PRP-Graph: Pairwise Ranking Prompting to LLMs with Graph Aggregation for Effective Text Re-ranking

Anonymous ACL submission

Abstract

Pairwise Ranking Prompting (PRP) demonstrates impressive effectiveness in zero-shot document re-ranking tasks with large language models (LLMs). However, in the existing methods, PRP only outputs the same label for the comparison results of different confidence intervals without considering the uncertainty of pairwise comparison, which implies an under-utilization of the generation probability information of LLMs. To bridge this gap, we propose PRP-Graph, a novel pairwise re-ranking approach, based on a refined scoring PRP unit that exploits the output probabilities of target labels to capture the degree of certainty of the comparison results. Specifically, the PRP-Graph consists of two stages, namely ranking graph construction and ranking graph aggregation. Extensive experiments conducted on the BEIR benchmark demonstrate the superiority of our approach over existing PRP-based methods. Comprehensive analysis reveals that the PRP-Graph displays strong robustness towards the initial ranking order and delivers exceptional re-ranking results with acceptable efficiency. Our code and data will be available.

1 Introduction

Large Language Models (LLMs) such as GPT-3 (Brown et al., 2020), FlanT5 (Wei et al., 2021), and PaLM (Chowdhery et al., 2023) have demonstrated exceptional performance across a diverse array of natural language processing tasks within the zero-shot paradigm (Agrawal et al., 2022; Kojima et al., 2022). Consequently, these LLMs have been adapted for zero-shot document ranking tasks, showcasing remarkable zero-shot ranking capabilities (Liang et al., 2022; Pradeep et al., 2023; Sun et al., 2023). The methodologies utilized LLMs in zero-shot ranking tasks focus on prompting the LLMs in providing relevance estimations for a query along with candidate documents (Liang et al., 2022; Qin et al., 2023).

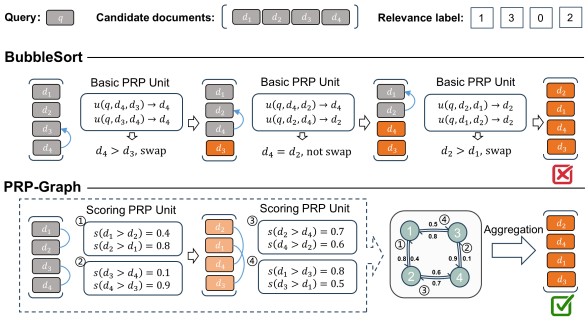

Figure 1: Schematic comparison of PRP-Graph with BubbleSort. In both basic and scoring PRP units, two compared documents d_i and d_j are fed into LLM twice using prompt texts $u(q, d_i, d_j)$ and $u(q, d_j, d_i)$. Different from just obtaining determined passage labels (namely, d_i) in basic PRP unit, PRP-Graph with scoring PRP unit captures the uncertainty (namely, $s(d_i > d_j)$) in document comparisons to iteratively construct a ranking graph, and further employs a graph aggregation strategy to obtain the final ranking.

Notably, recent research has shown that Pairwise Ranking Prompting (PRP) is effective for zero-shot re-ranking with moderate-sized, open-sourced LLMs and can produce state-of-the-art re-ranking performance with simple prompting and scoring mechanism (Qin et al., 2023). PRP involves building a pairwise prompting to determine which of two documents is more relevant to query with the help of LLMs, then using the PRP as a basic scoring unit to serve different ranking mechanisms.

Despite their effectiveness, we posit that existing PRP does not fully capitalize on the potential of LLMs. Firstly, current practices utilize PRP to simply ascertain the more relevant document relative to the query, disregarding a more granular analysis of generation probabilities, which represent the LLM's confidence in its judgment. Secondly, the inherent sensitivity of LLMs to the text orders in prompt (Lu et al., 2021) has led to the verification of consistency by alternating the pairwise order of documents within prompt and consulting the LLM

twice. Only consistent outputs from both inquiries are valid, which can degrade ranking quality. Finally, the comparison selection strategies in use, which are derived from sorting algorithms, lack the flexibility to enhance ranking quality through increased comparisons and are highly sensitive to the sequence order of input documents.

To this end, we propose an enhanced scoring Pairwise Ranking Prompt (PRP) unit and a novel pairwise approach, termed *PRP-Graph*, building upon the scoring PRP unit's performance. As illustrated in Figure 1, unlike the current PRP unit (Qin et al., 2023), which yields discrete judgments, our scoring PRP unit captures more granular correlation information for each pairwise order in prompt through the generation probability of judgment labels by LLMs, thus effectively utilizing the generative capability and order sensitivity of LLMs. PRP-Graph operates in two phases. The first phase continuously identifies document pairs for comparison, representing them as vertices connected by bidirectional edges within a ranking graph. The weights of vertices and edges are iteratively refined based on the scoring PRP unit's outputs. The second phase aggregates these signals across the ranking graph to produce a cohesive final document ranking that encapsulates the entire graph's sorting information.

We conduct experiments to evaluate our proposed PRP-Graph on Flan-T5 models (Chung et al., 2022). Our results on BEIR (Thakur et al., 2021) indicate that PRP-Graph surpasses existing PRP-based zero-shot re-ranking techniques. Further ablation studies confirm the effectiveness of individual components within our method, and our analyses demonstrate the influence of comparison rounds on PRP-Graph's performance as well as its robustness against initial ranking order.

Our contributions are threefold: 1) We introduce the PRP-Graph, a novel pairwise approach underpinned by an innovative scoring PRP unit that exploits the full potential of LLMs by leveraging the probability scores from pairwise ranking prompts. 2) Benefiting from the graph-based nature of PRP-Graph, which relies less on the initial rankings produced by the first-stage retriever, it demonstrates a more effective interpolation with BM25 than current PRP-based methods. 3) Extensive experimental validation shows that our PRP-Graph method significantly outperforms contemporary approaches, establishing it as a superior choice for zero-shot document re-ranking tasks.

