# BETTER HEADS, BETTER RANKERS: SELECTING AND TRAINING ATTENTION HEADS FOR LISTWISE RERANKING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Attention-based ranking has become a key paradigm for listwise ranking in large language models (LLMs). It reads the attention weights between internal query tokens and document tokens as the basis for ranking, thereby dispensing with autoregressive decoding and offering markedly better efficiency and interpretability than generative rankers. However, its effectiveness depends on two interrelated design choices: *which* attention heads supply the ranking signal, and *how* to perform training augmentation on these heads. Previous studies have not yet established a unified paradigm for these designs. In this paper, we conduct a systematic study of head selection and training for attention-based ranking. On the selection side, we show that prevailing practice of averaging per-head scores over a development set is brittle and the resulting set lacks overall representativeness. We instead propose an *frequency-based* criterion that can select adaptive, representative head sets. On the training side, we introduce a unified two-stage recipe that optimizes the selected heads. During supervised fine-tuning, **att-sft** augments the training objective with loss terms computed from the head-aggregated attention scores. During reinforcement learning, we first identify the discrepancies between training and inference, and then propose **att-dpo**, which can directly align preferences on the attention distribution. Across two backbones and several ranking benchmarks, the recipe consistently surpasses untrained and training-related baselines.

## 1 INSTRUCTION

Listwise document ranking is a foundational component of modern retrieval and retrieval-augmented generation pipelines: given a query and a set of candidate documents, the system must return an ordered list. The improvement of LLMs has given rise to three broad paradigms. *Generative rankers*, exemplified by RankGPT (Sun et al., 2023), decode the ordered document identifiers autoregressively. *FIRST*-style rankers (Reddy et al., 2024), which also need generation, sidestep full decoding by consuming only the first output token and ranking candidates according to its logits. *Attention-based rankers* (Chen et al., 2025) take a further step: they derive document scores directly from the internal attention weights, and return the permutations in descending order of those scores.

The attention-based paradigm is appealing for two reasons. First, it is *efficient*: a single forward pass over the concatenated query–documents input yields scores for all candidates simultaneously, with neither autoregressive decoding nor an output length that grows with the number of documents. Second, it is *interpretable and controllable*: because the ranking signal originates from identifiable attention heads, one can inspect, prune, or intervene on precisely the heads that drive the decision.

Despite this progress, two core design questions need to be explored. *Which attention heads are critical for ranking?* LLMs have many attention heads that handle different functions, and selecting the task-related heads can improve performance on corresponding task (Wu et al., 2025). For listwise ranking, the dominant approach scores every head on a development set and retains the top-$k$ heads by *average* score (Zhang et al., 2025; Tran et al., 2025). Although convenient, it is *brittle*: because the per-head score distribution over examples is heavy-tailed, a small number of extreme queries can inflate a head's mean. On the other hand, *how should the selected heads be trained?* Compared to other rankers (Pradeep et al., 2023a; Reddy et al., 2024) that already have well-established training



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

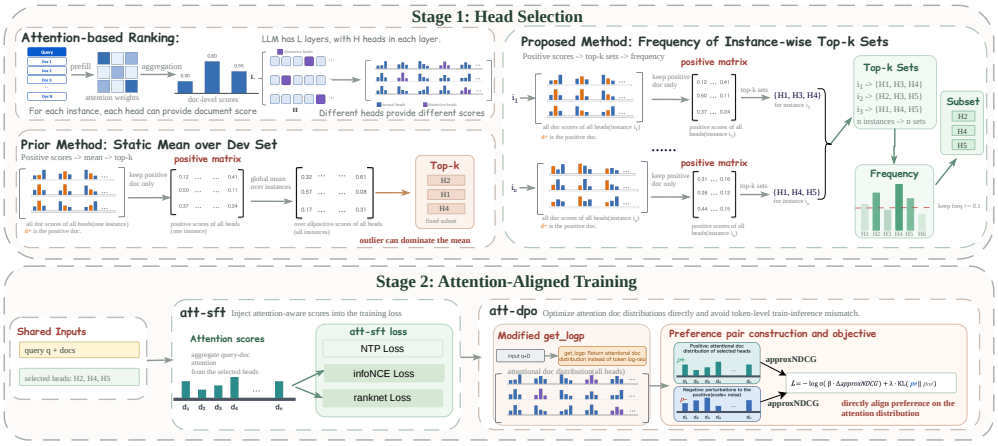



Figure 1: Framework overview: Frequency-based head selection, att-sft and att-dpo.

procedures, the potential for enhancing attention ranking through training remains to be explored. Some previous studies proposed strategies during the SFT (Li et al., 2026) or RL (Wang et al., 2026) training phases, but these approaches are limited to a single phase and only compare results to the baselines without training, which make them constrained.

In this paper, we revisit both questions and improve attention ranking by optimizing head selection and refining entire training pipeline, which is shown in figure 1. First, we propose a frequency-based selection: instead of accumulating scores, we perform a frequency analysis and make selections based on thresholds. The criterion is robust to outlier queries, adaptive in the number of heads it selects, and outperforms average-based selection. Based on the selected heads, we introduce a supervised fine-tuning stage, termed as **att-sft**, augments the training objective with an InfoNCE term and a RankNet term, both computed from the head-aggregated attention scores. For RL phase, we identify the discrepancies between training and inference that arise in standard preference optimization and propose preference alignment that directly targets the attention distribution. Our method, **att-dpo**, redefines the policy log-probability as an attention-derived document distribution, constructs preference pairs by perturbing that distribution, and optimizes an approxNDCG-based SimPO (Meng et al., 2024) objective. We demonstrate that the three components compose into a complete SFT–RL pipeline that improves attention-based listwise ranking on Qwen3-4B and Qwen3-8B backbones, surpassing both the untrained and the training-related baselines.

## 2 Related Works

**Listwise ranking with LLMs.** Listwise ranking with LLMs involves concatenating the query and all candidate documents in the prompt, and then having the LLM to produce ranking. RankGPT (Sun et al., 2023) established that instruction-tuned LLMs are strong zero-shot rerankers. RankVicuna (Pradeep et al., 2023a) and RankZephyr (Pradeep et al., 2023b), distilled this capability into smaller open models. To alleviate the cost of complete decoding, *FIRST*-style methods (Reddy et al., 2024) read the logits at the first output position. Chen et al. (2025) introduced ICR, demonstrating that the ranking signals from internal attention weights is reliable for document ranking. QRHead (Zhang et al., 2025) refined ICR by identifying a set of "query-relevant" heads and aggregating scores over only those heads. Later works (Tran et al., 2025; Jin et al., 2026) further improved the aggregation function and the handling of positional bias. In addition, some studies (Li et al., 2026; Wang et al., 2026) have proposed incorporating attention scores into the training phase. Our work comprehensively study attention ranking, including head selection and training.

**Head-level analysis and control.** A parallel line of research seeks to characterize the functional roles of individual attention heads. Voita et al. (2019) and Michel et al. (2019) established that many heads can be pruned with negligible loss. Wu et al. (2025) identified "retrieval heads" that are responsible for copy-from-context behavior in long-context LLMs, while Zhang et al. (2025) and Tran et al. (2025) isolated heads that are curial for ranking. Our work also investigates the functionality and control mechanisms of attention heads and proposes a tailored training framework.

# 3 METHOD

## 3.1 PROBLEM FORMULATION AND PRELIMINARIES

**Attention-based listwise ranking.** Given a sample consisting of query $q$ and a set of candidate documents $D = \{d_1, d_2, ..., d_n\}$, the goal of listwise ranking is to produce a sorted permutation $\hat{\pi}$. This process involves calculating relevance scores. For a decoder-only LLM $\pi_\theta$ with $L$ layers and $H$ heads per layer, $\mathcal{P} = \{(l, h) : 1 \leq l \leq L,\ 1 \leq h \leq H\}$ denotes the full pool of all heads. Chen et al. (2025) proposed the in-context ranking (ICR) paradigm, which aggregates the corresponding weights between query and document tokens to obtain document-level relevance scores as follows:

$$X = [\texttt{Instruction} \oplus d_1 \oplus d_2 \oplus ... \oplus d_n \oplus q] \qquad A = \pi_\theta(X) \tag{1}$$

$$\alpha_{d_t}^{(l,h)} = \frac{1}{|X_q|} \sum_{i \in X_q} \sum_{j \in X_{d_t}} A_{i,j}^{(l,h)} \tag{2}$$

where $X$ is the input tokens, and $A$ denotes the softmax-normalized attention matrix for input $X$ obtained from a single forward pass through $\pi_\theta$ . For token $i$ in the query tokens (denoted as $X_q$) and token $j$ in the tokens of $d_t$ (denoted as $X_{d_t}$, where $d_t \in D$), let $A_{i,j}^{(l,h)}$ denote the attention weight from token $i$ to token $j$ by attention head $h$ at layer $l$. Hereafter, $(l, h)$ will be used as a shorthand for the attention head $h$ at layer $l$. $\alpha_{d_t}^{(l,h)}$ is the relevance score of $d_t$ produced by head $(l, h)$. Relevance scores can be calculated in this manner for all documents to serve as the basis for ranking.

Given a set $\mathcal{H} \subseteq \{(l, h)\}$ of selected heads, the final document score is the mean over $\mathcal{H}$, and the predicted ranking follows by sorting in descending order:

$$\alpha_{d_t}(\mathcal{H}) \ = \ \frac{1}{|\mathcal{H}|} \sum_{(l,h) \in \mathcal{H}} \alpha_{d_t}^{(l,h)}, \qquad \hat{\pi} \ = \ \mathrm{argsort}_t \left(-\alpha_{d_t}(\mathcal{H})\right). \tag{3}$$

The quality of the ranking is thus highly affected by the choice of $\mathcal{H}$. In the implementation of ICR paradigm, the set $\mathcal{H}$ corresponds to all heads across all layers.

**Previous head selection.** Different heads in the model serve different functions, and select a specific set of heads that are dedicated to ranking and use only this subset to calculate the relevance scores may be helpful. Existing methods (Zhang et al., 2025; Tran et al., 2025) choose $\mathcal{H}$ as follows. The selection is based on a labeled development set $\mathcal{S} = \{(q_m, D_m, d_m^+\}_{m=1}^M$, where $d_m^+$ is the positive document in $D_m$. For each sample $m$ in the development set, they use the scoring function $F$ to obtain the score $s_m^{(l,h)}$ for each head $(l, h)$. The final score $s_{\text{avg}}^{(l,h)}$ is the average of all samples:

$$s_m^{(l,h)} = F(l, h, q_m, D_m, d_m^+) \qquad s_{\text{avg}}^{(l,h)} = \frac{1}{M} \sum_m s_m^{(l,h)} \tag{4}$$

Various studies have proposed different methods for designing scoring functions $F$, such as QR-Head(Zhang et al., 2025) and CoreHead(Tran et al., 2025). The calculations are as follows:

$$F_{\text{QR}}(l, h, q_m, D_m, d_m^+) = \alpha_{d_m^+}^{(l,h)} \qquad F_{\text{Core}}(l, h, q_m, D_m, d_m^+) = \frac{\exp(\alpha_{d_m^+}^{(l,h)}/t)}{\exp(\sum_{d_t \in D_m} \alpha_{d_t}^{(l,h)}/t)} \tag{5}$$

where $\alpha$ is the relevance score calculated as Eq.2, and $t$ is the temperature factor. After obtaining scores for each head $(l, h)$, the selection is made by retaining the $k$ heads with the highest $s_{\text{avg}}^{(l,h)}$:

$$\mathcal{H}_{\text{avg}}(k) = \text{Top-}k_{(l,h) \in \mathcal{P}}\ s_{\text{avg}}^{(l,h)}. \tag{6}$$

We refer to this baseline as $\text{TopK}_{\text{avg}}(k)$. $k$ is typically set to 8 or 16. They have proved that ranking solely based on the doc scores of this subset $\mathcal{H}_{\text{avg}}(k)$ yields better results than the entire set $\mathcal{P}$.

