ChainRank-DPO: Chain Rank Direct Preference Optimization for LLM Rankers

Anonymous ACL submission

Abstract

Large language models (LLMs) have demonstrated remarkable effectiveness in text reranking through works like RankGPT, leveraging their human-like reasoning about relevance. However, supervised fine-tuning for ranking often diminishes these models' general-purpose capabilities, including the crucial reasoning abilities that make them valuable for ranking. We introduce a novel approach integrating Chain-of-Thought prompting with an SFT-DPO (Supervised Fine-Tuning followed by Direct Preference Optimization) pipeline to preserve these capabilities while improving ranking performance. Our experiments on TREC 2019 and 2020 Deep Learning datasets show that our approach outperforms the state-of-the-art RankZephyr while maintaining strong performance on the Massive Multitask Language Understanding (MMLU) benchmark, demonstrating effective preservation of general-purpose capabilities through thoughtful fine-tuning strategies. Our code and data will be publicly released upon the acceptance of the paper.

1 Introduction

Text reranking is a vital task in information retrieval (Liu, 2009; Hasanain, 2018), crucial for search engines (Li et al., 2024), conversational AI (Becker et al., 2012), and recommendation systems (Zhao and Liu, 2024). Large language models (LLMs) excel in reranking due to their reasoning and human-like thinking capabilities, enabling them to handle complex queries and ambiguous contexts. RankGPT (Achiam et al., 2023) has set a high standard in listwise reranking (Ma et al., 2019), leveraging its interpretability and systematic reasoning (Brown et al., 2020) to advance reranking technologies. Building on the success of RankGPT, several models have sought to distill its output into smaller, task-optimized models through supervised fine-tuning (SFT). RankVicuna (Pradeep et al., 2023a), for instance, uses distilled data generated exclusively from GPT-3.5, focusing on efficient fine-tuning to improve ranking performance while maintaining a lightweight architecture. Moreover, RankZephyr (Pradeep et al., 2023b) employs a more comprehensive approach utilizing distilled data from GPT-3.5 and GPT-4, ensuring the quality and reliability of training data to achieve superior ranking results. Although these models achieve state-of-the-art performance in specific benchmarks, they highlight a key challenge in ranking: the trade-off between optimizing task-specific performance and preserving the broader reasoning capabilities of LLMs. Addressing this challenge requires novel strategies to balance specialization with generalization, ensuring that ranking models retain their versatility across diverse tasks.

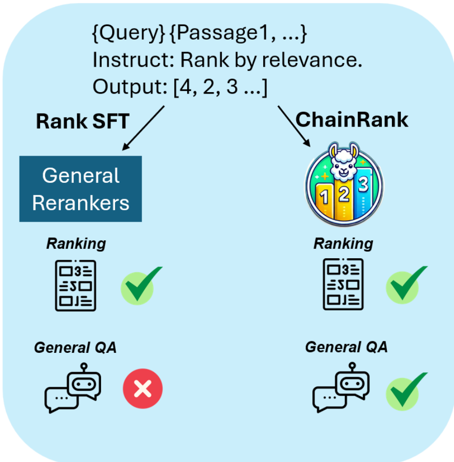

Figure 1: While both methods can rerank passages, general methods struggle with simple QA tasks, whereas ChainRank successfully solves them step by step.

Our investigations in Figure 1 reveal that while RankVicuna and RankZephyr exhibit strong performance in listwise ranking tasks, they have forfeited their mathematical reasoning capabilities following fine-tuning. This is evident from the figure, where the models fail to produce coherent solutions to mathematical problems, often outputting unrelated or nonsensical numerical sequences. Moreover, further analysis uncovered that these models have also lost their general text-generating capabilities. Regardless of the input prompt, the models struggle to produce meaningful or contextually appropriate responses, highlighting a significant trade-off introduced by their specialized fine-tuning for ranking tasks. Furthermore, while they exhibit comprehension of listwise ranking, they encountered difficulties with pairwise ranking, as their pairwise outcomes did not correspond with those generated by the listwise ranking (Peng et al., 2024).

To address this limitation, we propose ChainRank strategy, built on LLaMA3-8b-instruct (Dubey et al., 2024). Using the same training data as RankZephyr and RankGPT$_{3.5}$/RankGPT$_4$ as teacher models, we introduced a chain-of-thought (CoT) prompt to guide sequential passage ranking. We then implemented ChainRank-DPO, enhancing the model's reasoning abilities for superior ranking performance. While traditional Direct Preference Optimization (DPO) (Rafailov et al., 2024) performs well on chat benchmarks, it struggles with long-chain reasoning tasks like math and ranking due to error propagation. Inspired by Step-DPO (Lai et al., 2024), we designed a novel DPO framework using overlapping ranking orders as the reward function, allowing better error correction. ChainRank preserves general text generation and reasoning capabilities, overcoming limitations in RankVicuna and RankZephyr. It outperforms both on TREC 2019/2020 Deep Learning Tracks (Craswell et al., 2021, 2020) and BEIR benchmarks (Thakur et al., 2021). We summarize the contributions of this work as follows:

- We propose a novel chain-of-thoughts instruction (reranking) tuning that enables LLMs to rank passages based on relevance step by step.

- To our best knowledge, this is the first work to introduce the Supervised Fine-Tune–DPO pipeline in the context of text reranking.

- Using the same training data, our model outperforms both RankZephyr and the teacher model, RankGPT$_4$, while retaining the general capabilities of LLMs.

USER: I will provide you with {num} passages, each indicated by a numerical identifier []. Rank the passages based on their relevance to the search query: {query}.
[1] {passage 1}
[2] {passage 2}
...
[{num}] {passage {num}}
Search Query: {query}.
Rank the {num} passages by selecting the most relevant passage at each step from the remaining passages. After choosing the most relevant passage, remove it from the pool and continue ranking until all passages are ordered.
Instructions:
Start with the most relevant passage and select it from the full list.
For each following step, pick the most relevant passage from the remaining passages only.
List the selected passages by their identifiers at each step, one after the other, until all passages are ranked.
Example Output:
Step 1: [4]
Step 2: [4, 2]
Step 3: [4, 2, 3]
...
step {num}: [4,2,3,15,...,14]
Final Answer: [4, 2, 3,..., 14]
Only respond with each step and the final answer, ensuring each passage is included once and ranked in descending relevance.

Figure 2: ChainRank Chain-of-Thought (CoT) reranking prompt guiding the model to rank passages based on relevance to a query iteratively. The prompt ensures step-by-step selection, removal, and ordering of passages, with an example illustrating the expected output format.

2 Methodology

2.1 CoT Reranking Prompt Design

Recent advancements in listwise text reranking, such as RankVicuna (Pradeep et al., 2023a), RankZephyr (Pradeep et al., 2023b), and related methods (Liu et al., 2024), build on zero-shot techniques (Ma et al., 2023; Sun et al., 2023), using prompt templates to reorder documents for metrics like nDCG (Järvelin and Kekäläinen, 2002). Given a query q and documents $d_1, d_2, ..., d_n$, the goal is to return a reordered list (e.g., $[1] > [2] > [3] ... > [20]$).

Our ChainRank strategy frames listwise reranking as a chain-of-thought (CoT) reasoning task, selecting the most relevant document iteratively until all are ranked. The CoT prompt for LLaMA is shown in Figure 2.

2.2 Training Dataset

We train ChainRank model using 35k GPT-3.5 and 5k GPT-4 labeled instances from (Pradeep

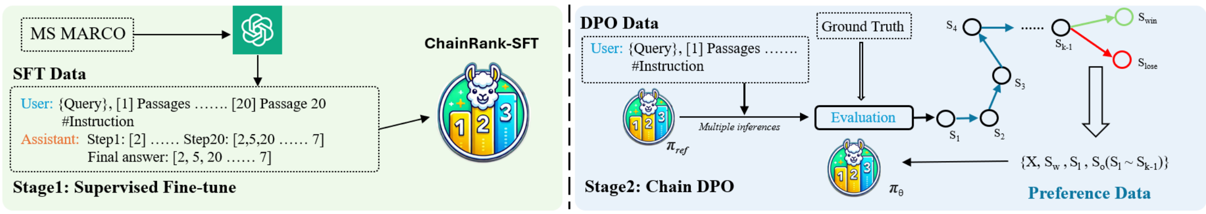

Figure 3: Training framework of our model. *Left* : Stage 1 aims to teach the student model to perform text reranking. *Right* : In stage 2, we use the same prompt format to generate multiple answers, then pick the chosen and rejected answers based on the number of overlapped steps, finally, we utilize the preference data to perform Chain DPO training.

et al., 2023b), derived from randomly selected MS MARCO v1 queries. Pyserini (Lin et al., 2021) retrieved 20 BM25 candidates per query, which $\text{RankGPT}_{3.5}$ and RankGPT_4 ordered as teacher models. Malformed generations, such as missing or duplicate identifiers, were removed to improve data quality. We randomly selected 90% of the query-document pairs for Stage 1 SFT training and 10% for Stage 2 DPO training, denoted as SFT data and DPO data.

2.3 Training Stage 1: Supervised Fine-Tune

As shown in Figure 3, during the Supervised Fine-Tuning (SFT) phase, we use SFT data, maintaining a zero-shot setup since RankGPT and ChainRank do not rely on human-labeled data.

Before fine-tuning, the original LLaMA3-8B-Instruct model (Dubey et al., 2024) failed to produce meaningful CoT reranking, simply replicating the example format without relevance-based results, highlighting the need for fine-tuning. The model weights are publicly available on HuggingFace.[1]

We fully fine-tune the 8B parameter LLaMA-3 model for three epochs with a batch size of 128, a 5×10^{-6} learning rate, and bfloat16 format. Training on four NVIDIA A100 80GB GPUs took approximately 39 hours.

2.4 Training Stage 2: Chain DPO

After Stage 1, the ChainRank-SFT model generates three ranking predictions (y) on prompts (x) from DPO training data. Predictions are evaluated by overlapping ranking orders with ground-truth labels, creating a preference dataset with prompts (x), chosen steps (s_w), rejected steps (s_l), and overlapping steps (s_o). Unlike prior methods, the final ranking y comprises a sequence of reasoning steps, $y = s_1, s_2, \ldots, s_n$, where each step is conditioned on prior steps $\pi(s_k|x; s_{1:k-1})$. Overlapping steps (s_o) are tracked until a divergence is found, after which only the initial contiguous overlaps are included in s_o, and later steps are categorized as chosen (s_w) or rejected (s_l). This results in a dataset of (x, s_w, s_l, s_o).

The objective maximizes the likelihood of correct steps (s_w) while minimizing incorrect ones (s_l), using the loss:

$$\mathcal{L}(\theta) = -\mathbb{E}_{(x,s_w,s_l,s_o)\sim D}\Big[\log\sigma\Big(\beta\log\frac{\pi_\theta(s_w \mid x; s_o)}{\pi_{ref}(s_w \mid x; s_o)} - \beta\log\frac{\pi_\theta(s_l \mid x; s_o)}{\pi_{ref}(s_l \mid x; s_o)}\Big)\Big],$$

where $\pi_\theta(s \mid x; s_o)$ represents the model's probabilities, optimized to favor correct steps while leveraging reference model probabilities (π_{ref}) for stable training and better generalization, especially in zero-shot scenarios. This improves CoT reasoning's sequential decision-making and model consistency. The Chain DPO stage is trained on four NVIDIA A100 80GB GPUs over one epoch, taking approximately eight hours.

3 Experiments

3.1 Research Questions

Before conducting the experiments, we formulate several research questions to ensure that ChainRank could effectively address the challenges:

-**RQ1:** Does the CoT instruction tuning improve the step-by-step relevance ranking of passages compared to traditional ranking methods?

-**RQ2:** How does our model's performance on text reranking tasks compare with existing models like RankZephyr and RankGPT_4 across various datasets and task settings?

-**RQ3:** Does introducing DPO improve the performance and robustness of our model?

3.2 Evaluation Benchmarks

We evaluate ranking capabilities using TREC DL19 (Craswell et al., 2020) and DL20 (Craswell et al.,

[1]https://huggingface.co/meta-LLaMA/Meta-LLaMA-3-8B-Instruct

2021) Tracks, derived from MS MARCO V1 (Bajaj et al., 2016), with human-annotated relevance labels. TREC DL19 and DL20 contain 43 and 54 queries, respectively, each paired with 100 candidate passages retrieved by BM25. Additionally, we include cross-domain datasets from BEIR (Thakur et al., 2021), not seen by the fine-tuned LLMs, for broader analysis. Using a sliding window strategy with a size of 20 and a stride of 10, we rerank 100 passages per query.

