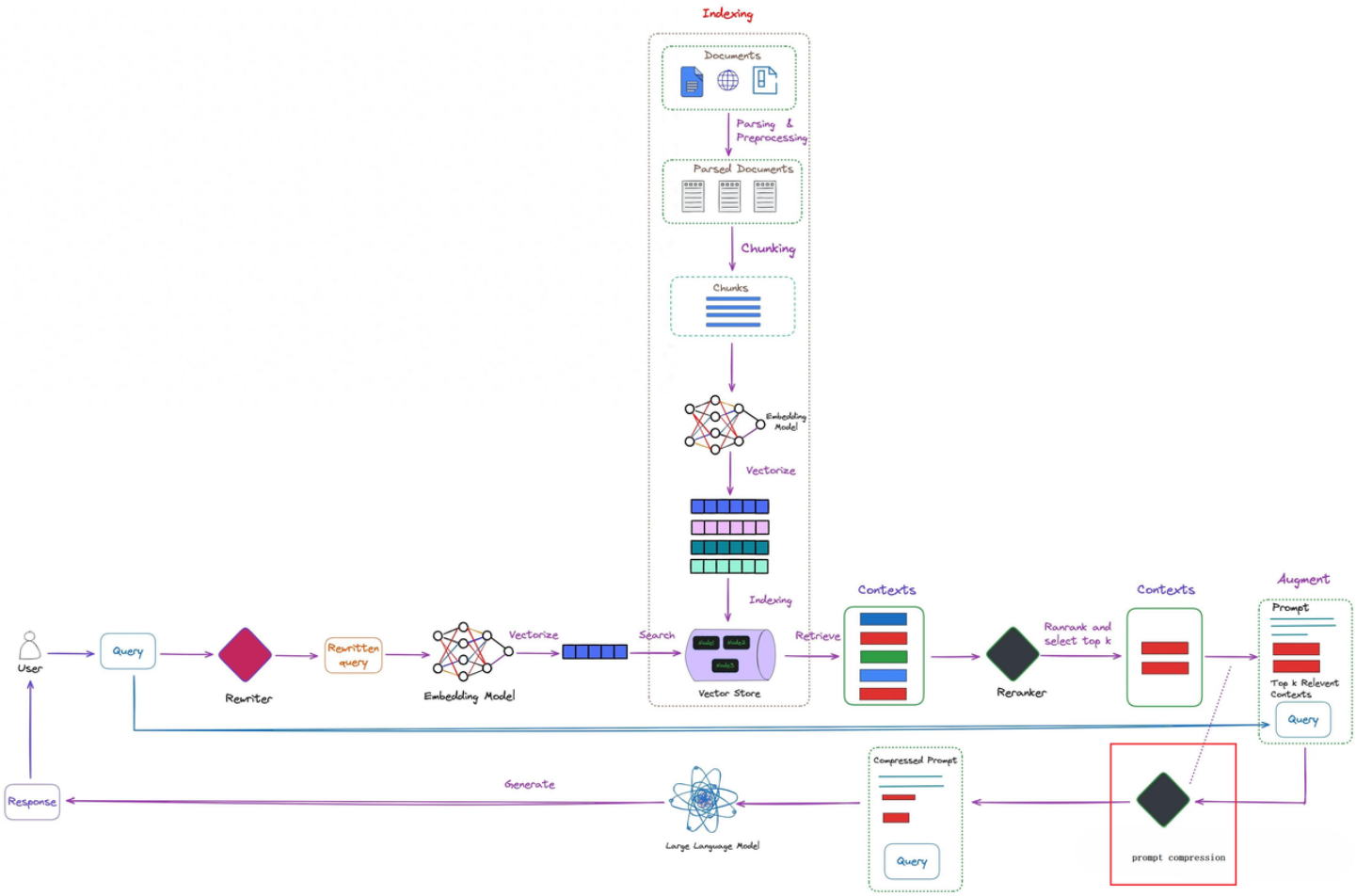
Rerank1

Rerank作为RAG模型中在retrieve和generation之间的一个重要环节，主要负责在大范围检索完成后对候选文档进行再精排序，从而提升最终大模型生成结果的质量和关联性。



rerank的作用主要包括：

- RAG粗排返回的文档质量和相关性都较差，rerank能过滤掉与用户问题相关性较低的文档，以及噪声和不相关的信息。
- rerank能帮助大模型更好地理解和利用检索到的信息，强化相关文档的影响，从而提升生成结果的相关性和准确性。