## 3.2 FREQUENCY-BASED HEAD SELECTION

**Motivation.** The summary statistic $s_{\text{avg}}^{(l,h)}$ in Eq.4 is a mean, and means are not robust. First, the distribution of per-head scores across examples is heavy-tailed, so a single query can elevate a head's

average even when that head is unhelpful on the majority of queries, thereby masking heads that are dependably useful on the bulk of the data. Second, the cutoff $k$ is chosen arbitrarily and does not adapt to the backbone. Different models may concentrate ranking-relevant behavior in markedly different numbers of heads. Lin et al. (2026) also observed these issues, but rather than studying hidden states, we propose a frequency-based head selection method to optimize the selection criteria.

**Our criterion: count usefulness, do not average it.** Our selection method is also based on the labeled development set $\mathcal{S} = \{(q_m, D_m, d_m^+\}_{m=1}^M$ and we replace the mean by a *frequency*. Specifically, for each example $m$ we first form its own top-$k$ head set $\mathcal{T}_k^{(m)}$ from the per-example scores $s_m^{(l,h)}$, and then measure how often each head enters these per-example sets:

$$\mathcal{T}_k^{(m)} \;=\; \text{Top-}k_{(l,h)\in\mathcal{P}}\; s_m^{(l,h)}, \qquad f(l,h) \;=\; \frac{1}{M}\sum_{m=1}^{M} \mathbf{1}\Big[(l,h) \in \mathcal{T}_k^{(m)}\Big] \;\in\; [0,1]. \tag{7}$$

The final head set keeps every head whose frequency clears a threshold,

$$\mathcal{H}^\star \;=\; \big\{(l,h) \in \mathcal{P} : f(l,h) \geq \tau\big\}, \qquad \tau = 0.1 \text{ by default.} \tag{8}$$

**Robustness and adaptivity.** The criterion is both robust and adaptive. First, it is a bounded, rank-based statistic. Changing a head's score on one example alters $f(l,h)$ by at most $1/M$, whereas it can shift the mean $\bar{g}(l,h)$ arbitrarily, so outliers cannot inflate a head, unlike average-based selection. Moreover, the threshold $\tau$ in Eq.8 does not fix $|\mathcal{H}^\star|$ a priori: it stays small when ranking behavior is concentrated in a few recurring heads and grows when the behavior is distributed, adapting the set size to each backbone, which transfers far more easily across models than an absolute count $k$. In contrast, $k$ in Eq. 7 only sets the granularity of the per-example vote and is far less sensitive, so we fix it and vary only $\tau$. We report the sensitivity in §5.1. $k$ is set to 8 for all models.

We refer to the frequency-based head selection method as $\text{Freq}(k,\tau)$. Compared to the baseline $\text{TopK}_{\text{avg}}(k)$, $\text{Freq}(k,\tau)$ does not introduce any additional computational overhead, while the results can be obtained by performing a single forward pass on each example. Therefore, head selection costs $O(M)$ forward passes in total and is performed once, offline, before any training.

### 3.3 ATT-SFT: SFT WITH ATTENTION-AWARE LOSSES

Given $\mathcal{H}^\star$, the supervised fine-tuning stage adapts $\pi_\theta$ so that the heads in $\mathcal{H}^\star$ produce cleaner, better-separated document scores. The key idea is to supervise *the attention scores*.

