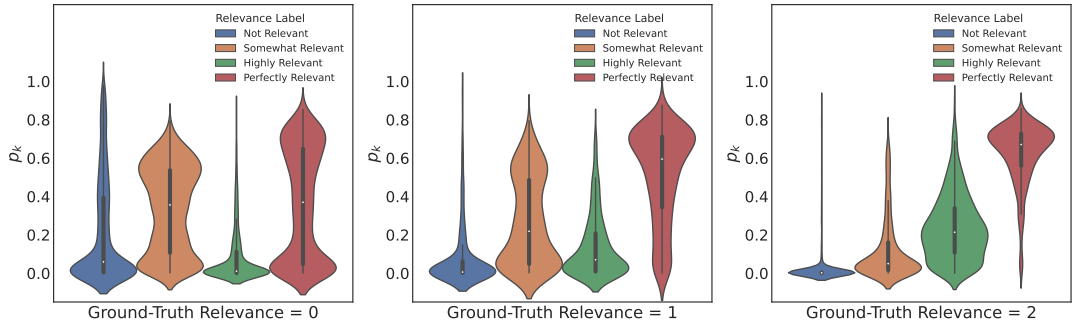


Figure 6: Distribution of marginal probability p_k of each relevance label in RG-4L for query-document pairs with different ground-truth labels on Covid data set

Table 6: Overall ranking performances of FLAN PaLM2 XS and FLAN UL2 measured by NDCG@10 on BEIR data sets. The best performances are bolded.

Model	Method	Covid	Touche	DBPedia	SciFact	Signal	News	Robust04	NFCorpus	Average
FLAN PaLM2 XS	RG-2L	0.7769	0.2549	0.4228	0.6826	0.2892	0.4229	**0.4947**	0.3756	0.4649
	RG-3L	0.7936	0.2554	0.4235	0.6810	0.2931	**0.4374**	0.4933	**0.3777**	0.4694
	RG-4L	0.7969	0.2598	0.4277	0.6681	0.3004	0.4326	0.4772	0.3773	0.4675
	RG-S$(0, 2)$	0.7819	0.2535	0.4141	**0.7135**	0.2791	0.4356	0.4579	0.3711	0.4633
	RG-S$(0, 4)$	**0.8119**	**0.2885**	**0.4386**	0.7102	**0.3097**	0.4341	0.4559	0.3763	**0.4781**
FLAN UL2	RG-2L	0.7769	**0.2737**	0.4047	0.5626	0.2822	0.4573	0.5421	0.3756	0.4594
	RG-3L	0.7998	0.2555	0.4303	0.7007	0.2928	0.4698	**0.5582**	0.3757	0.4853
	RG-4L	**0.8030**	0.2477	**0.4336**	0.7186	0.3047	**0.4710**	0.5575	**0.3775**	0.4892
	RG-S$(0, 2)$	0.7915	0.2546	0.4252	0.7341	0.2997	0.4700	0.5497	0.3702	0.4869
	RG-S$(0, 4)$	0.7969	0.2641	0.4325	**0.7391**	**0.3129**	0.4557	0.5454	0.3708	**0.4897**

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 different k values, and different strategies to derive ranking scores. Table 7 illustrates the results.

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 the single-stage RankGPT. However, note that the LLM used in these experiments are different, so the difference might also be explained by the model difference.

Figure 7 also plots the performance of rating scale methods ranking score derivation methods. It can be observed that the ranking performance of using PR to derive ranking scores is more sensitive to the selection of k than using ER.

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-3L-ER	FLAN PaLM2 S	0.8065	0.2650	0.4013	0.7671	0.3142	0.4890	0.5660	0.3849	0.4992
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-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-PR	FLAN PaLM2 S	0.8065	0.2634	0.4032	0.7745	0.3202	0.4816	0.5681	0.3860	0.5005
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$(0, 3)$-ER	FLAN PaLM2 S	0.7936	0.2720	0.4092	0.7434	0.3240	0.4817	0.5662	0.3868	0.4971
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$(0, 5)$-ER	FLAN PaLM2 S	0.8088	0.2702	0.4217	0.7475	0.3266	0.4734	0.5666	0.3871	0.5002
RG-S$(0, 6)$-ER	FLAN PaLM2 S	0.7898	0.2720	0.4260	0.7529	0.3288	0.4734	0.5687	0.3864	0.4997
RG-S$(0, 7)$-ER	FLAN PaLM2 S	0.7873	0.2695	0.4225	0.7557	0.3263	0.4848	0.5659	0.3831	0.4994
RG-S$(0, 8)$-ER	FLAN PaLM2 S	0.7971	0.2730	0.4254	0.7463	0.3239	0.4722	0.5647	0.3853	0.4985
RG-S$(0, 9)$-ER	FLAN PaLM2 S	0.7910	0.2746	0.4160	0.7465	0.3017	0.4679	0.5644	0.3871	0.4936
RG-S$(0, 10)$-ER	FLAN PaLM2 S	0.7576	0.2496	0.3738	0.7310	0.2771	0.4779	0.5642	0.3655	0.4746
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$(0, 4)$-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

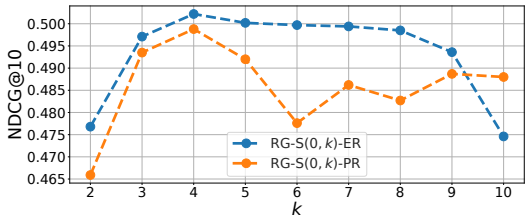


Figure 7: Comparing rating scale relevance generation with different strategies to derive ranking scores.

Table 8: 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

F 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., 2020a). 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 8. 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.

G Prompts

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

G.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}`

G.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. (2023), Sun et al. (2023) and Qin et al. (2024).

> For the following query and document, judge whether they are relevant. Output "Yes" or "No".
>
> Query: `{query}`
> Document: `{document}`
> Output:

G.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:

G.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:

G.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:

G.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: