R²LLMs: Retrieval and Ranking with LLMs

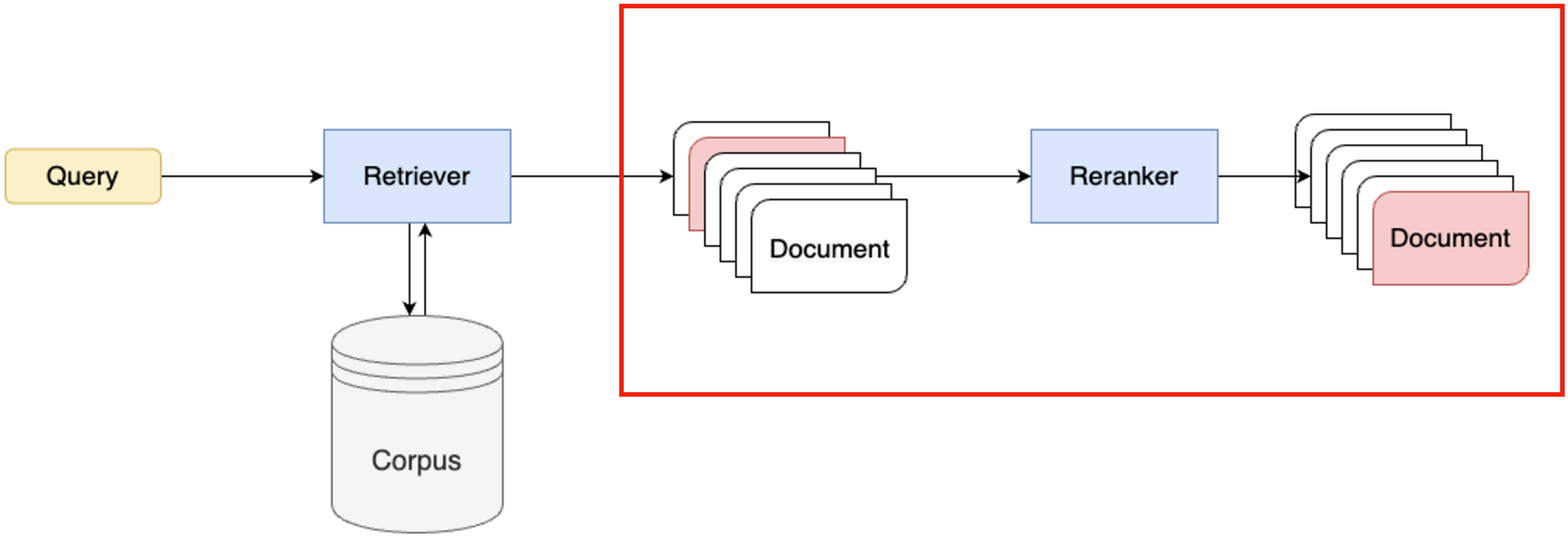
Guido Zuccon[1], **Shengyao** Zhuang[2], **Xueguang** Ma[3]

[1] ielab, The University of Queensland, Australia & Google Research Australia
[2] ielab, CSIRO, Australia
[3] The University of Waterloo

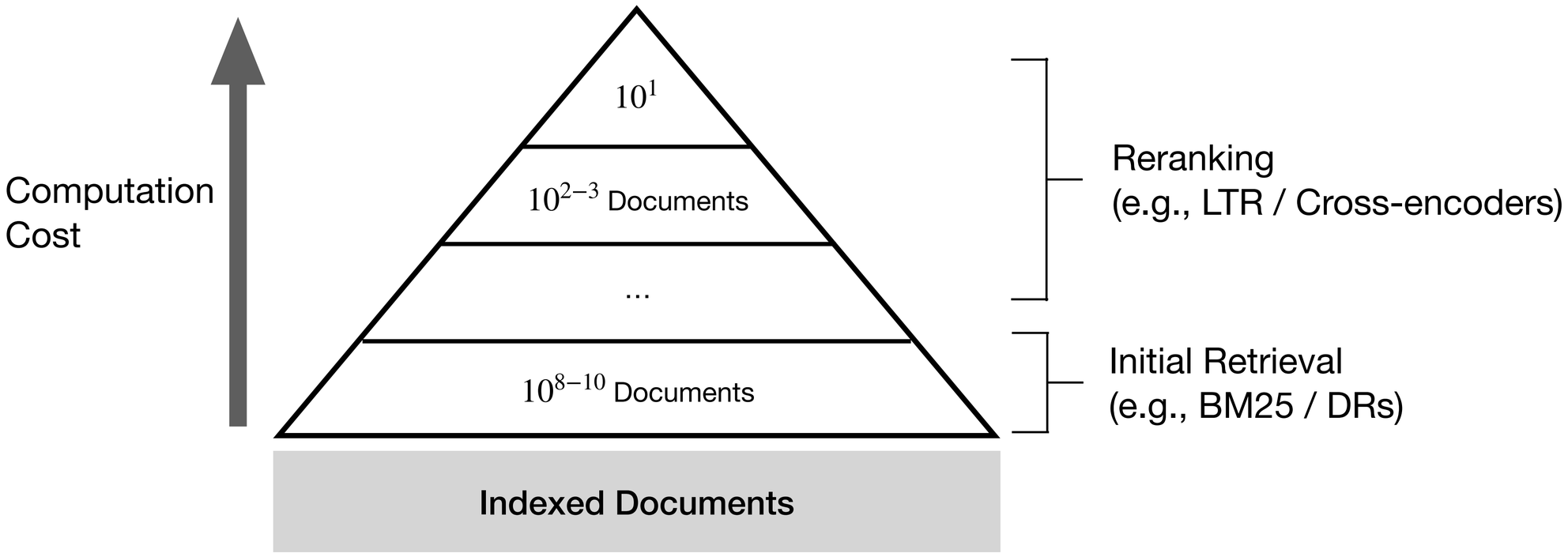
https://ielab.io/tutorials/r2llms.html

Part 3: LLM-based Rankers

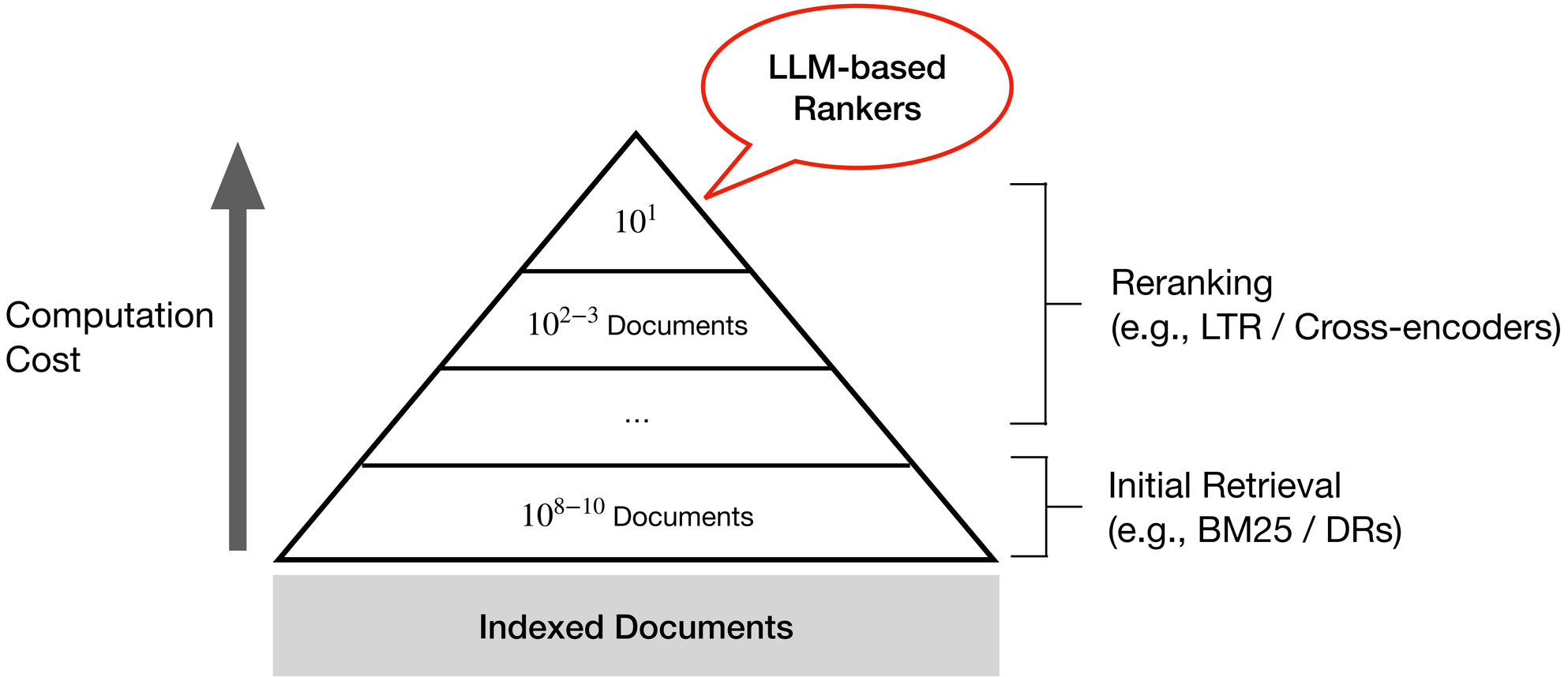
PART 3: LLM-based Rankers



PART 3: LLM-based Rankers

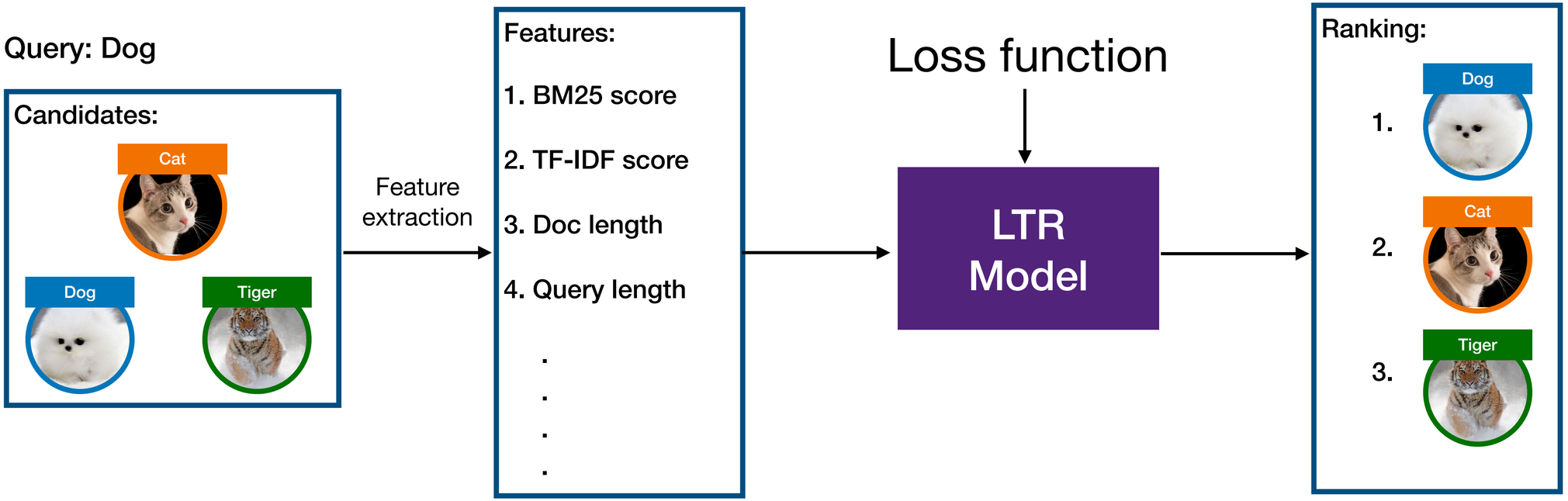


PART 3: LLM-based Rankers



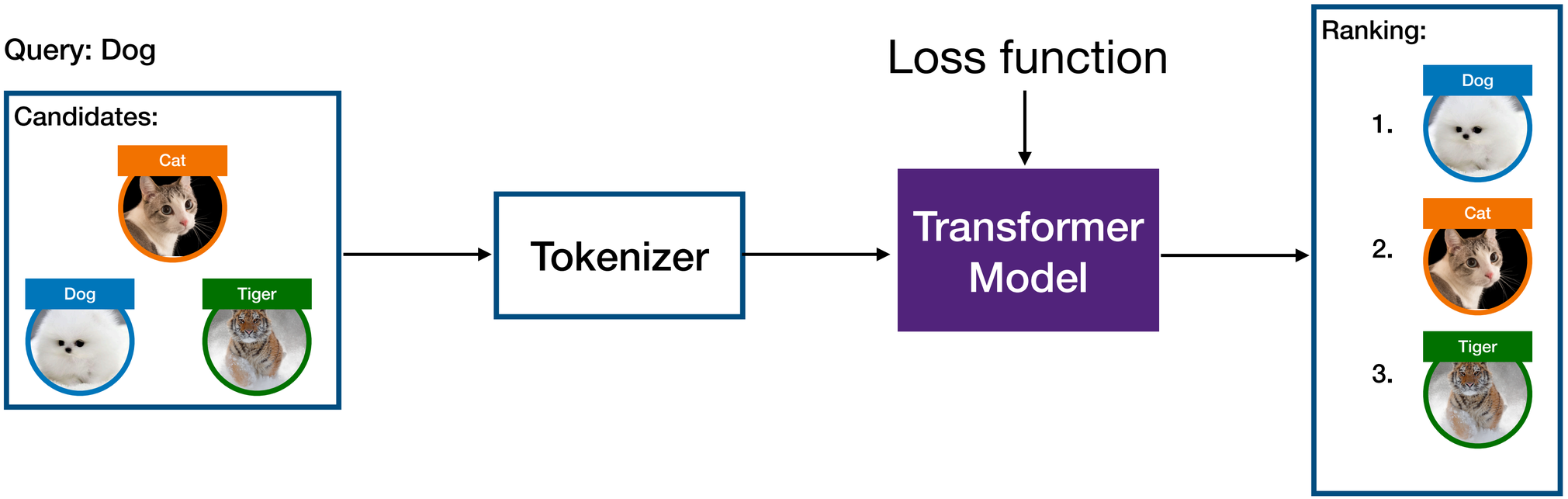
Recap of LTR and Cross-encoders

- Learning to Rank (LTR):



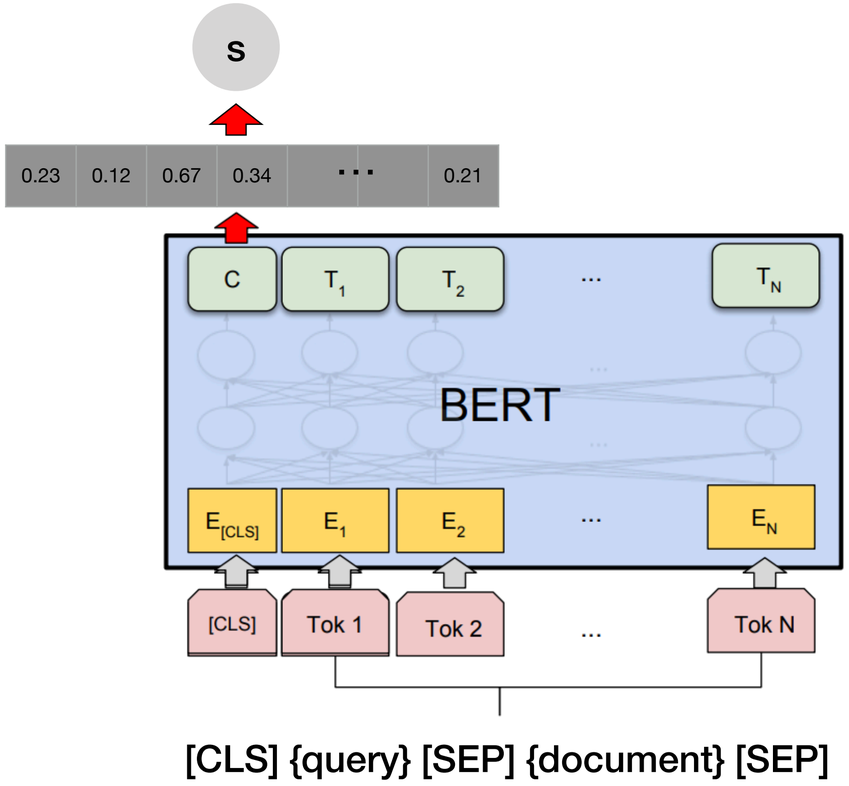
Recap of LTR and Cross-encoders

- Transformer ranker



Recap of LTR and Cross-encoders

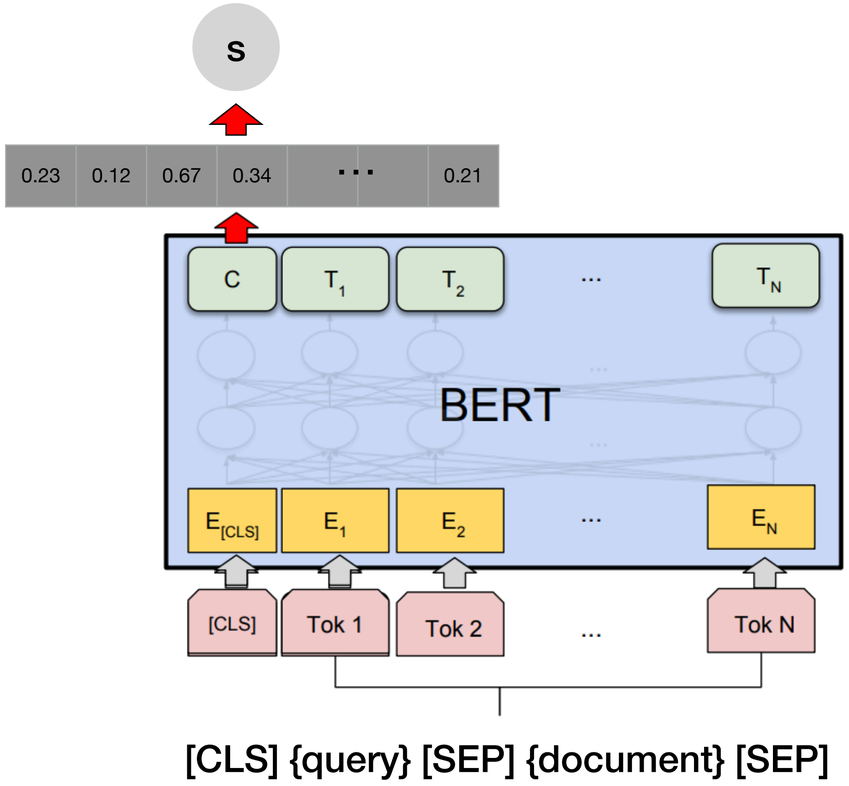
- monoBERT: cross-encoder architecture



Nogueira, R., Yang, W., Cho, K., Lin, J., 2019. Multi-Stage Document Ranking with BERT. *arXiv preprint*

Recap of LTR and Cross-encoders

- monoBERT: cross-encoder architecture

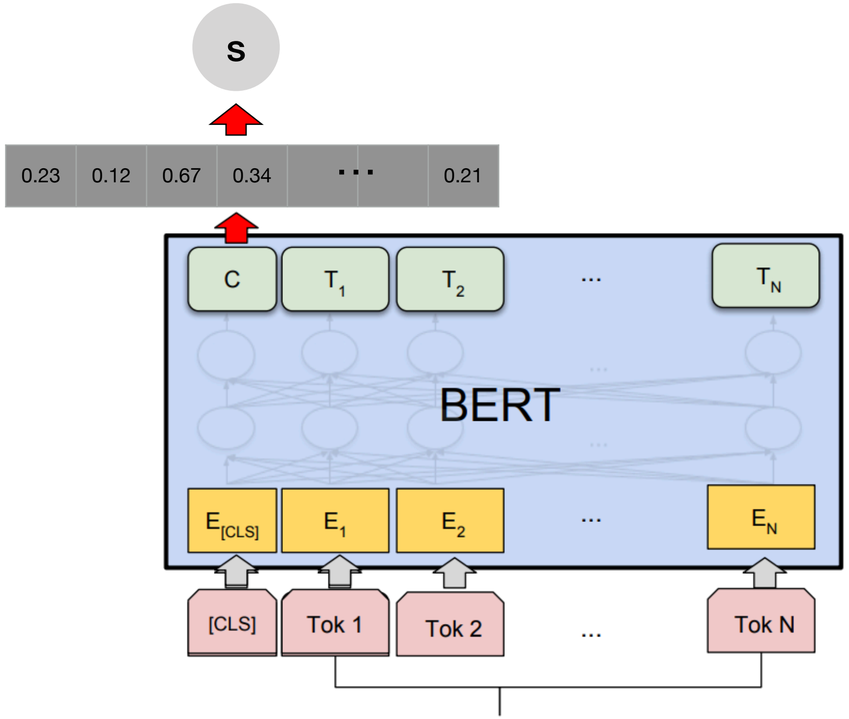


LLM-based cross-encoder?

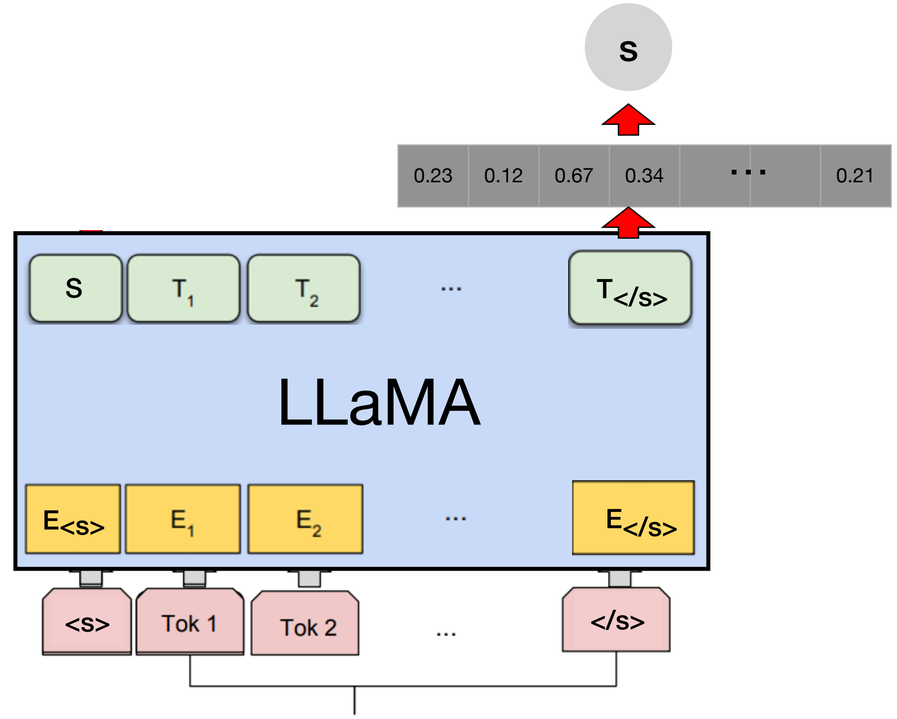
Nogueira, R., Yang, W., Cho, K., Lin, J., 2019. Multi-Stage Document Ranking with BERT. *arXiv preprint*

LLM-based Cross-encoder

- RankLLaMA



[CLS] {query} [SEP] {document} [SEP]



query: {query} document: {document} </s>

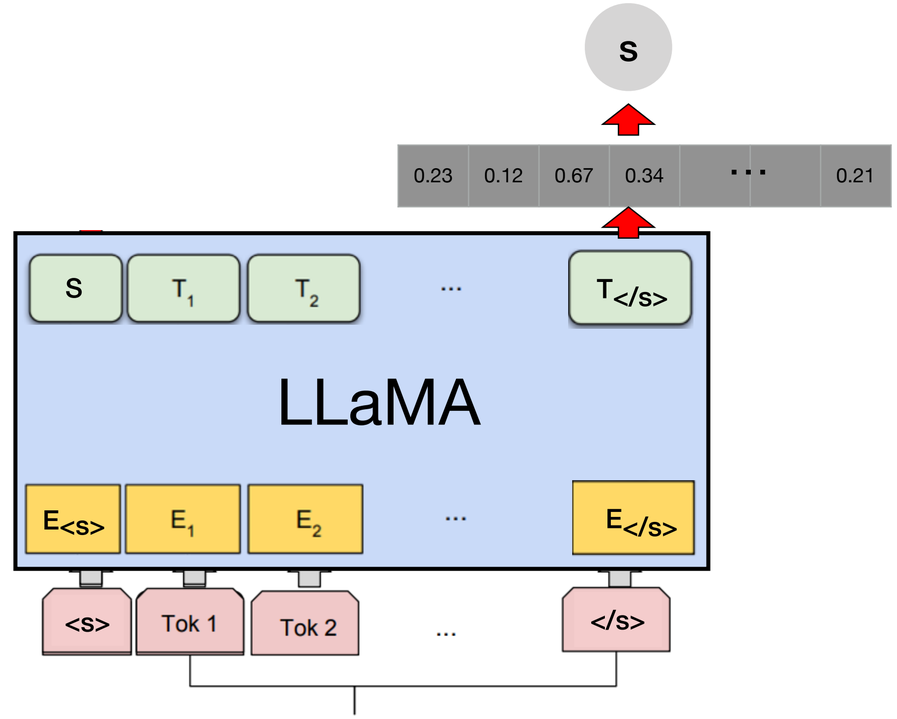
Ma, X., Wang, L., Yang, N., Lin, J., 2023. Fine-Tuning LLaMA for Multi-Stage Text Retrieval. *SIGIR*

LLM-based Cross-encoder

- RankLLaMA

$$\text{input} = \text{‘query: } \{Q\} \text{ document: } \{D\}\text{</s>’}$$
$$\text{Sim}(Q, D) = \text{Linear}(\text{Decoder}(\text{input})[-1])$$

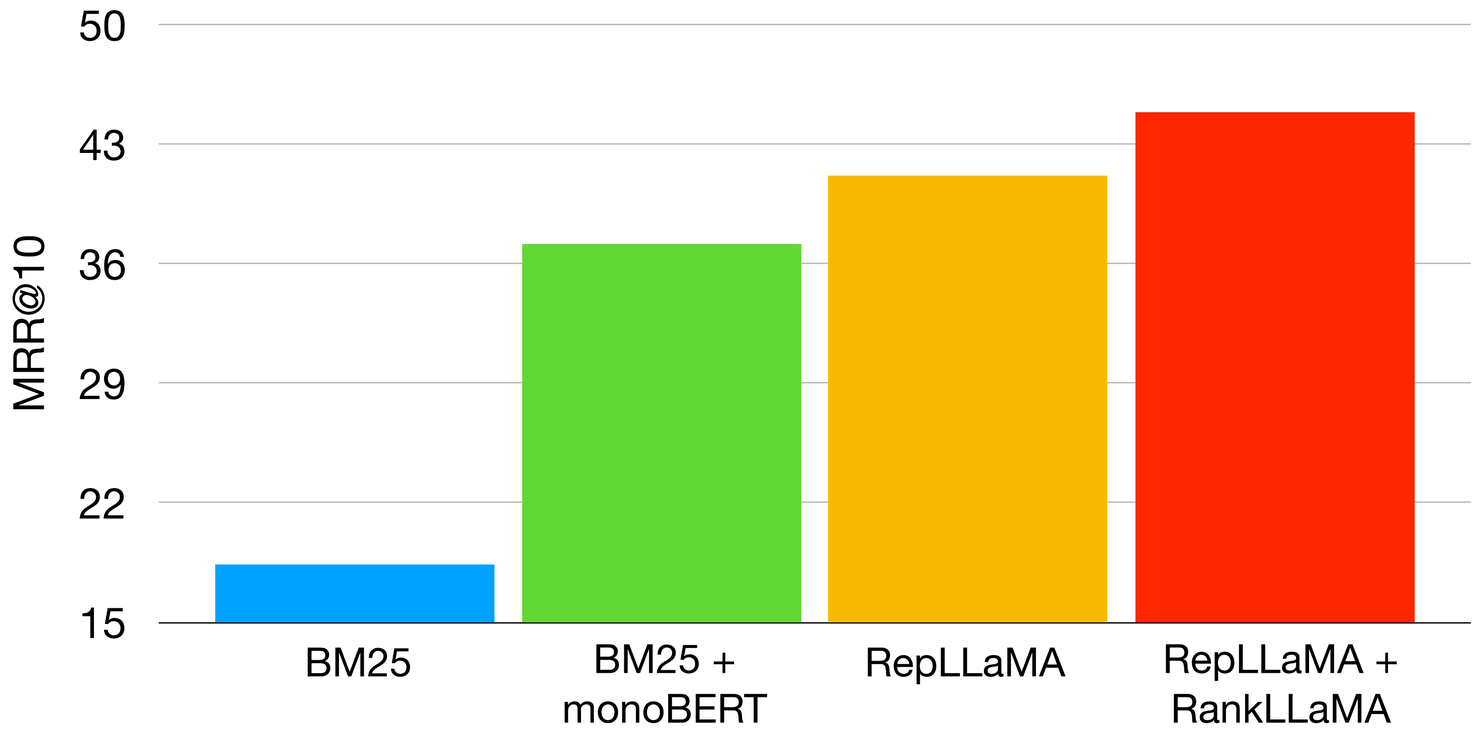
$$\mathcal{L}(Q, D^+, \{D_\text{N}\}) = -\log p(D = D^+ \mid Q) =$$
$$-\log \frac{\exp(\text{Sim}(Q, D^+))}{\exp(\text{Sim}(Q, D^+)) + \sum_{D_i^- \in \{D_\text{N}\}} \exp(\text{Sim}(Q, D_i^-))}$$



Ma, X., Wang, L., Yang, N., Lin, J., 2023. Fine-Tuning LLaMA for Multi-Stage Text Retrieval. *SIGIR*

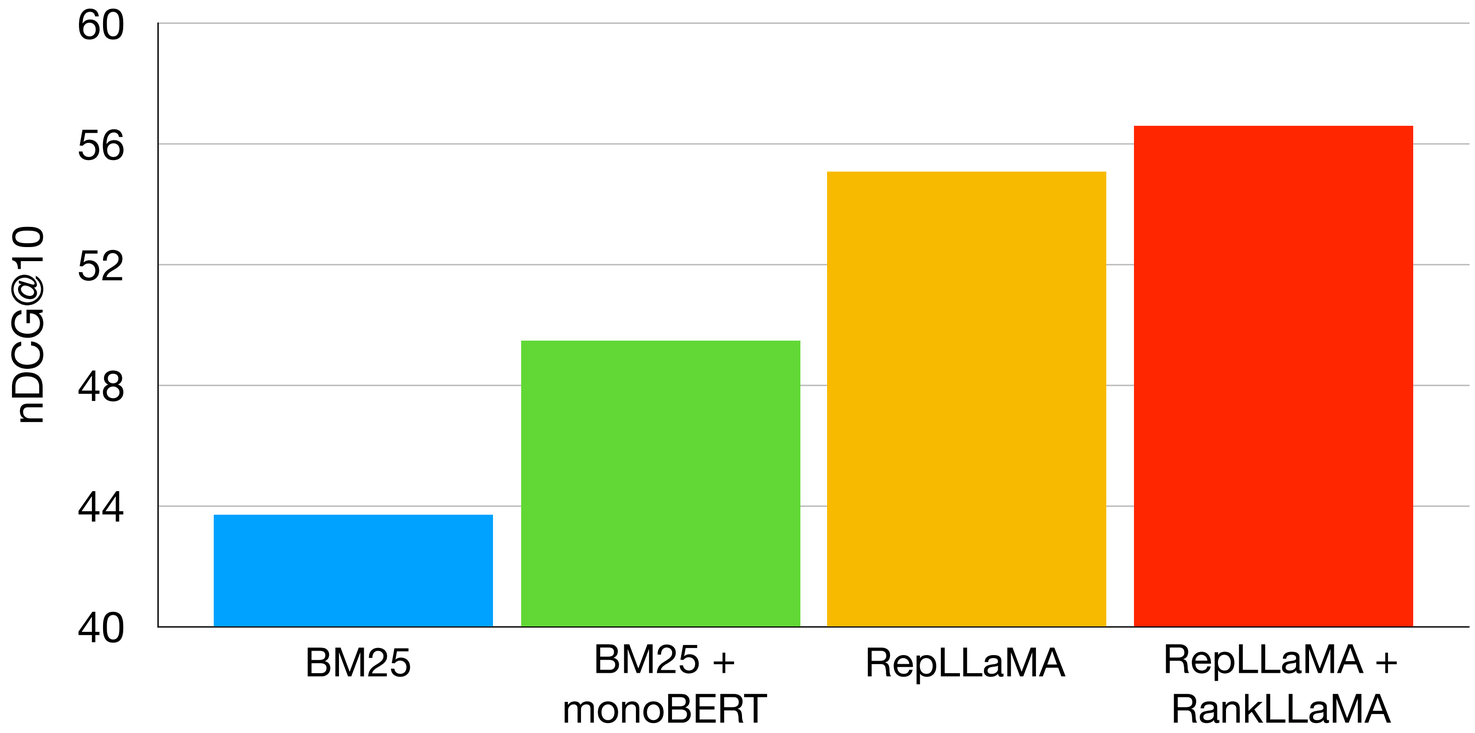
RankLLaMA Results

- In-domain results: MSMARCO



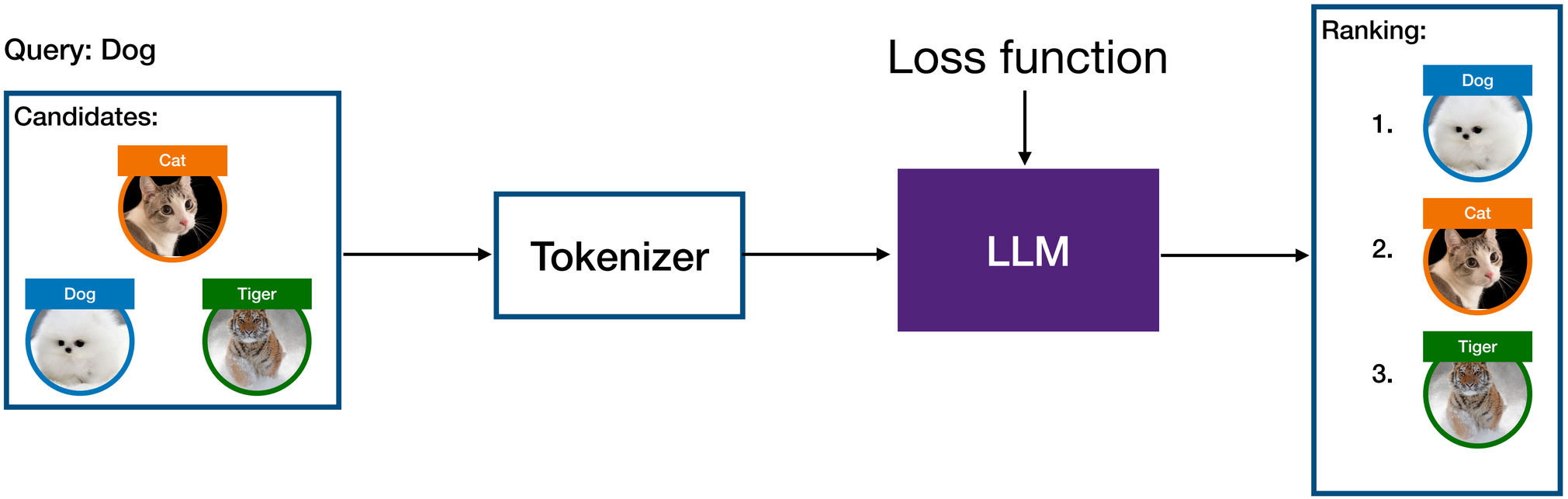
RankLLaMA Results

- Out-of-domain results: BEIR



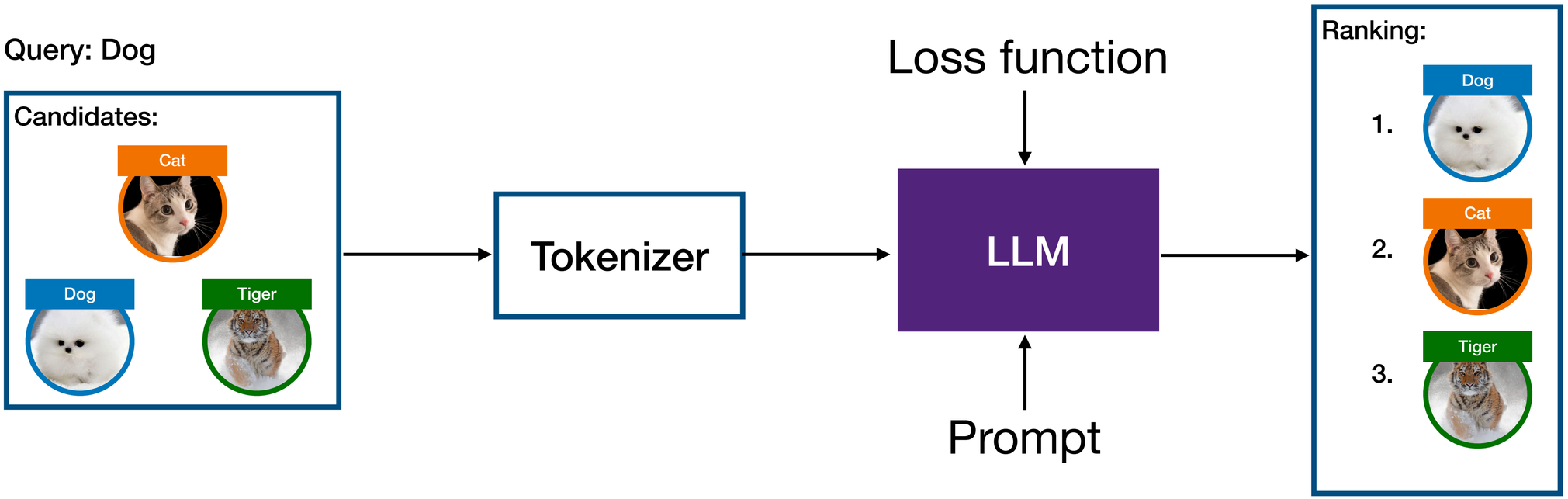
LLM-based Cross-encoder

- LLM-based ranker

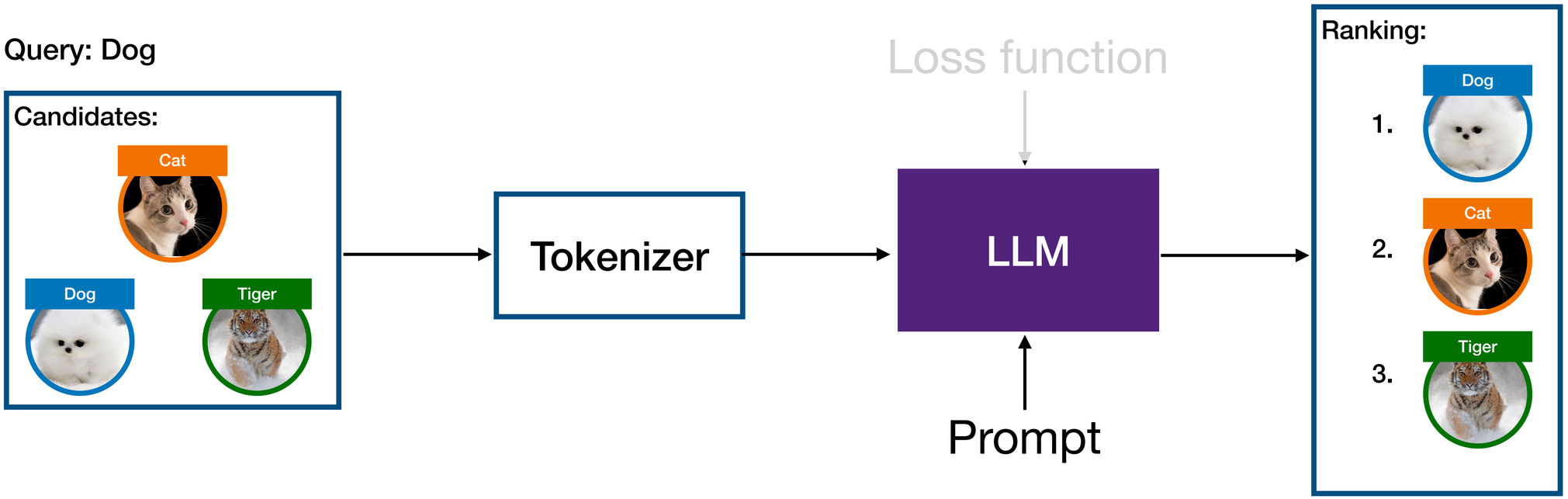


LLM-based Cross-encoder

- LLM-based ranker



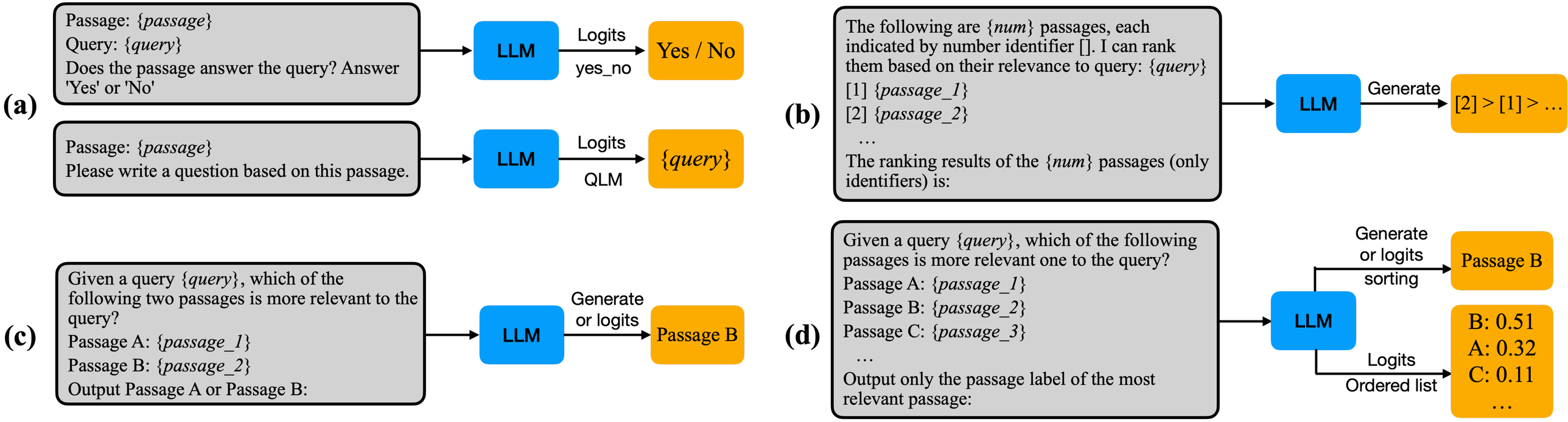
Zeroshot LLM-based Rankers (LLM as Rankers)



- All are “zero-shot”: i.e. once you obtained the pre-trained, instruction tuned LLM, no need to do supervised fine-tuning.

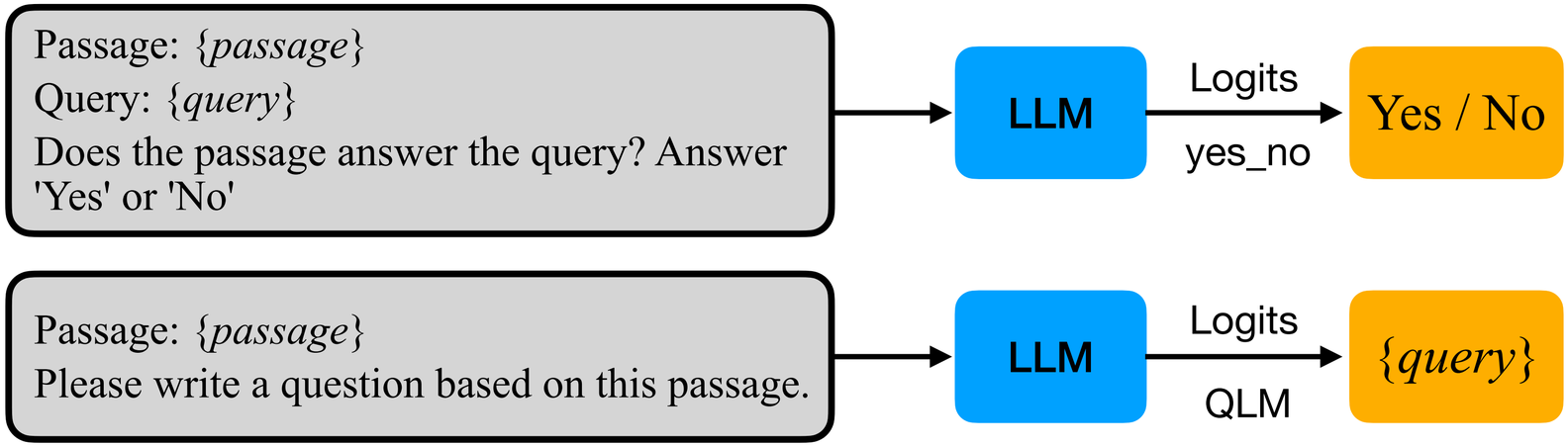
- Input could be one or multiple documents, depending on the prompt.

Zeroshot LLM-based Rankers: 3.5 types



Zhuang, S., Zhuang, H., Koopman, B. and Zuccon, G., 2023. A Setwise Approach for Effective and Highly Efficient Zero-shot Ranking with Large Language Models. *SIGIR*

Pointwise



- Yes / No label generation.
- Query generation likelihood.

Liang, P., Bommasani, R., Lee, T., Tsipras, D., Soylu, D., Yasunaga, M., Zhang, Y., Narayanan, D., Wu, Y., Kumar, A. and Newman, B., 2022. Holistic evaluation of language models. *arXiv.*
Sachan, D.S., Lewis, M., Joshi, M., Aghajanyan, A., Yih, W.T., Pineau, J. and Zettlemoyer, L., 2022. Improving passage retrieval with zero-shot question generation. *EMNLP.*
Zhuang, S., Liu, B., Koopman, B., Zuccon, G., 2023. Open-source Large Language Models are Strong Zero-shot Query Likelihood Models for Document Ranking. *EMNLP*.

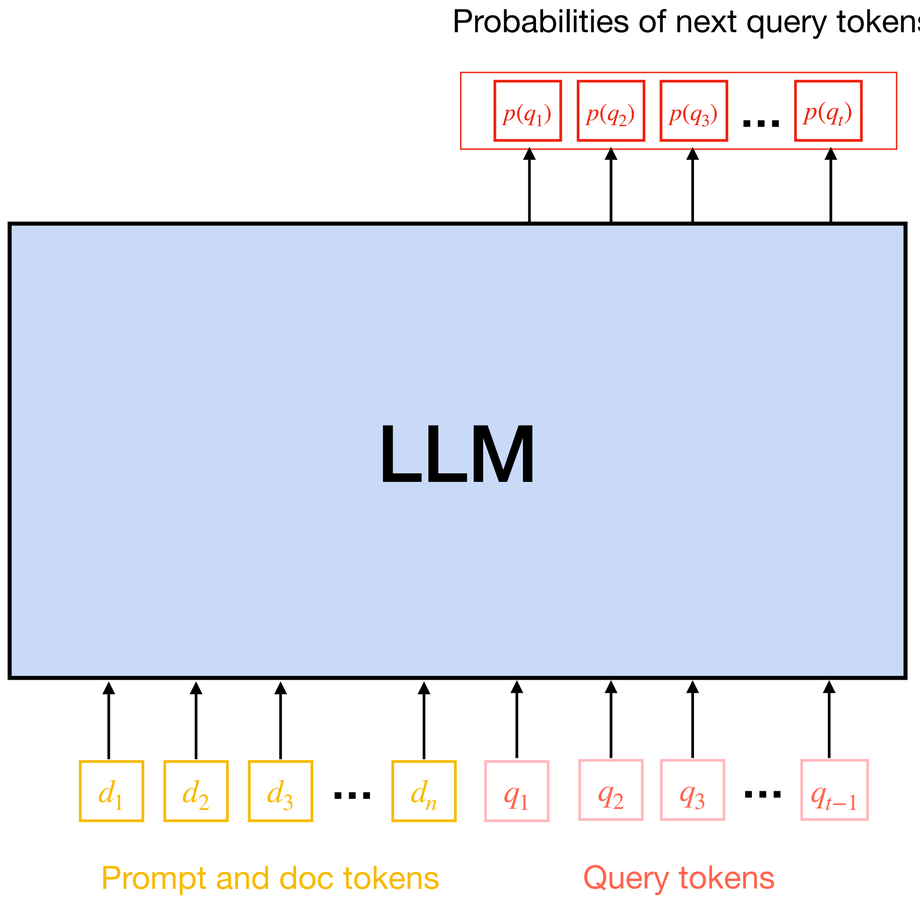
Pointwise

- Query generation likelihood.

Passage: {*passage*}
Please write a question based on this passage.

Rank by query likelihood:

$$P(Q \mid D) = \sum_{i}^{t} \log p(q_i)$$



Sachan, D.S., Lewis, M., Joshi, M., Aghajanyan, A., Yih, W.T., Pineau, J. and Zettlemoyer, L., 2022. Improving passage retrieval with zero-shot question generation. *EMNLP.*
Zhuang, S., Liu, B., Koopman, B., Zuccon, G., 2023. Open-source Large Language Models are Strong Zero-shot Query Likelihood Models for Document Ranking. *EMNLP*.

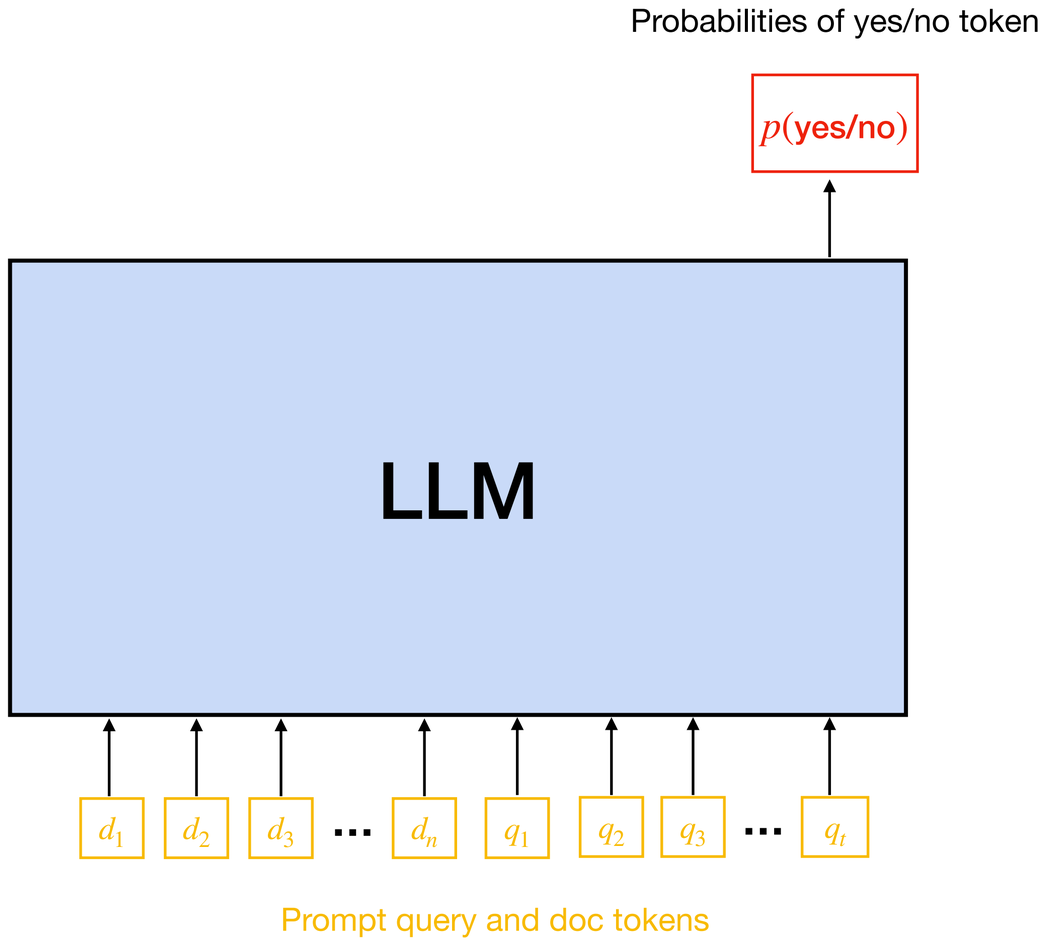
Pointwise

- Yes / No label generation.

Passage: {*passage*}
Query: {*query*}
Does the passage answer the query? Answer 'Yes' or 'No'

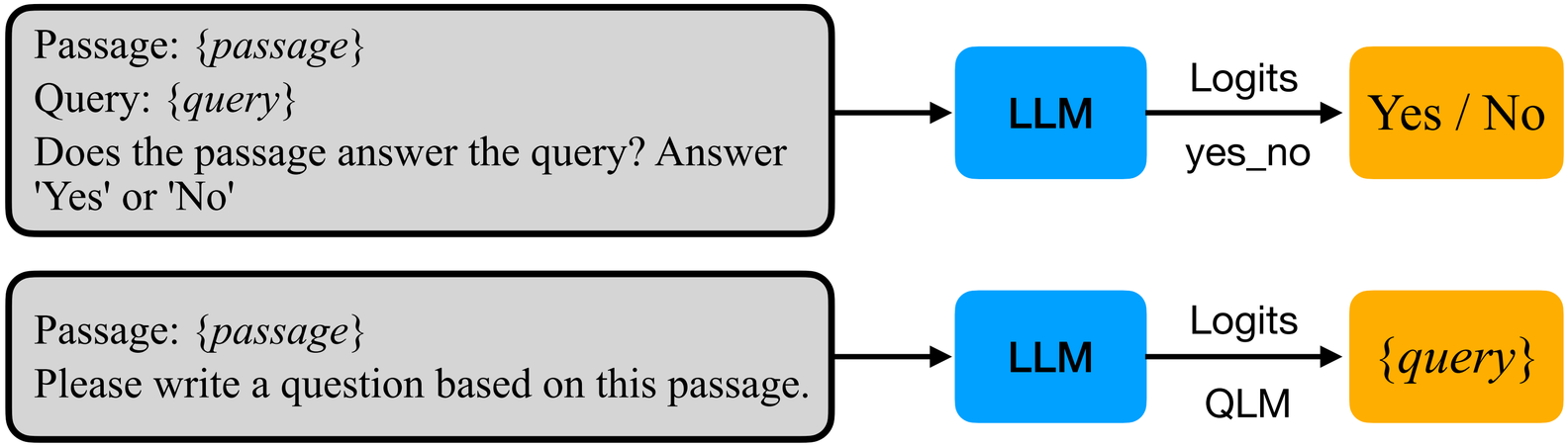
Rank by normalised yes probability:

$$P(\textbf{yes} \mid Q, D) = \frac{\log p(\textbf{yes})}{\log p(\textbf{yes}) + \log p(\textbf{no})}$$



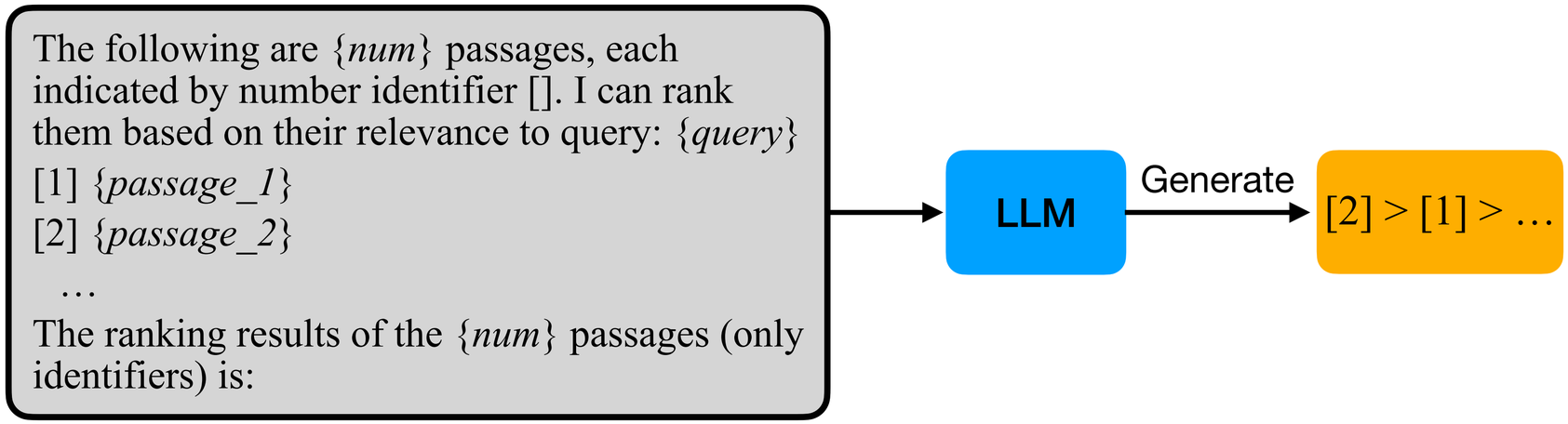
Liang, P., Bommasani, R., Lee, T., Tsipras, D., Soylu, D., Yasunaga, M., Zhang, Y., Narayanan, D., Wu, Y., Kumar, A. and Newman, B., 2022. Holistic evaluation of language models. *arXiv.*

Pointwise



- Each query-doc pair can be processed in parallel.
- It needs access to the model weights (need token logits).

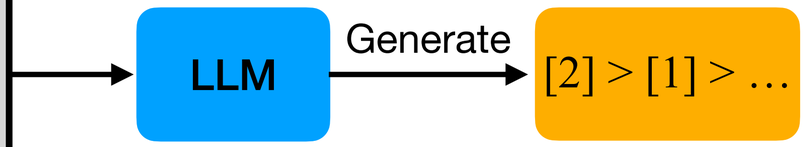
Listwise



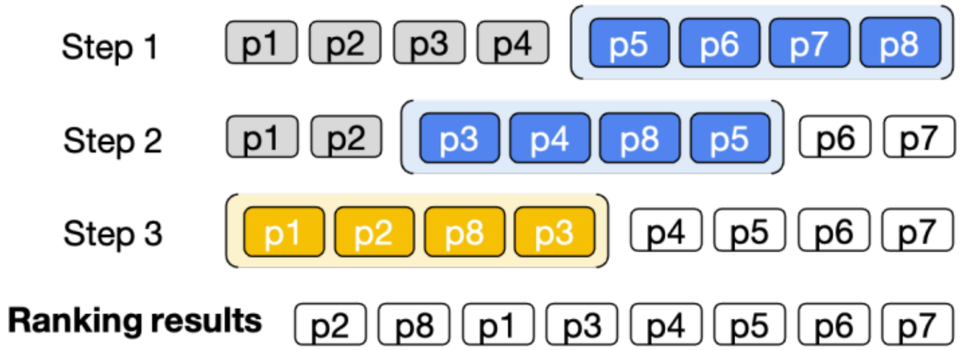
Ma, X., Zhang, X., Pradeep, R. and Lin, J., 2023. Zero-Shot Listwise Document Reranking with a Large Language Model. *arXiv*.
Pradeep, R., Sharifymoghaddam, S. and Lin, J., 2023. RankVicuna: Zero-Shot Listwise Document Reranking with Open-Source Large Language Models. *arXiv*.
Sun, W., Yan, L., Ma, X., Ren, P., Yin, D. and Ren, Z., 2023. Is ChatGPT Good at Search? Investigating Large Language Models as Re-Ranking Agent. *EMNLP.*

Listwise

The following are {*num*} passages, each indicated by number identifier []. I can rank them based on their relevance to query: {*query*}
[1] {*passage_1*}
[2] {*passage_2*}
…
The ranking results of the {*num*} passages (only identifiers) is:

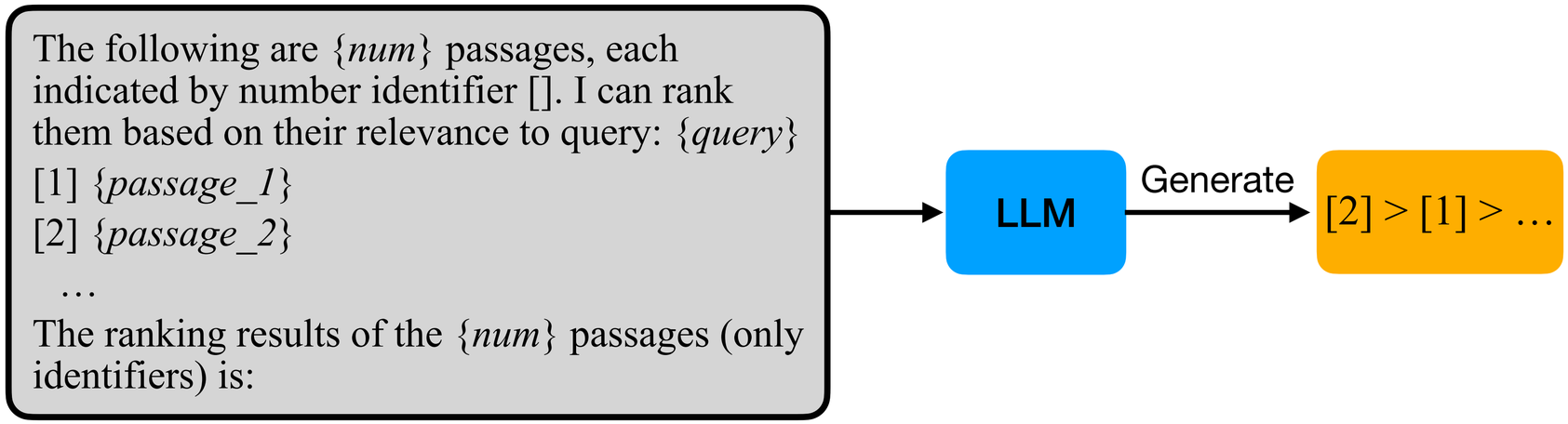


- Sliding window



Ma, X., Zhang, X., Pradeep, R. and Lin, J., 2023. Zero-Shot Listwise Document Reranking with a Large Language Model. *arXiv*.
Pradeep, R., Sharifymoghaddam, S. and Lin, J., 2023. RankVicuna: Zero-Shot Listwise Document Reranking with Open-Source Large Language Models. *arXiv*.
Sun, W., Yan, L., Ma, X., Ren, P., Yin, D. and Ren, Z., 2023. Is ChatGPT Good at Search? Investigating Large Language Models as Re-Ranking Agent. *EMNLP.*

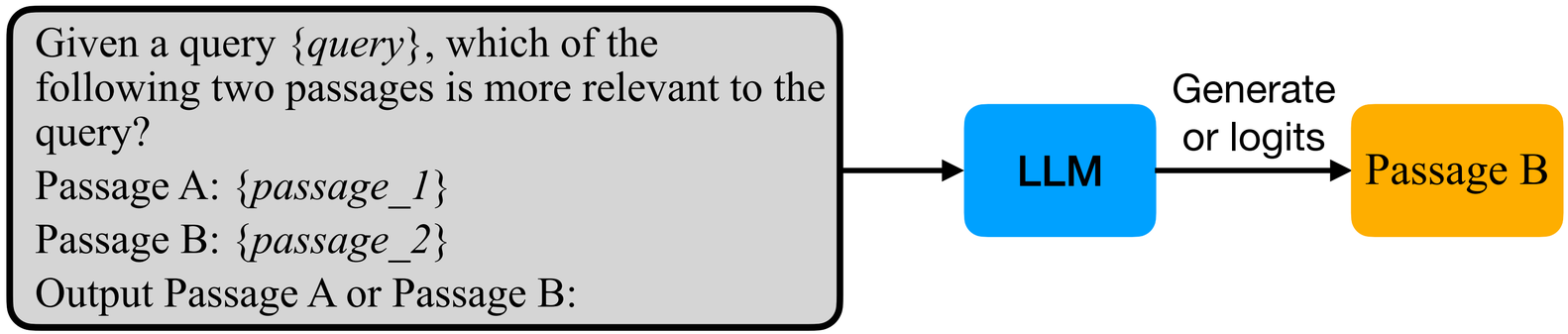
Listwise



- Cannot parallel.
- No need access to the model weights, pure generation.

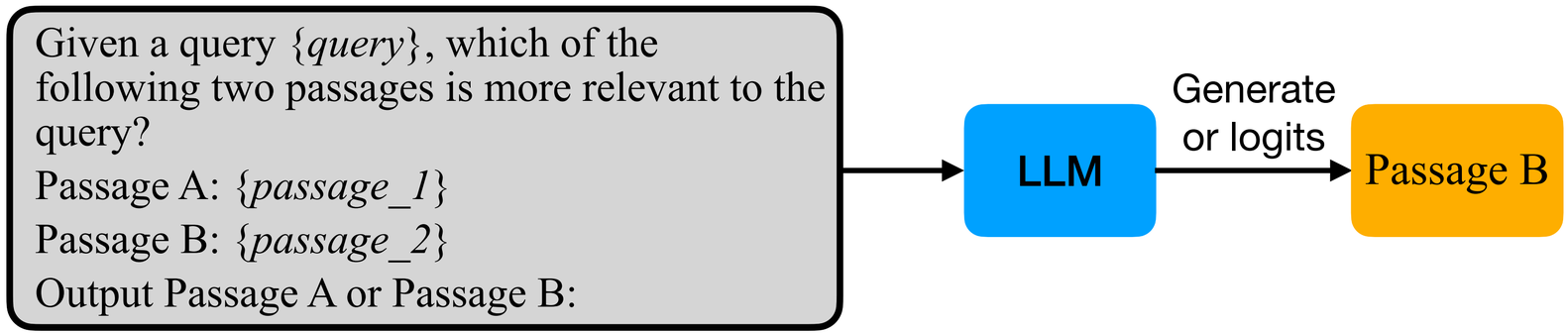
Ma, X., Zhang, X., Pradeep, R. and Lin, J., 2023. Zero-Shot Listwise Document Reranking with a Large Language Model. *arXiv*.
Pradeep, R., Sharifymoghaddam, S. and Lin, J., 2023. RankVicuna: Zero-Shot Listwise Document Reranking with Open-Source Large Language Models. *arXiv*.
Sun, W., Yan, L., Ma, X., Ren, P., Yin, D. and Ren, Z., 2023. Is ChatGPT Good at Search? Investigating Large Language Models as Re-Ranking Agent. *EMNLP.*

Pairwise



Qin, Z., Jagerman, R., Hui, K., Zhuang, H., Wu, J., Shen, J., Liu, T., Liu, J., Metzler, D., Wang, X. and Bendersky, M., 2023. Large language models are effective text rankers with pairwise ranking prompting. *NAACL-findings*.

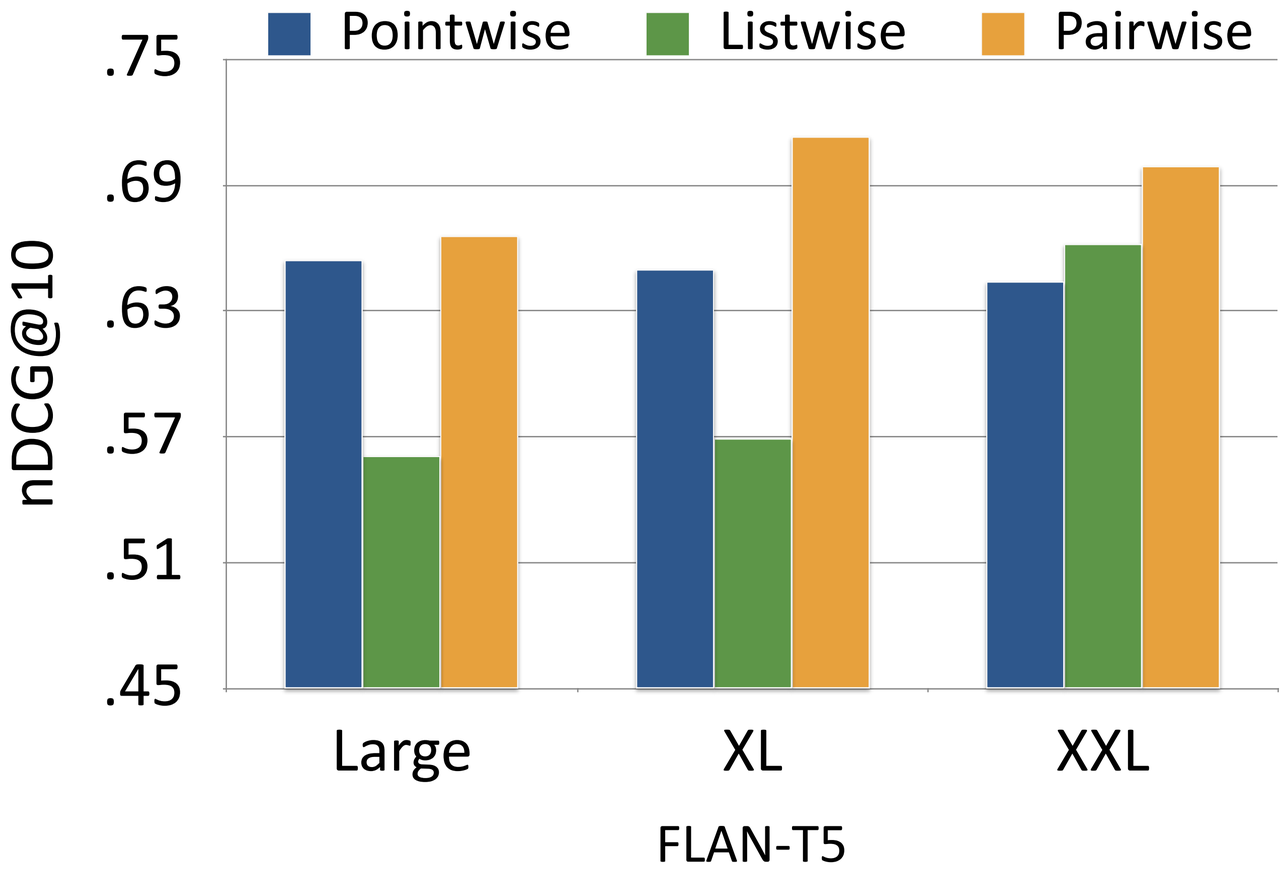
Pairwise



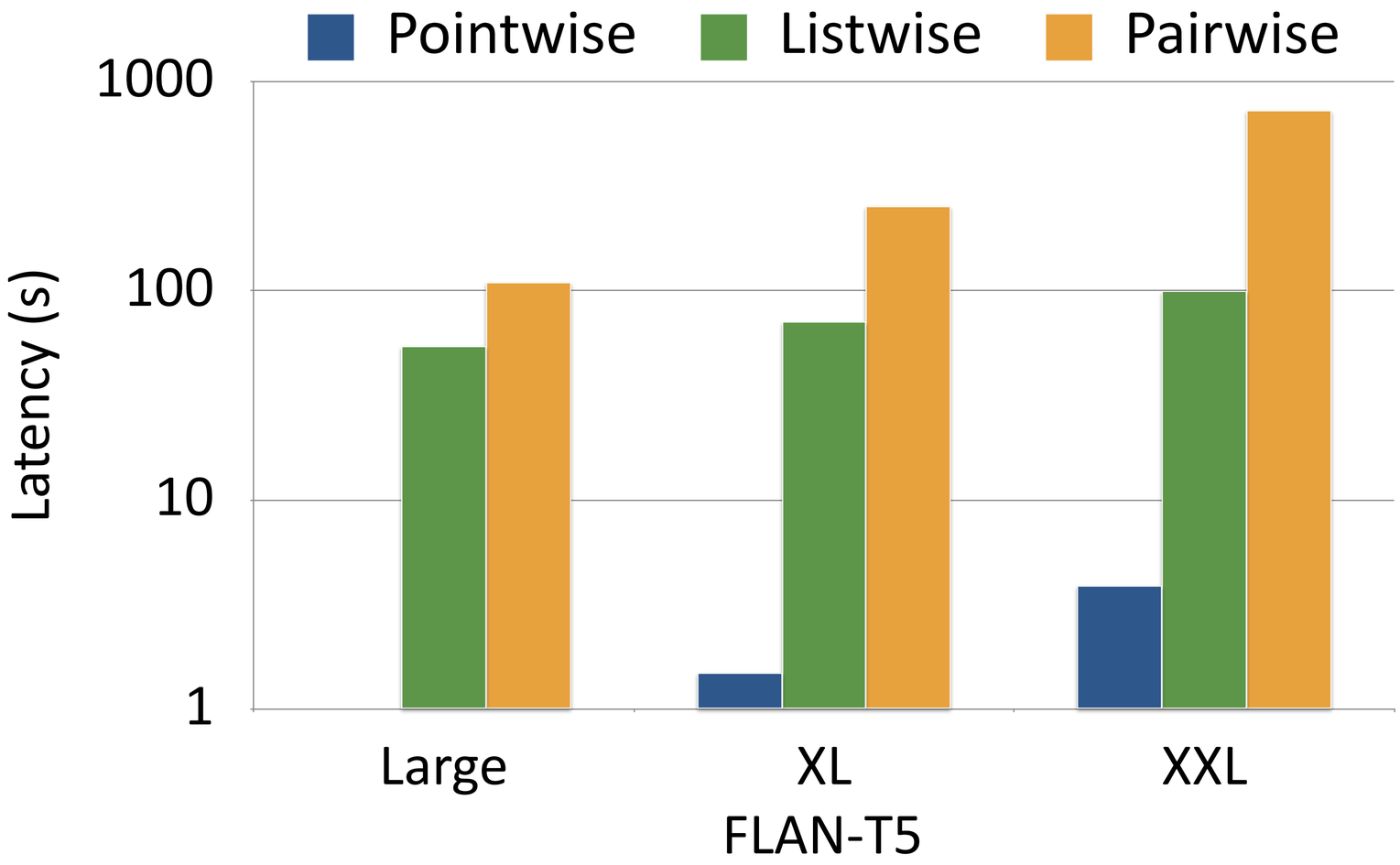
- **Scoring**: Compare all the pairwise preferences. This can be done in parallel.
- Both generation and logits are supported.

Qin, Z., Jagerman, R., Hui, K., Zhuang, H., Wu, J., Shen, J., Liu, T., Liu, J., Metzler, D., Wang, X. and Bendersky, M., 2023. Large language models are effective text rankers with pairwise ranking prompting. *NAACL-findings*.

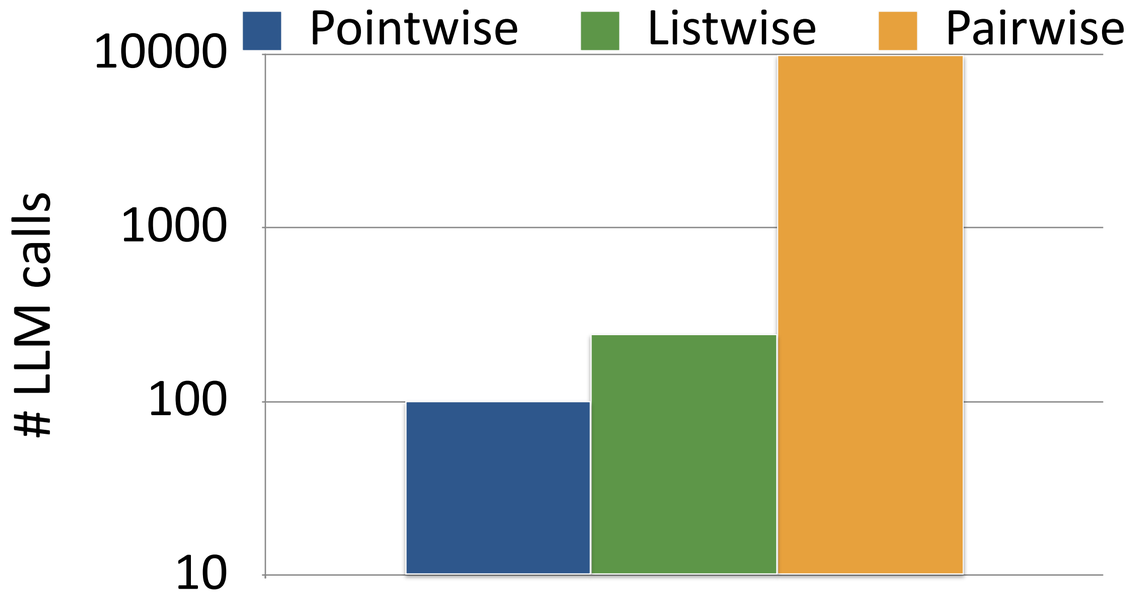
Effectiveness

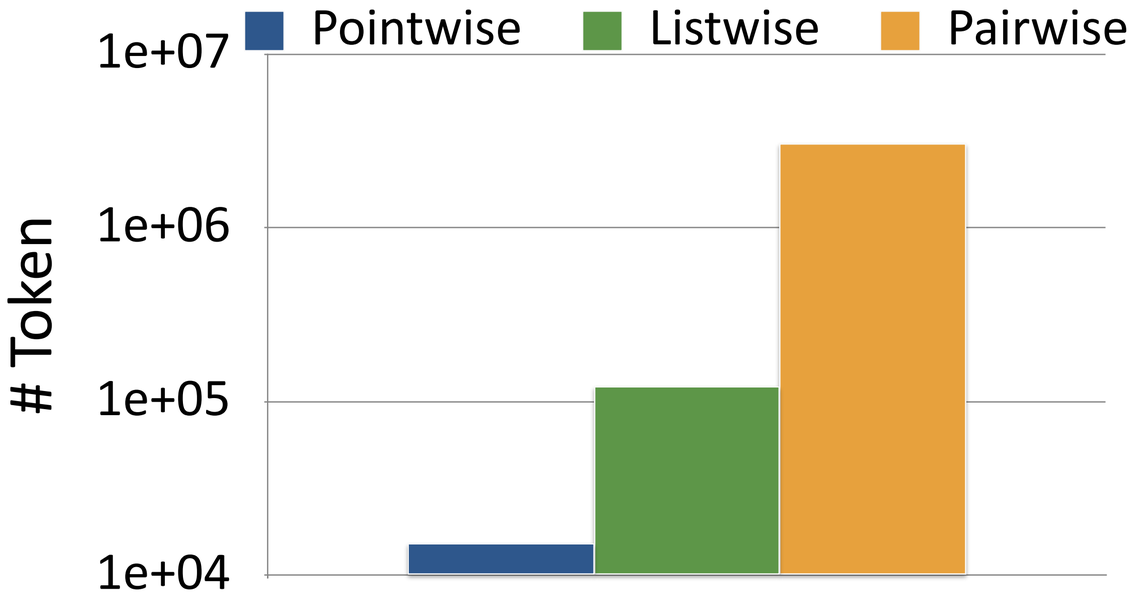


Efficiency


Pointwise
Listwise
Pairwise
1000
100
10
1
Latency (s)
Large
XL
XXL
FLAN-T5

Efficiency


Pointwise
Listwise
Pairwise
10000
1000
100
10
LLM calls

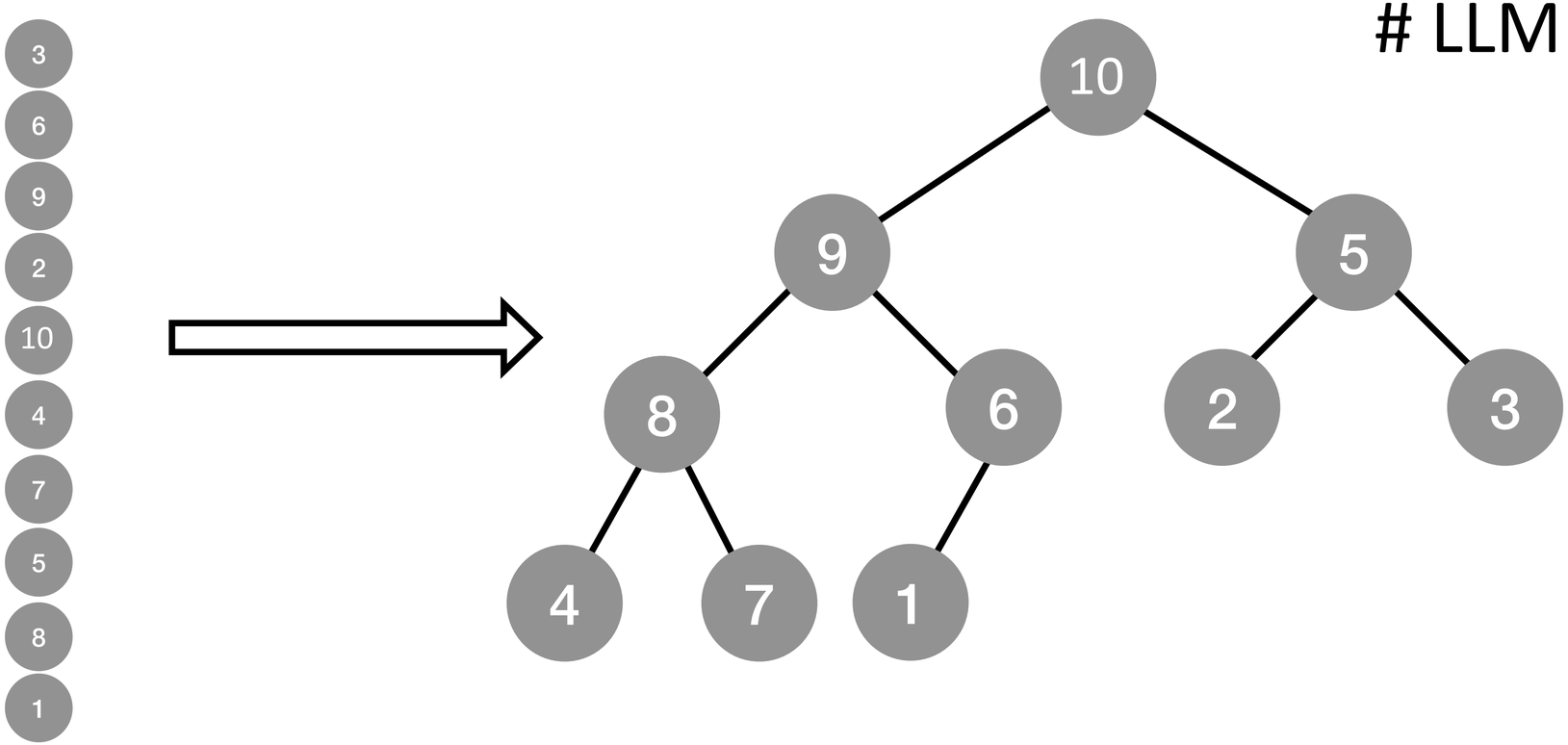

Pointwise
Listwise
Pairwise
1e+07
1e+06
1e+05
1e+04
Token

Pairwise

- **Sorting-based ranking: Heapsort**

  Step 1: Build max heap tree

LLM call: $O(nlogn)$

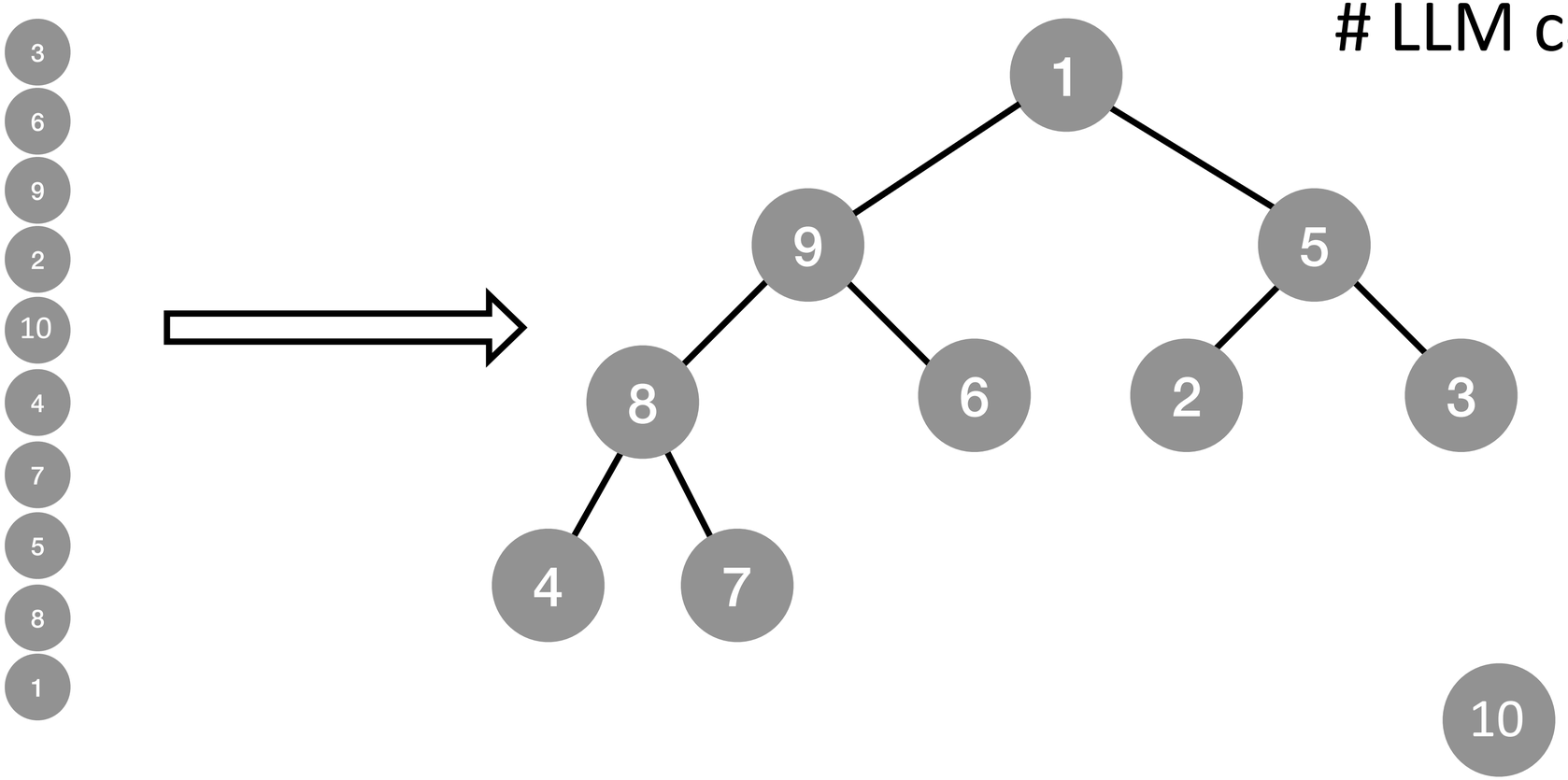


Pairwise

- **Sorting-based ranking: Heapsort**

  Step 1: Build max heap tree

LLM call: $O(nlogn)$

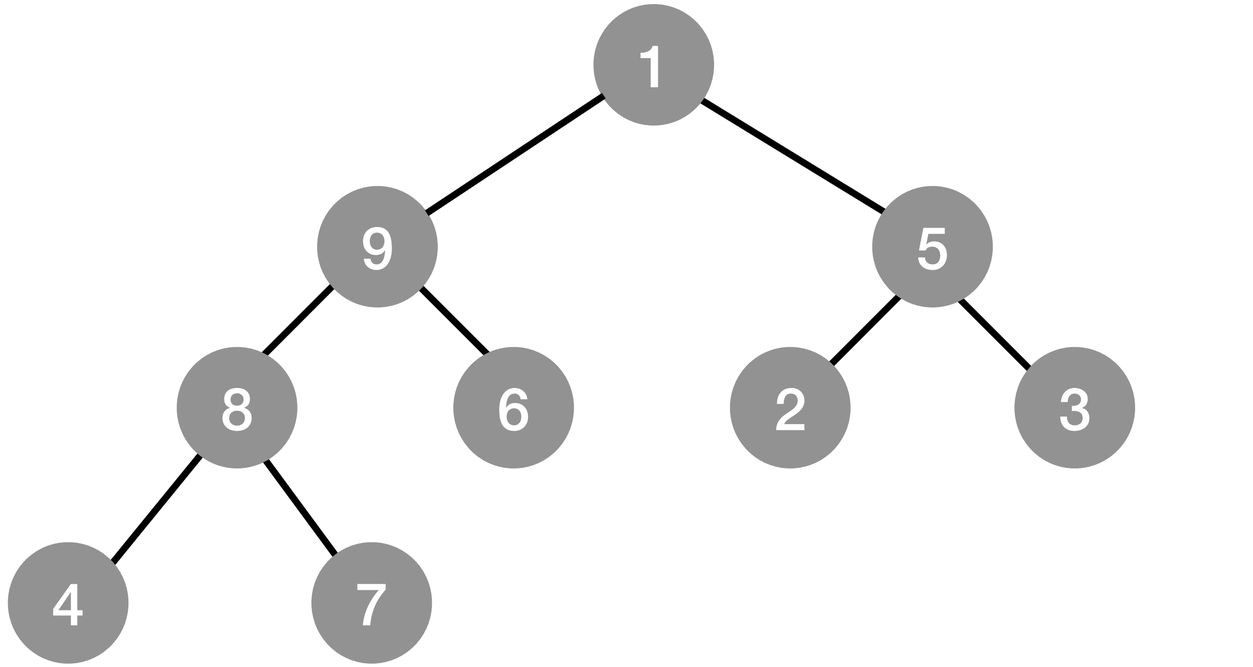


Pairwise

- **Sorting-based ranking: Heapsort**

  Step 2: heapify the tree

LLM call: 0



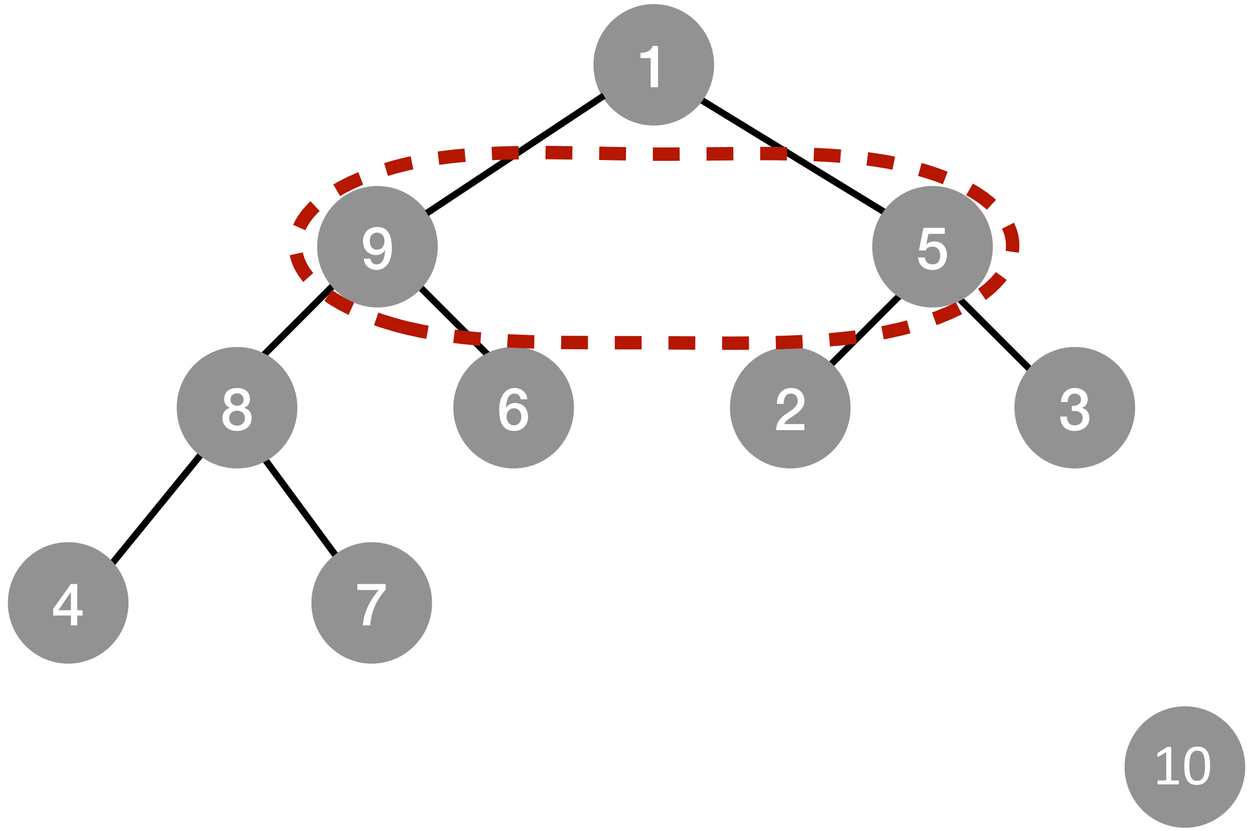
10

Pairwise

- **Sorting-based ranking: Heapsort**

  Step 2: heapify the tree

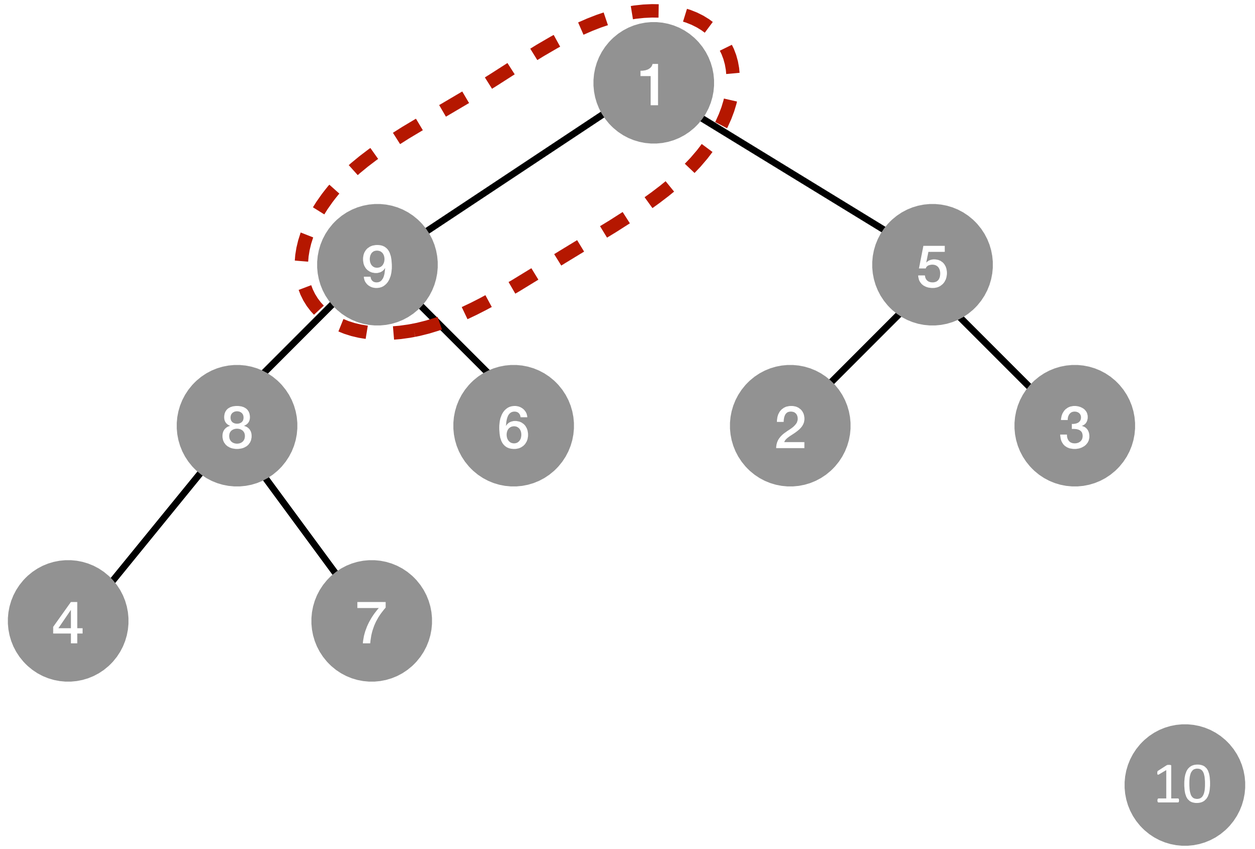
LLM call: 1



Pairwise

- **Sorting-based ranking: Heapsort**

  Step 2: heapify the tree

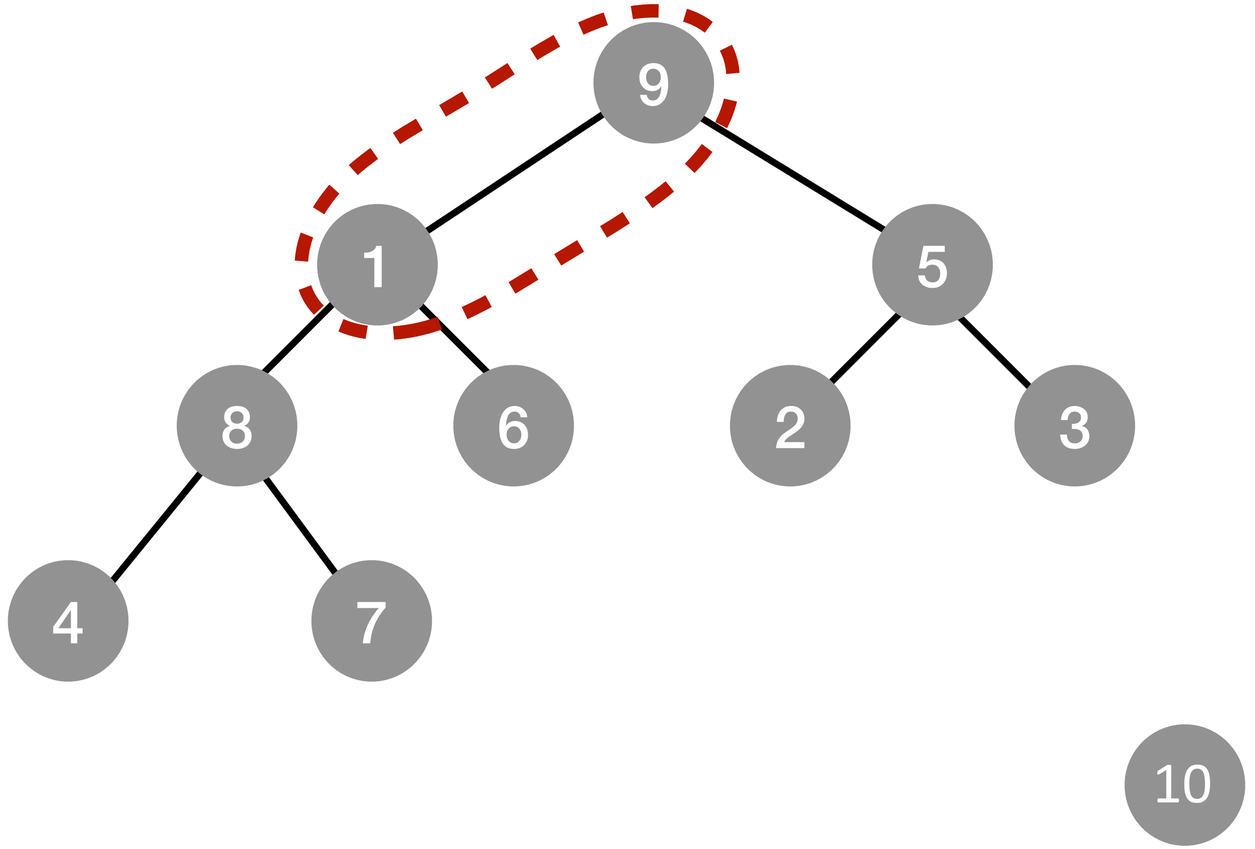


LLM call: 2

Pairwise

- **Sorting-based ranking: Heapsort**

  Step 2: heapify the tree



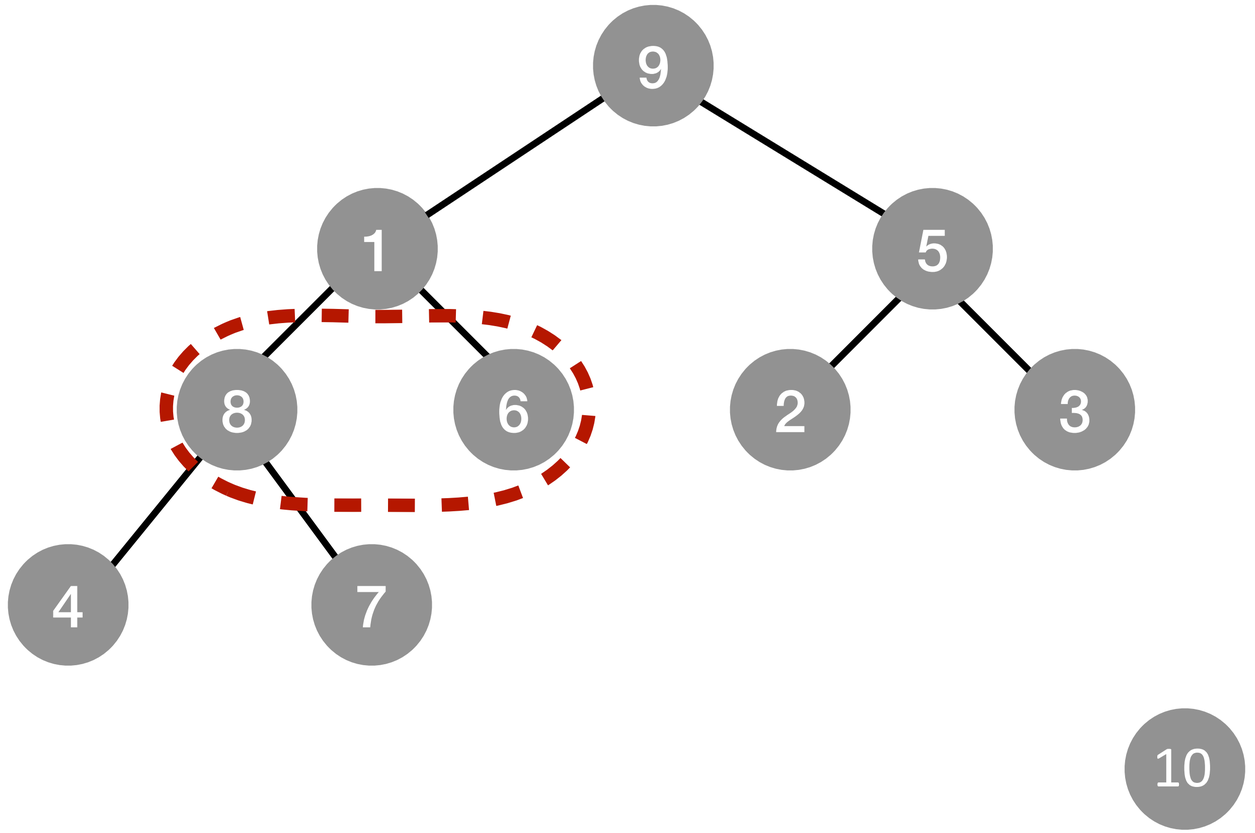
LLM call: 2

Pairwise

- **Sorting-based ranking: Heapsort**

  Step 2: heapify the tree

LLM call: 3

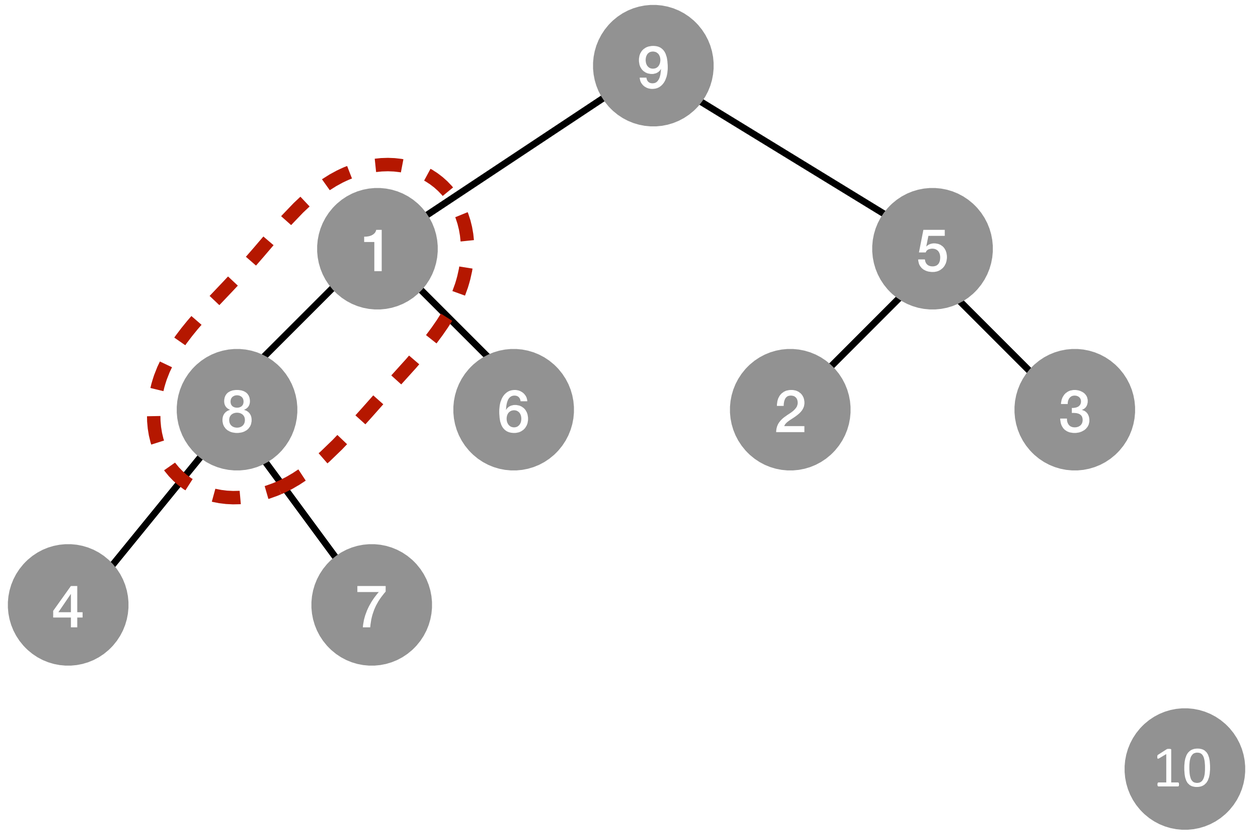


Pairwise

- **Sorting-based ranking: Heapsort**

  Step 2: heapify the tree

LLM call: 4

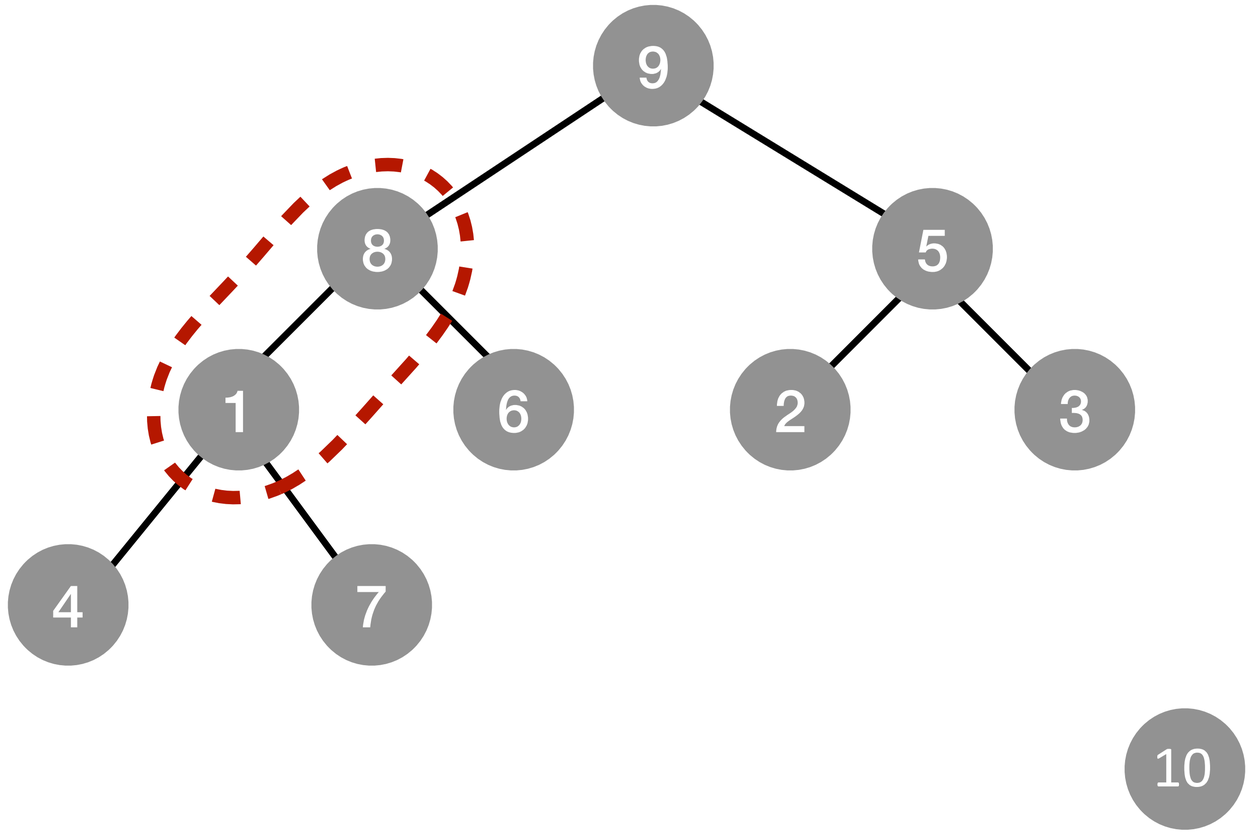


Pairwise

- **Sorting-based ranking: Heapsort**

  Step 2: heapify the tree

LLM call: 4

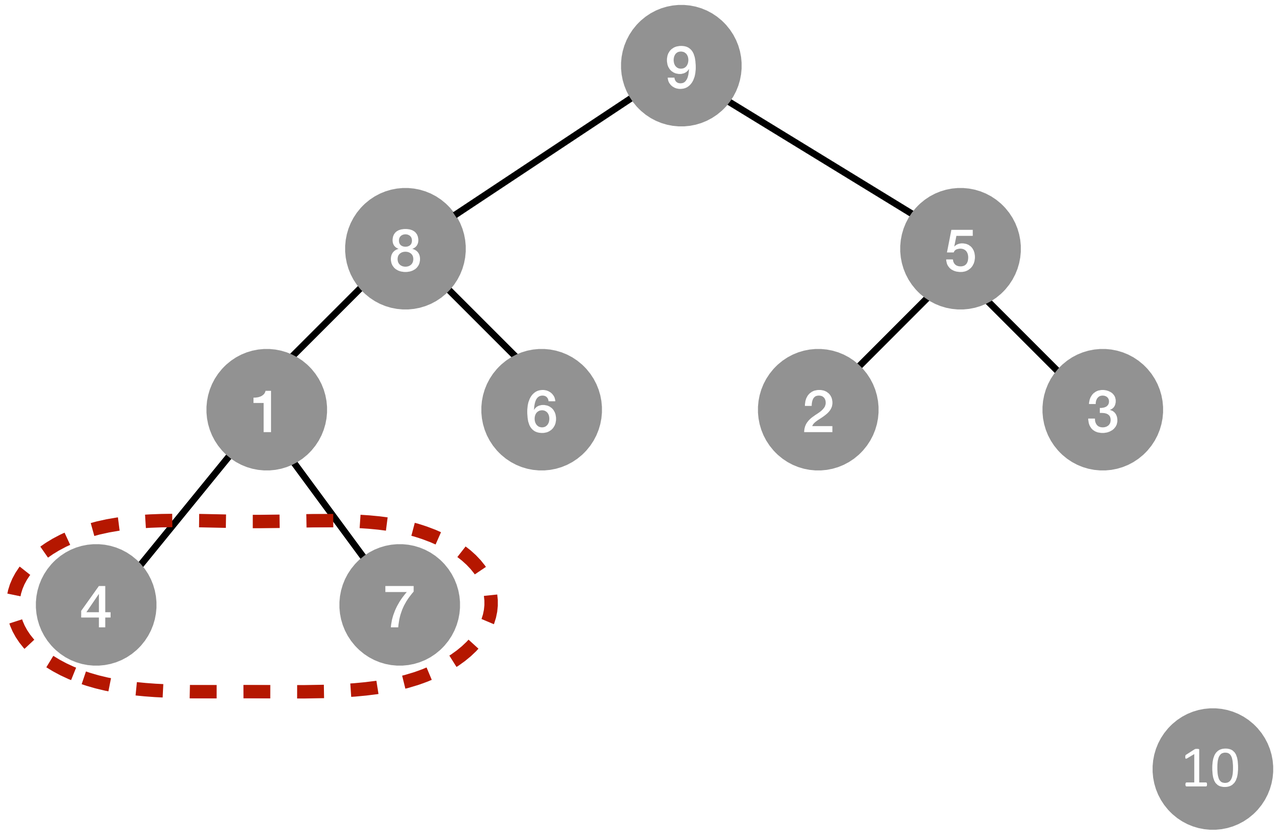


Pairwise

- **Sorting-based ranking: Heapsort**

  Step 2: heapify the tree

LLM call: 5

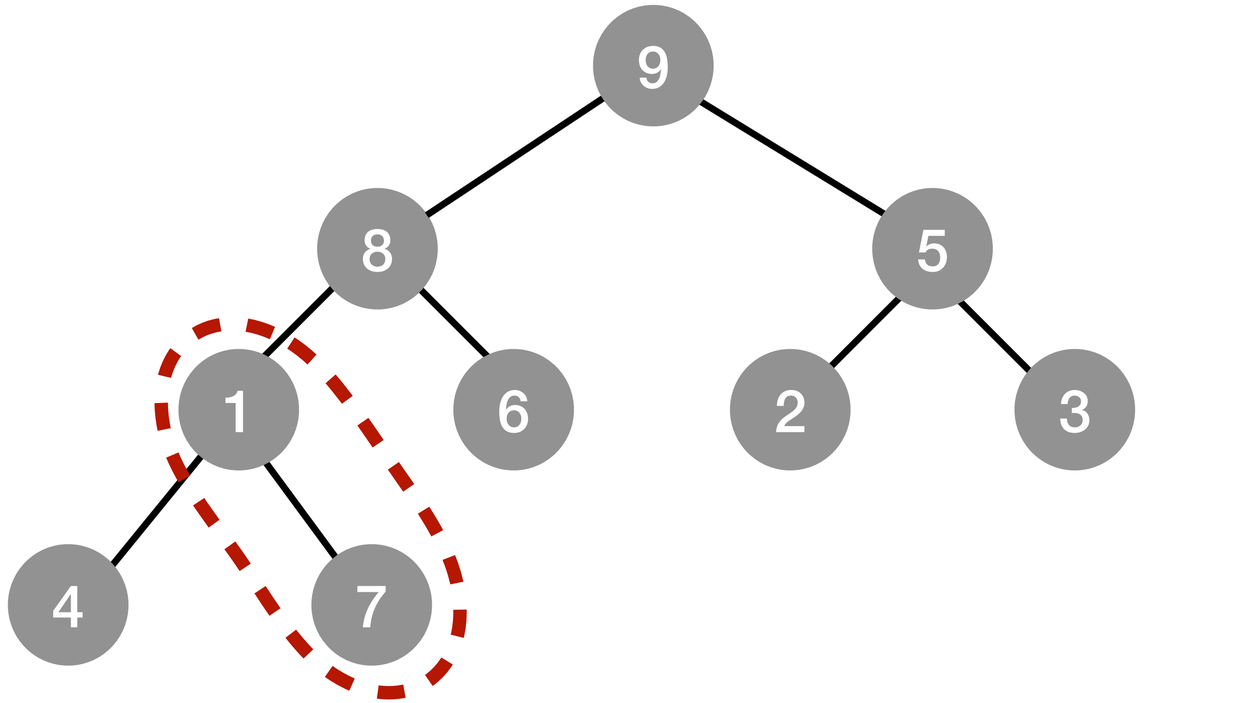


Pairwise

- **Sorting-based ranking: Heapsort**

  Step 2: heapify the tree

LLM call: 6



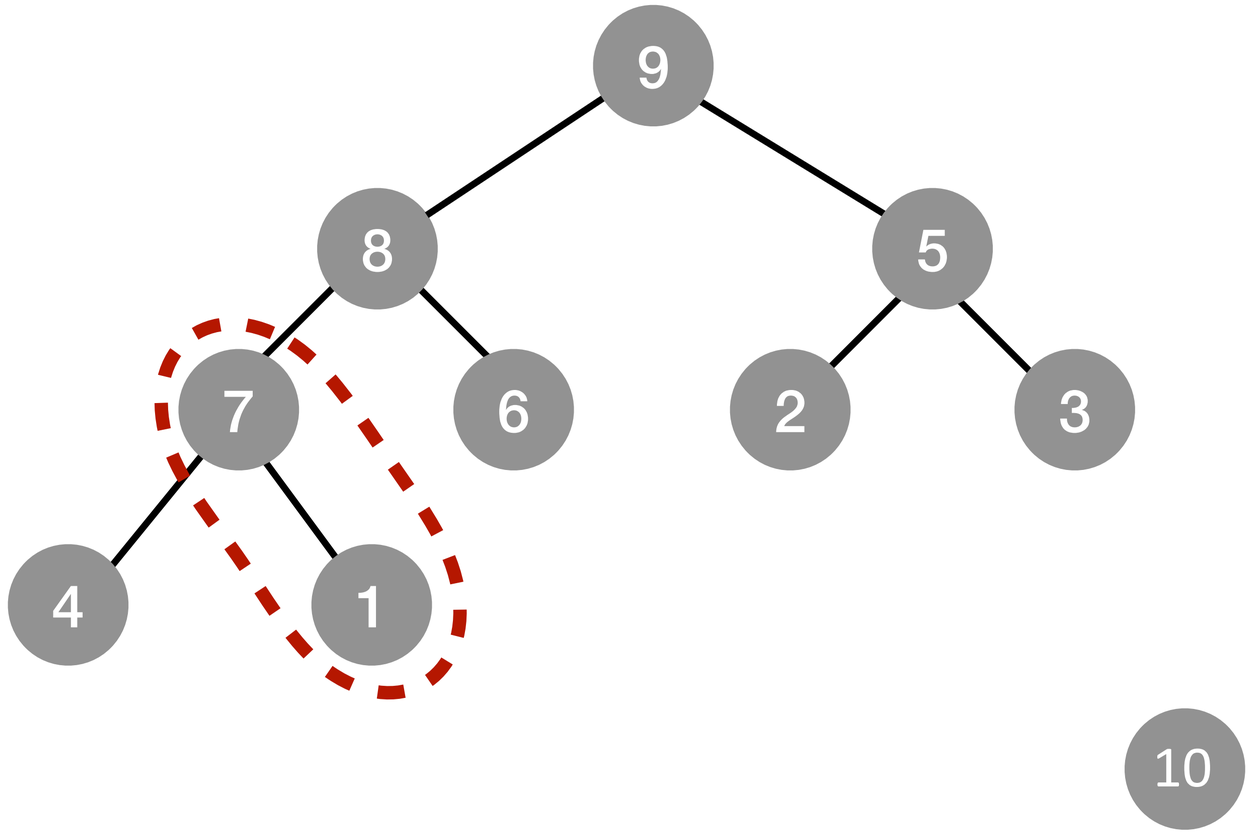
10

Pairwise

- **Sorting-based ranking: Heapsort**

  Step 2: heapify the tree

LLM call: 6

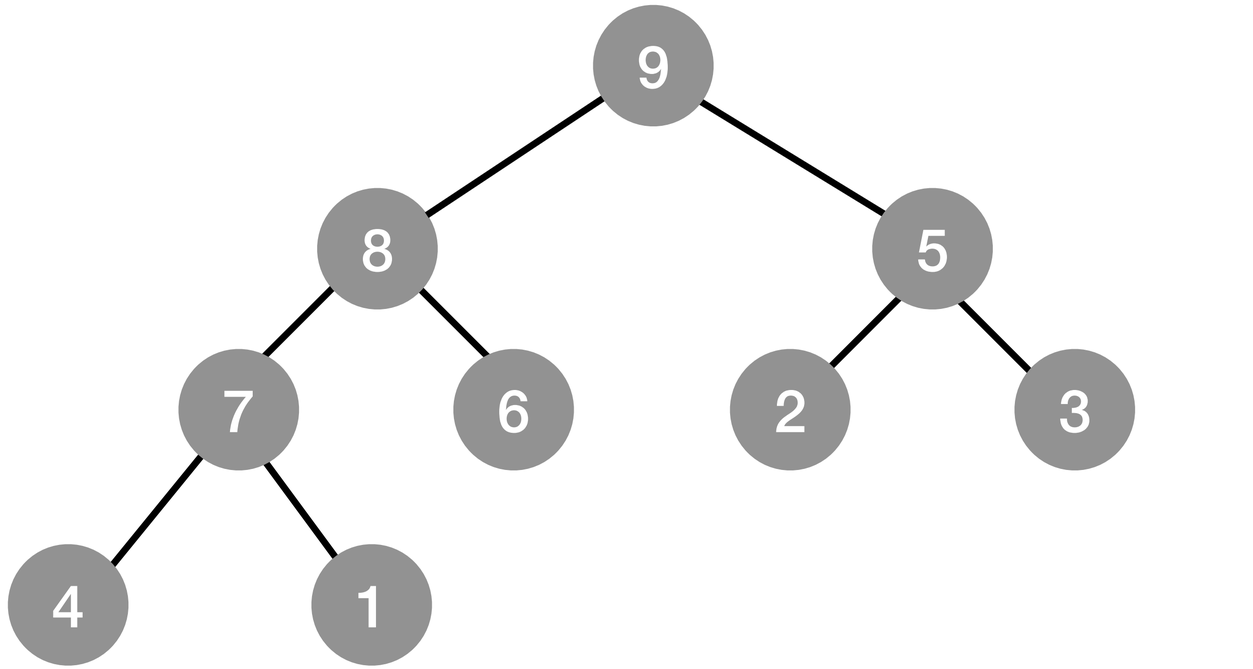


Pairwise

- **Sorting-based ranking: Heapsort**

  Step 2: heapify the tree

LLM call: 6



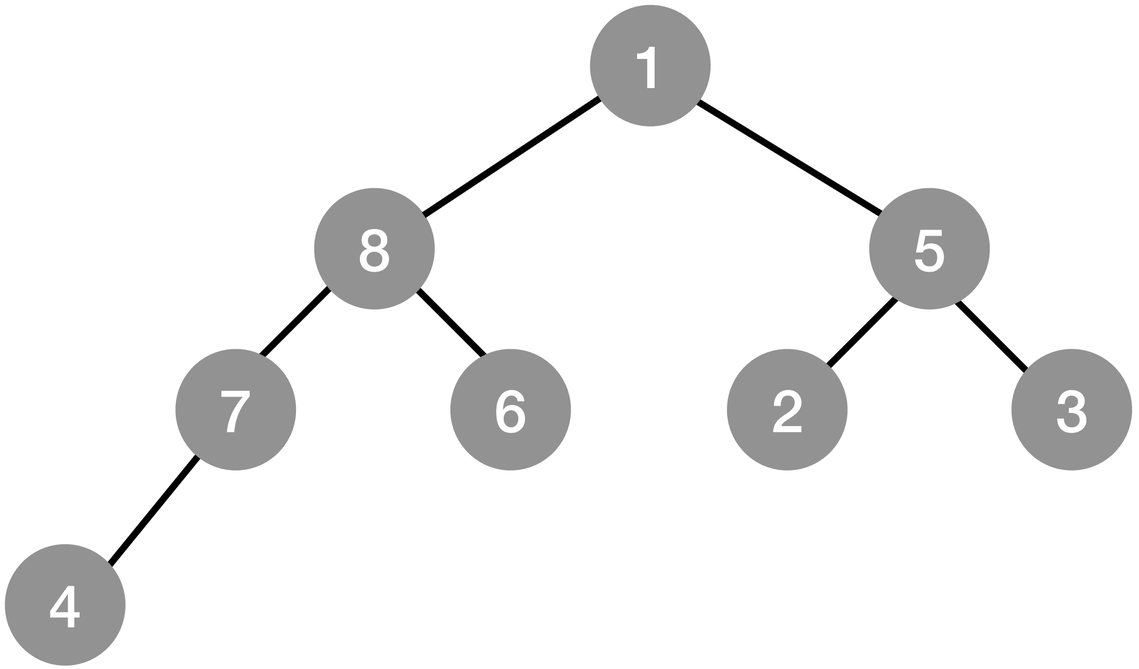
10

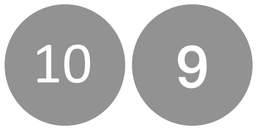
Pairwise

- **Sorting-based ranking: Heapsort**

  Step 2: heapify the tree

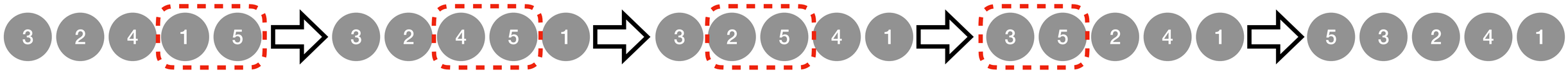
LLM call: 6



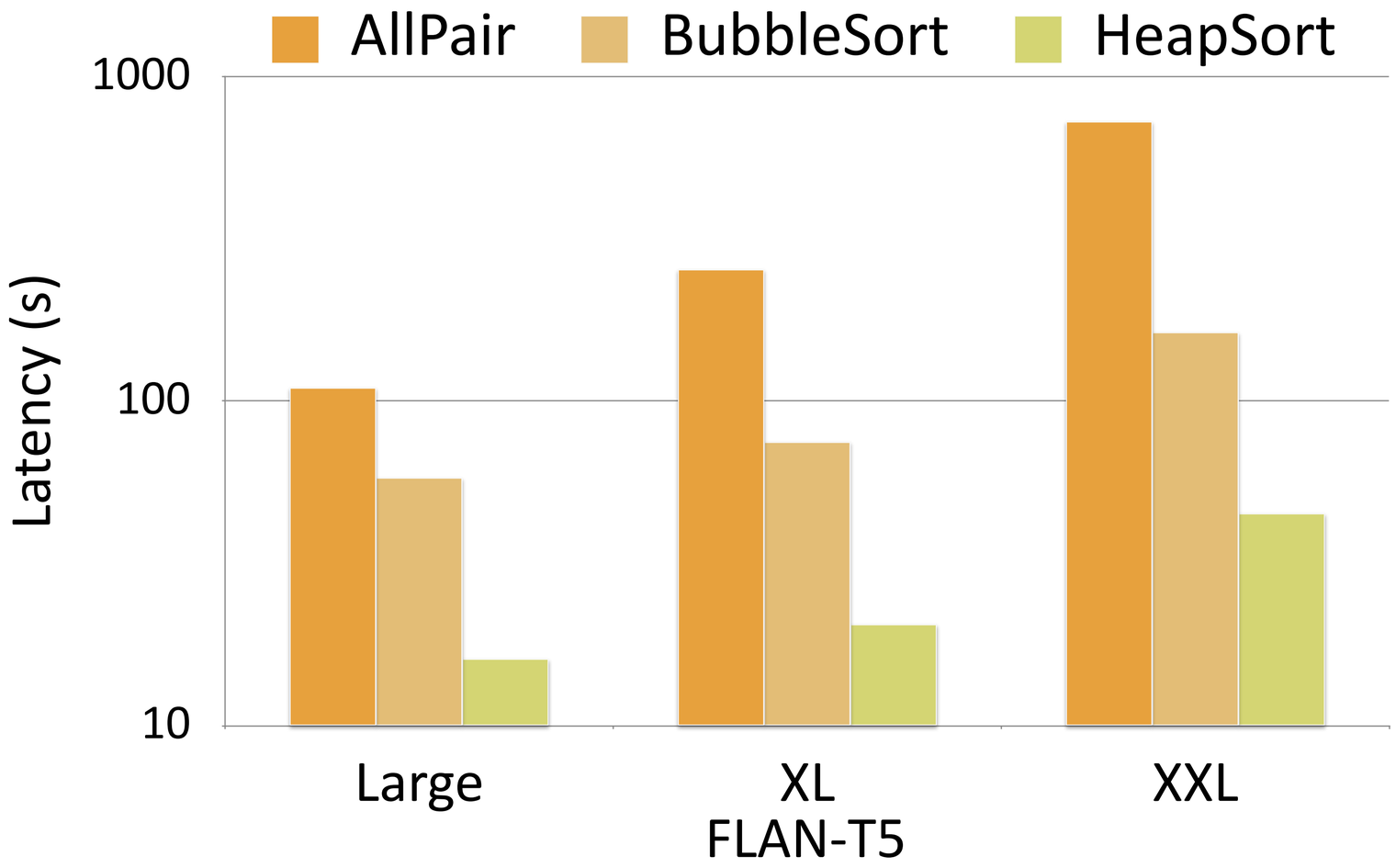


Pairwise

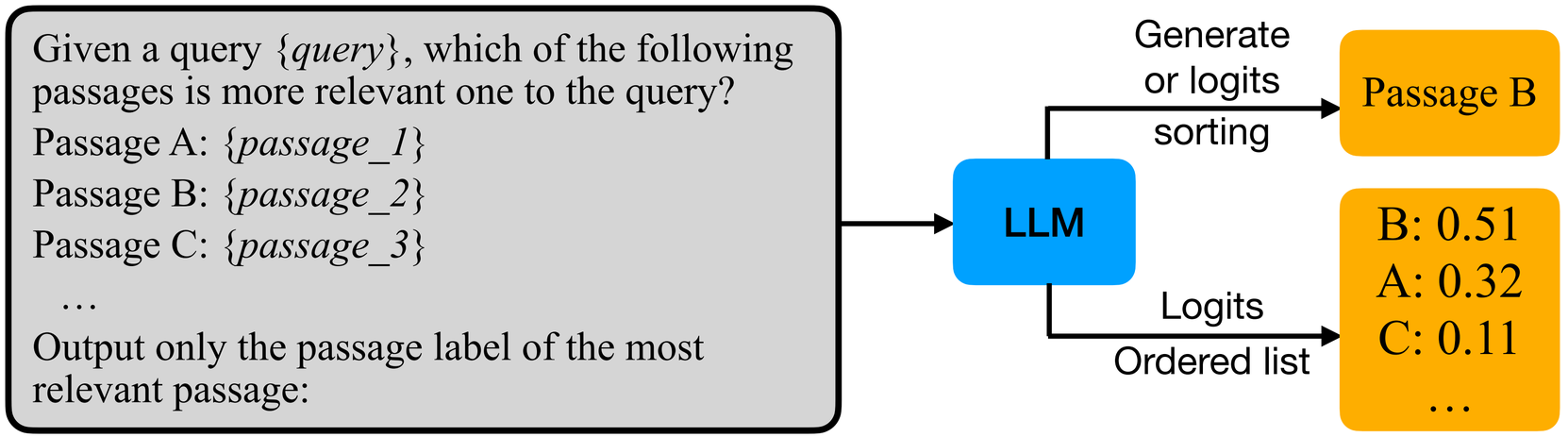
- **Sorting-based ranking: Bubblesort**



Pairwise Efficiency



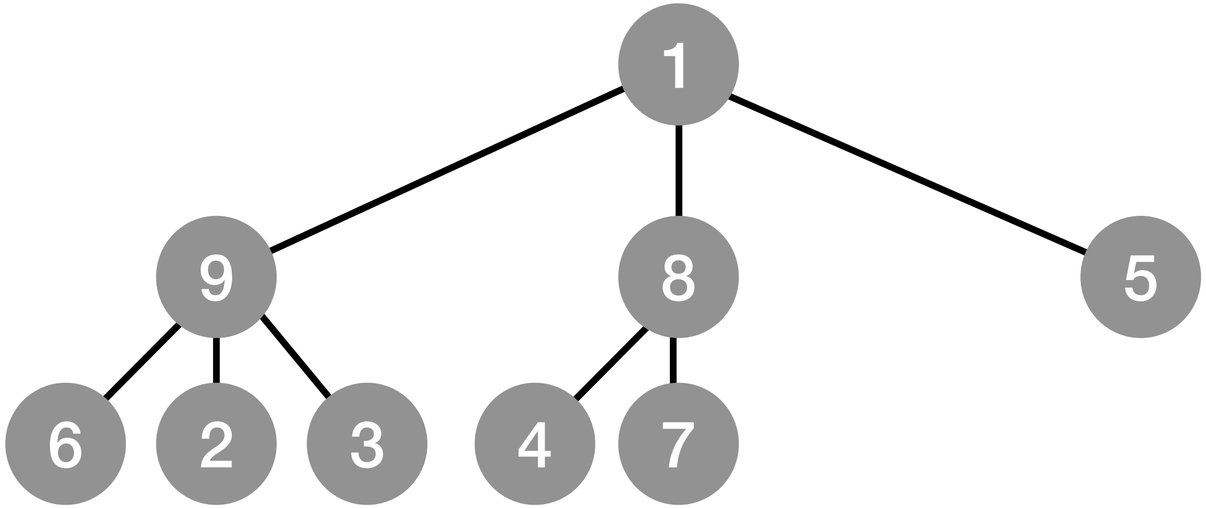
LLMs as Rankers: Setwise



Zhuang, S., Zhuang, H., Koopman, B. and Zuccon, G., 2023. A Setwise Approach for Effective and Highly Efficient Zero-shot Ranking with Large Language Models. *SIGIR*