**Data Preparation and Preprocessing.** Our training does not require any special data structure, and apply traditional listwise data. For a training example consisting of a query $q$, documents $D$ and a gold permutation $\pi^{\text{gold}}$, we first obtain the document-level scores using the selected head sets, $\alpha(\mathcal{H}^\star) = \{\alpha_{d_1}(\mathcal{H}^\star), \alpha_{d_2}(\mathcal{H}^\star), ..., \alpha_{d_n}(\mathcal{H}^\star)\}$. Based on $\alpha(\mathcal{H}^\star)$, we applied max-min normalization and scaling to alleviate instability issue, as suggested by Li et al. (2026):

$$\alpha^{\text{norm}} = \frac{\alpha(\mathcal{H}^\star) - \min(\alpha(\mathcal{H}^\star))}{\max(\alpha(\mathcal{H}^\star)) - \min(\alpha(\mathcal{H}^\star))} \tag{9}$$

**Loss terms.** The att-sft objective combines three complementary terms:

- *Next-token prediction* ($\mathcal{L}_{\text{ntp}}$): the standard causal language-modeling loss on the target answer. It anchors the model to its pretrained distribution and preserves the generative-ranking ability, preventing the attention-side terms from collapsing the representation.
- *InfoNCE* ($\mathcal{L}_{\text{infonce}}$): a contrastive term that treats $d_+$ as the positive key and the $\{d_{-,j}\}$ as in-batch negatives, pulling query attention toward the relevant document. For each training example, we treat the document ranked first in the gold permutation $\pi^{\text{gold}}$ as a positive example $d_+$, and the rest as negative examples $\{d_{-,j}\}$.

  $$\mathcal{L}_{\text{infonce}} \;=\; -\log \frac{\exp(\alpha_+^{\text{norm}}/\tau_a)}{\exp(\alpha_+^{\text{norm}}/\tau_a) + \sum_j \exp(\alpha_{-,j}^{\text{norm}}/\tau_a)} \tag{10}$$

  where the temperature $\tau_a$ controls how sharply probability mass concentrates on the positive. This term calibrates the *absolute* separation between the positive and the negatives.

- *Weighted RankNet* (Burges et al., 2005) ($\mathcal{L}_{\text{ranknet}}$): a pairwise loss that penalizes every score inversion relative to the gold order, with a FIRST-style (Reddy et al., 2024) weighting that emphasizes the top of the ranking. For each ordered pair $(i, j)$ with $\pi^{\text{gold}}(i) < \pi^{\text{gold}}(j)$ (i.e., $d_i$ should outrank $d_j$),

$$\mathcal{L}_{\text{ranknet}} \;=\; \sum_{i \prec j} w_{ij}\;\log\big(1+\exp\big(-\frac{(\alpha_i^{\text{norm}}-\alpha_j^{\text{norm}})}{\tau_a}\big)\big). \tag{11}$$

$$w_{ij} = 1/\log_2(i+2) + 1/\log_2(j+2), \tag{12}$$

Whereas InfoNCE only distinguishes positive from negative, RankNet supervises the *full relative order* among all documents, which provides fine-grained supervisory signals. The implement $w_{ij}$ is substitutable, and we analyze this in the ablation experiments in §5.2.

**Combined objective.** The overall att-sft loss is a weighted sum,

$$\mathcal{L}_{\text{att-sft}} \;=\; \lambda_1\,\mathcal{L}_{\text{ntp}} + \lambda_2\,\mathcal{L}_{\text{infonce}} + \lambda_3\,\mathcal{L}_{\text{ranknet}}, \tag{13}$$

with $\lambda_1, \lambda_2, \lambda_3 \geq 0$. Here $\mathcal{L}_{\text{ntp}}$ keeps the model competitive as a generative ranker, whereas the two attention-side terms shape the heads in $\mathcal{H}^\star$. Since the two attention-related loss terms are computed during the prefill stage, att-sft adds negligible overhead over plain SFT. The gradient flow of the objective is detailed in appendix A.1.

### 3.4 ATT-DPO: RL FOR ATTENTION RANKING

After att-sft, the selected heads already yield reasonable rankings. We now push their performance further with a preference-based reinforcement-learning stage. The central difficulty is that the mismatch of training and inference. In this section, we first make this precise, then correct it.