用LLM来做rerank

现有的涉及LLM的重排方法大致可以分为三类：微调LLM来重排，使用prompt让LLM进行重排，以及利用LLM做训练数据的增强。

微调LLM来重排

由于在LLM预训练阶段缺少rerank意识，因此需要在任务相关的排序数据集上进行微调（例如MS MARCO passage ranking dataset），这种数据集针对每个条目都包含了相关和不相关的信息。微调

LLM又可以分为以下两类：微调LLM做生成模型和重排模型

1、微调LLM做生成模型

给定用户问题query和检索到的内容document，微调LLM生成“真”或“假”标签来表示两者是否相关。在推理阶段，使用softmax函数对预测为“真”和“假”的概率进行处理，query和document的相关性就是针对softmax后预测为真的概率。

比如DuoT5方法是一种典型的排序微调方法。在该方法中，对于一组输入（查询q和两个候选文档 d_i 、 d_j ），模型会判断哪个文档更相关。如果文档 d_i 比文档 d_j 更接近查询q，则模型返回“真”，否则返回“假”。在推理阶段，模型会计算查询q与每个候选文档之间的相关性，或者计算一个候选文档与所有其他文档的相关性，并返回该文档的相关性值。最终，所有候选文档的相关性值会被计算平均值或直接排序，以确定查询的最终排序结果。

2、微调LLM做重排模型

微调LLM做生成模型面临以下问题：做rerank时希望模型输出数值而不是标签。因此考虑微调LLM做重排模型的方法，比如RankT5方法直接计算query和document之间的相关性，并使用pairwise或listwise排序损失来优化。

使用prompt让LLM进行重排

随着大模型参数量的激增，微调大模型也变得困难。可以通过prompt工程来提升rerank效果，这种方法可以分为三种： pointwise, listwise, pairwise。

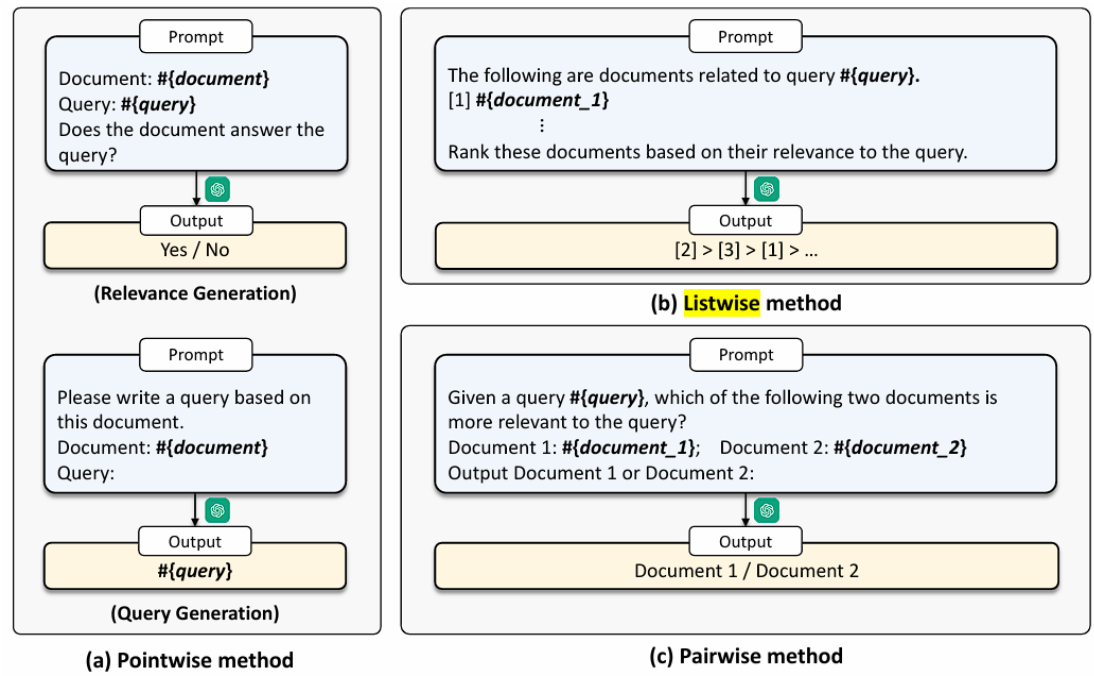


Fig. 6. Three types of prompting-based reranking methods: (a) pointwise methods that consist of relevance generation (upper) and query generation (lower), (b) listwise methods, and (c) pairwise methods.