Setwise

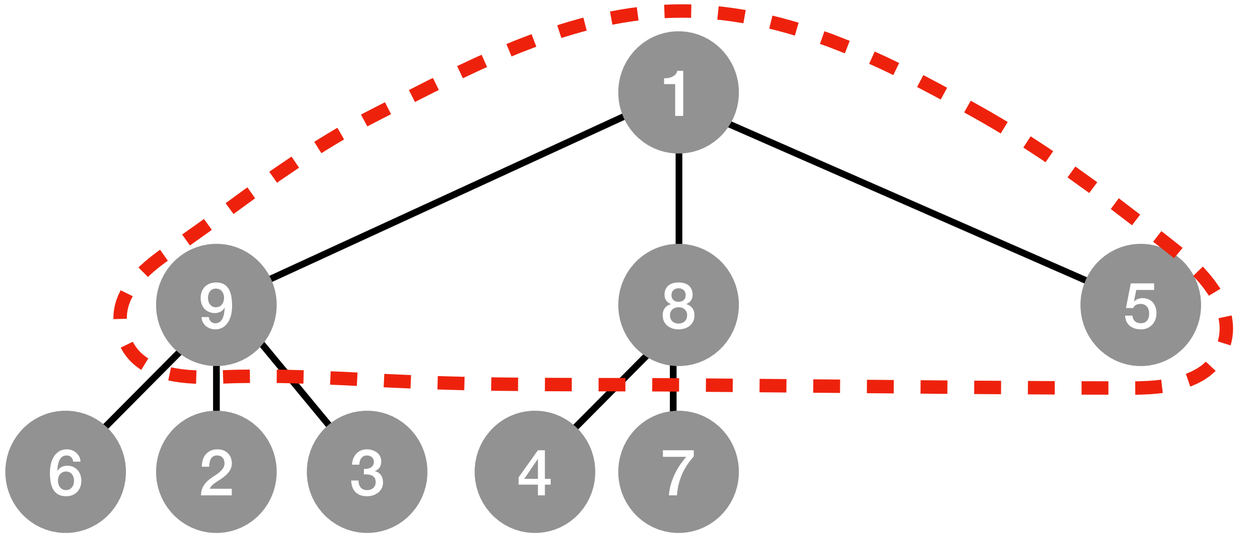
- **Sorting-based ranking: Heapsort**



Setwise

- **Sorting-based ranking: Heapsort**

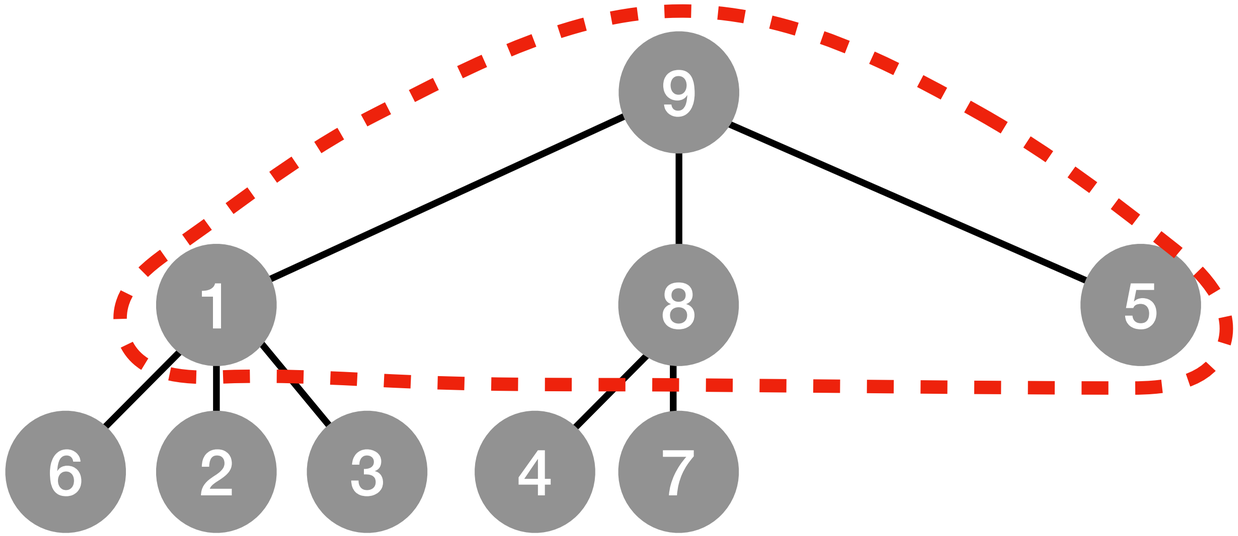
LLM call: 1



Setwise

- **Sorting-based ranking: Heapsort**

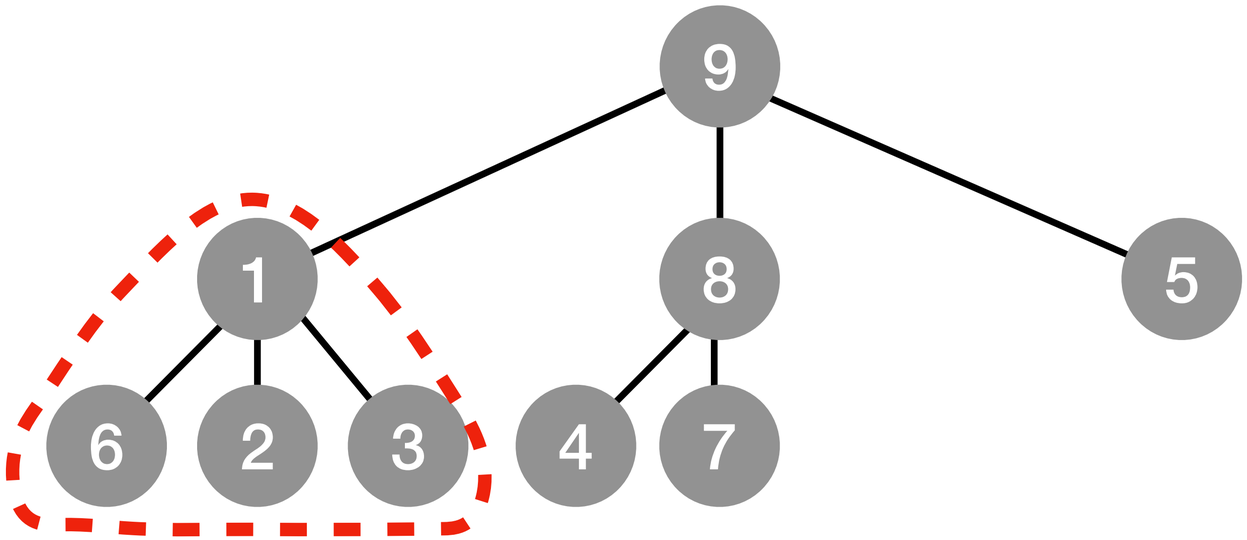
LLM call: 1



Setwise

- **Sorting-based ranking: Heapsort**

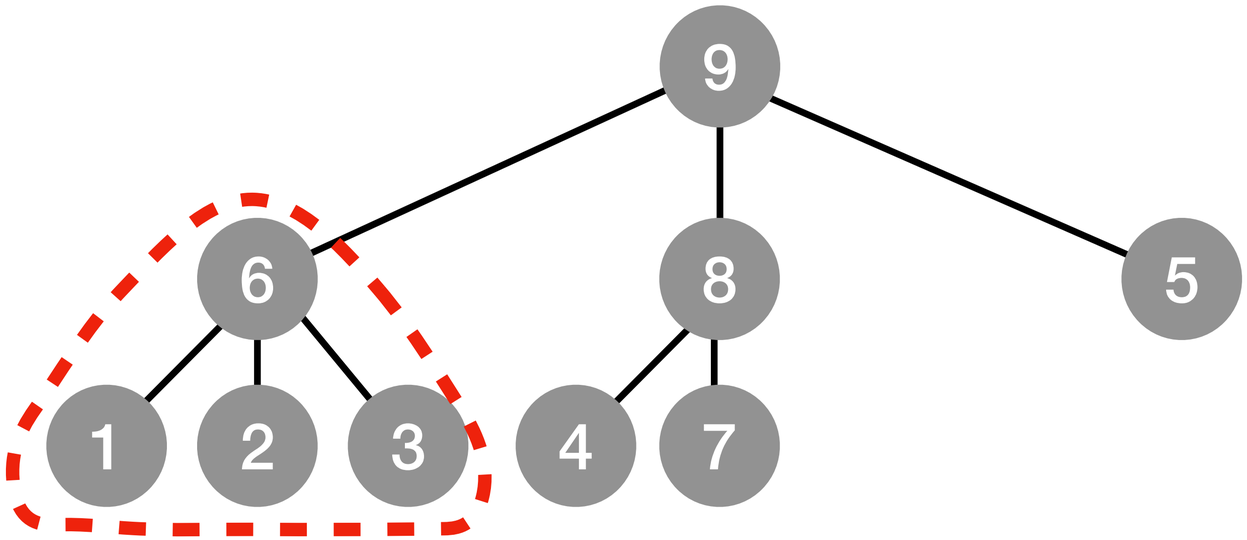
LLM call: 2



Setwise

- **Sorting-based ranking: Heapsort**

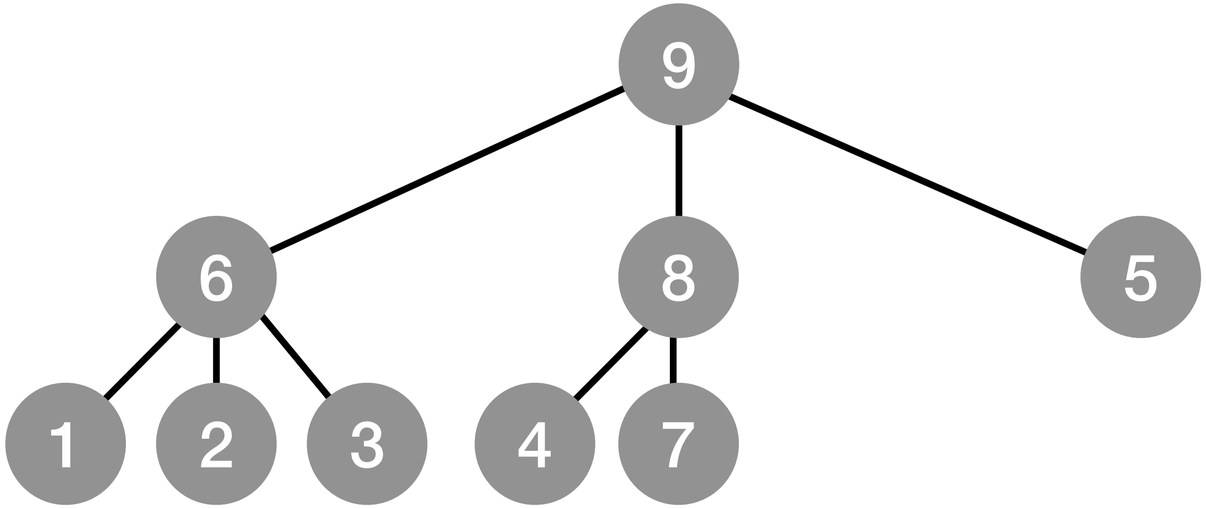
LLM call: 2



Setwise

- **Sorting-based ranking: Heapsort**

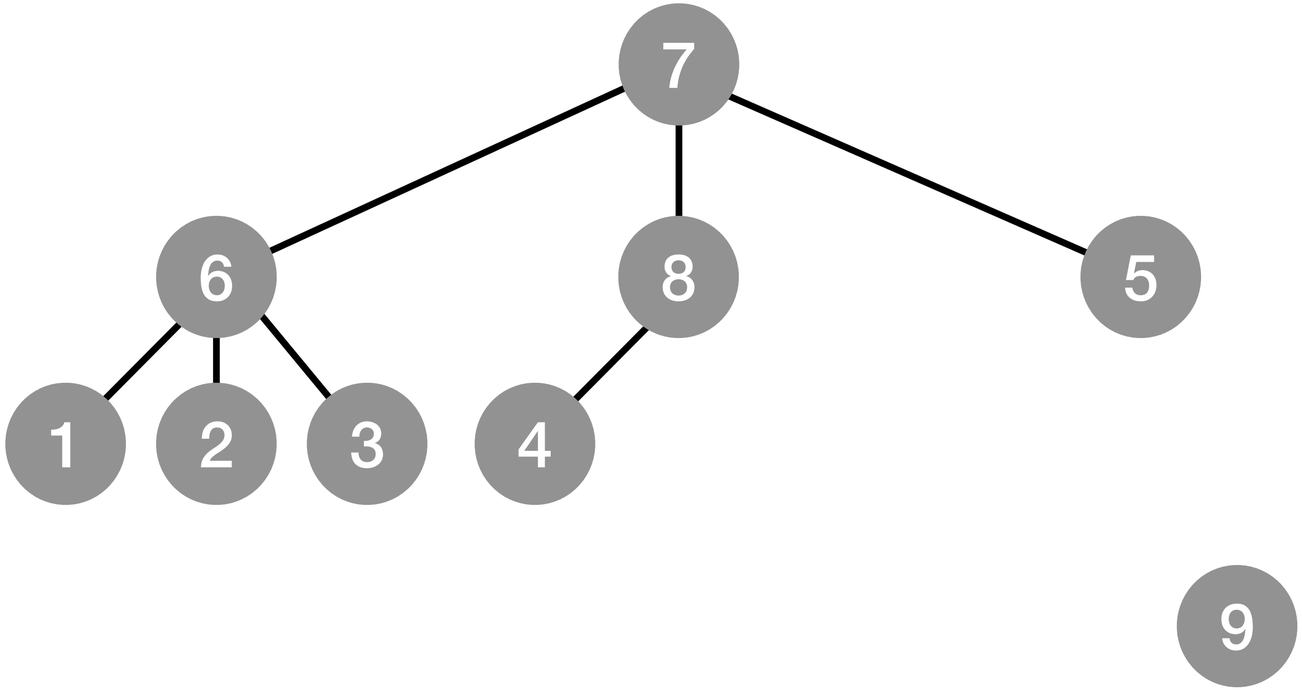
LLM call: 2



Setwise

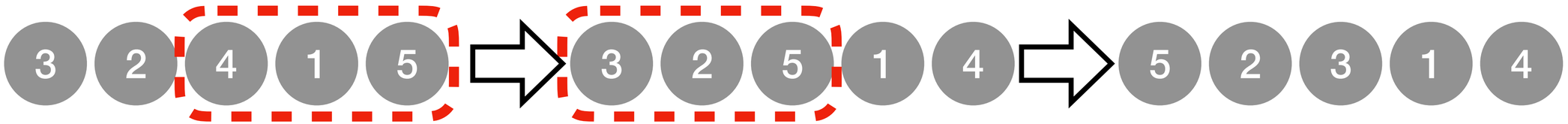
- **Sorting-based ranking: Heapsort**

LLM call: 2

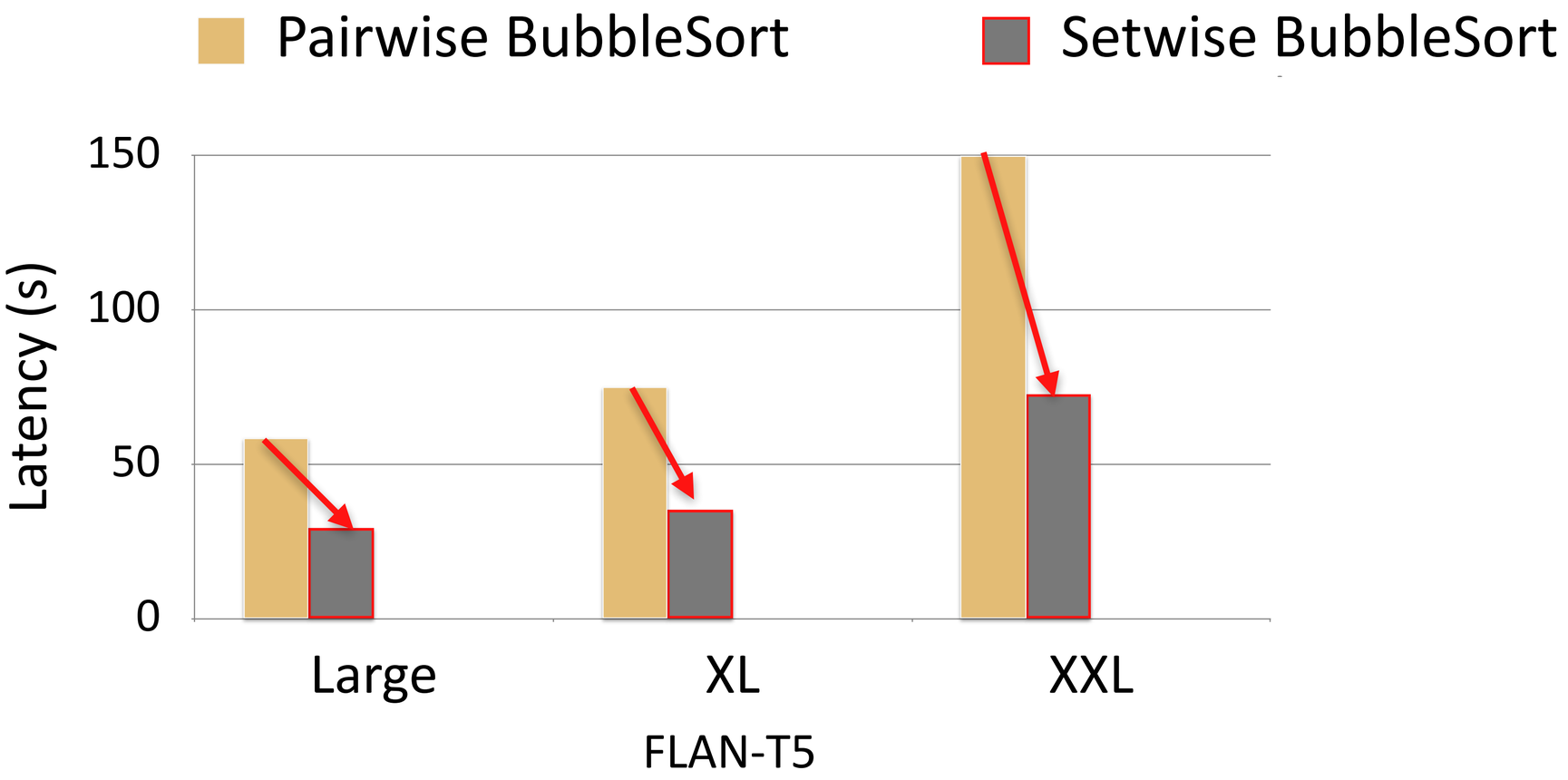


Setwise

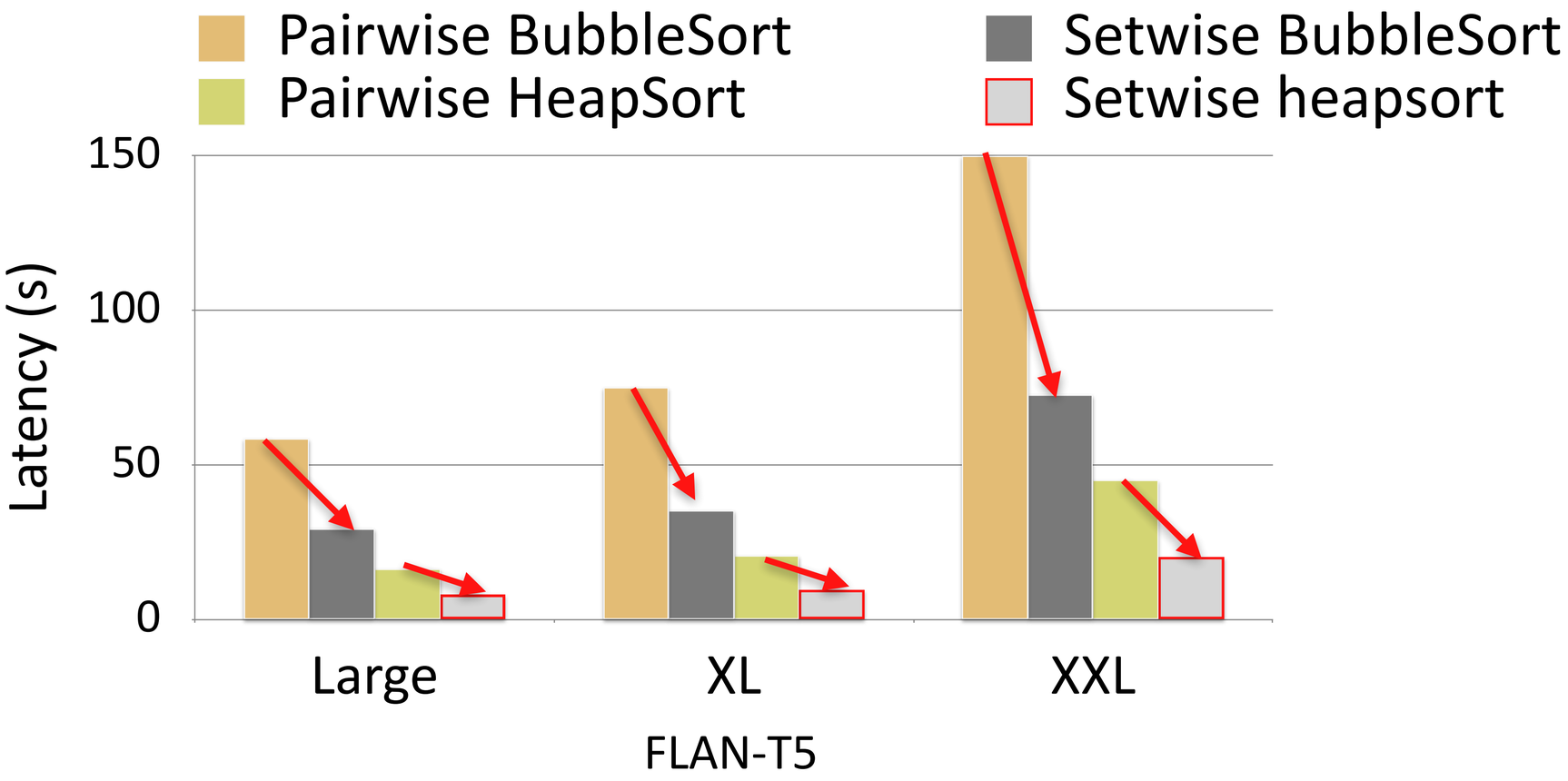
- **Sorting-based ranking: Bubblesort**



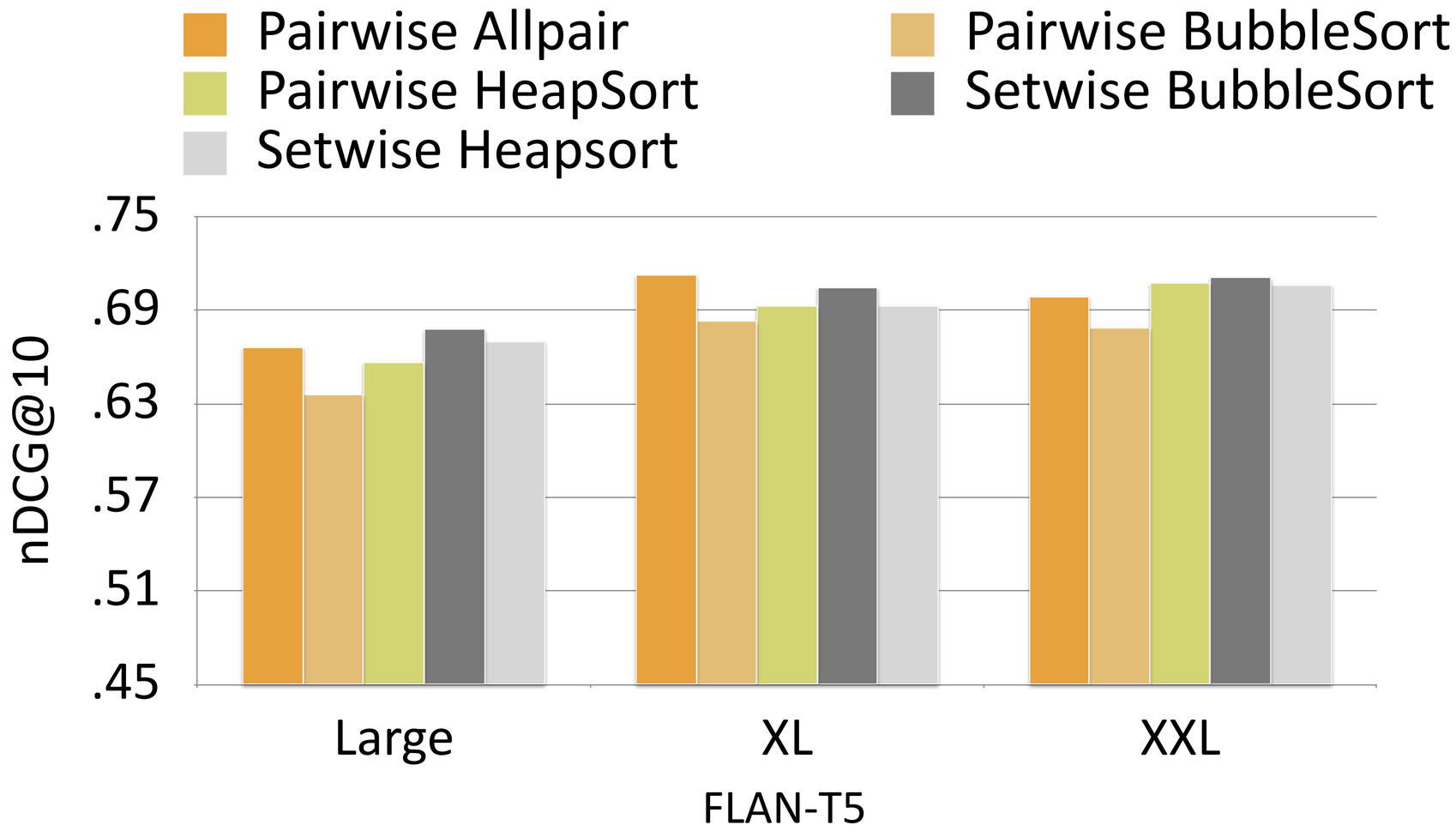
Setwise Efficiency



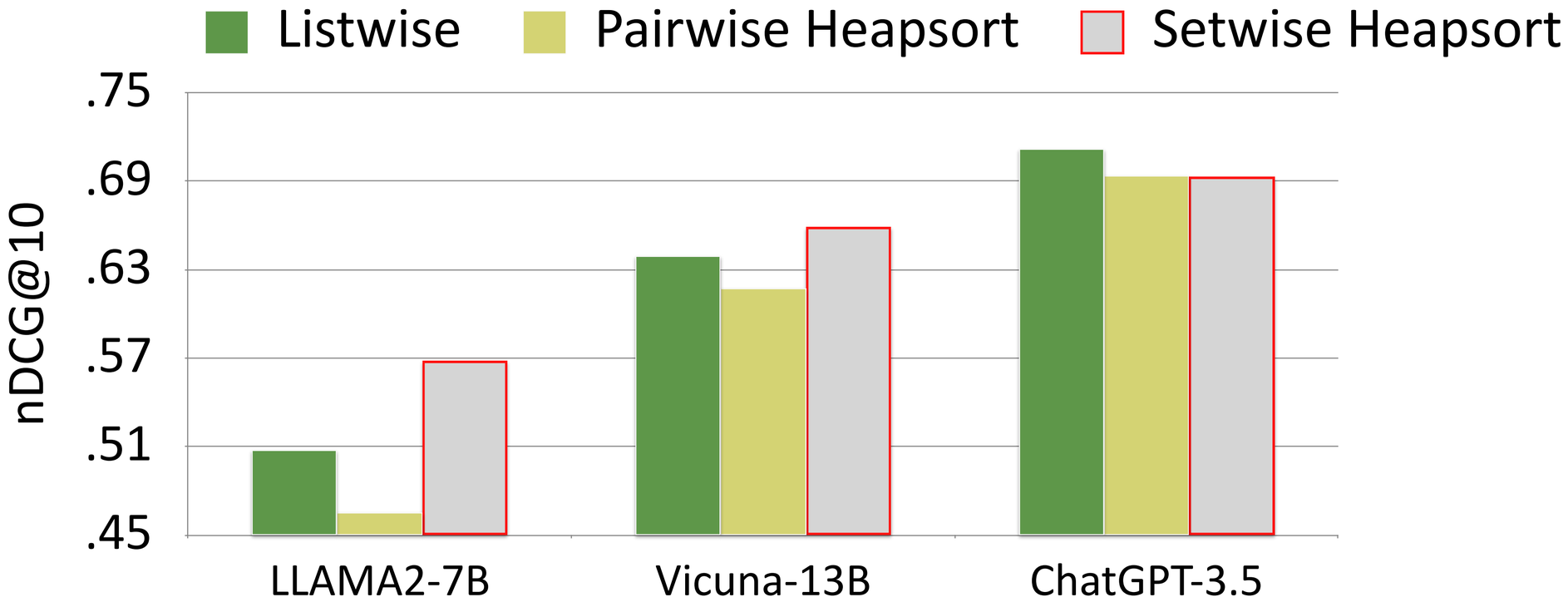
Pairwise Efficiency



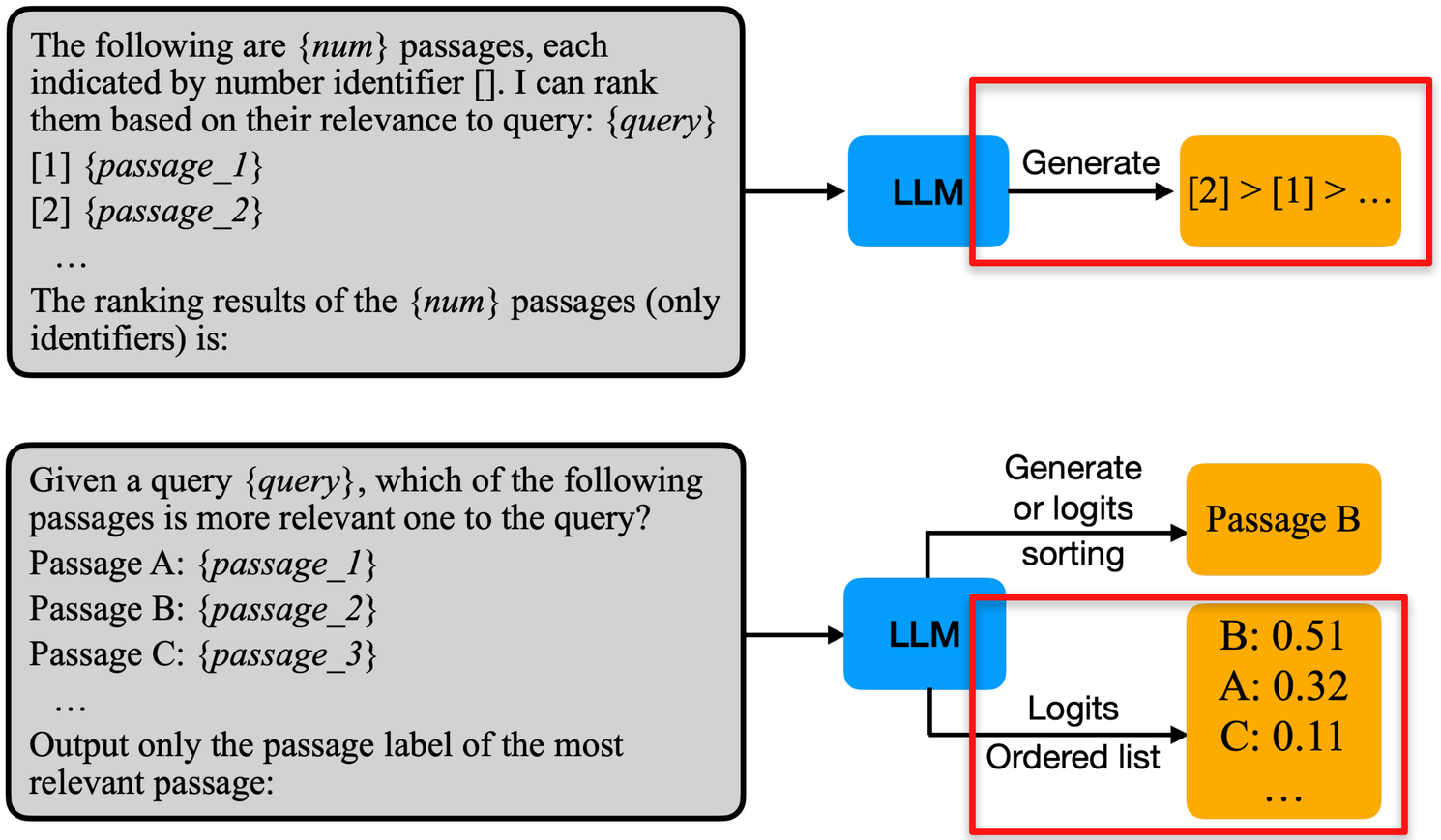
Setwise Effectiveness



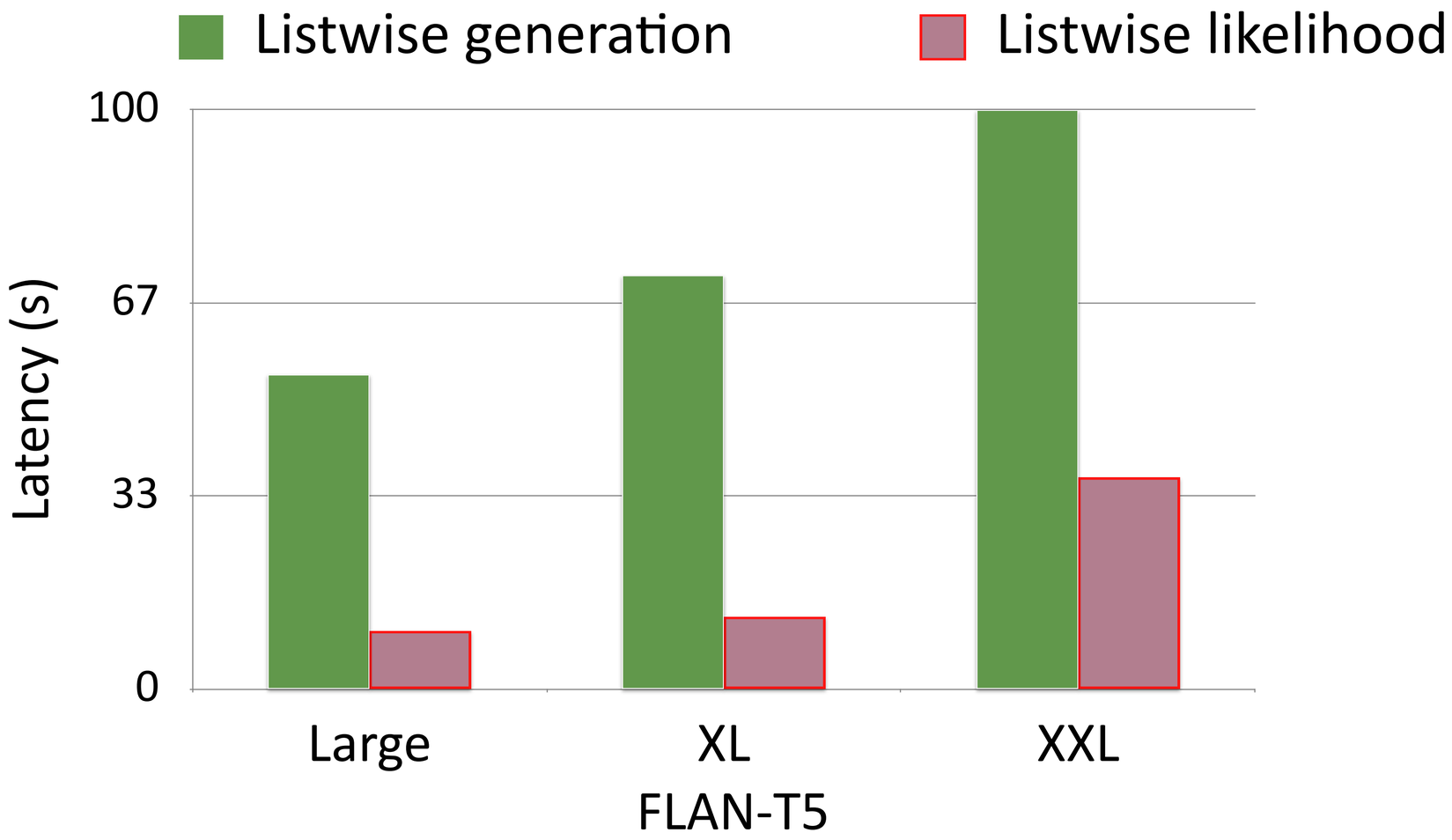
Setwise Effectiveness



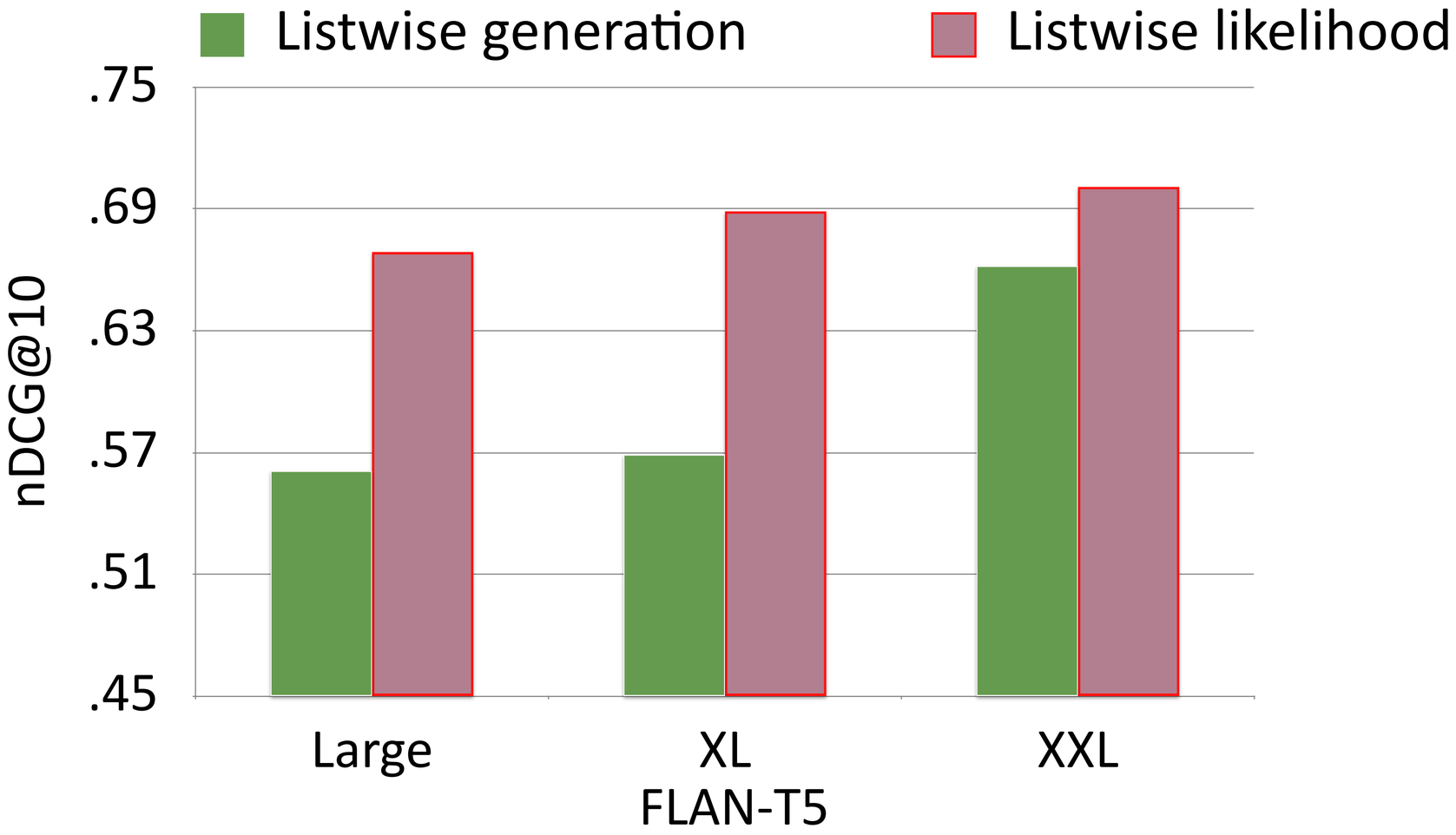
Listwise Ranking with Setwise Prompting



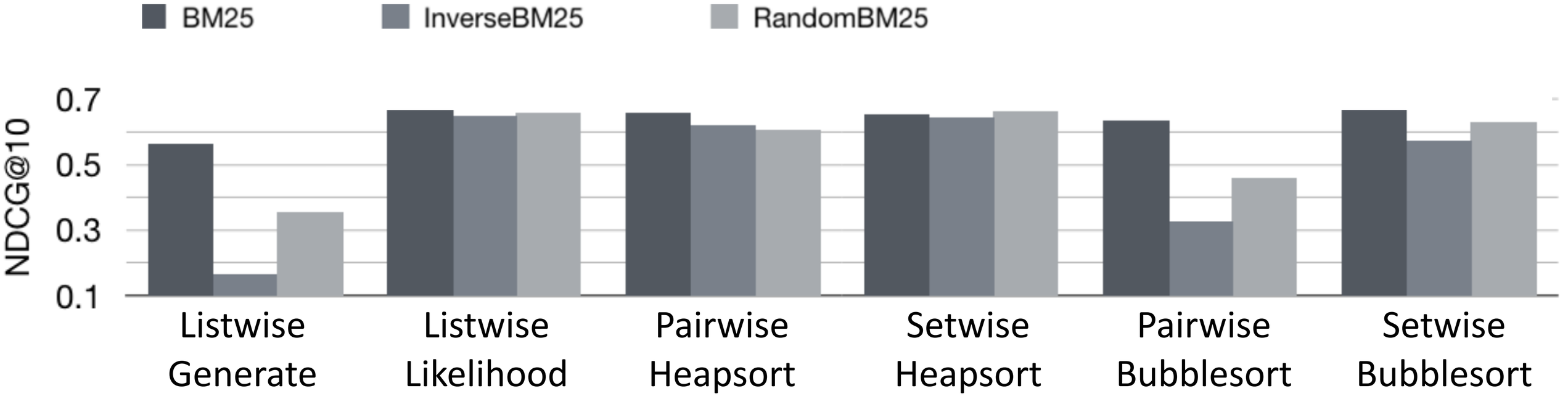
Efficiency


Listwise generation
Listwise likelihood
100
67
33
0
Latency (s)
Large
XL
XXL
FLAN-T5

Effectiveness



Sensitivity to the Initial Ranking



Properties

Methods	Logits	Generate
pointwise.qlm	✓	
pointwise.yes_no	✓	
listwise.generation		✓
listwise.likelihood	✓	
pairwise.allpair	✓	✓
pairwise.heapsort	✓	✓
pairwise.bubblesort	✓	✓
setwise.heapsort	✓	✓
setwise.bubblesort	✓	✓

Properties

Methods	Logits	Generate	Batching
pointwise.qlm	✓		✓
pointwise.yes_no	✓		✓
listwise.generation		✓	
listwise.likelihood	✓		
pairwise.allpair	✓	✓	✓
pairwise.heapsort	✓	✓	
pairwise.bubblesort	✓	✓	
setwise.heapsort	✓	✓	
setwise.bubblesort	✓	✓	

Properties

Methods	Logits	Generate	Batching	Top-k
pointwise.qlm	✓		✓	
pointwise.yes_no	✓		✓	
listwise.generation		✓		✓
listwise.likelihood	✓			✓
pairwise.allpair	✓	✓	✓	
pairwise.heapsort	✓	✓		✓
pairwise.bubblesort	✓	✓		✓
setwise.heapsort	✓	✓		✓
setwise.bubblesort	✓	✓		✓

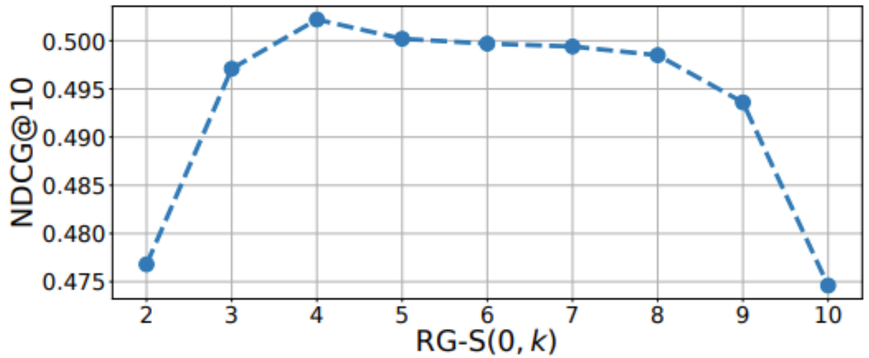
Properties

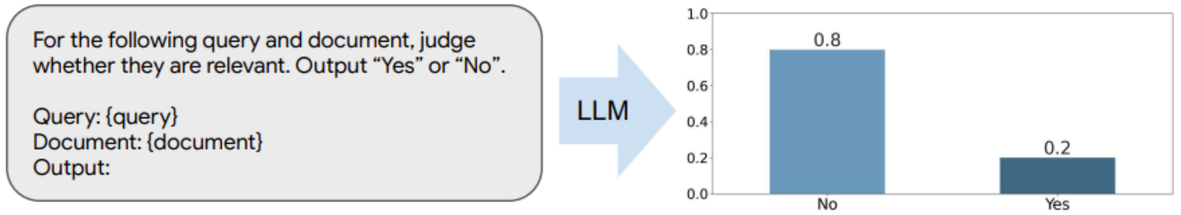
Methods	Logits	Generate	Batching	Top-k	# LLM calls
pointwise.qlm	✓		✓		$O(N)$
pointwise.yes_no	✓		✓		$O(N)$
listwise.generation		✓		✓	$O(r * (N/s))$
listwise.likelihood	✓			✓	$O(r * (N/s))$
pairwise.allpair	✓	✓	✓		$O(N^2 - N)$
pairwise.heapsort	✓	✓		✓	$O(k * \log_2 N)$
pairwise.bubblesort	✓	✓		✓	$O(k * N)$
setwise.heapsort	✓	✓		✓	$O(k * \log_c N)$
setwise.bubblesort	✓	✓		✓	$O(k * (N/(c-1)))$

Zeroshot LLM-based Rankers' Variations

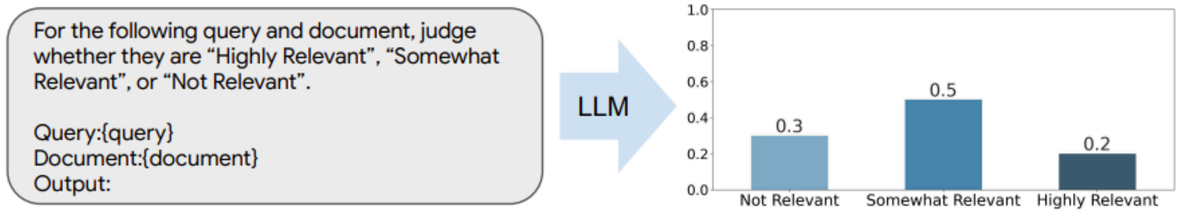
Pointwise Variation

- Beyond yes and no.

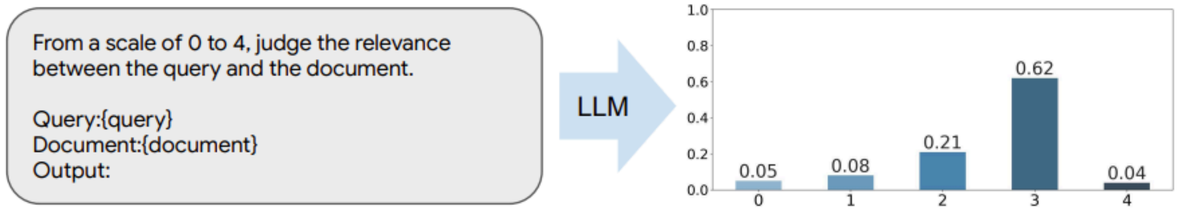




(a) Yes-No relevance generation



(b) Fine-grained relevance label generation

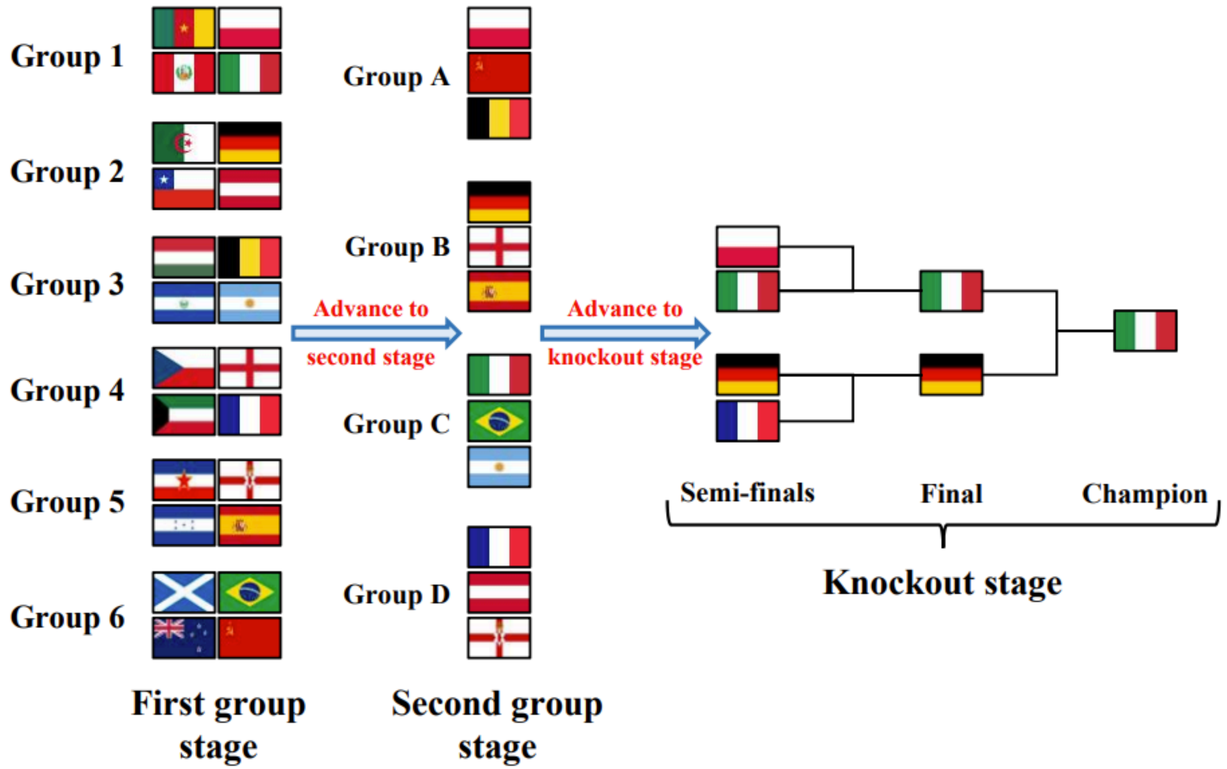


(c) Rating scale relevance generation

Zhuang, H., Qin, Z., Hui, K., Wu, J., Yan, L., Wang, X., Bendersky, M. 2024. Beyond Yes and No: Improving Zero-Shot LLM Rankers via Scoring Fine-Grained Relevance Labels, NAACL

TourRank

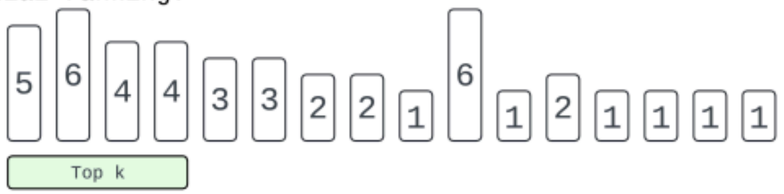
- Sport tournaments scoring



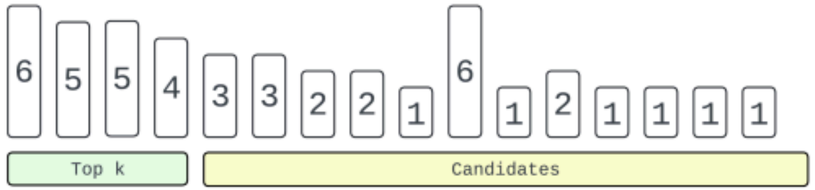
Chen, Y., et al. 2025. TourRank: Utilizing Large Language Models for Documents Ranking with a Tournament-Inspired Strategy, TheWebConf

Setwise Insertion sort

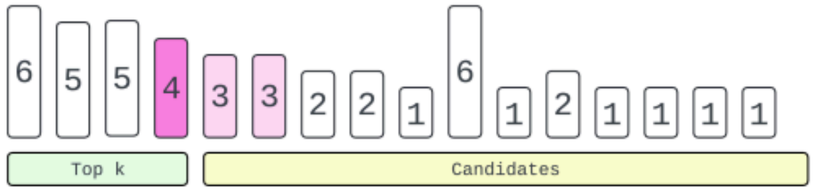
0. Start with a partially sorted list based on an initial ranking.