2 Preliminaries

As aforementioned, several recent text re-ranking approaches implement Pairwise Ranking Prompting (PRP) as an atomic ranking unit to extract specific textual labels via the generative capabilities or logit outputs of LLMs. This section provides an initial overview of the PRP construct, subsequently outlining the landscape of text re-ranking approaches that integrate the basic PRP unit.

Basic PRP unit. The prompt in PRP is shown in Figure 2(a), wherein it asks the LLM to answer which document is more relevant to the query. Akin to (Qin et al., 2023), we denote this PRP prompt text as $u(q, d_i, d_j)$ for a query q and two documents d_i and d_j. As seen in Figure 2(b), the basic PRP unit, applied in existing work (Qin et al., 2023; Zhuang et al., 2023b), is only designed to obtain an answer to a question in the prompt. Specifically, there are two modes to obtain the answer, the first is generation mode wherein LLM directly generates the passage label ("Passage A" or "Passage B"), and the second is scoring mode wherein the log-likelihood of LLM generates the target label is used for a scoring comparison for the answer. As it is known that LLMs can be sensitive to text orders in the prompt (Lu et al., 2021), the order of these two documents within the prompt will be swapped and inquire the LLM twice, namely $u(q, d_i, d_j)$ and $u(q, d_j, d_i)$. Only if the output of two inquiries is consistent, the LLM's judgment will have a clear preference as shown in Figure 2(b).

Text re-ranking via basic PRP unit. Given a set of N candidate documents, two principal approaches leveraging the PRP unit to achieve re-ranking have emerged: the exhaustive Allpair method and the sorting-based method (Qin et al., 2023). The Allpair approach applies the PRP unit to every possible document pair, calculating a cumulative score for each document based on pairwise preferences, yet it incurs a substantial $O(N^2)$ computational cost and does not assure the most effective result according to previous work (Qin et al., 2023). On the other hand, sorting algorithms such as Heapsort and Bubblesort utilize LLM-derived pairwise preferences for ordering documents. These methods improve efficiency (e.g. Heapsort operates at $O(N \log N)$ complexity), but their effectiveness is bounded by the reduced comparative scope and sensitivity to the initial ranking order (Zhuang et al., 2023b).

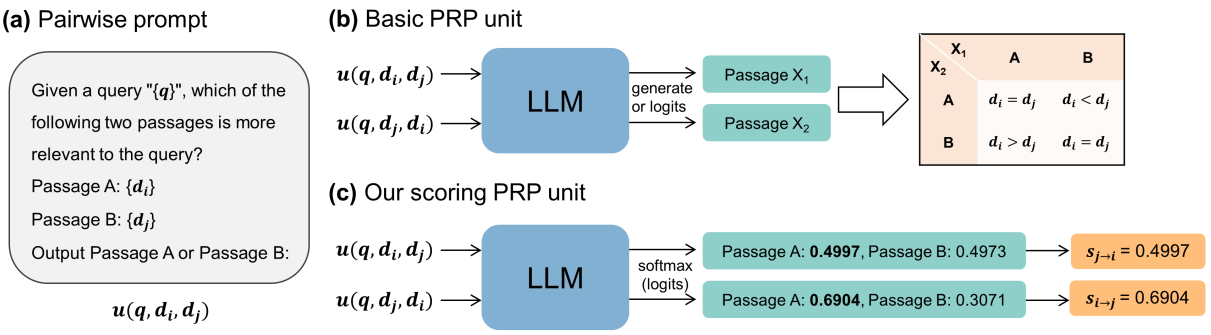

Figure 2: The introduction of PRP unit. **(a)** The pairwise prompt, for query q and the documents d_i and d_j involved in the comparison, is denoted as $u(q, d_i, d_j)$. **(b)** The basic PRP unit, as used in recent approaches, performs LLM inference twice for d_i and d_j in the transposition order then the results of the two inferences are integrated to make a judgment. **(c)** Our scoring PRP unit, considers the probability of generating "Passage A" as the probability that the current preorder document is more relevant to the query than the subsequent document.

3 Method

Our PRP-Graph introduces a refined scoring PRP unit that leverages LLMs' output probabilities for target labels, transforming these outputs into numerical scores for enhanced text ranking. The PRP-Graph operates in two main stages: ranking graph construction and ranking graph aggregation. In the first stage, document pairs are selectively compared to form a ranking graph with documents as vertices linked by bidirectional edges, where the scoring PRP unit informs the iterative updating of weights. The second stage performs an aggregation of these weighted interactions to derive the final document ranking, fully encapsulating the sorting information across the graph.

3.1 Scoring PRP Unit

Although a scoring mode exists within the basic PRP framework, as outlined in Section 2, it employs log-likelihood merely to reach a comparative decision, without utilizing the full extent of the numerical information provided. In contrast, our new scoring PRP unit, illustrated in Figure 2(c), capitalizes on two probability values from the LLM outputs to gauge the degree of comparison between documents.

Mirroring the basic PRP, our scoring PRP unit also prompts the LLM twice with both $u(q, d_i, d_j)$ and $u(q, d_j, d_i)$, swapping the document positions. During decoding, a *softmax* function is applied to the LLM's logit outputs. The resulting probability of generating label "A" serves as the score of "Passage A" over "Passage B" in relevance to the query q. For instance, given the input $u(q, d_i, d_j)$, $s_{j\to i}$ represents the score that document d_i is more relevant compared to d_j. Hence, the scoring PRP unit yields a pair of scores reflecting relative relevance as described in Eq. 1, in contrast to the basic PRP unit's singular comparative outcome.

$$\begin{aligned} s_{j\to i} &= \text{softmax}(\text{LLM}(u(q, d_i, d_j)))[\text{"Passage A"}] \\ s_{i\to j} &= \text{softmax}(\text{LLM}(u(q, d_j, d_i)))[\text{"Passage A"}] \end{aligned} \tag{1}$$

wherein ["Passage A"] corresponds to the probability that LLM generates "A", as "Passage" serves as conditional probability or the input to the Decoder.