We use the Massive Multitask Language Understanding (MMLU) benchmark (Hendrycks et al., 2020) to evaluate whether our model retains its text understanding abilities after fine-tuning for ranking tasks. MMLU tests proficiency across 57 diverse subjects, from elementary to advanced professional topics. We compare our model's performance on MMLU with baseline LLMs to ensure ranking fine-tuning does not compromise general-purpose language understanding and generation capabilities, maintaining its versatility in broader tasks.

3.3 Results

Table 1 compares model performance on TREC, BEIR, and MMLU benchmarks. Our 8B ChainRank model outperforms baselines, including RankGPT_4 [2], even with 90% of the training data, and surpasses RankLLaMA3 under identical training conditions, showcasing the effectiveness of our CoT reranking prompt. In the DPO stage, ChainRank-DPO improves across all datasets with just one training epoch, enhancing robustness to passage variations (Appendix B).

On MMLU, ChainRank matches LLaMA3's performance, while RankLLaMA3 shows a slight drop and RankVicuna performs poorly. RankZephyr has completely lost its ability to generate meaningful outputs, it receives a score of 0 on MMLU. Additional examples are in Appendix D.

We present a figure to illustrate the trade-off between performance and inference cost when generating ranking orders at varying step intervals. Notably, our method achieves comparable latency to RankLLaMA3 (26.5 vs. 25.0 seconds). By utilizing parallel distributed evaluation, we further reduce inference latency to 6.73 seconds on 4 A100 GPUs, demonstrating the efficiency of our approach.

[2]RankGPT is the only closed-source and largest model in the table.

Models	TREC		BEIR			MMLU
	DL19	DL20	NFC	COVID	FIQA	AVG.
RankGPT_4	0.746	0.708	**0.406**	0.749	0.333	**0.864**
BM25	0.506	0.480	0.325	0.595	0.236	N/A
Contriever	0.616	0.599	0.328	0.596	0.329	N/A
Gemma-7B	0.533	0.530	0.338	0.573	0.236	0.649
LLaMA3	0.641	0.621	0.269	0.628	0.216	0.662
RankZephyr	0.742	0.709	0.331	0.592	0.230	0.000
RankVicuna	0.668	0.655	0.338	0.592	0.236	0.373
RankLLaMA3	0.730	0.635	0.335	0.672	0.330	0.628
ChainRank-SFT	0.752	0.714	0.353	0.766	0.342	0.663
ChainRank-DPO	**0.755**	**0.717**	0.358	**0.772**	**0.342**	0.663

Table 1: Performance of different models on TREC (nDCG@10) and MMLU (exact match score) benchmarks. All the reranking tasks are based on BM25 retrieval results. The **bold** values in the table highlight the best performance across the respective benchmarks.

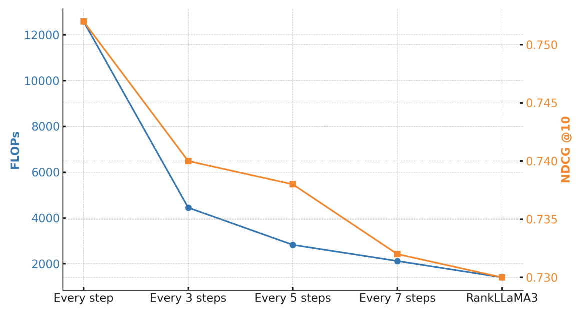

Figure 4: FLOPs and nDCG@10 performance across different step intervals. The blue line represents the inference cost (FLOPs), while the orange line shows the ranking performance (NDCG@10). As the step interval increases, FLOPs decrease significantly, with a slight drop in nDCG@10 scores.

4 Conclusion and Future Work

In this paper, we propose a novel approach, ChainRank, a novel zero-shot listwise text reranking model built on LLaMA3. We demonstrated the effectiveness of CoT reranking prompt and our SFT–Chain DPO pipeline. The results of our experiments show that our model achieves superior performance compared to many open-source and closed-source LLMs, such as RankZephyr and RankGPT_4.