1、Pointwise

relevance generation方法直接要求LLM输出“真”或“假”，query-document的相关性分数为：

$$\text{score} = \begin{cases} 1 + p(\text{Yes}), & \text{if LLMs output Yes} \\ 1 - p(\text{No}), & \text{if LLMs output No} \end{cases} \quad (1)$$

query generation基于document生成一个预测query，然后使用生成的真实query中token的平局对数似然来衡量相关性分数：

$$\text{score} = \frac{1}{|q|} \sum_i \log p(q_i | q_{<i}, d, \mathcal{P}), \quad (2)$$

其中|q|表示query的token数，d表示document，P表示预测prompt。

- 举例：Discrete Prompt Optimization via Constrained Generation for Zero-shot Re-ranker

定义 $\rho*$ 作为指导LLM生成最接近于用户query的prompt：

$$\rho^* = \arg\max_{\rho} \mathbb{E}_{(d_i, q_i) \in D}[P(q_i | d_i, \rho)], \quad (2)$$

其中D包含了所有的用户query和其对应的检索到的相关document。为了解决寻找最优prompt $\rho*$ 的问题，本文用基于鉴别器的条件生成方法解决，该方法遵循贝叶斯公式：

$$P(\rho_t | D, \rho_{1:t-1}) \propto P_{M_D}(D_s | \rho_{1:t}) P_{M_G}(\rho_t | \rho_{1:t-1}), \quad (3)$$

其中 M_D 是zero-shot的重拍器作为鉴别器， M_G 作为decoder-only的大模型作为生成器， D_s 为数据集D的子集。

鉴别器 M_D 用于衡量prompt能否指导大模型生成好的query， $P_{M_D}(D_s|\rho)$ 表示query-document对 (q_i, d_i)之间的相关性期望:

$$P_{M_D}(D_s | \rho) = \mathbb{E}_{(d_i, q_i) \in D_s}[P_{M_D}(q_i | d_i, \rho)]. \quad (4)$$

由于直接计算公式3中词表中所有的token耗时过长，因而生成器 M_G 只从鉴别器衡量过的prompt进行采样。生成器采用beam search的方式选取每一轮的token。整体训练如下：

Algorithm 1: Co-Prompt: a beam search-based prompt generation algorithm with a discriminator and a generator. D_s: document-query pairs, B: beam width, L: maximum prompt length, N: the number of final prompts, $\mathcal{V}$: vocabulary set

Require: $D_s, B, L, \mathcal{V}$
begin
  $P_1 \leftarrow \{\text{Start-Token}\}$
  for $t = 1, \ldots, L$ **do**
    $P_{t+1} \leftarrow \emptyset$
    foreach $\rho_{1:t} \in P_t$ **do**
      $S_{t+1} \leftarrow \underset{K=B, \rho_{t+1} \in \mathcal{V}}{topK} P_{M_G}(\rho_{t+1} | \rho_{1:t})$
      $P_{t+1} \leftarrow P_{t+1} \cup \{\rho_{1:t+1} | \rho_{1:t} \oplus \rho_{t+1} \in S_{t+1}\}$
    end
    $P_{t+1} \leftarrow \underset{K=B, \rho_{1:t+1} \in P_{t+1}}{topK} P_{M_D}(D_s | \rho_{1:t+1})$
  end
  $P \leftarrow \cup_{t \in [1,L]} P_t$
  $R \leftarrow \underset{K=N, \rho \in P}{topK} P_{M_D}(D_s | \rho)$
  return R
end

2、Listwise

LLM的输入是用户query和一些检索到的document，要求LLM对其进行排序。由于LLM的上下文长度有限，这里也会采用一些Longcentext的方法（例如滑动窗口分批次排序）。使用GPT-4做LLM的方法取得了比较好的性能。

- 举例：Zero-Shot Listwise Document Reranking with a Large Language Model

作者提出使用如下prompt来让LLM实现document的重排，方括号后生成一系列按相关性重新排序后的passage id。为了解决输入长度的限制，作者采用**滑动窗口**的方法。

```
Passage1 = {passage_1}
...
Passage10 = {passage_10}
Query = {query}
Passages = [Passage1, ..., Passage10]
Sort the Passages by their relevance to the Query.
Sorted Passages = [
```