3.2 Ranking Graph Construction

Addressing the challenge of quantifying the significance of pairwise comparison outcomes in a full ranking list, we draw parallels from the world of international chess tournaments. In these tournaments, players partake in multiple rounds of one-on-one matches, with the Swiss-system (Csató, 2013) tournament being a prevalent format. Here, players are matched in each round according to their current standings, and over successive rounds, they accumulate points that contribute to their final rankings. Inspired by this system, we develop a ranking graph construction strategy, encapsulated in Algorithm 1, which mirrors the dynamic pairing and point accumulation process of the Swiss-system tournament to discern the importance of pairwise comparisons for document ranking.

Given a query q and an initially retrieved top-N document ranking list $\mathcal{D} = [d_1, ..., d_N]$ using a retrieval algorithm like BM25 (Robertson et al., 2009), we aim to construct a ranking graph $\mathcal{G}$ to collect pairwise ranking relationships between documents. This is carried out by performing pairwise comparisons of documents, facilitated by our scor-

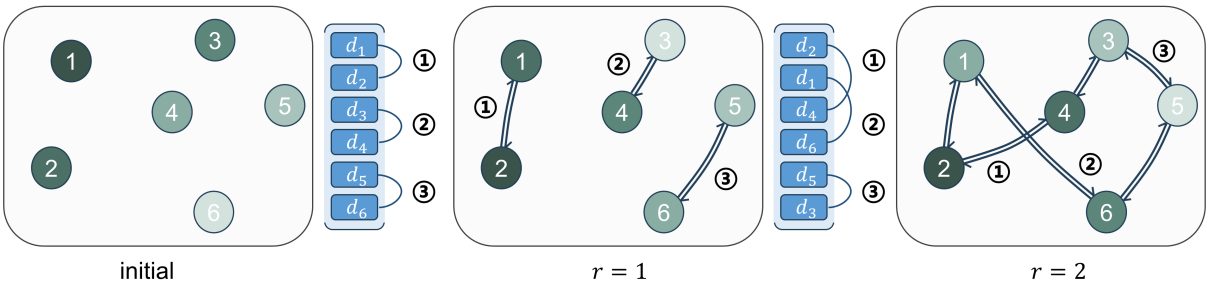

Figure 3: An example of building a ranking graph with six candidate documents ($N = 6$) and performing two comparison rounds ($R = 2$). The dots represents the corresponding d_i, the darker color represents the higher current ranking, the blue list represents the current ranking list, and the line between two documents in the list represents the comparison document pair selected according to the ranking graph construction algorithm.

ing PRP unit that leverages preference scores from an LLM. For this purpose, we conduct R rounds of comparisons across the ranking list $\mathcal{D}$. Each document's ranking score in the r-th round is denoted as $\mathcal{S}^r = [S_1^r, ..., S_N^r]$, with the initial scores set as $\mathcal{S}^0 = [1, 1 - \frac{1}{N}, 1 - \frac{2}{N}, \ldots, \frac{1}{N}]$ to reflect their starting position in the ranking list.

In each round, a document d_i is selected in sequence from the ranking list $\mathcal{D}$, and paired with the closest subsequent document d_j within the range $[d_{i+1}, \ldots, d_N]$ that has neither been compared with d_i in any previous round nor within the current round r. Upon selecting a pair of documents d_i and d_j, they are input into the scoring PRP unit, as described in Eq. 1, to obtain the preference scores, denoted as $s_{j \to i}$ and $s_{i \to j}$, from the LLM. Consequently, two edges are constructed within the graph $\mathcal{G}$, connecting d_j to d_i and d_i to d_j, each weighted by $s_{j \to i}$ and $s_{i \to j}$ respectively.

After the pairwise comparison between documents d_i and d_j in the r-th round, we utilize the preference scores, $s_{j \to i}$ and $s_{i \to j}$, derived from the LLM to adjust the vertex weights, S_i^r and S_j^r. The update adheres to the predefined score accumulation rule presented in Eq. 2.

$$
\begin{aligned}
S_i^r &= S_i^{r-1} + s_{j \to i} \times \frac{S_j^{r-1}}{r} \\
S_j^r &= S_j^{r-1} + s_{i \to j} \times \frac{S_i^{r-1}}{r}
\end{aligned}
\tag{2}
$$

The rationale for this score updating process is predicated on the notion that $s_{j \to i}$ represents the score secured by the pre-ranked document d_i against its subsequent counterpart d_j. To account for the varying challenges posed by different documents, the difficulty level associated with d_j, denoted as S_j^{r-1}/r, is incorporated into the calculation. This quantity reflects the normalized weight of d_j from the previous round and scales the points added to d_i based on the relative difficulty of the comparison. In cases where the PRP scores are equivalent, the adjustment rewards the documents proportionally to the strength of their adversaries, thereby ensuring a more nuanced representation of each comparison's contribution within the global ranking context.

After each comparison round, the document list $\mathcal{D}$ is updated and sorted by the new scores $\mathcal{S}^r$ from highest to lowest. Once all R rounds are complete, we create the final ranking graph $\mathcal{G}$. In this graph, documents are the vertices, and the comparison scores $s_{j \to i}$ and $s_{i \to j}$ are the weights of the two directed edges, representing the pairwise ranking relationships, between them. These edge weights are fixed after being set, since each document pair is compared only once. Figure 3 illustrates how this ranking graph is constructed.