In future work, we plan to explore the application of our method to other models, such as Mistral, Zephyr, and LLaMA3.1, to evaluate its generalizability and performance across different architectures. Additionally, our future research could incorporate higher-quality datasets with varying numbers of passages per instance to enhance the diversity and robustness of the training data. Expanding the dataset in this manner could lead to improved overall performance.

5 Limitations

While this study provides valuable insights, several limitations should be acknowledged. Firstly, the need to include long examples of CoT formatting (Step1: ...) in our prompts leads to increased training and inference times.

Secondly, although LLaMA3 has a larger context window (8192 tokens) compared to Vicuna and Zephyr (4096 tokens), we are still limited by the maximum number of passages in our training data, which is set to 20. As a result, we are restricted to ranking within a window size of 20 and require nine passes to complete the reranking of 100 passages.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Lee Becker, Martha Palmer, Sarel van Vuuren, and Wayne H. Ward. 2012. Question ranking and selection in tutorial dialogues. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP, BEA@NAACL-HLT 2012, June 7, 2012, Montréal, Canada*, pages 1–11. The Association for Computer Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *CoRR*, abs/2005.14165.

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

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

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Maram Hasanain. 2018. Automatic ranking of information retrieval systems. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018, Marina Del Rey, CA, USA, February 5-9, 2018*, pages 749–750. ACM.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118*.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446.

Xin Lai, Zhuotao Tian, Yukang Chen, Senqiao Yang, Xiangru Peng, and Jiaya Jia. 2024. Step-dpo: Step-wise preference optimization for long-chain reasoning of llms. *arXiv preprint arXiv:2406.18629*.

Yongqi Li, Xinyu Lin, Wenjie Wang, Fuli Feng, Liang Pang, Wenjie Li, Liqiang Nie, Xiangnan He, and Tat-Seng Chua. 2024. A survey of generative search and recommendation in the era of large language models. *CoRR*, abs/2404.16924.

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*.

Qi Liu, Bo Wang, Nan Wang, and Jiaxin Mao. 2024. Leveraging passage embeddings for efficient listwise reranking with large language models. *arXiv preprint arXiv:2406.14848*.

Tie-Yan Liu. 2009. Learning to rank for information retrieval. *Found. Trends Inf. Retr.*, 3(3):225–331.

Fan Ma, Haoyun Yang, Haibing Yin, Xiaofeng Huang, Chenggang Yan, and Xiang Meng. 2019. Online learning to rank in a listwise approach for information retrieval. In *IEEE International Conference on Multimedia and Expo, ICME 2019, Shanghai, China, July 8-12, 2019*, pages 1030–1035. IEEE.

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*.

Zhiyuan Peng, Xuyang Wu, Qifan Wang, Sravanthi Rajanala, and Yi Fang. 2024. Q-PEFT: query-dependent parameter efficient fine-tuning for text reranking with large language models. *CoRR*, abs/2404.04522.

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

Ronak Pradeep, Sahel Sharifymoghaddam, and Jimmy Lin. 2023b. Rankzephyr: Effective and robust zero-shot listwise reranking is a breeze! *arXiv preprint arXiv:2312.02724*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

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

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. 2024. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogenous benchmark for zero-shot evaluation of information retrieval models. *CoRR*, abs/2104.08663.

Yu Zhao and Fang Liu. 2024. A survey of retrieval algorithms in ad and content recommendation systems. *CoRR*, abs/2407.01712.

A Baseline Selection

To illustrate the efficacy of ChainRank, we select several baseline models, including the existing retriever models, BM25 (Robertson and Zaragoza, 2009) and Contriever (Izacard et al., 2021), zero-shot prompt-based LLMs, RankGPT$_4$ and Gemma-7B (Team et al., 2024), fine-tuned LLM-based models (RankVicuna, RankZephyr), as well as our base model LLaMA3-8B-Instruct. To show the effectiveness of our CoT reranking prompt, we train a model with a similar paradigm to RankVicuna but use our base model LLaMA3-8B-Instruct, denoted as RankLLaMA3.