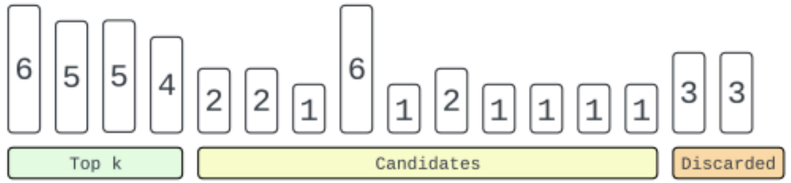
1. Use setwise.heapsort to sort the top-k elements.



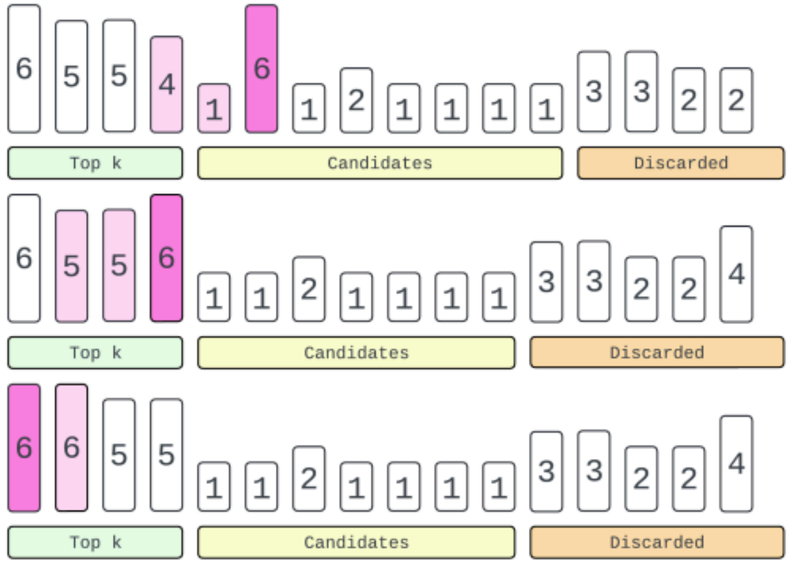
2. Compare small candidate sets with the smallest element in top-k (the "guard").



3. Discard candidates if the guard is the largest.



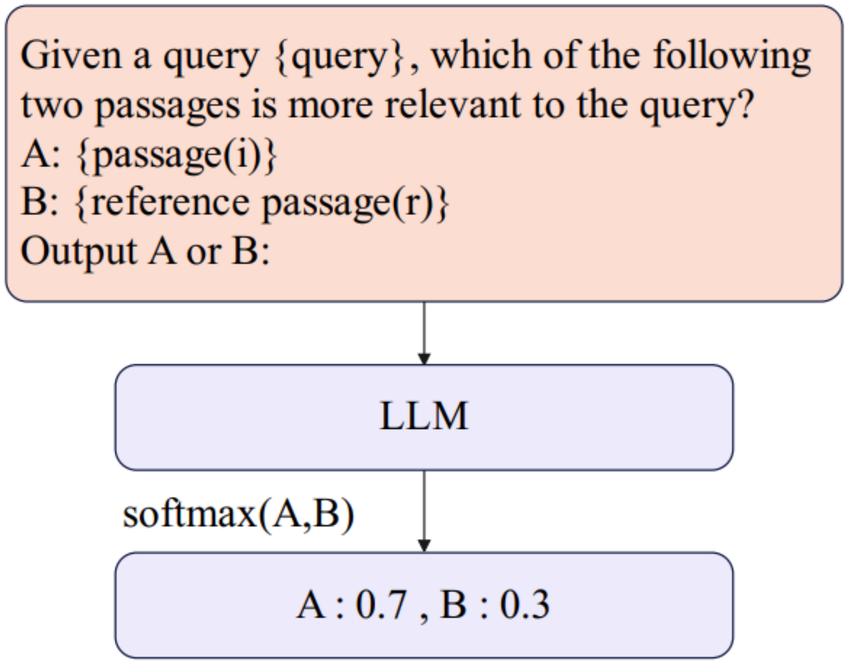
4. If at least one candidate is larger than the guard, discard the guard, insert the candidate into top k, and fix the order.

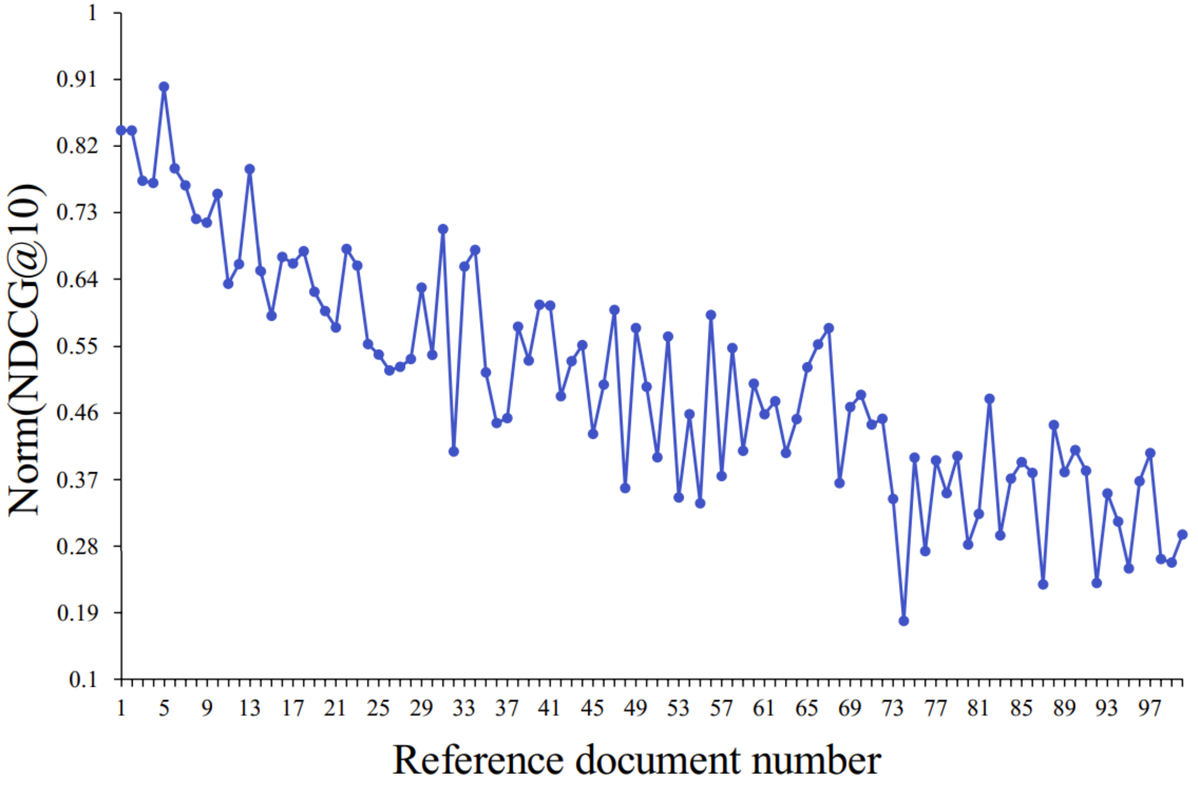


5. Repeat steps 2–4 until the candidate list is empty.

Podolak, J., Peric, L., Janicijevic, M., Petcu, R. 2025. Beyond Reproducibility: Advancing Zero-shot LLM Reranking Efficiency with Setwise Insertion. SIGIR

Combine Pointwise and Pairwise





Li, J., Hu, X., Zhao, Y., Zhuang, S., Zhang, H. 2025. Beyond Reproducibility: Advancing Zero-shot LLM Reranking Efficiency with Setwise Insertion. arXiv
Long, K., Li, S., Xu, C., Tang, J., Wang, T. 2025. Precise Zero-Shot Pointwise Ranking with LLMs through Post-Aggregated Global Context Information. SIGIR
Wisznia, J., Bolanos, C., Tollo, J., Marraffini, G., Gianolini, A., Hsueh, N., Corro, L. 2025, Are Optimal Algorithms Still Optimal? Rethinking Sorting in LLM-Based Pairwise Ranking with Batching and Caching. ACL

Impact of Prompt Variations on LLM Rankers

- Prompts proposed for different LLM ranker methods vary largely
- Not just in terms of instruction for ranking, but also for (unrelated) additional wordings, e.g. role playing, and ordering of components (e.g. passage first, then query — or vice versa?)
- What are the effects of prompt wordings on methods? What makes a good prompt?

Method	Prompt
PRP, Qin et al.	Passage: {text} Query: {query} Does the passage answer the query?
RankGPT, Sun et al.	You are RankGPT, an intelligent assistant that can rank passages based on their relevancy to the query. I will provide you with num passages, each indicated by number identifier []. Rank the passages based on their relevance to query: {query}. {PASSAGES} Search Query: {query}. Rank the num passages above based on their relevance to the search query. The passages should be listed in descending order using identifiers. The most relevant passages should be listed first. The output format should be [] > [], e.g., [1] > [2]. Only response the ranking results, do not say any word or explain.

Role Playing

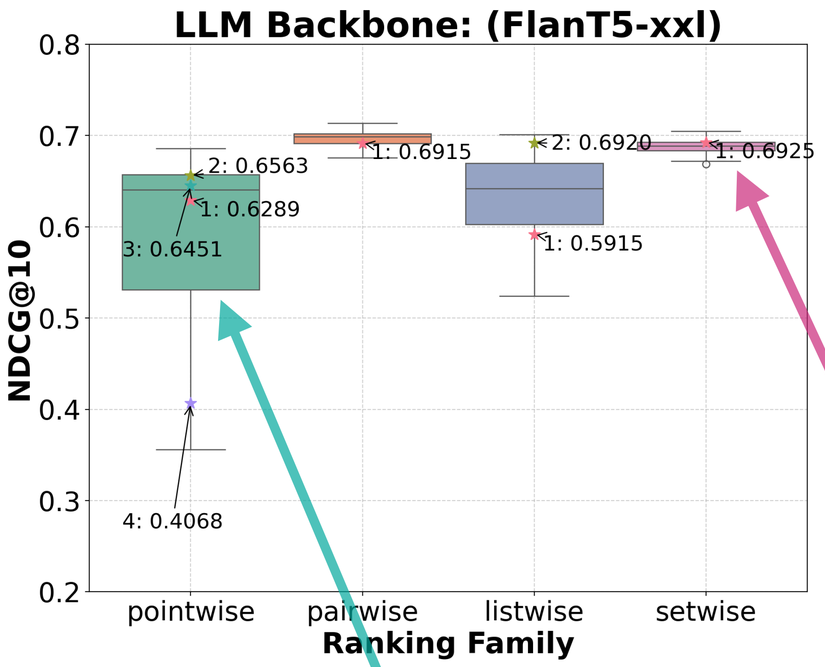
Formatting Instr.

Restriction on Output

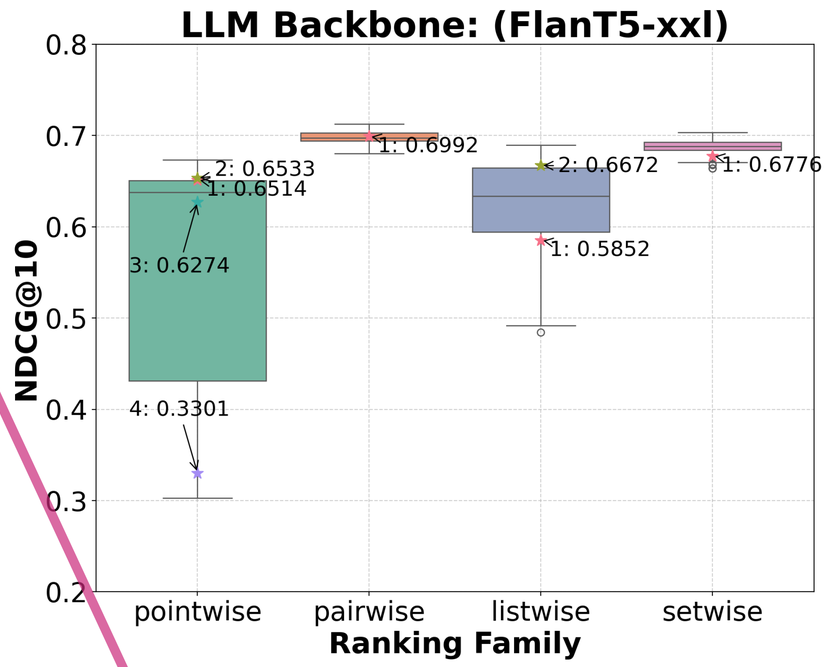
Sun, S., Zhuang, S., Wang, S. and Zuccon, G., 2024. An Investigation of Prompt Variations for Zero-shot LLM-based Rankers. *ECIR*

Impact of Prompt Variations on LLM Rankers

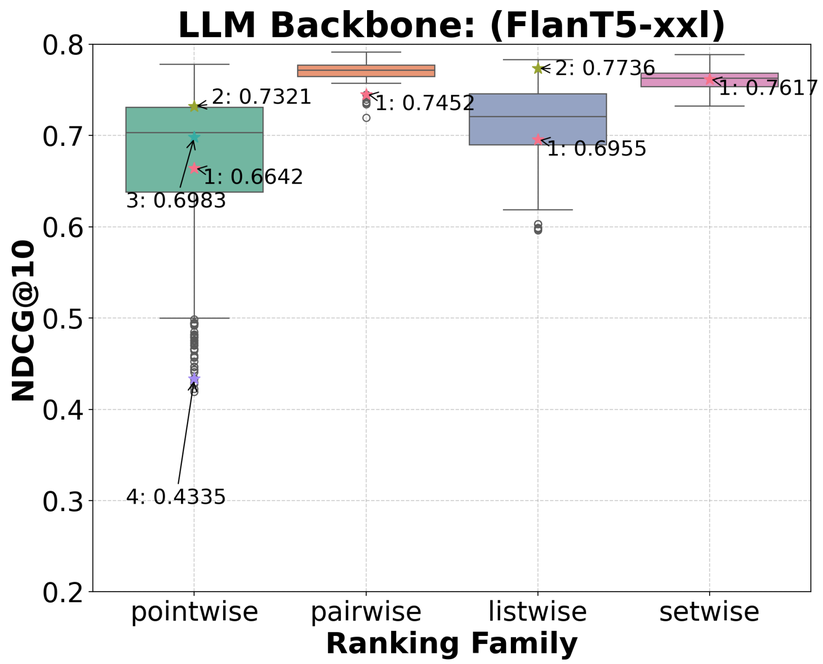
TREC DL 2019



TREC DL 2020



COVID (BEIR)



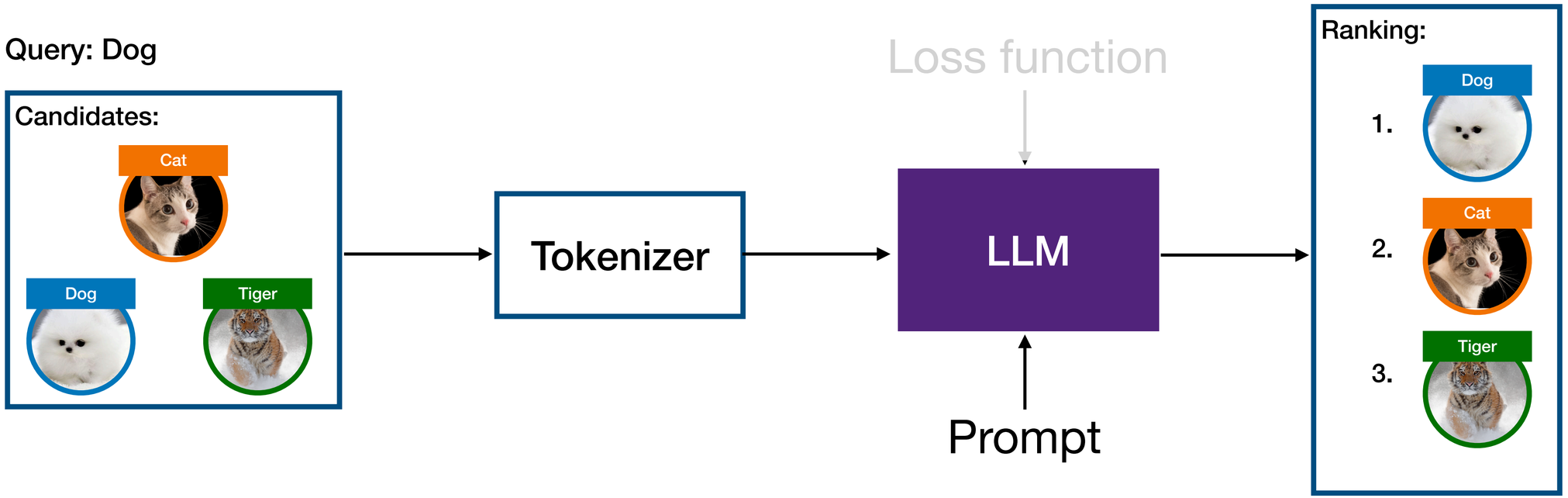
Pointwise, listwise often display large variations of effectiveness across different prompts

Pairwise, set wise very robust across different prompts

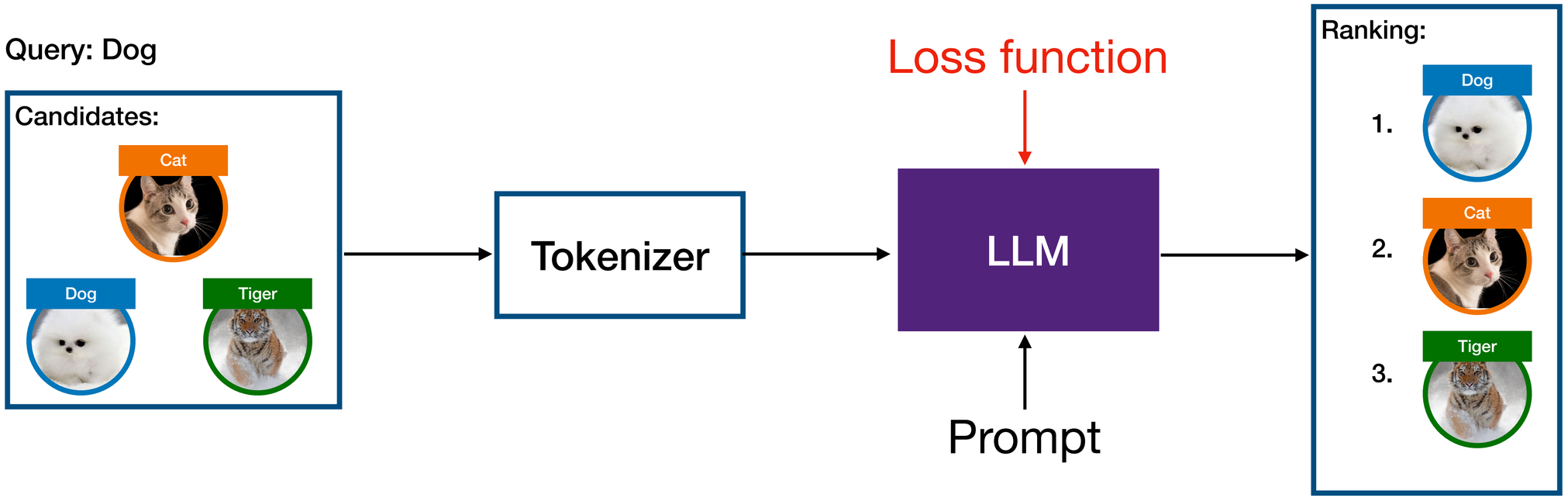
For all approaches, there are better prompts than the one in the original paper

Sun, S., Zhuang, S., Wang, S. and Zuccon, G., 2024. An Investigation of Prompt Variations for Zero-shot LLM-based Rankers. *ECIR*

LLM as Rankers



LLM as Rankers



Distillation

- RankZephyr: 7B listwise ranker distilled from large GPT Models, outperforms GPT Models

Pradeep, R., Sharifymoghaddam, S., Lin, J. 2023. RankZephyr: Effective and Robust Zero-Shot Listwise Reranking is a Breeze!. arXiv

Distillation

- RankZephyr: 7B listwise ranker distilled from large GPT Models, outperforms GPT Models
  - Stage1: 100k training queries from MS MARCO, with rankings provided by ChatGPT 3.5.
    - Excludes malformed generations.
    - Has randomly shuffled input order.

Pradeep, R., Sharifymoghaddam, S., Lin, J. 2023. RankZephyr: Effective and Robust Zero-Shot Listwise Reranking is a Breeze!. arXiv

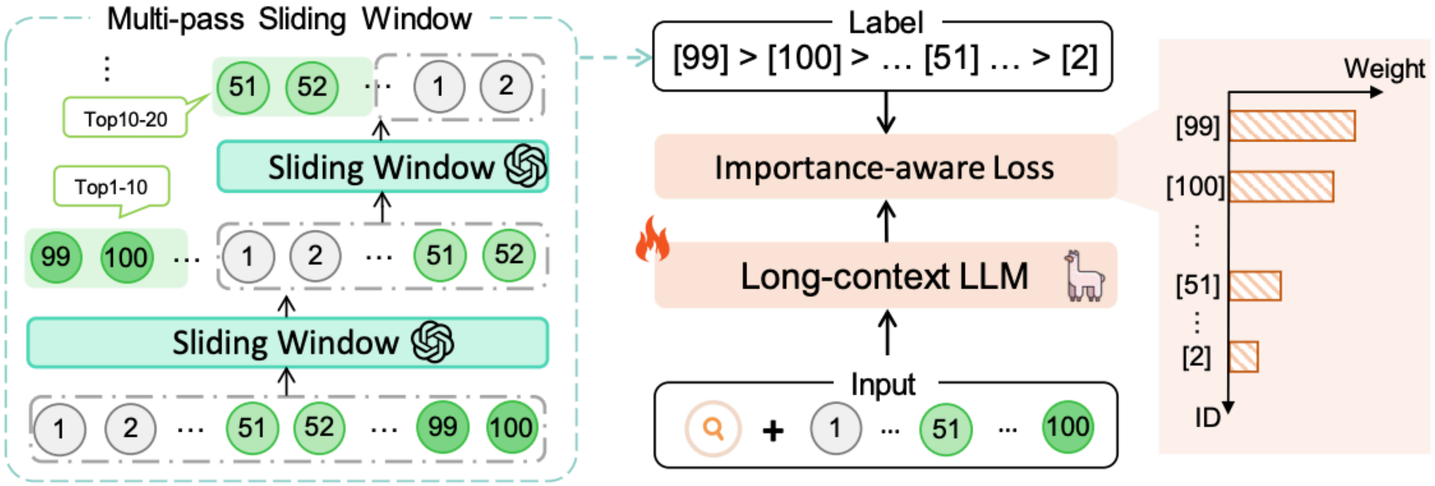
Distillation

- RankZephyr: 7B listwise ranker distilled from large GPT Models, outperforms GPT Models
  - Stage1: 100k training queries from MS MARCO, with rankings provided by ChatGPT 3.5.
    - Excludes malformed generations.
    - Has randomly shuffled input order.
  - Stage2: 5k queries with rankings from GPT-4.5.
    - Diverse queries to maximize query embedding differences.

Pradeep, R., Sharifymoghaddam, S., Lin, J. 2023. RankZephyr: Effective and Robust Zero-Shot Listwise Reranking is a Breeze!. arXiv

Distillation

- Sliding Windows vs Full ranking distillation.

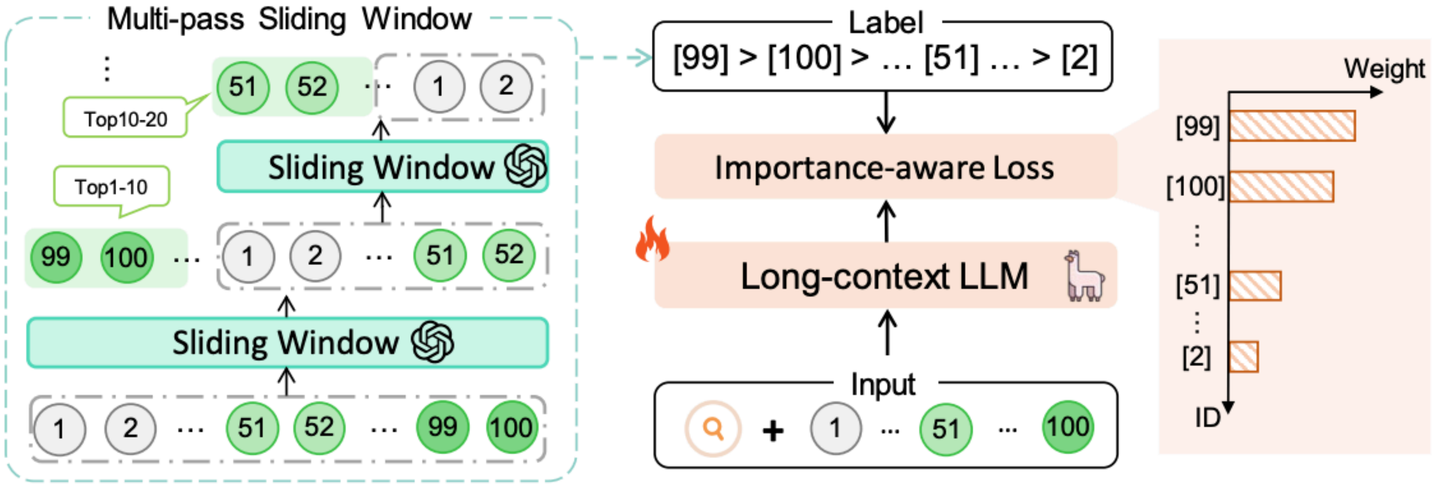


Models	Strategy	DL19	DL20	Covid	DBPedia	SciFact	NFCorpus	Signal	Robust04	Touche	News	Avg.
Zero-shot												
GPT-4o	Sliding	**74.78**	69.52	**83.41**	**45.56**	**77.41**	**39.67**	**34.20**	**60.25**	32.26	**51.92**	**53.09**
	Full	73.94	**70.03**	82.10	43.31	74.85	39.00	32.63	55.95	30.42	47.96	50.78
SFT from GPT-4o												
RankMistral_{20}	Sliding	70.34	69.58	80.86	42.52	75.82	38.38	33.90	54.69	**37.18**	**51.42**	51.85
RankMistral_{100}	Full	**72.55**	**71.29**	**82.24**	**43.54**	**77.04**	**39.14**	**33.99**	**57.91**	34.63	50.59	**52.40**

Liu W, Ma, X., Zhu, Y., Zhao, Z., Wang, S., Yin, D., Dou, Z. 2025. Sliding Windows Are Not the End: Exploring Full Ranking with Long-Context Large Language Models. ACL

Distillation

- Sliding Windows vs Full ranking distillation.

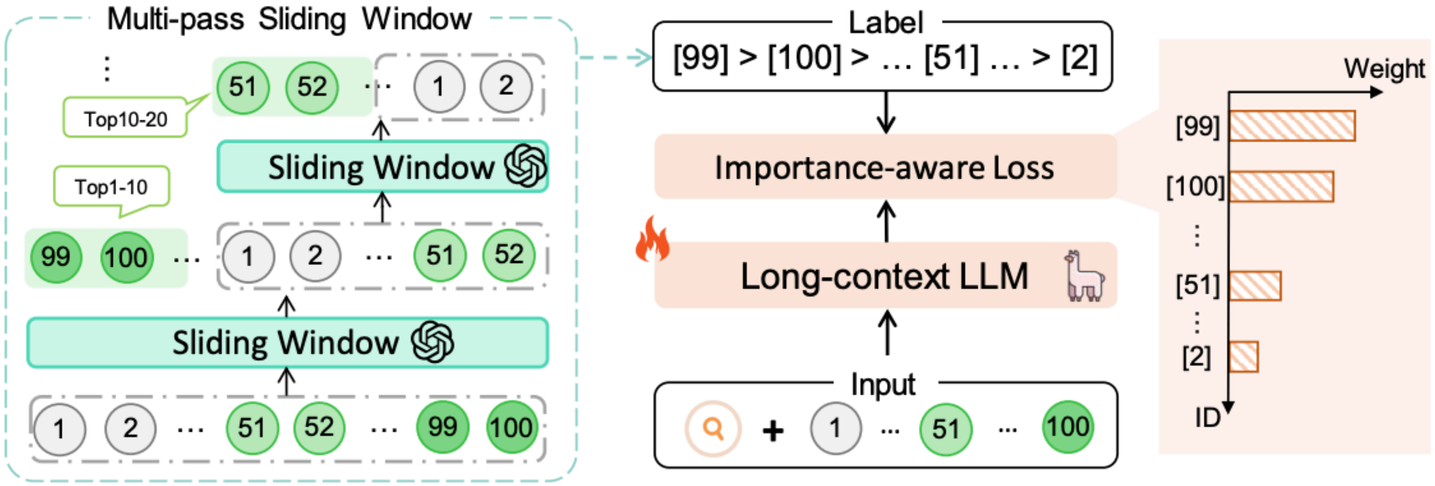


Models	Strategy	DL19	DL20	Covid	DBPedia	SciFact	NFCorpus	Signal	Robust04	Touche	News	Avg.
Zero-shot												
GPT-4o	Sliding	**74.78**	69.52	**83.41**	**45.56**	**77.41**	**39.67**	**34.20**	**60.25**	32.26	**51.92**	**53.09**
	Full	73.94	**70.03**	82.10	43.31	74.85	39.00	32.63	55.95	30.42	47.96	50.78
SFT from GPT-4o												
RankMistral_{20}	Sliding	70.34	69.58	80.86	42.52	75.82	38.38	33.90	54.69	**37.18**	**51.42**	51.85
RankMistral_{100}	Full	**72.55**	**71.29**	**82.24**	**43.54**	**77.04**	**39.14**	**33.99**	**57.91**	34.63	50.59	**52.40**

Liu W, Ma, X., Zhu, Y., Zhao, Z., Wang, S., Yin, D., Dou, Z. 2025. Sliding Windows Are Not the End: Exploring Full Ranking with Long-Context Large Language Models. ACL

Distillation

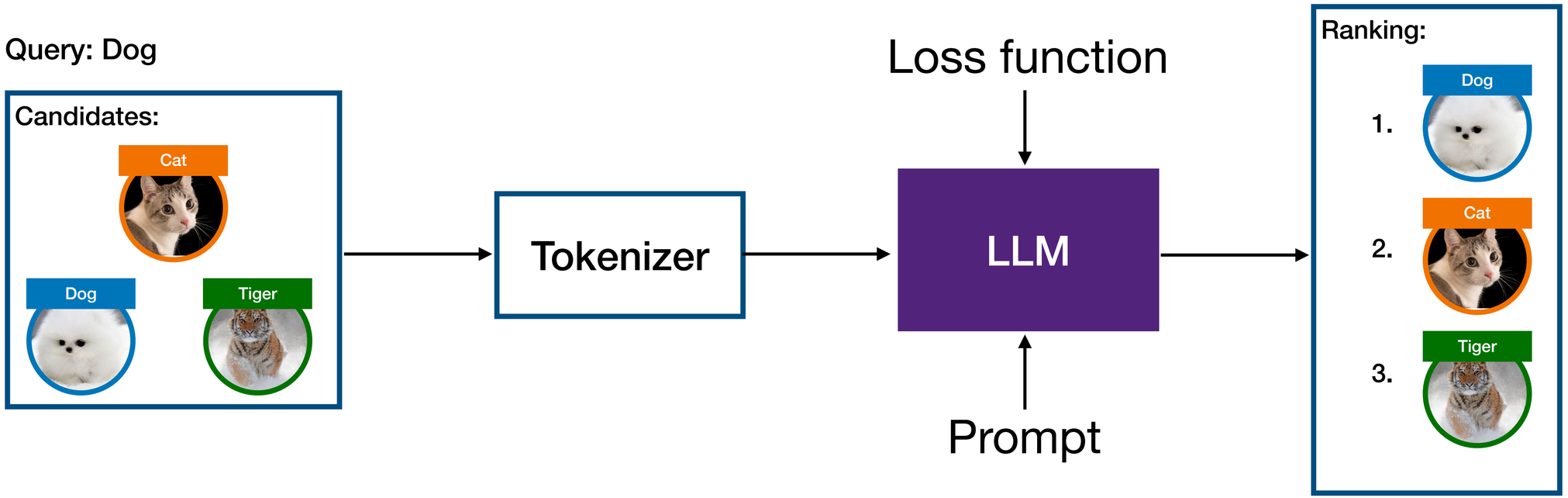
- Sliding Windows vs Full ranking distillation.



Models	Strategy	DL19	DL20	Covid	DBPedia	SciFact	NFCorpus	Signal	Robust04	Touche	News	Avg.
Zero-shot												
GPT-4o	Sliding	**74.78**	69.52	**83.41**	**45.56**	**77.41**	**39.67**	**34.20**	**60.25**	32.26	**51.92**	**53.09**
	Full	73.94	**70.03**	82.10	43.31	74.85	39.00	32.63	55.95	30.42	47.96	50.78
SFT from GPT-4o												
RankMistral_{20}	Sliding	70.34	69.58	80.86	42.52	75.82	38.38	33.90	54.69	**37.18**	**51.42**	51.85
RankMistral_{100}	Full	**72.55**	**71.29**	**82.24**	**43.54**	**77.04**	**39.14**	**33.99**	**57.91**	34.63	50.59	**52.40**

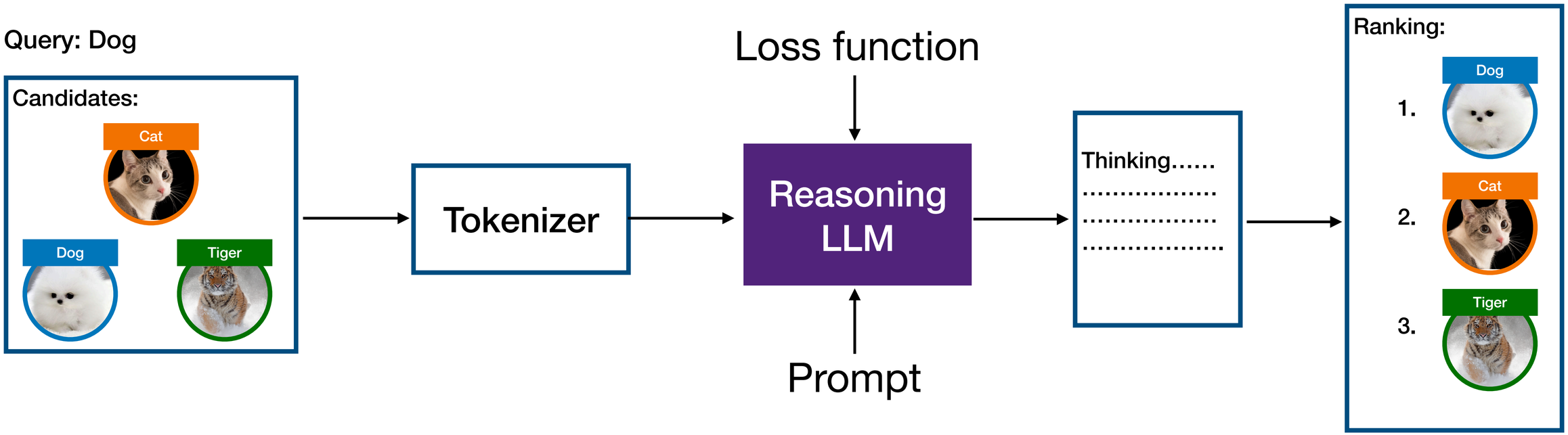
Liu W, Ma, X., Zhu, Y., Zhao, Z., Wang, S., Yin, D., Dou, Z. 2025. Sliding Windows Are Not the End: Exploring Full Ranking with Long-Context Large Language Models. ACL

LLM as Rankers



Reasoning LLMs

Reasoning Rankers



Why Reasoning?