3.3 Ranking Graph Aggregation

In the development of a ranking graph $\mathcal{G}$ for a particular query q across a suite of documents $\mathcal{D}$, initial ranking outcomes are derived from the cumulative scores $\mathcal{S}^R$. However, these results are not yet refined by the structural nuances of the graph, which are essential for weighting each document's ranking score more effectively. To address this, we introduce a graph-based aggregation method, conceptualized from the PageRank algorithm (Xing and Ghorbani, 2004). Unlike the traditional application of PageRank, which assigns importance to web pages based on the quantity of links, our adaptation focuses on the interplay of edge weights in the graph to revise the scores of the vertices according to Eq. 3.

Within the constructed ranking graph $\mathcal{G}$ for query q, each document is represented as a vertex i, cor-

Algorithm 1 Ranking Graph Construction

Input: a query q, a top-N document ranking list $\mathcal{D} = [d_1, \ldots, d_N]$ for q, the scoring PRP unit based on LLM

Parameter: the number of documents N, the total rounds of comparisons R, the scores of $\mathcal{D}$ in r-th round $\mathcal{S}^r = [S_1^r, \ldots, S_N^r]$ initialized with $\mathcal{S}^0 = [1, 1-\frac{1}{N}, 1-\frac{2}{N}, \ldots, \frac{1}{N}]$

Output: a ranking graph $\mathcal{G}$ on $\mathcal{D}$ for q

for r from 1 to R **do**
  for i from 1 to N **do**
    if d_i has not been compared in r-th round yet **then**
      for j from $i+1$ to N **do**
        if d_j has not been compared in r-th round and has also not been compared with d_i before **then**
          break
        end if
      end for
      $s_{j\to i}, s_{i\to j}$ = Scoring PRP Unit (q, d_i, d_j), as in Eq. 1
      Add two edges between d_i and d_j into $\mathcal{G}$ with $s_{j\to i}$ and $s_{i\to j}$ as the weights of edges
      $S_i^r = S_i^{r-1} + s_{j\to i} \times \frac{S_j^{r-1}}{r}$
      $S_j^r = S_j^{r-1} + s_{i\to j} \times \frac{S_i^{r-1}}{r}$
    end if
  end for
  Re-sort $\mathcal{D}$ by $\mathcal{S}^r$ from highest to lowest
end for
return the ranking graph $\mathcal{G}$

responding to document d_i. The vertex value $s(i)$, indicative of the aggregated score of d_i, is initially set by its BM25 score to incorporate precise matching signals. Subsequently, vertex values are iteratively recalculated following the order of descending BM25 scores as per Eq. 3.

$$s(i) = df \times [\sum_{j \in In(i)} \frac{s(j)}{\sum_{k \in Out(j)} w_{jk}} \times w_{ji}] + \frac{1-df}{N} \tag{3}$$

where N denotes the total number of vertices within $\mathcal{G}$, df is the damping factor, $In(i)$ signifies the set of vertices with edges directed towards vertex i, $Out(j)$ includes vertices with edges emanating from vertex j, and w_{ij} is the weight of the edge linking vertex i to vertex j.

The iterative process persists until the fluctuation in vertex ranking scores is below a specified threshold δ, whereupon the results are deemed to have stabilized. The final vertex scores then serve as the basis for re-ranking the documents in $\mathcal{D}$. In line with the methods outlined in (Wang et al., 2021), our approach, inherently semantic and generative, is best utilized in conjunction with a precision-oriented matching method like BM25. This hybridized strategy is optimized to achieve a balance between semantic coherence and precise matching. In situations where a development set is not available, we determine the interpolation ratio through cross-validation.

4 Experiments

4.1 Experimental Setup

Benchmark and metric To evaluate the effectiveness of our PRP-Graph on zero-shot text re-ranking, we carry out a series of experiments on BEIR benchmark (Thakur et al., 2021). BEIR is a well-established document ranking benchmark composed of multiple datasets, which cover a wide range of domains, like news and medicine, and involve various retrieval tasks, such as question answering and entity search. Details regarding the datasets we employed can be found in Appendix A. In the evaluation, each approach is tasked with re-ranking 100 documents initially retrieved by a first-stage BM25 retriever (Robertson et al., 2009). The performance of various approaches is assessed using NDCG@10, which serves as the official evaluation metric for the employed BEIR benchmark.

Baselines We consider two prevailing PRP-based methods as our comparative baselines: Allpair and Sorting-based comparisons as in (Qin et al., 2023; Zhuang et al., 2023b). For sorting-based comparisons, we have selected HeapSort and BubbleSort algorithms, which are founded on time-honored sorting techniques and have been validated for their exceptional effectiveness (Zhuang et al., 2023b). Our implementation replicates the experimental framework employed by (Zhuang et al., 2023b). For a fair comparison, we interpolate both the baseline methods and our PRP-Graph with sparse retrieval model BM25, and the interpolation parameters are determined by ten rounds of cross-checking, which determines parameters on a separate validation set in each fold.