B Window Size

Models	Window/Stride Size		
	2/1	10/5	20/10
Gemma-7B	0.541	0.593	0.533
LLaMA3	0.549	0.660	0.641
RankZephyr	0.562	0.612	0.742
RankVicuna	0.551	0.655	0.668
RankLLaMA3	0.567	0.663	0.666
ChainRank-SFT	0.697	0.737	0.755
ChainRank-DPO	0.671	0.755	0.758

Table 2: Performance of different models with different window size and step size, BM25 performs all of the retrieval stage, and all of the evaluations are performed on DL19

C Explanation of CoT strategies

To evaluate the trade-off between performance and inference cost, we trained our model to generate ranking orders after every 1, 3, 5, and 7 steps. An additional instruction was introduced to guide the model in returning ranking orders at these specified intervals. For comparison, we also included the performance of RankLLaMA3 to highlight the effectiveness of our CoT strategies

D Examples of general LLM tasks

To illustrate the degradation in general text understanding and generation capabilities of RankVicuna and Rankzephyr, we provide two examples below. Due to the length of ChainRank's response, we have omitted the middle portion of the answer.

D.1 Example 1

Prompt

Compose an engaging travel blog post about a recent trip to Hawaii, highlighting cultural experiences and must-see attractions.

RankVicuna

Ultimately, showcasing Hawaii's rich history and cultural significance.
Title: Embracing Hawaiian Culture: Discovering Historical Landmarks and Rich Traditions on the Big Island

RankZephyr

[1] > [2] > [3] > [4] > [5] > [6] > [7] > [8] > [9] > [10] > [2] > [11] > [12] > [13]

ChainRank

Aloha and Mahalo: Discovering the Aloha Spirit in Hawaii
As I stepped off the plane in Honolulu, I couldn't help but feel a sense of excitement and anticipation wash over me. I had dreamed of visiting Hawaii for years, and finally, I was here. The warm tropical air enveloped me, and the sweet scent of plumeria filled my senses as I made my way to my hotel. Little did I know, this trip would be a journey not only to breathtaking natural wonders but also to a rich cultural heritage that would leave me feeling grateful and inspired.
Immersing in the Local Culture:
(Content omitted for brevity.)
Must-See Attractions:
(Content omitted for brevity.)
A Taste of Aloha:
(Content omitted for brevity.)
Conclusion:
(Content omitted for brevity.)
Practical Tips and Recommendations:

- Visit Pearl Harbor and USS Arizona Memorial in the morning to avoid the crowds.
- Try local specialties like poke, laulau, and shave ice.
- Take a surf lesson or snorkeling trip to explore the islands' stunning coastlines.
- Visit the Bishop Museum to learn about the history and geology of Hawaii.
- Explore the islands' many farmers' markets for fresh produce and local products.
- Learn traditional Hawaiian crafts, like leis and ukulele strings, from local artisans.

Mahalo (thank you) for joining me on this journey to the Aloha State. I hope my story has inspired you to plan your own trip to Hawaii and experience the warmth and beauty of this incredible destination.

D.2 Example 2

Prompt

Can you help me write a resignation letter to my current employer, while leaving on good terms and expressing gratitude for the opportunities provided?

RankVicuna

(Nothing was generated from the model.)

RankZephyr

[1] > [2] > [3] > [4] > [5] > [6] > [7] > [8] > [9] > [10] > [2] > [3] > [4] > [5] > [6]

ChainRank

I'd be happy to help you write a resignation letter to your current employer. Here's a sample template you can use as a guide:

Dear [Recipient's Name],
I am writing to inform you of my decision to resign from my position as [Your Position] at [Company Name], effective [Date of Last Day of Work]. I want to express my deepest gratitude to you and the entire [Company Name] team for the opportunities I've had during my time here.
(Content omitted for brevity.)
Please don't hesitate to reach out to me if you need anything after my departure. I can be reached at [Your Email Address] or [Your Phone Number].
Sincerely,
[Your Signature]

Some tips to keep in mind when writing your resignation letter:

- Be professional and respectful: Use a formal tone and language throughout the letter.
- Express gratitude: Let your employer know how much you appreciate the opportunities you've had and the experience you've gained.
- Offer to help with the transition: Show that you're committed to making the transition as smooth as possible by offering to complete outstanding tasks and assist with the handover of your responsibilities.
- Keep it brief: Aim for a letter that is no more than one page in length.
- Proofread: Make sure to proofread your letter carefully to ensure it is free of errors and typos.

Remember to customize the letter to fit your own circumstances and experiences. Good luck with your resignation, and I hope it leads to a positive and successful transition for you!