3、Pairwise

Listwise的方法只有在使用很大的模型（例如GPT4）才能取得良好的性能，并且对document在prompt中的顺序敏感，当document随机时，其效果甚至差于BM25。

pairwise方法利用了大模型天生擅长做对比的特点，采用一些ranking算法来对所有document进行排序。

- 举例：Large Language Models are Effective Text Rankers with Pairwise Ranking Prompting

本文提出的pairwise ranking prompting (PRP)支持生成式和打分式的输出，但是生成式可能生成无关内容，所以主要讨论生成式。PRP的输入为入为 $u(q, d_1, d_2)$ 的三元组形式，并且利用LLM对输入顺序敏感的特点，同一个三元组会变换顺序输入到模型两次，若两次结果相反，则认为两个document得分一样。

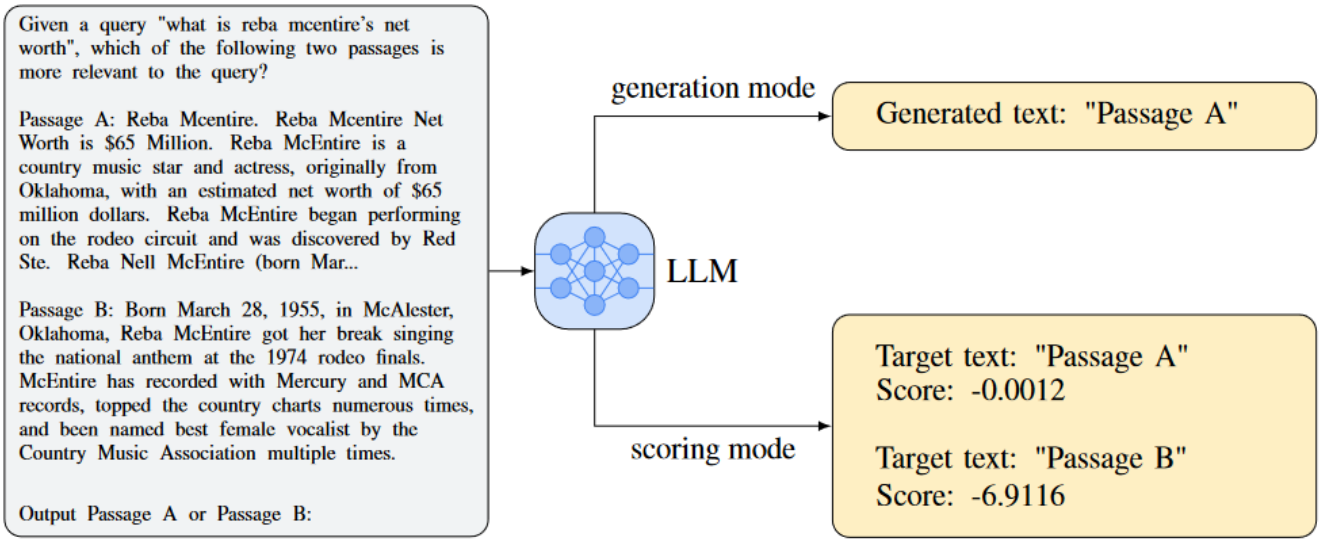


Figure 2: An illustration of pairwise ranking prompting. The scores in scoring mode represent the log-likelihood of the model generating the target text given the prompt.

基于PRP，本文还提出三种变体：

1. PRP-Allpair：对所有的document都进行比较，缺点是时间复杂度 $O(N^2)$
2. PRP-Sorting:使用快排或者堆排等算法，时间复杂度 $O(NlogN)$
3. PRP-Sliding-K:类似于冒泡排序，但是由于rerank只关心top K的文档，这里K比较小，总体复杂度 $O(KlogN)$

利用LLM做训练数据的增强

即利用LLM（如GPT）来生成或增强训练数据，并通过这些增强的数据来训练或优化排序模型。一些典型方法如下：

1. **ExaRanker**：利用GPT生成查询-文档对的解释，训练seq2seq排序模型。
2. **InPars-Light**：通过prompt要求大模型根据文档生成查询。
3. **ChatGPT-RetrievalQA**：基于用户查询生成文档。
4. **GPT生成排名列表迁移到小模型**：用GPT生成文档的排序列表，然后训练较小的模型。

参考：

1. Large Language Models for Information Retrieval: A Survey
2. Discrete Prompt Optimization via Constrained Generation for Zero-shot Re-ranker
3. Zero-Shot Listwise Document Reranking with a Large Language Model

4. Large Language Models are Effective Text Rankers with Pairwise Ranking Prompting