	Method	Covid	Robust04	Touche	SciFact	Signal	News	DBPedia	NFCorpus	FiQA	HotpotQA	NQ	Avg. (w/o IB)
	BM25	59.5	40.7	44.2	67.9	33.1	39.5	31.8	32.2	23.6	63.3	30.6	42.4
Flan-T5-L	HeapSort	76.1	44.4	44.9	70.0	34.5	42.9	41.3	33.9	31.4	67.6	51.0	48.9 (45.8)
	BubbleSort	71.4	43.9	45.4	69.0	36.2	44.0	41.6	34.1	29.7	66.6	49.5	48.3 (48.0)
	Allpair	77.0	47.5	44.0	70.4	36.1	44.2	43.8	34.1	33.2	68.3	52.7	50.1 (45.3)
	PRP-Graph-10	76.3	48.6	44.5	70.9	35.8	47.7	44.9	34.7	33.2	68.3	53.1	50.7 (47.2)
	PRP-Graph-20	78.1	49.1	44.5	70.4	35.7	47.8	45.6	35.3	33.7	68.2	54.7	51.2 (47.7)
	PRP-Graph-40	78.8	48.9	45.8	70.7	35.8	48.0	45.8	35.2	34.5	68.2	55.5	**51.5** (47.7)
Flan-T5-XL	HeapSort	77.9	55.0	43.9	73.7	35.5	47.1	41.7	35.2	38.3	69.8	55.7	52.2 (49.9)
	BubbleSort	76.3	55.3	44.0	73.4	35.9	48.6	43.2	35.9	38.3	69.5	55.3	52.3 (51.8)
	Allpair	77.7	54.8	44.7	72.2	35.1	48.6	43.0	35.9	38.2	70.1	56.6	52.4 (49.9)
	PRP-Graph-10	75.9	53.3	45.2	74.2	35.7	48.5	42.8	35.6	36.9	71.8	55.3	52.3 (49.3)
	PRP-Graph-20	77.0	54.0	45.6	74.8	35.9	49.0	43.9	35.4	38.4	72.3	56.3	53.0 (50.0)
	PRP-Graph-40	77.5	54.5	44.8	74.4	36.1	50.5	44.6	35.7	38.8	72.4	57.0	**53.3** (50.5)
Flan-T5-XXL	HeapSort	73.8	54.3	44.8	73.8	34.3	47.1	40.5	35.4	40.0	70.9	54.9	51.8 (50.6)
	BubbleSort	73.3	55.0	45.3	75.5	34.5	49.1	42.0	36.2	39.6	70.5	54.6	52.3 (52.2)
	Allpair	76.4	55.4	44.9	74.7	34.1	49.7	41.1	36.0	40.5	71.5	55.9	52.7 (50.4)
	PRP-Graph-10	76.3	53.2	43.9	75.6	35.6	47.2	42.6	36.0	39.7	71.8	54.8	52.4 (48.8)
	PRP-Graph-20	76.7	54.0	42.5	74.2	36.0	49.7	43.4	36.2	40.3	72.5	56.1	52.9 (49.5)
	PRP-Graph-40	78.1	54.2	45.8	75.4	35.8	49.3	43.7	36.5	41.2	72.8	56.6	**53.6** (50.2)

Table 1: NDCG@10 on BEIR for different PRP re-ranking approaches on different sizes of model. The best results are highlighted in boldface. "w/o IB" denotes the results are not interpolated with BM25 results.

Implementation details For the retrieval of the top-100 candidate documents, we employ the Pyserini Python library (Lin et al., 2021) with default settings to utilize the BM25 model. In line with previous works (Qin et al., 2023; Zhuang et al., 2023b), we conduct evaluations on Flan-T5 models of three sizes [1], including Flan-T5-Large (L) with 780M parameters, Flan-T5-XL with 3B parameters, and Flan-T5-XXL with 11B parameters, to explore the effectiveness of our method on LLMs with different parameter sizes. Besides, we use the same prompt following the existing literature, as introduced in Section 2, for all PRP-based re-ranking methods. Regarding the number of comparison rounds R performed at the ranking graph construction step, we mainly report the evaluation results when $R = 10, 20$, and 40, and put the exploration of the setting of R for different comparison rounds in the Analysis Section 4.3. For the ranking graph aggregation step, the damping factor df in Eq. 3 is conventionally set to 0.85, and the convergence threshold δ is set to 1e-6.

4.2 Results

Table 1 presents an empirical evaluation of different PRP-based re-ranking approaches on the BEIR benchmark. In the table, HeapSort, BubbleSort, and Allpair are the existing PRP-based re-ranking methods as baselines. We report the results of our PRP-Graph method for $R = 10$, $R = 20$, and $R = 40$, denoted as PRP-Graph-10, PRP-Graph-20, and PRP-Graph-40, respectively. To compare the results before and after interpolation with BM25 results, we also include the average uninterpolated results, labeled as **"w/o IB"**.

PRP-Graph demonstrates a clear advantage over current PRP approaches in different model sizes. On three different sizes of models Flan-T5-L, Flan-T5-XL, and Flan-T5-XXL, PRP-Graph-10 with a small number of comparison rounds can perform on par with the best baseline Allpair, while PRP-Graph-40 with more comparison rounds outperforms Allpair by margins of +1.4, +0.9, and +0.9 in terms of average NDCG@10, respectively. An exploration of the number of comparison rounds R in the ranking graph construction step can be found in Section 4.3.

PRP-Graph demonstrates a more effective interpolation with BM25. As shown in Table 1, the interpolation with BM25 provides a certain improvement for all PRP methods. However, compared to all baselines, PRP-Graph demonstrates a more effective interpolation with BM25. For example, PRP-Graph-40 obtains a +3.4 gain, while HeapSort, BubbleSort, and Allpair obtain +1.2, +0.1, and +2.3 at Flan-T5-XXL model after the interpolation. This enhancement can be attributed to the graph-based nature of PRP-Graph, which relies less on the initial rankings produced by BM25. In contrast, HeapSort and BubbleSort are dependent on the BM25 rankings throughout their entire sorting process. This also explains the outstanding performance of Allpair among baselines, since it does not take initial ranking into account.

[1] https://huggingface.co/docs/transformers/model_doc/flan-t5

Method	$R = 10$	$R = 20$	$R = 40$
PRP-Graph	50.7	51.2	51.5
w/o nearest selection	50.6	51.1	51.2
w/o bidirectional edge (larger)	48.7	49.0	49.9
w/o bidirectional edge (subtract)	48.6	48.7	49.3
w/o exact matching initialization	50.3	50.7	51.1
w/o ranking aggregation	50.3	50.7	51.1

Table 2: Ablation results of PRP-Graph with Flan-T5-Large model on BEIR benchmark.