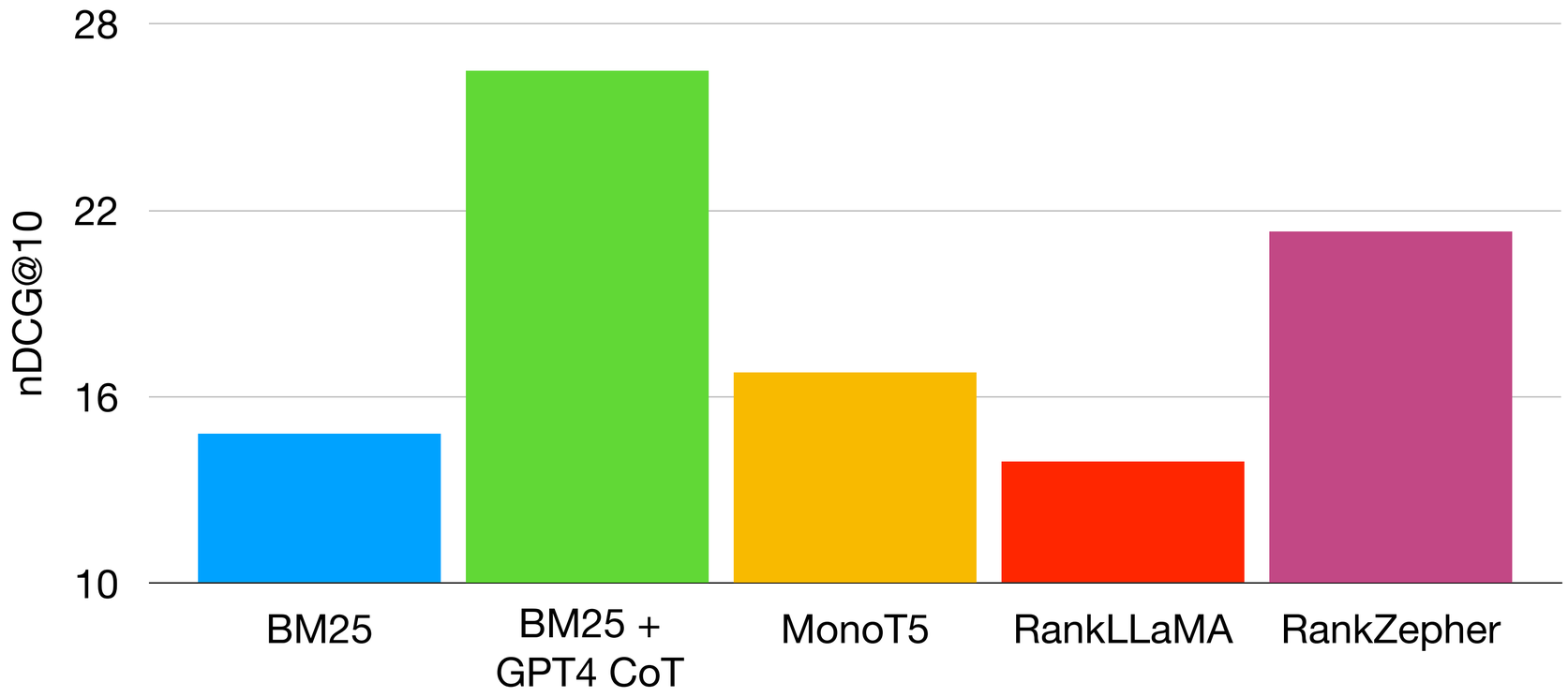
BRIGHT Benchmark

Dataset	Total Number			Avg. Length		Source		Examples
	Q	$\mathcal{D}$	$\mathcal{D}^+$	Q	$\mathcal{D}$	Q	$\mathcal{D}$	
StackExchange								
Biology	103	57,359	3.6	115.2	83.6	StackExchange post	Web pages: article, tutorial, news, blog, report ...	Tab. 20
Earth Science	116	121,249	5.3	109.5	132.6			Tab. 21
Economics	103	50,220	8.0	181.5	120.2			Tab. 22
Psychology	101	52,835	7.3	149.6	118.2			Tab. 23
Robotics	101	61,961	5.5	818.9	121.0			Tab. 24
Stack Overflow	117	107,081	7.0	478.3	704.7			Tab. 25
Sustainable Living	108	60,792	5.6	148.5	107.9			Tab. 26
Coding								
LeetCode	142	413,932	1.8	497.5	482.6	Coding question	Coding Q&Sol	Tab. 27
Pony	112	7,894	22.5	102.6	98.3	Coding question	Syntax Doc	Tab. 28
Theorems								
AoPS	111	188,002	4.7	117.1	250.5	Math Olympiad Q	STEM Q&Sol	Tab. 29
TheoremQA-Q	194	188,002	3.2	93.4	250.5	Theorem-based Q	STEM Q&Sol	Tab. 30
TheoremQA-T	76	23,839	2.0	91.7	354.8	Theorem-based Q	Theorems	Tab. 31

Su, H., et al., 2025. BRIGHT: A REALISTIC AND CHALLENGING BENCHMARK FOR REASONING-INTENSIVE RETRIEVAL. ICLR

Why Reasoning?

BRIGHT Benchmark



Reasoning Rankers

- JudgeRank: Zeroshot pointwise reasoning ranker.

Query Analysis Prompt

You will be presented with a/an *{query name}*.

Your task consists of the following step:

1. Analyze the *{query name}*:
- Carefully read each sentence of the *{query name}*.
- Identify the core problem or question being asked.

Here is the *{query name}*:
{query}

(a). Query Analysis Prompt of JudgeRank

Document Analysis Prompt

You will be presented with a/an *{query name}*, an analysis of the query, and a/an *{doc name}*.

Your task consists of the following steps:
1. Analyze the *{doc name}*:
- Thoroughly examine each sentence of the *{doc name}*.
- List all sentences from the *{doc name}* that *{definition of relevance}* the *{query name}*.
- Briefly explain how each sentence listed *{definition of relevance}* the *{query name}*.

2. Assess overall relevance:
- If the *{doc name}*, particularly the relevant sentences (if applicable), *{definition of relevance}* the *{query name}*, briefly explain why.
- Otherwise, briefly explain why not.

Here is the *{query name}:*
{query}

Here is the analysis of the *{query name}:*
{query analysis}

Here is the *{doc name}:*
{doc}

(b). Document Analysis Prompt of JudgeRank

Judgment Prompt

You will be presented with a/an *{query name}*, an analysis of the *{query name}*, a/an *{doc name}*, and an analysis of the *{doc name}*.

Your task is to assess if the *{doc name}* *{definition of relevance}* the *{query name}* in one word:
- Yes: If the *{doc name}* *{definition of relevance}* the *{query name}*.
- No: Otherwise.

Important: Respond using only one of the following two words without quotation marks: Yes or No.

Here is the *{query name}:*
{query}

Here is the analysis of the *{query name}:*
{query analysis}

Here is the *{doc name}:*
{doc}

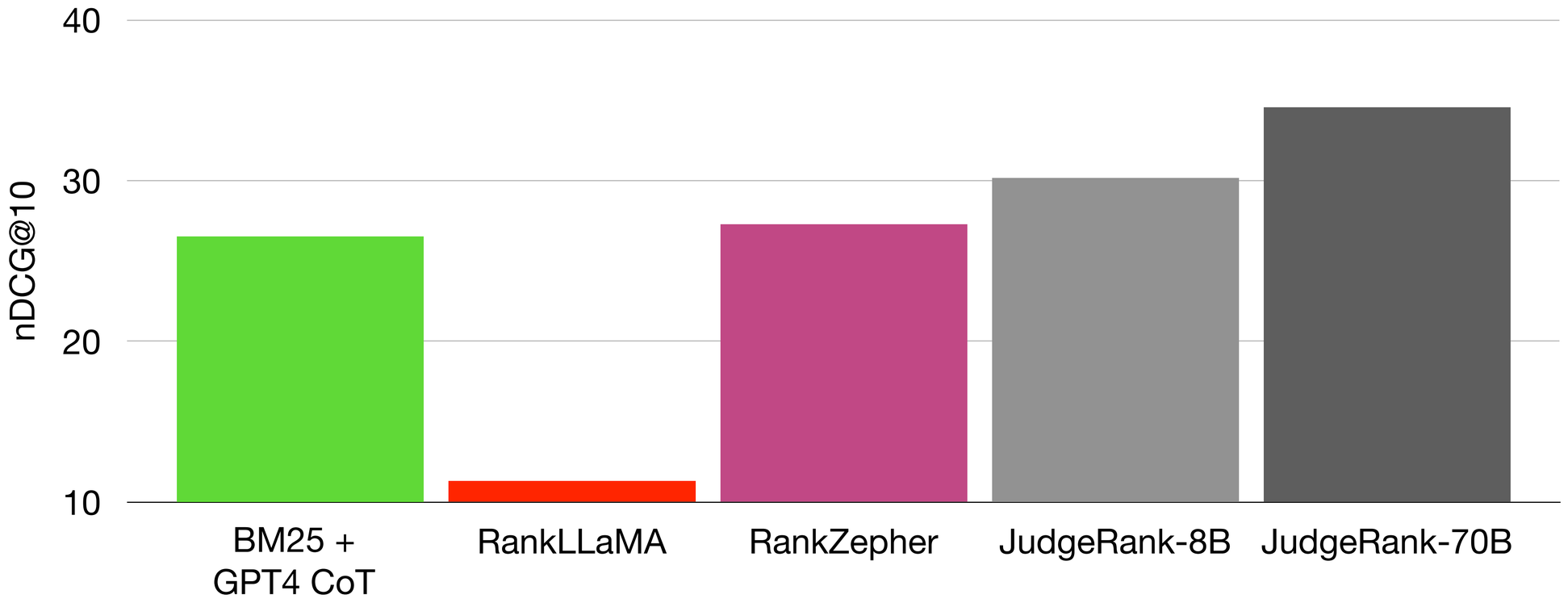
Here is the analysis of the *{doc name}:*
{doc analysis}

(c). Judgment Prompt of JudgeRank

Niu, T., Joty, S., Liu, Y., Xiong, C., Zhou, Y., Yavuz, S. 2024. JudgeRank: Leveraging Large Language Models for Reasoning-Intensive Reranking. arXiv

Reasoning Rankers

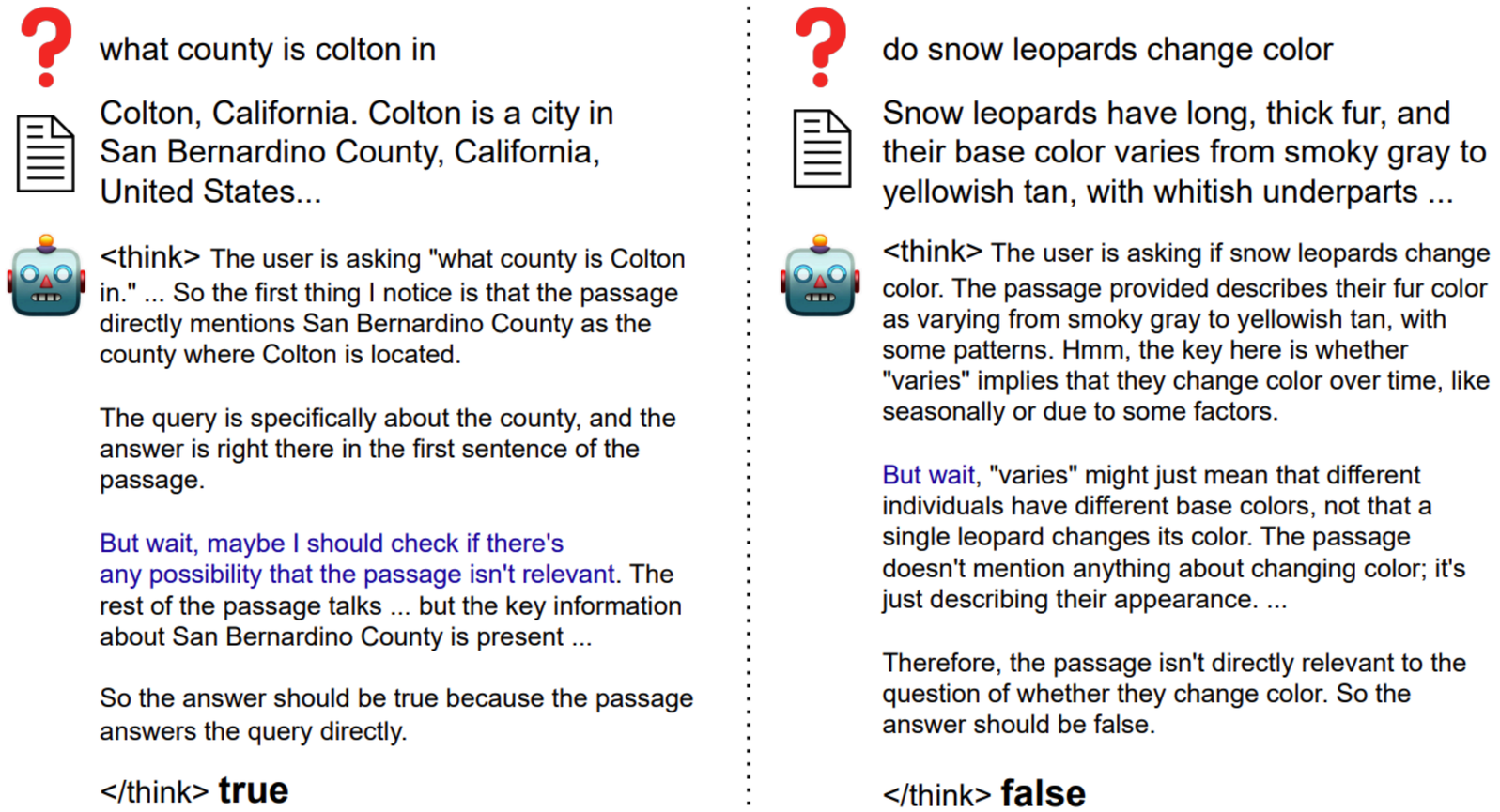
- JudgeRank: BRIGHT results



Niu, T., Joty, S., Liu, Y., Xiong, C., Zhou, Y., Yavuz, S. 2024. JudgeRank: Leveraging Large Language Models for Reasoning-Intensive Reranking. arXiv

Reasoning Rankers

- Rank1: Pointwise reasoning ranker distilled from DeepSeek-R1



Weller, O., Ricci, K., Yang, E., Yates, A., Lawrie, D., Durme, B. 2025, Rank1: Test-Time Compute for Reranking in Information Retrieval. arXiv

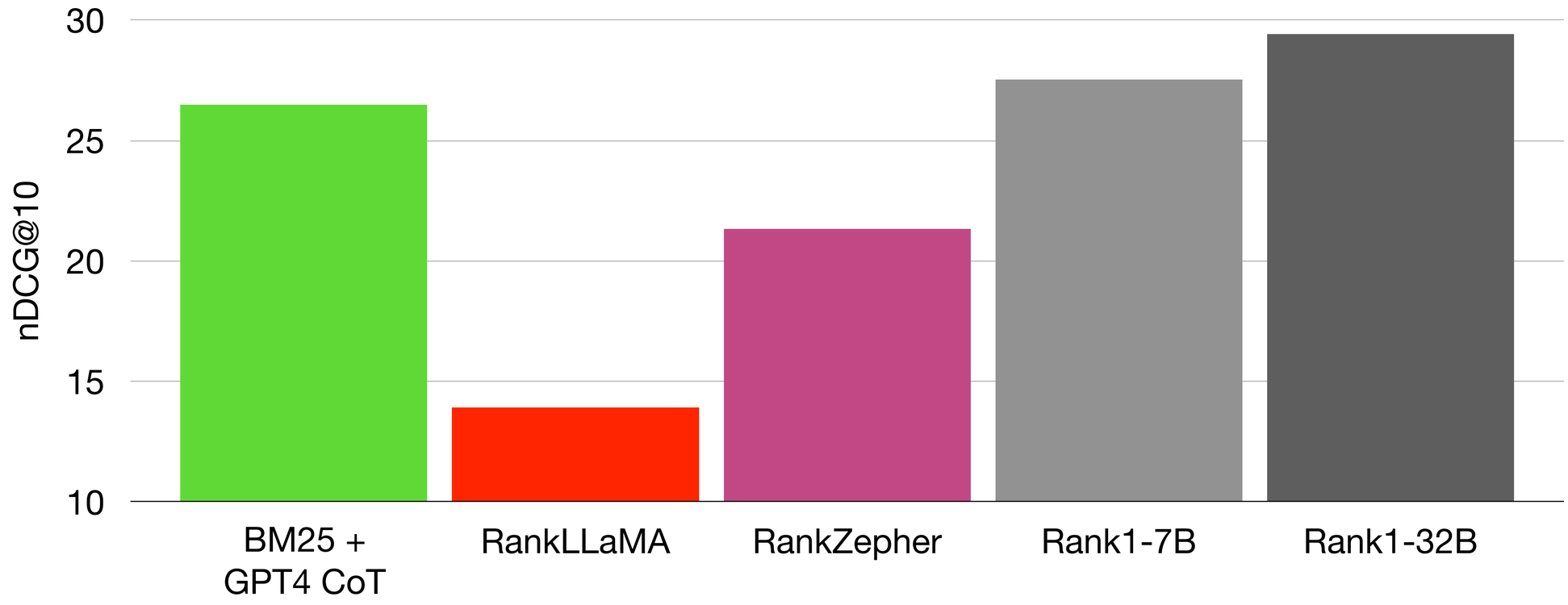
Reasoning Rankers

- Rank1: Pointwise reasoning ranker distilled from DeepSeek-R1
  - Positive and negative data points from MSMARCO.
    - Labelled postive docs, hard/easy negatives from retrievers and rerankers
  - Extensively filtered.
    - R1 disagreed on positives and easy negatives.
    - A model trained on first round to filter out disagreed hard negatives.

Weller, O., Ricci, K., Yang, E., Yates, A., Lawrie, D., Durme, B. 2025, Rank1: Test-Time Compute for Reranking in Information Retrieval. arXiv

Reasoning Rankers

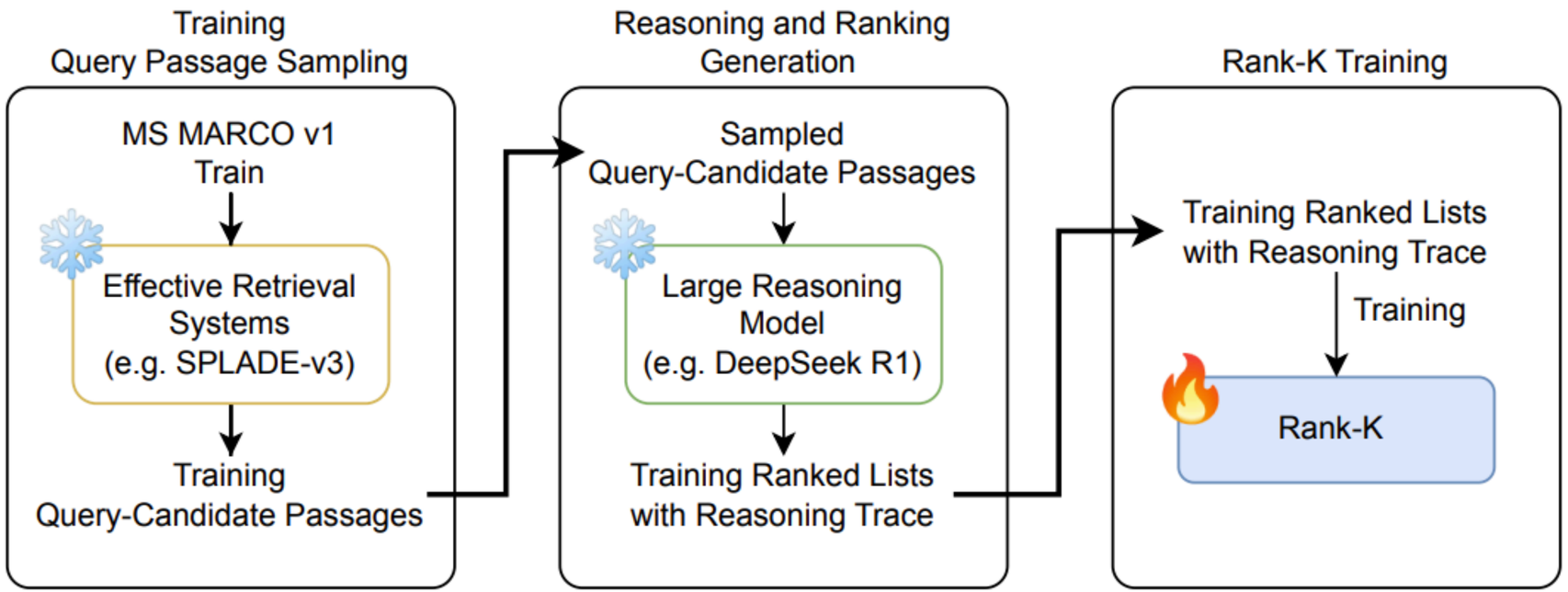
- Rank1: BRIGHT results



Weller, O., Ricci, K., Yang, E., Yates, A., Lawrie, D., Durme, B. 2025, Rank1: Test-Time Compute for Reranking in Information Retrieval. arXiv

Reasoning Rankers

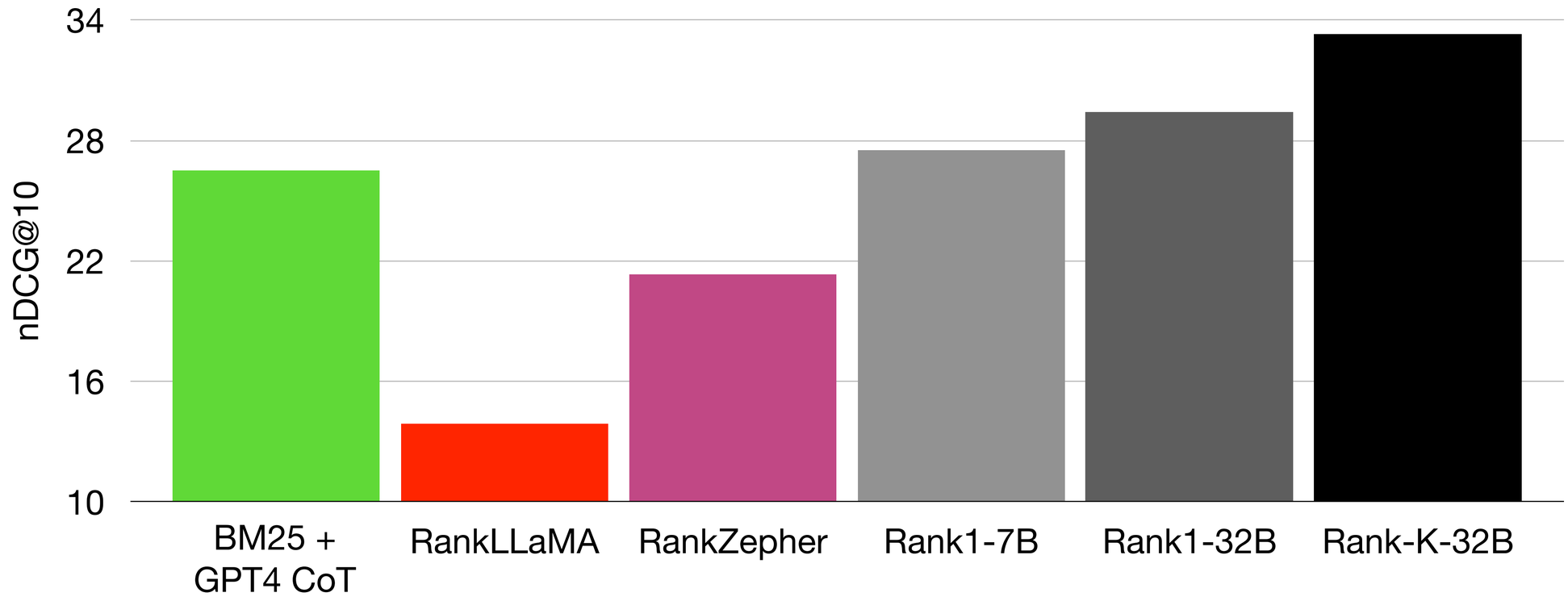
- Rank-K: Listwise reasoning ranker distilled from DeepSeek-R1



Yang, E., Yates, A., Ricci, K., Weller, O., Chari, V., Durme, B., Lawrie, D. 2025, Rank-K: Test-Time Reasoning for Listwise Reranking. arXiv

Reasoning Rankers

- Rank-K: BRIGHT results



Yang, E., Yates, A., Ricci, K., Weller, O., Chari, V., Durme, B., Lawrie, D. 2025, Rank-K: Test-Time Reasoning for Listwise Reranking. arXiv

Reasoning Rankers

- Rank-R1: Reinforcement Learning trained setwise reasoning ranker

Given a query {*query*}, which of the following passages is more relevant one to the query?
Passage A: {*passage_1*}
Passage B: {*passage_2*}
Passage C: {*passage_3*}
…
Output only the passage label of the most relevant passage:

SYSTEM:
A conversation between User and Assistant. The user asks a question, and the Assistant solves it. The assistant first thinks about the reasoning process in the mind and then provides the user with the answer. The reasoning process and answer are enclosed within <think> </think> and <answer> </answer> tags, respectively, i.e., <think> reasoning process here </think> <answer> answer here </answer>.

USER:
Given the query: "{query}", which of the following documents is most relevant?
[1] {document1}
[2] {document2}
....
[20] {document20}
After completing the reasoning process, please provide only the label of the most relevant document to the query, enclosed in square brackets, within the answer tags. For example, if the third document is the most relevant, the answer should be: <think> reasoning process here </think> <answer>[3]</answer>.

Reasoning Rankers

- Rank-R1: Reinforcement Learning trained setwise reasoning ranker

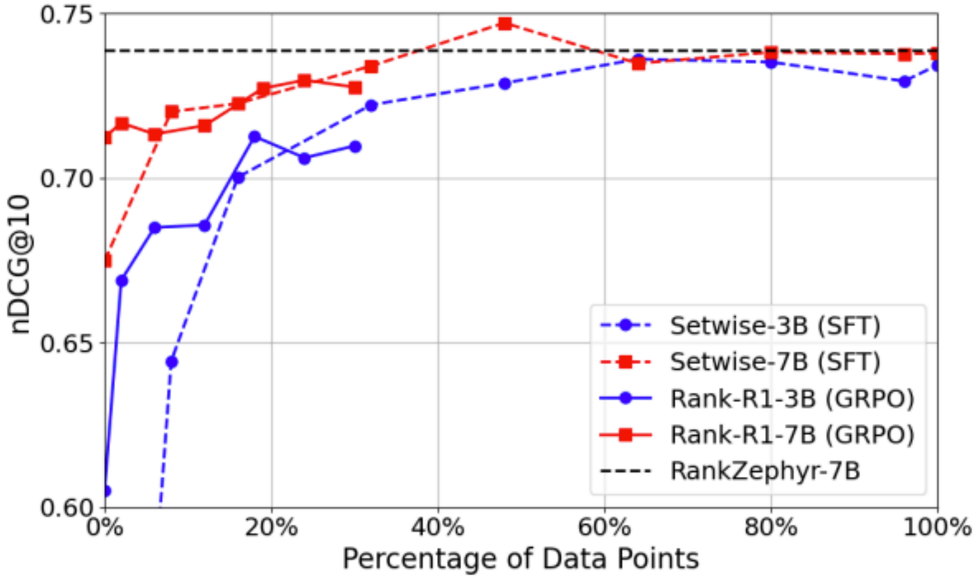
  - GRPO RL algorithm.
  - MSMARCO training data.
  - Reward: 1 if the labelled relevant document is selected and the format matches <think> </think> <answer> </answer>, otherwise 0

SYSTEM:
A conversation between User and Assistant. The user asks a question, and the Assistant solves it. The assistant first thinks about the reasoning process in the mind and then provides the user with the answer. The reasoning process and answer are enclosed within <think> </think> and <answer> </answer> tags, respectively, i.e., <think> reasoning process here </think> <answer> answer here </answer>.

USER:
Given the query: "{query}", which of the following documents is most relevant?
[1] {document1}
[2] {document2}
....
[20] {document20}
After completing the reasoning process, please provide only the label of the most relevant document to the query, enclosed in square brackets, within the answer tags. For example, if the third document is the most relevant, the answer should be: <think> reasoning process here </think> <answer>[3]</answer>.

Reasoning Rankers

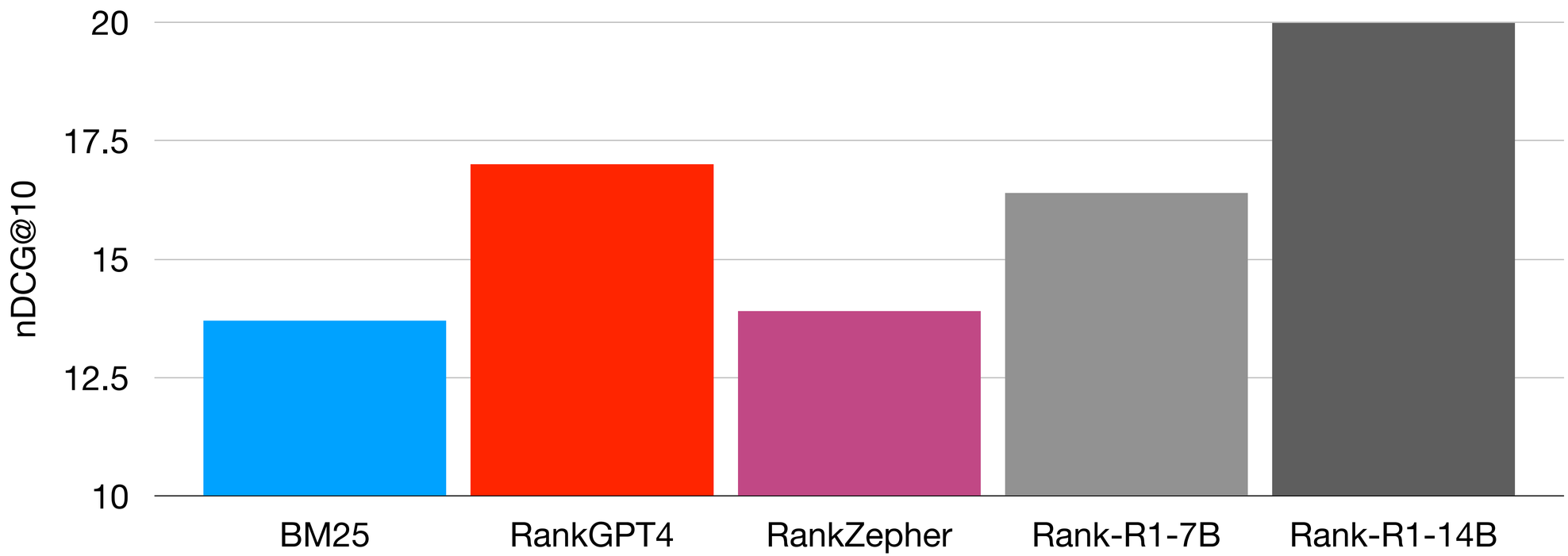
- Rank-R1: Reinforcement Learning trained setwise reasoning ranker



Zhuang, S., Ma, X., Koopman, B., Lin, J., Zuccon, G. 2025. Rank-R1: Enhancing Reasoning in LLM-based Document Rerankers via Reinforcement Learning. arXiv

Reasoning Rankers

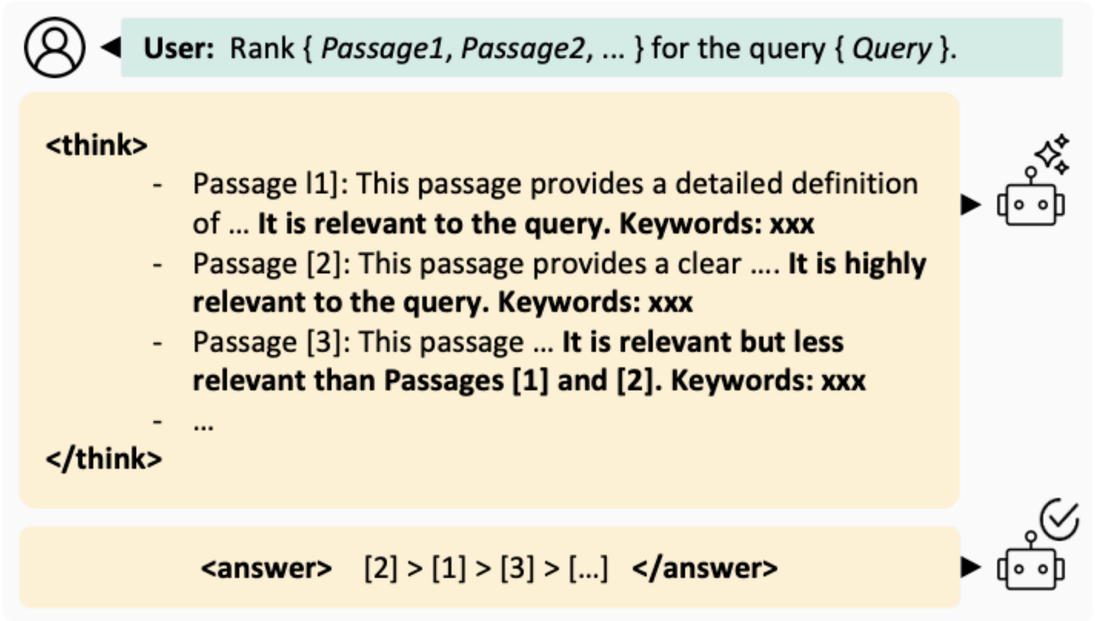
- Rank-R1: BRIGHT results

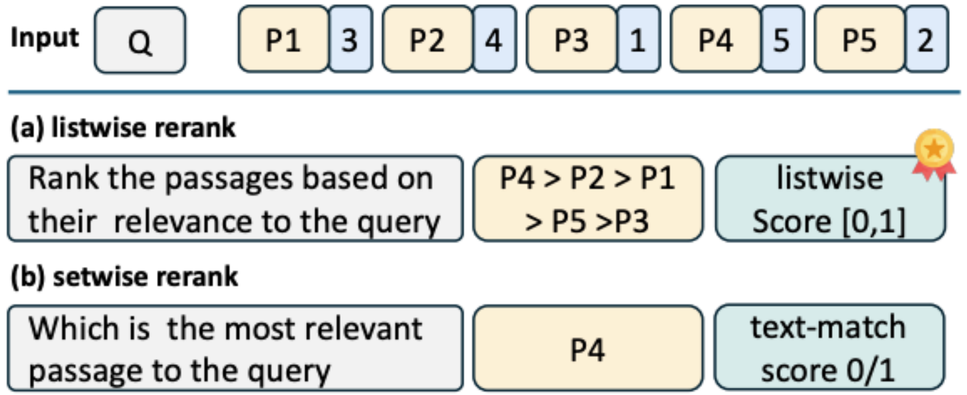


Zhuang, S., Ma, X., Koopman, B., Lin, J., Zuccon, G. 2025. Rank-R1: Enhancing Reasoning in LLM-based Document Rerankers via Reinforcement Learning. arXiv

Reasoning Rankers

- REARANK: Reinforcement Learning trained Listwise reasoning ranker





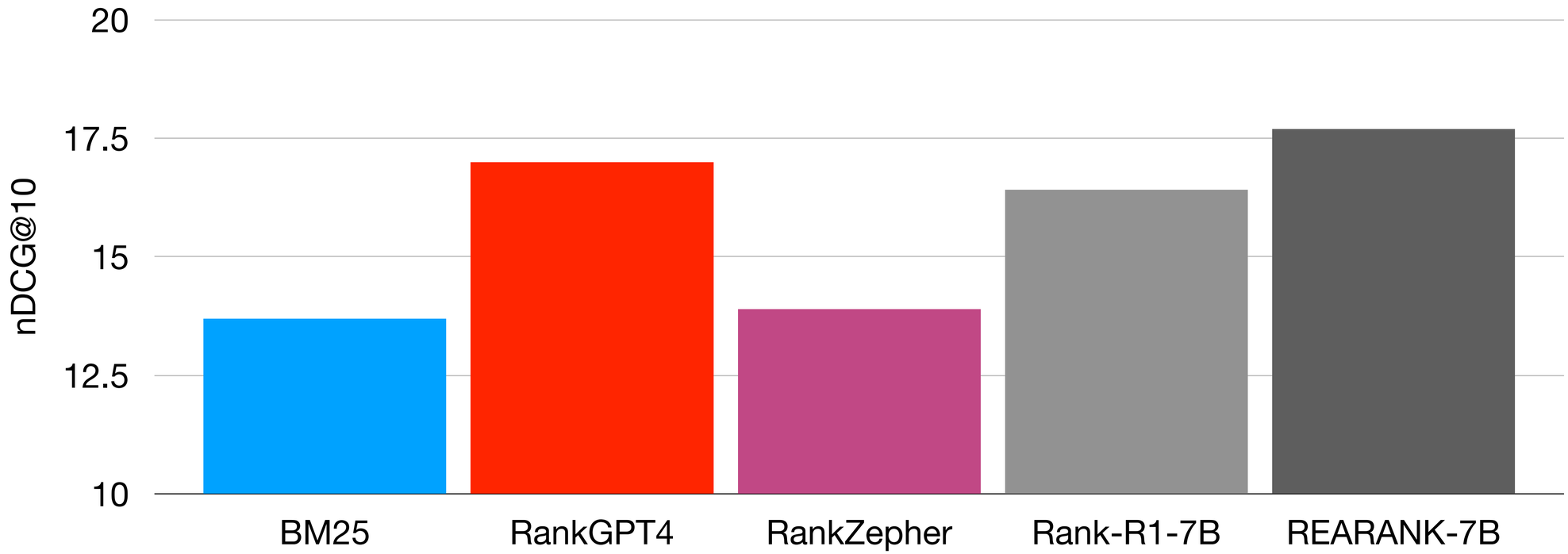
$$r_{\text{rank}} = \frac{r_{\text{rerank}} - r_{\text{init}}}{r^* - r_{\text{init}}},$$

- 179 queries from MSMARCO v2, with fine-grained relevance judgments (0-3)

Zhang, L., Wang, B., Qiu, X., Reddy, S., Agrawal, A. REARANK: Reasoning Re-ranking Agent via Reinforcement Learning. arXiv

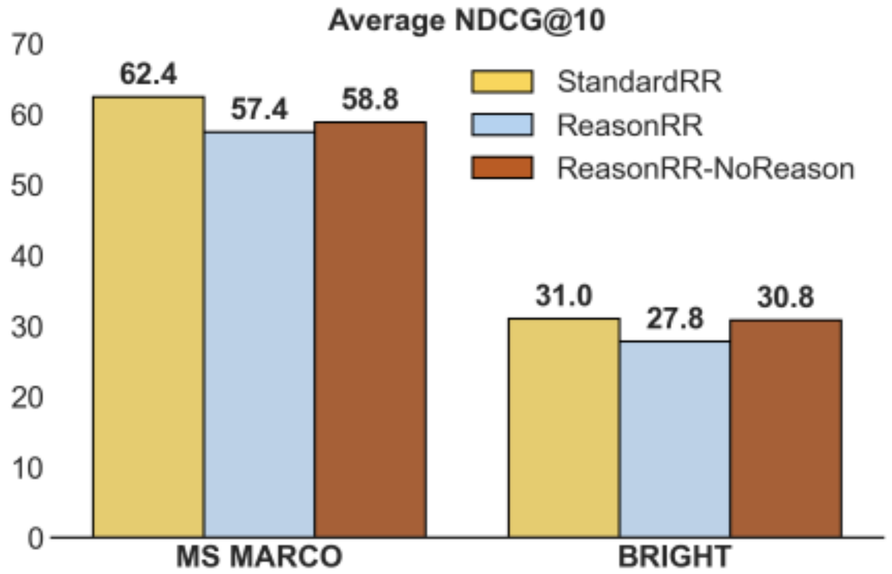
Reasoning Rankers

- REARANK: BRIGHT results



Zhang, L., Wang, B., Qiu, X., Reddy, S., Agrawal, A. REARANK: Reasoning Re-ranking Agent via Reinforcement Learning. arXiv

Is Reasoning Truly Necessary for Ranking?