4.3 Analysis

Ablation study. Herein, to verify the effectiveness of different components or settings in our PRP-Graph method, we design the following experiments as shown in Table 2. During the construction of the ranking graph, **"w/o nearest selection"** denotes the document comparison pairs in each round are randomly selected during the graph construction rather than finding the nearest documents in the ranking list (as in lines 4 through 8 in Algorithm 1), which exhibits a discernible reduction of effectiveness in NDCG@10; **"w/o bidirectional edge (larger)"** and **"w/o bidirectional edge (subtract)"** denote unidirectional edges rather than bidirectional edges are conducted in graph $\mathcal{G}$ using the larger weight and the subtraction between two weights, respectively. Their performance drops substantially, which indicates the benefit of our choice of bidirectional edges to build the ranking graph. As for the ranking graph aggregation, **"w/o exact matching initialization"** denotes initializing the weights of vertices in graph $\mathcal{G}$ with the final score $\mathcal{S}^R$ from the graph construction phase rather than the BM25 exact matching scores; **"w/o ranking aggregation"** denotes the results only after the construction of the ranking graph without the aggregation step, namely using the ranking scores in $\mathcal{S}^R$ interpolated with BM25. Both suffered the same degree of performance decline.

The impact of comparison rounds. During the ranking graph construction stage, we performed R rounds of pairwise comparison of candidate documents. To understand the influence of the number of R on the ranking efficacy, we conduct a comprehensive empirical study, the results of which are encapsulated in Figure 4. This figure illustrates the ranking performance (average NDCG@10) on the BEIR benchmark when subjected to varying comparison round configurations for different model sizes. We also mark the results of Allpair, the best-performing baseline, in the figure to illustrate

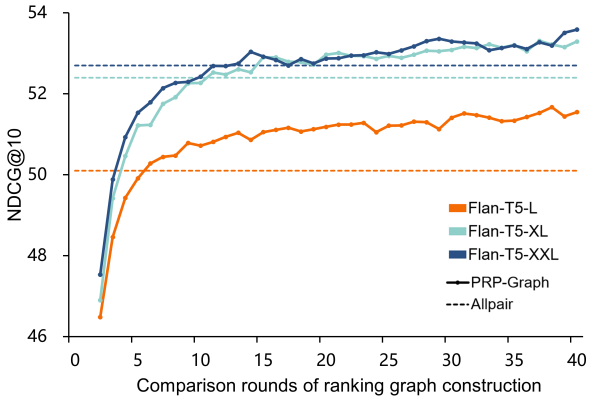

Figure 4: Results on BEIR benchmark of PRP-Graph with different rounds R of ranking graph construction.

Method	NDCG@10	#Inference
HeapSort	48.9	213.2
BubbleSort	48.3	746.3
Allpair	50.1	4908.2
PRP-Graph-10	50.7	489.1
PRP-Graph-20	51.2	965.5
PRP-Graph-40	51.5	1843.2

Table 3: The effectiveness and efficiency trade-offs on BEIR benchmark for PRP-based methods on Flan-T5-L model. "#Inference" denotes the average number of LLM inferences per query.

how many rounds of comparison PRP-Graph needs to perform to outperform it. Our investigation revealed that there is a clear trend demonstrating an increase in the number of comparison rounds, progressing from 2 up to around 15. After 15 rounds, the ranking effect exhibits an improvement in oscillation with the increase of R.

The trade-offs between effectiveness and efficiency. As depicted in Table 3, HeapSort is efficient and presents acceptable trade-offs among baselines. BubbleSort, on the other hand, faces challenges in both efficiency and effectiveness domains. Allpair, while demonstrating superior effectiveness compared to the two methods mentioned earlier, incurs significant overhead and still has room for improvement. In contrast, our PRP-Graph significantly surpasses these methods by achieving a better balance between ranking effectiveness and computational efficiency. Notably, PRP-Graph-10 provides a +0.6 improvement in NDCG@10 over the leading Allpair baseline while concurrently reducing the computational overhead by 90.0%. Furthermore, for scenarios where ranking effectiveness is paramount, the PRP-Graph provides flexibility to increase the number of comparison rounds, as

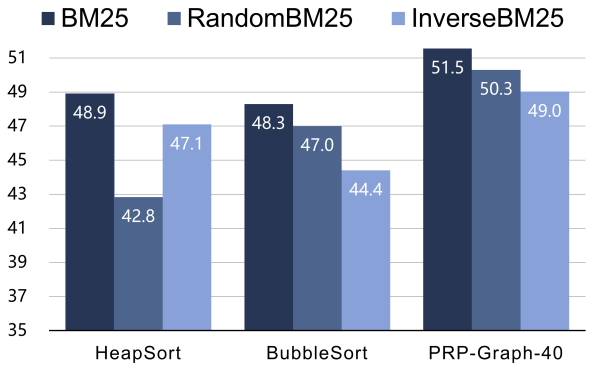

Figure 5: Results of different initial ranking order.

evidenced by PRP-Graph-20 and PRP-Graph-40, which yield even more precise ranking outcomes.

The sensitivity to the initial ranking. The effectiveness of sorting methods is notably influenced by the order of the input rankings (Qin et al., 2023; Zhuang et al., 2023b). To delve into this aspect concerning our approach, we explore three types of orderings of the initial document list. Our results on Flan-T5-L are illustrated in Figure 5. Specifically, "BM25" denotes the ranking order retrieved by a first-stage BM25 retriever, "RandomBM25" denotes the randomly shuffled BM25 ranking, and "InverseBM25" denotes the inverted BM25 ranking. It is observed that diverse initial ranking orders hurt both the sorting method and our PRP-Graph. But PRP-Graph's overall impact is smaller and still maintains the best results. We also show the uninterpolated results in Appendix B.

5 Related Work

In the realm of zero-shot text re-ranking with LLMs, there are three main prompting approaches categorized broadly into Pointwise (Liang et al., 2022; Sachan et al., 2022), Listwise (Ma et al., 2023; Pradeep et al., 2023; Sun et al., 2023), and Pairwise (Qin et al., 2023; Zhuang et al., 2023b). Pointwise approaches rank candidate documents based on prompting LLMs for query-document pair relevance judgment, such as prompting LLMs to generate whether the provided query-document pairs are relevant (Liang et al., 2022; Zhuang et al., 2023a) and using the LLMs generation likelihood of query for the corresponding document as ranking score (Muennighoff, 2022; Sachan et al., 2022). Listwise approaches involve sorting a list of documents at once, wherein LLMs are prompted to generate a ranked list of document labels based on the relevance of their corresponding documents to the query. Current listwise approaches (Sun et al., 2023; Ma et al., 2023) use sliding window approaches to work within the input length limits of LLMs, involving re-ranking the candidate document window from the bottom of the entire candidate documents list and progressing upwards.

Pairwise approaches determine the relevance order of two documents for a query at a time, wherein LLMs are prompted with a query alongside a pair of documents to choose the label indicating which document is more relevant to the query, namely Pairwise Ranking Prompting (PRP). Exiting methods for obtaining the final re-ranking through pairwise comparisons between candidate documents can be classified into *aggregation* and *sorting* methods. Allpair is a basic aggregation method (Qin et al., 2023), which conducts pairwise comparisons on all possible pairings within the candidate documents and aggregates the result of each comparison into the ranking score of the document. Other aggregation methods exist focusing on sampling from all possible pairings to increase efficiency (Mikhailiuk et al., 2021; Gienapp et al., 2022), but it hasn't been used in LLMs yet. Sorting methods rely on pairwise comparisons as a basic comparison unit to run sorting algorithms, such as Heapsort and Bubblesort (Qin et al., 2023; Zhuang et al., 2023b). These algorithms make use of efficient data structures to compare document pairs selectively.

However, existing methods only utilize PRP to obtain the same label for the comparison results of different confidence intervals without considering the uncertainty of pairwise comparisons. In contrast, our proposed PRP-Graph, based on a new scoring PRP unit that quantifies the certainty of the pairwise comparisons from LLMs, exploits the potential of LLMs on zero-shot re-ranking.

6 Conclusion

In our work, we propose a novel PRP-Graph approach based on a refined scoring Pairwise Ranking Prompting (PRP) unit to promote the potential of LLMs on text re-ranking tasks. Benefiting from the consideration of the uncertainty of pairwise comparison in scoring PRP unit, the PRP-Graph builds a ranking graph via the comparison results of the scoring PRP unit and aggregates global signals of the ranking graph, surpassing existing PRP-based zero-shot re-ranking techniques on the BEIR benchmark. In future work, we plan to explore a more efficient ranking graph aggregation strategy.

Limitations

Firstly, in the ranking graph aggregation stage of PRP-Graph, we only use an aggregation method based on weighted PageRank, more other graph learning strategies could be included to extend the effectiveness of PRP-Graph, such as the HITS algorithm (Sherin et al., 1998). Secondly, in the ranking graph construction step of PRP-Graph, the number of comparison rounds R we explore is limited to 40 for efficiency reasons. Even though our analysis of comparison rounds shows the effectiveness of PRP-Graph, more rounds are still unexplored. Finally, we conducted experiments on the Flan-T5 series of models in line with previous works (Qin et al., 2023; Zhuang et al., 2023b). We can also extend the experiments to a more diverse range of LLMs, such as LLaMA (Touvron et al., 2023), to evaluate our approach.

References

Monica Agrawal, Stefan Hegselmann, Hunter Lang, Yoon Kim, and David Sontag. 2022. Large language models are zero-shot clinical information extractors. *arXiv preprint arXiv:2205.12689*.

Alexander Bondarenko, Maik Fröbe, Meriem Beloucif, Lukas Gienapp, Yamen Ajjour, Alexander Panchenko, Chris Biemann, Benno Stein, Henning Wachsmuth, Martin Potthast, et al. 2020. Overview of touché 2020: argument retrieval. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction: 11th International Conference of the CLEF Association, CLEF 2020, Thessaloniki, Greece, September 22–25, 2020, Proceedings 11*, pages 384–395. Springer.

Vera Boteva, Demian Gholipour, Artem Sokolov, and Stefan Riezler. 2016. A full-text learning to rank dataset for medical information retrieval. In *Advances in Information Retrieval: 38th European Conference on IR Research, ECIR 2016, Padua, Italy, March 20–23, 2016. Proceedings 38*, pages 716–722. Springer.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

László Csató. 2013. Ranking by pairwise comparisons for swiss-system tournaments. *Central European Journal of Operations Research*, 21:783–803.

Lukas Gienapp, Maik Fröbe, Matthias Hagen, and Martin Potthast. 2022. Sparse pairwise re-ranking with pre-trained transformers. In *Proceedings of the 2022 ACM SIGIR International Conference on Theory of Information Retrieval*, pages 72–80.

Faegheh Hasibi, Fedor Nikolaev, Chenyan Xiong, Krisztian Balog, Svein Erik Bratsberg, Alexander Kotov, and Jamie Callan. 2017. Dbpedia-entity v2: a test collection for entity search. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1265–1268.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

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: An easy-to-use python toolkit to support replicable ir research with sparse and dense representations. *arXiv preprint arXiv:2102.10073*.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2021. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. *arXiv preprint arXiv:2104.08786*.

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*.

Macedo Maia, Siegfried Handschuh, André Freitas, Brian Davis, Ross McDermott, Manel Zarrouk, and Alexandra Balahur. 2018. Www'18 open challenge: financial opinion mining and question answering. In *Companion proceedings of the the web conference 2018*, pages 1941–1942.

Aliaksei Mikhailiuk, Clifford Wilmot, Maria Perez-Ortiz, Dingcheng Yue, and Rafał K Mantiuk. 2021. Active sampling for pairwise comparisons via approximate message passing and information gain maximization. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 2559–2566. IEEE.

Niklas Muennighoff. 2022. Sgpt: Gpt sentence embeddings for semantic search. *arXiv preprint arXiv:2202.08904*.