	StackExchange							Coding		Theorem-based			Avg.
	Bio.	Earth.	Econ.	Psy.	Rob.	Stack.	Sus.	Leet.	Pony	AoPS	TheoQ.	TheoT.	
BM25 + GPT-4 CoT	53.6	54.1	24.3	38.7	18.9	27.7	26.3	19.3	17.6	3.9	19.2	20.8	27.0
+ Qwen2.5-1.5B													
StandardRR	**37.0**	**21.7**	**16.8**	23.1	**16.1**	10.0	**26.3**	2.6	**30.6**	1.8	**16.1**	**26.1**	**19.0**
ReasonRR	32.5	20.3	12.3	**25.5**	11.1	**15.3**	23.5	**6.6**	12.3	**3.4**	10.6	13.7	15.6
+ Qwen2.5-3B													
StandardRR	**41.6**	27.1	**20.9**	31.9	**22.2**	16.9	**30.3**	**13.2**	**42.0**	2.7	16.2	30.6	**24.6**
ReasonRR	37.3	**27.8**	20.7	**33.1**	18.3	**24.3**	25.2	11.3	26.2	**4.7**	**20.7**	**34.0**	23.6
+ Qwen2.5-7B													
StandardRR	**47.1**	**38.0**	**28.1**	**44.1**	**26.1**	**29.5**	**36.5**	**19.3**	**37.5**	4.6	**22.4**	**39.4**	**31.0**
ReasonRR	47.0	35.4	24.0	35.2	20.0	25.2	31.0	15.1	36.0	**5.9**	22.2	36.6	27.8

Jedidi, N., Chuang, Y., Glass, J., Lin, 2025. Jimmy. Don't "Overthink" Passage Reranking: Is Reasoning Truly Necessary?. arXiv

Is Reasoning Truly Necessary for Ranking?

- Seems not is not necessary for simple queries. (Results on MS MARCO and BEIR).
- Seems not necessary when base model is “small”.
- Think about latency.

In this part we covered

- SFT: From monoBERT to RankLLaMA.
- Zero-shot LLM rankers: Pointwise, Pairwise, Listwise, Setwise.
- Reasoning rankers: distillation and RL

End of Part 3

...

Part 4: Challenges, Opportunities and Resources

LLM-Rankers and LLM-Judge (autoraters): two sides of the same coin

Rankers, Judges, and Assistants: Towards Understanding the Interplay of LLMs in Information Retrieval Evaluation

Krisztian Balog
Google DeepMind
Stavanger, Norway
krisztianb@google.com

Donald Metzler
Google DeepMind
Mountain View, USA
metzler@google.com

Zhen Qin
Google DeepMind
Mountain View, USA
zhenqin@google.com

Abstract

Large language models (LLMs) are increasingly integral to information retrieval (IR), powering ranking, evaluation, and AI-assisted content creation. This widespread adoption necessitates a critical examination of potential biases arising from the interplay between these LLM-based components. This paper synthesizes existing research and presents novel experiment designs that explore how LLM-based rankers and assistants influence LLM-based judges. We provide the first empirical evidence of LLM judges exhibiting significant bias towards LLM-based rankers. Furthermore, we observe limitations in LLM judges' ability to discern subtle system performance differences. Contrary to some previous findings, our pre-
liminary
content
of the L
initial g
of LLMs

the potential risks alongside the undeniable benefits. Could this heavy reliance on LLMs across content creation, retrieval, ranking, evaluation, etc., inadvertently introduce or amplify biases within these systems?

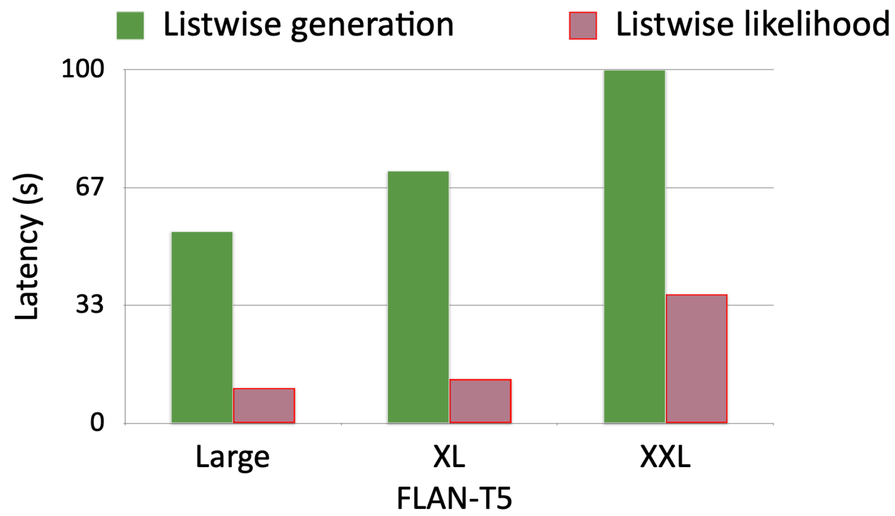
Recent research has begun to explore some of these emerging issues. For example, studies have shown that LLMs can exhibit biases in their output, favoring LLM-generated content over human-generated ones [10], and perpetuating biases present in their training data [15, 21, 33]. Furthermore, LLM-based rating systems have been found to be susceptible to manipulation [2], may not accurately reflect human preferences [28], and demonstrate self-inconsistency [50]. Additionally, the phenomenon of "model
on synthetic
n of quality
s for assess-
ons ranging
from complete rejection of LLMs for relevance assessment [48] to

LLM judges exhibiting significant bias towards LLM rankers

- LLM Rankers: assess the relevance of a doc to a q to score doc
- LLM Judge: assess the relevance of a doc to a q to label the doc for search evaluation
- Also in LLM Judge labelling can be pointwise, pairwise, listwise

Challenges & Opportunities: Latency, Costs, Scalability & Deployability

From Part 3: Rankers Efficiency at Inference



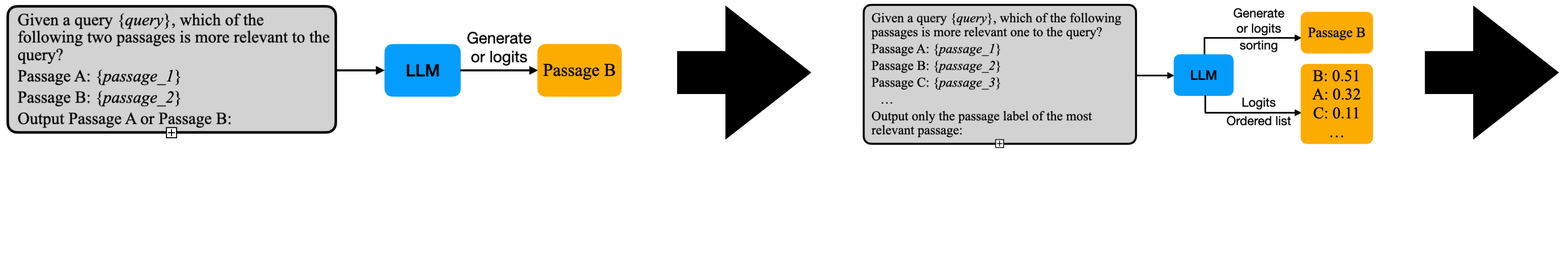
- Impractical latency
- High infrastructure and inference costs

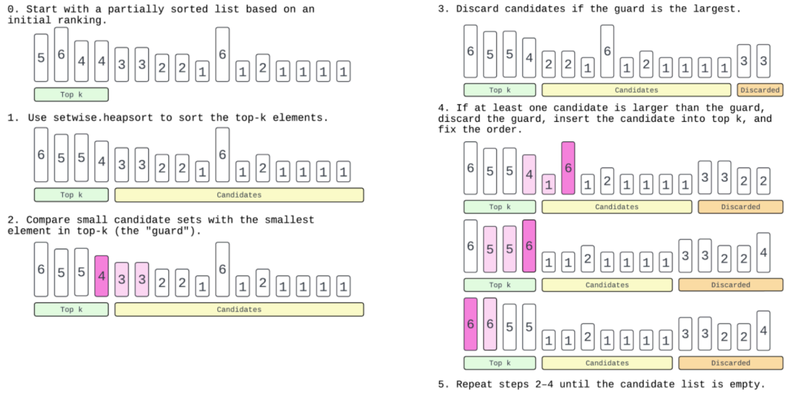
Deepseek R1 Gemini 2.5 Pro Claude 3.7 Sonnet OpenAI o1 Qwen3

Reasoning further exacerbates latency, costs

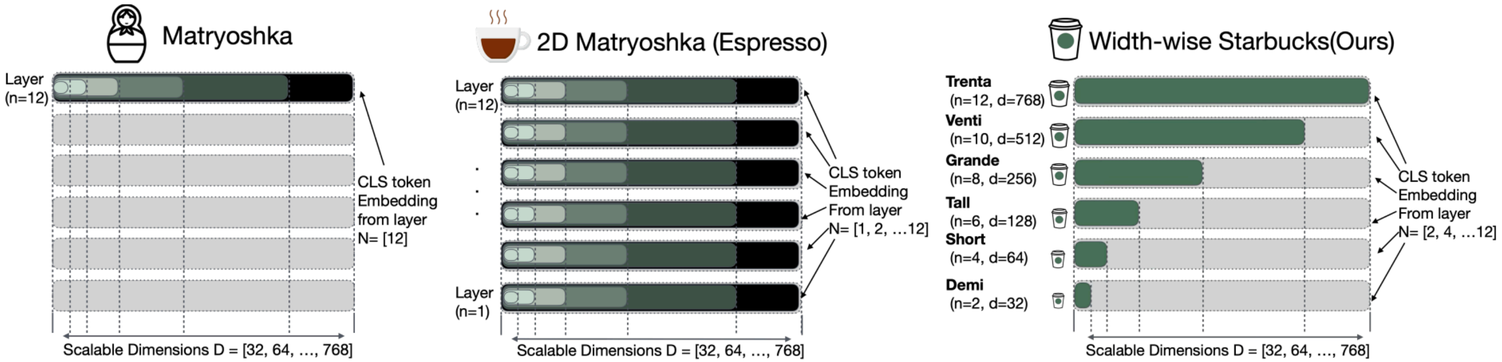
Challenges & Opportunities: Latency, Costs, Scalability & Deployability

Methods Efficiencies

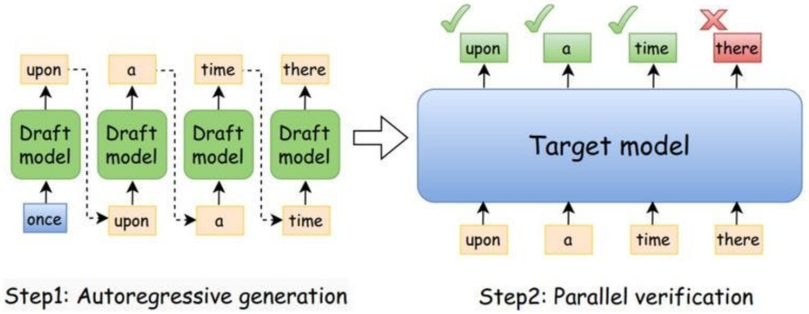




Backbone Efficiencies



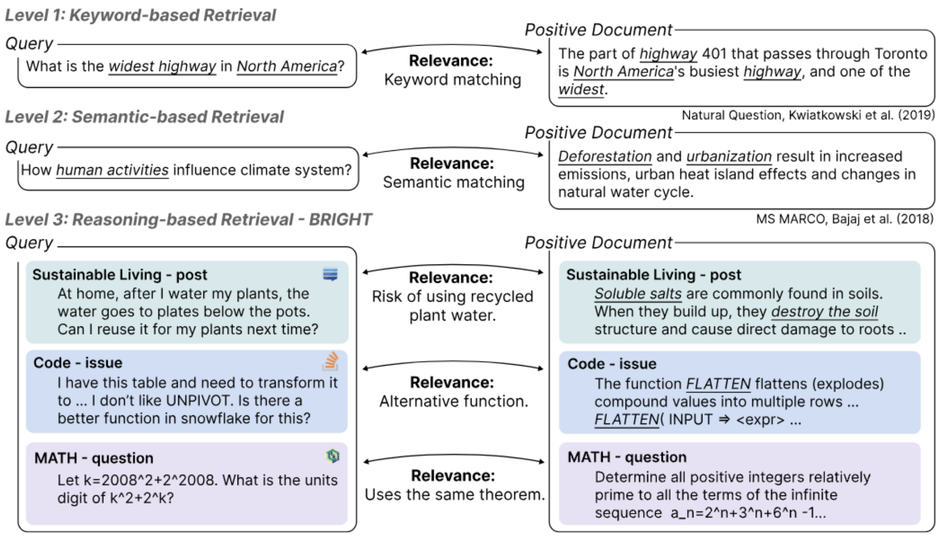
Matryoshka Representation Learning



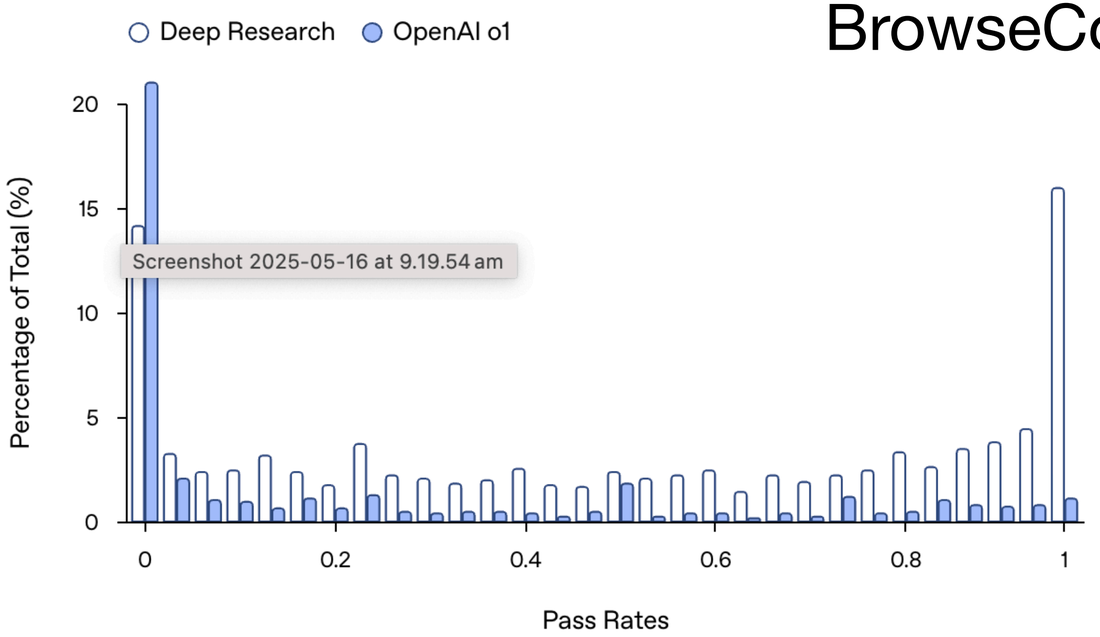
Speculative Decoding

Challenges & Opportunities: Evaluation of more complex retrieval & ranking tasks

BRIGHT



BrowseComp



- Complex, reasoning intensive tasks; but current evaluation resources have limits:
  - Very artificial user requests
  - Noisy labels
  - Lack of reference corpus for retrieval and ranking (BrowseComp

Challenges & Opportunities: Personalise retrieval and ranking

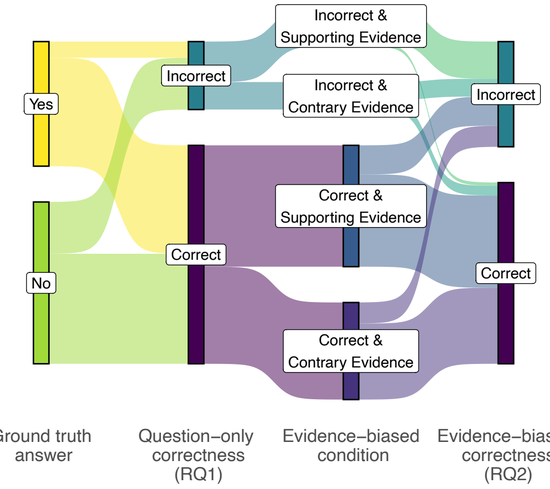
- LLMs prompt instructions offer the possibility to easily and explicitly integrate user preferences on how search works
  - Limited datasets, no retrieval/ranking focus

    Salemi & Zamani, LaMP-QA: A Benchmark for Personalized Long-form Question Answering. arXiv preprint arXiv:2506.00137. 2025

Challenges & Opportunities: robustness to attacks and biases

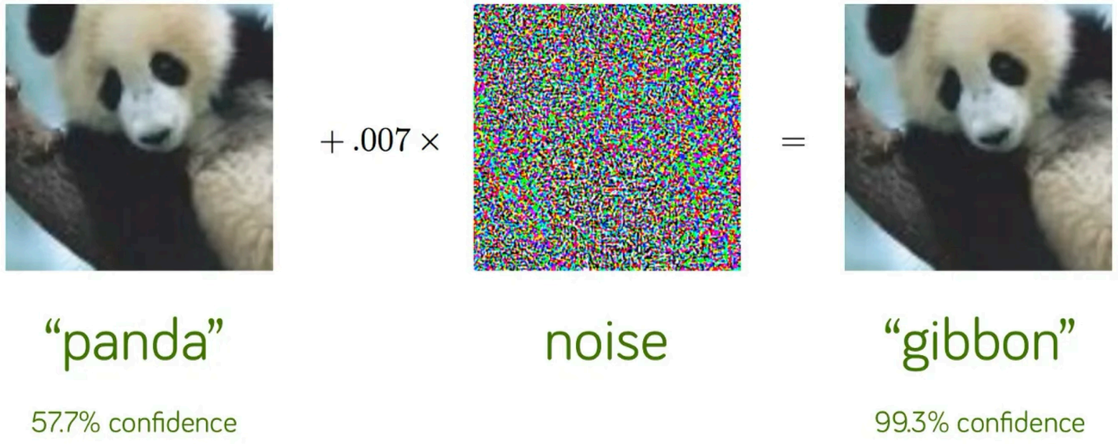
LLMs encode many types of biases, which are hardly identified, controlled

Koopman&Zuccon "Humans are more gullible than LLMs in believing common psychological myths." arXiv 2025



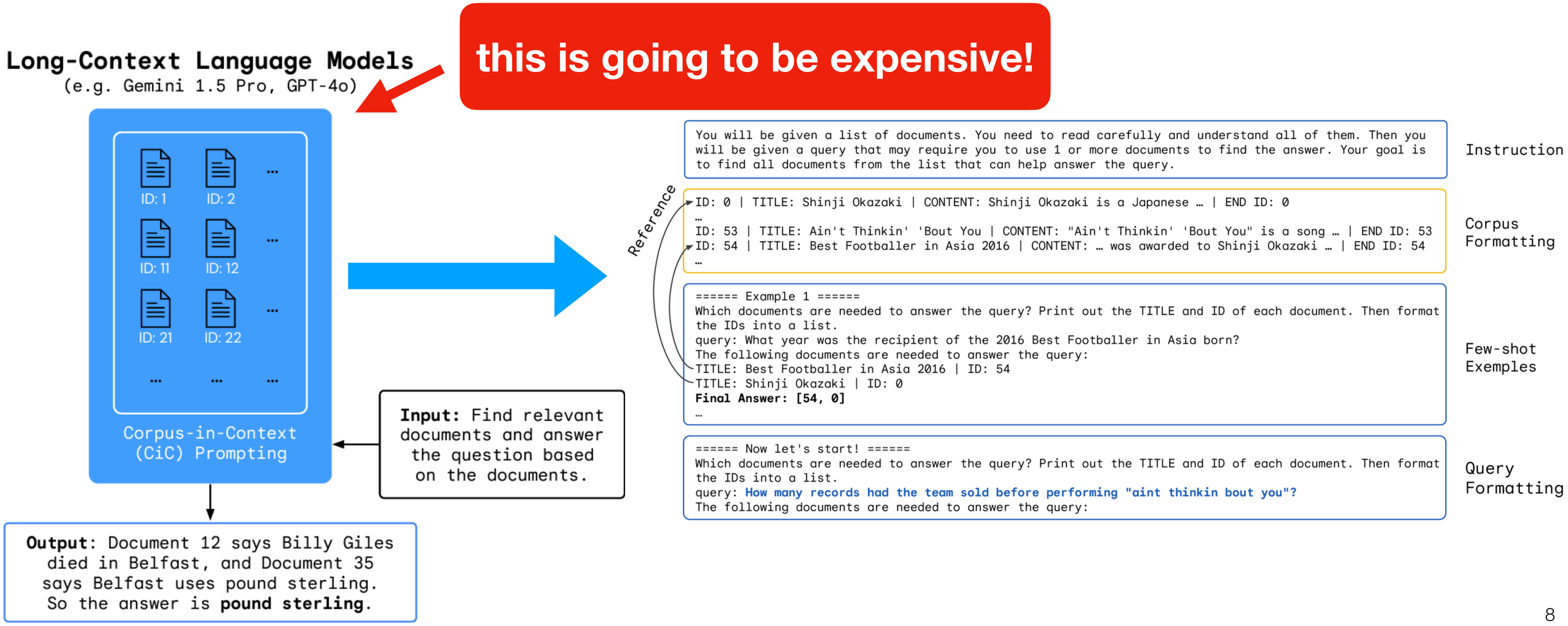
Koopman&Zuccon "Dr ChatGPT tell me what I want to hear: How different prompts impact health answer correctness." EMNLP 2023

Attacks are possible; effects not well understood yet, unclear how to identify

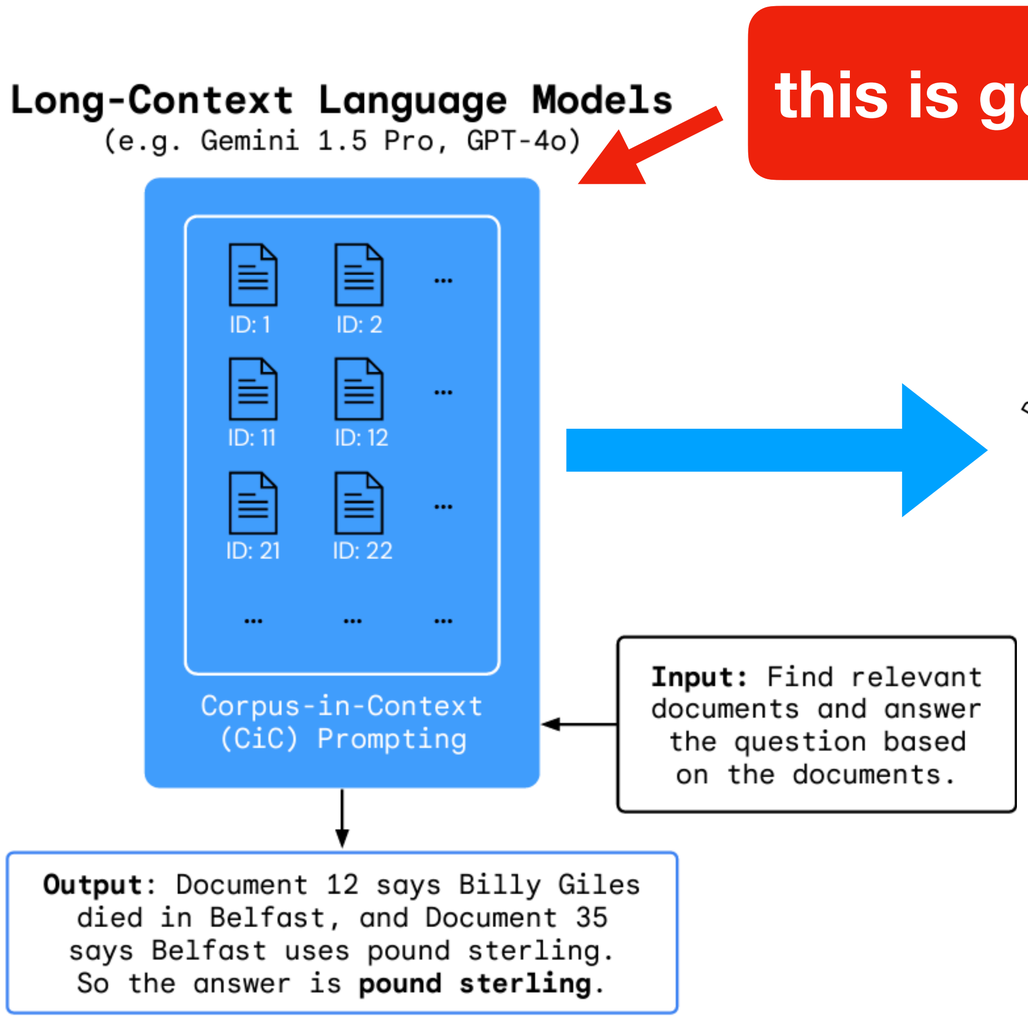


Zhuang,, et al. "Document screenshot retrievers are vulnerable to pixel poisoning attacks." SIGIR 2025

Challenges & Opportunities: LC-LLMs to rule it all



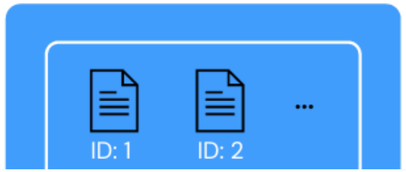
Challenges & Opportunities: LC-LLMs to rule it all



Limitations Our experiments were constrained by the computational resources and financial costs associated with utilizing LCLMs. The entire LOFT 128k test sets contain around 35 datasets $\times$ 100 prompts $\times$ 128k tokens $=$ 448M input tokens, which cost \$1,568 for Gemini 1.5 Pro, \$2,240 for GPT-4o, and \$6,720 for Claude 3 Opus at the time of writing. To reduce costs, we also release dev sets, which are 10x smaller and can be evaluated with around \$200 using Gemini 1.5 Pro or GPT-4o. We also expect LLM API prices to decrease over time. Another limitation of this work is that we focused on evaluating the quality of LCLMs, and leave efficiency considerations for future work. We could not measure the efficiency improvements from prefix caching [20] due to API constraints at the time of writing. Without caching, the Gemini 1.5 Pro API has a median latency of roughly four seconds for 32k input tokens, twelve seconds for 128k input tokens, and 100 seconds for 1 million input tokens. This speed is likely slower than specialized retrievers or SQL databases; the promising quality results on LOFT encourage further investigation into optimizing LCLMs efficiency. Additionally, our retrieval and RAG tasks was limited to 1 million tokens, which still leaves a large gap from real-world applications that may involve several million or even billions of documents.

Challenges & Opportunities: LC-LLMs to rule it all

Long-Context Language Models
(e.g. Gemini 1.5 Pro, GPT-4o)



this is going to be expensive!

But…

You will be given a list of documents. You need to read carefully and understand all of them. Then you will be given a query that may require you to use 1 or more documents to find the answer. Your goal is to find all documents from the list that can help answer the query.

Instruction

Reference

ID: 0 | TITLE: Shinji Okazaki | CONTENT: Shinji Okazaki is a Japanese … | END ID: 0
…
ID: 53 | TITLE: Ain't Thinkin' 'Bout You | CONTENT: "Ain't Thinkin' 'Bout You" is a song … | END ID: 53
ID: 54 | TITLE: Best Footballer in Asia 2016 | CONTENT: … was awarded to Shinji Okazaki … | END ID: 54
…

Corpus Formatting

====== Example 1 ======
Which documents are needed to answer the query? Print out the TITLE and ID of each document. Then format the IDs into a list.
query: What year was the recipient of the 2016 Best Footballer in Asia born?
The following documents are needed to answer the query:
TITLE: Best Footballer in Asia 2016 | ID: 54
TITLE: Shinji Okazaki | ID: 0
Final Answer: [54, 0]
…

Few-shot Exemples

====== Now let's start! ======
Which documents are needed to answer the query? Print out the TITLE and ID of each document. Then format the IDs into a list.
query: **How many records had the team sold before performing "aint thinkin bout you"?**
The following documents are needed to answer the query:

Query Formatting

You will be given a list of documents. You need to read carefully and understand all of them. Then you will be given a query that may require you to use 1 or more documents to find the answer. Your goal is

Instruction

Limitations Our experiments were constrained by the computational resources and financial costs associated with utilizing LCLMs. The entire LOFT 128k test sets contain around 35 datasets × ns = 448M input tokens, which cost $1,568 for Gemini 1.5 Pro, $2,240 for Claude 3 Opus at the time of writing. To reduce costs, we also release dev er and can be evaluated with around $200 using Gemini 1.5 Pro or GPT-4o. PI prices to decrease over time. Another limitation of this work is that we e quality of LCLMs, and leave efficiency considerations for future work. e efficiency improvements from prefix caching [20] due to API constraints Without caching, the Gemini 1.5 Pro API has a median latency of roughly ut tokens, twelve seconds for 128k input tokens, and 100 seconds for 1 is speed is likely slower than specialized retrievers or SQL databases; the on LOFT encourage further investigation into optimizing LCLMs efficiency. l and RAG tasks was limited to 1 million tokens, which still leaves a large ications that may involve several million or even billions of documents.

Likely to improve further in future

Challenges & Opportunities: The Role of Reasoning

- Initial steps to do reasoning for ranking — but the contribution to effectiveness is still unclear
    - Likely due to lack of adequate datasets to pick up these signals
- How to do reasoning for retrieval?
    - ReasonIR does not do architectural changes to integrate reasoning in DRs

Where from here? Pointers to resources

Large Language Models for Information Retrieval: A Survey

Yutao Zhu, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng
Haonan Chen, Zheng Liu, Zhicheng Dou, and Ji-Rong Wen

Abstract—As a primary means of information acquisition, information retrieval (IR) systems, such as search engines, have integrated themselves into our daily lives. These systems also serve as components of dialogue, question-answering, and recommender systems. The trajectory of IR has evolved dynamically from its origins in term-based methods to its integration with advanced neural models. While the neural models excel at capturing complex contextual signals and semantic nuances, thereby reshaping the IR landscape, they still face challenges such as data scarcity, interpretability, and the generation of contextually plausible yet potentially inaccurate responses. This evolution requires a combination of both traditional methods (such as term-based sparse retrieval methods with rapid response) and modern neural architectures (such as language models with powerful language understanding capacity). Meanwhile, the emergence of large language models (LLMs), typified by ChatGPT and GPT-4, has revolutionized natural language processing due to their remarkable language understanding, generation, generalization, and reasoning abilities. Consequently, recent research has sought to leverage LLMs to improve IR systems. Given the rapid evolution of this research trajectory, it is necessary to consolidate existing methodologies and provide nuanced insights through a comprehensive overview. In this survey, we delve into the confluence of LLMs and IR systems, including crucial aspects such as query rewriters, retrievers, rerankers, and readers. Additionally, we explore promising directions, such as search agents, within this expanding field.

4 Sep 2024

Where from here? Pointers to resources

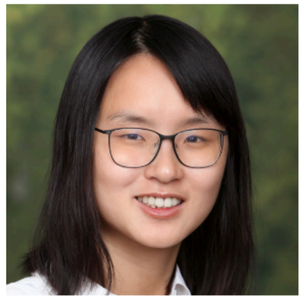
Large Language Models for Information

Yutao Zhu,

ACL 2023 Tutorial:

Retrieval-based Language Models and Applications



Akari Asai[1], Sewon Min[1], Zexuan Zhong[2], Danqi Chen[2]

[1]University of Washington, [2]Princeton University

Abstract—As a primar
themselves into our dai
The trajectory of IR ha
While the neural mode
they still face challenge
responses. This evoluti
response) and modern
the emergence of large
to their remarkable lan
sought to leverage LLM
existing methodologies
of LLMs and IR system
promising directions, su

4 Sep 2024

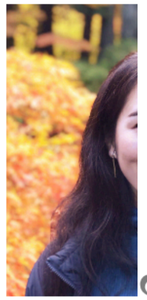
Where from here? Pointers to resources

Large Language Models for Information

ACL 2023 Tutorial:

Yutao Zhu, **Retrieval-based Language Models and Applications**

Abstract—As a primar
themselves into our dai
The trajectory of IR ha
While the neural mode
they still face challenge
responses. This evoluti
response) and modern
the emergence of large
to their remarkable lan
sought to leverage LLM
existing methodologies
of LLMs and IR system
promising directions, su



Akar

Information Retrieval Meets Large Language Models: A Strategic Report from Chinese IR Community

Qingyao AI[a], Ting BAI[b], Zhao CAO[c], Yi CHANG[d], Jiawei CHEN(✉)[e], Zhumin CHEN[f], Zhiyong CHENG[g], Shoubin DONG[h], Zhicheng DOU[i], Fuli FENG[j], Shen GAO[f], Jiafeng GUO[k], Xiangnan HE(✉)[j], Yanyan LAN[a], Chenliang LI[l], Yiqun LIU[a], Ziyu LYU[m], Weizhi MA[a], Jun MA[f], Zhaochun REN[f], Pengjie REN[f], Zhiqiang WANG[n], Mingwen WANG[o], Ji-Rong WEN[i], Le WU[p], Xin XIN[f], Jun XU[i], Dawei YIN[q], Peng ZHANG(✉)[r], Fan ZHANG[j], Weinan ZHANG[s], Min ZHANG[a], Xiaofei ZHU[t]

[a]*Tsinghua University*, [b]*Beijing University of Posts and Telecommunications*, [c]*Huawei Technologies Ltd. Co*, [d]*Jilin University*, [e]*Zhejiang University*, [f]*Shandong University*, [g]*Shandong Artificial Intelligence Institute*, [h]*South China University of Technology*, [i]*Renmin University of China*, [j]*University of Science and Technology of China*, [k]*Institute of Computing Technology, Chinese Academy of Sciences*, [l]*Wuhan University*, [m]*Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences*, [n]*Shanxi University*, [o]*Jiangxi Normal University*, [p]*Hefei University of Technology*, [q]*Baidu Inc.*, [r]*Tianjin University*, [s]*Shanghai Jiao Tong University*, [t]*Chongqing University of Technology*

Tools for research on LLM-retrievers

- **FlagEmbedding**: The BGE series, RAG-Retrieval: The Stella-embedding series
  https://github.com/FlagOpen/FlagEmbedding

- **Tevatron**: LLM/VLLM retriever and reranker finetuning
  https://github.com/texttron/tevatron

- **PyLate**: ColBERT style mutli-vector retriever training and inference
  https://github.com/lightonai/pylate

- **SentenceTransformer**: Support Most open source embedding and rerank model for inference and training
  https://github.com/UKPLab/sentence-transformers

- **Search-R1, Verl-Tool**: RL with Search
  https://github.com/PeterGriffinJin/Search-R1

Tools for research on LLM-rankers

- LLM-rankers: https://github.com/ielab/llm-rankers
  - Reference implementations of Zero-shot Pointwise, Listwise, Pairwise, Setwise, Rank-R1
- RankLLM: https://github.com/castorini/rank_llm
  - Python toolkit for rerankers, with a focus on listwise reranking; includes reference implementations of BERT backbones methods
- LLM4Ranking: https://github.com/liuqi6777/llm4ranking
  - Similar to LLM-rankers: Reference implementations of LLM-rankers, supports open-source or closed-source API-based LLMs, includes fine-tuning scripts

It’s time for Questions / Comments?