Ronak Pradeep, Sahel Sharifymoghaddam, and Jimmy Lin. 2023. Rankvicuna: Zero-shot listwise document reranking with open-source large language models. *arXiv preprint arXiv:2309.15088*.

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*.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

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*.

Kevin M Sherin, James M Sinacore, Xiao-Qiang Li, Robert E Zitter, and Amer Shakil. 1998. Hits: a short domestic violence screening tool for use in a family practice setting. *FAMILY MEDICINE-KANSAS CITY-*, 30:508–512.

Ian Soboroff, Shudong Huang, and Donna Harman. 2018. Trec 2018 news track overview. In *TREC*, volume 409, page 410.

Axel Suarez, Dyaa Albakour, David Corney, Miguel Martinez, and José Esquivel. 2018. A data collection for evaluating the retrieval of related tweets to news articles. In *Advances in Information Retrieval: 40th European Conference on IR Research, ECIR 2018, Grenoble, France, March 26-29, 2018, Proceedings 40*, pages 780–786. Springer.

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 heterogenous benchmark for zero-shot evaluation of information retrieval models. *arXiv preprint arXiv:2104.08663*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

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. Overview of the trec 2004 robust retrieval track.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020. Fact or fiction: Verifying scientific claims. *arXiv preprint arXiv:2004.14974*.

Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Douglas Burdick, Darrin Eide, Kathryn Funk, Yannis Katsis, Rodney Kinney, et al. 2020. Cord-19: The covid-19 open research dataset. *ArXiv*.

Shuai Wang, Shengyao Zhuang, and Guido Zuccon. 2021. Bert-based dense retrievers require interpolation with bm25 for effective passage retrieval. In *Proceedings of the 2021 ACM SIGIR international conference on theory of information retrieval*, pages 317–324.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Wenpu Xing and Ali Ghorbani. 2004. Weighted pagerank algorithm. In *Proceedings. Second Annual Conference on Communication Networks and Services Research, 2004.*, pages 305–314. IEEE.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.

Honglei Zhuang, Zhen Qin, Kai Hui, Junru Wu, Le Yan, Xuanhui Wang, and Michael Berdersky. 2023a. Beyond yes and no: Improving zero-shot llm rankers via scoring fine-grained relevance labels. *arXiv preprint arXiv:2310.14122*.

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*.

Dataset	Relevancy	#Query	Type
TREC-COVID	3-level	50	Bio-Medical IR
Robust04	Binary	249	News Retrieval
Touché-2020	3-level	49	Argument Retrieval
SciFact	Binary	300	Fact Checking
Signal	3-level	97	Tweet Retrieval
TREC-News	5-level	57	News Retrieval
DBPedia	3-level	400	Entity Retrieval
NFCorpus	3-level	323	Bio-Medical IR
FiQA-2018	Binary	648	Question-Answering
NQ	Binary	3452	Question-Answering
HotpotQA	Binary	7405	Question-Answering

Table 4: Statistics of the datasets.

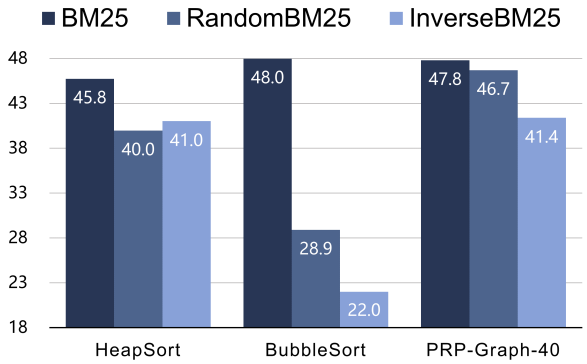

Figure 6: Uninterpolated results of different initial ranking order.

A Datasets

We conducted our experiments on eleven datasets within the BEIR benchmark (Thakur et al., 2021). **TREC-COVID** (Voorhees et al., 2021) is an ad-hoc search challenge based on the CORD-19 dataset,which containing scientific articles related to the COVID-19 pandemic (Wang et al., 2020). **Robust04** (Voorhees) offers a robust dataset, focusing on evaluating on poorly performing topics. **Touché-2020** (Bondarenko et al., 2020), Task 1, tackles a conversational argument retrieval task. **SciFact** (Wadden et al., 2020) verifies scientific claims using evidence from the research literature, including scientific paper abstracts. **Signal-1M** (Suarez et al., 2018) pertains to Tweets task, retrieving relevant tweets for a given news article title. **TREC-NEWS** (Soboroff et al., 2018) 2019 track involves background linking. **DBPedia-Entity-v2** (Hasibi et al., 2017) stands as an established entity retrieval dataset. **NFCorpus** (Boteva et al., 2016) comprises natural language queries harvested from NutritionFacts (NF). **FiQA-2018** (Maia et al., 2018), Tasks 2, involves opinion-based question-answering from StackExchange posts. **Natural Questions** (Kwiatkowski et al., 2019) contains Google search queries and documents with paragraphs and answer spans within Wikipedia articles. **HotpotQA** (Yang et al., 2018) contains multi-hop like questions which require reasoning over multiple paragraphs to find the correct answer. The statistics of the datasets are displayed in Table 4.

B Uninterpolated results of different initail ranking

To show the effect of interpolation on the sensitivity of the initial ranking order of the different methods, we show uninterpolated results of different initail ranking in Figure 6. Compared with the results after interpolation in Figure 5, different initial ranking improved after interpolation. Among them, BubbleSort is most affected by initial ranking before interpolation, and interpolation also improves it the most on "RandomBM25" and "InverseBM25". Our PRP-Graph-40 maintains a consistent distribution of results on both uninterpolated and interpolated results, and has the best results for both "RandomBM25" and "InverseBM25" among the uninterpolated results. This further indicates the robustness of our method to the initial ranking order.