**The discrepancies between training and inference of standard DPO.** For a preferred completion $y_+$ and a dispreferred completion $y_-$, standard DPO (Rafailov et al., 2023) minimizes

$$\mathcal{L}_{\text{DPO}} = -\log\sigma\left(\beta\big[\underbrace{\log\tfrac{\pi_\theta(y_+|x)}{\pi_{\text{ref}}(y_+|x)}}_{\text{token log-ratio}} - \log\tfrac{\pi_\theta(y_-|x)}{\pi_{\text{ref}}(y_-|x)}\big]\right), \tag{14}$$

where each $\log\pi(y|x) = \sum_t \log\pi(y_t|x, y_{<t})$ is a sum of *token* log-probabilities. Eq. 14 is aligned with a *generative* ranker: it teaches the model to *decode* the string $y_+$ more readily than $y_-$. Our inference procedure, however, never decodes. It consumes the document-level distribution induced by $\alpha(\mathcal{H}^\star)$ through only prefill. There is no reason the token log-ratio from generation phase should track the ordering from prefill phase. Optimizing it affects inference only indirectly.

**Redefining the policy log-probability.** To align training with inference, we replace the token-sequence probability by the *attention-derived document distribution* that inference actually uses. For an input $x = [\texttt{Instruction} \oplus d_1 \oplus d_2 \oplus ... \oplus d_n \oplus q]$, we define

$$\log\pi_\theta(y \mid x) \;\coloneqq\; \log\mathbf{p}_\theta(x), \qquad \mathbf{p}_\theta(x)_i = \frac{\exp\big(\alpha_{d_i}(\mathcal{H}^\star)/\tau_a\big)}{\sum_j \exp\big(\alpha_{d_j}(\mathcal{H}^\star)/\tau_a\big)}, \tag{15}$$

so that the "response" $y$ is now an entire attentional distribution over documents rather than a token sequence. The reference model $\pi_{\text{ref}}$ is defined identically, but with $\alpha$ computed from the frozen checkpoint. Crucially, $\mathbf{p}_\theta(x)$ is consistent the object that Eq.3 sorts at test time.

**Preference construction by distribution-level perturbation.** Because $y$ is a distribution rather than a decodable sequence, we can avoid additional construction of training data and build preference pair based on the distribution. We hold the original $\mathbf{p}_\theta(x)$ as the positive and construct the negative by *perturbing the distribution itself*. Specifically, our perturbation method involves scaling and then adding Gaussian noise to the pre-softmax score $\alpha(\mathcal{H}^\star)$ before renormalizing. This ensures that the negative remain inferior to the positive while retaining some features:

$$\alpha_{d_i}^{\text{pert}} = \lambda\;\alpha_{d_i}(\mathcal{H}^\star)) + \varepsilon_i \qquad \mathbf{p}_- = \frac{\exp\big(\alpha_{d_i}^{\text{pert}}(\mathcal{H}^\star)/\tau_a\big)}{\sum_j \exp\big(\alpha_{d_j}^{\text{pert}}(\mathcal{H}^\star)/\tau_a\big)} \tag{16}$$

where $\lambda$ is the scaling factor and $\varepsilon_i \sim \mathcal{N}(0, \sigma_p^2)$ is the Gaussian noise. Overall, the positive $\mathbf{p}_+$ is the unperturbed $\mathbf{p}_\theta(x)$ and the negative $\mathbf{p}_-$ is its perturbed counterpart. Since the perturbation acts on the *output distribution* and not on the input tokens, both members share a single forward pass, which is what makes the scheme cheap and stable relative to sampling alternative token sequences.

**Training objective.** Traditional DPO compares the joint probability of generation, whereas we obtain the attention-derived document distribution. Therefore, we propose a new method for evaluating the distribution. We score a candidate distribution with the differentiable approxNDCG (Bruch et al., 2020), which admits gradients through the sort, and see the appendix A.2 for the calculation details. Given the distribution $\mathbf{p}_+$, $\mathbf{p}_-$ and gold labels $\pi_{\text{gold}}$, we define the preference margin:

$$\Delta\text{aNDCG} \;=\; \text{approxNDCG}(\mathbf{p}_+, \pi_{\text{gold}}) - \text{approxNDCG}(\mathbf{p}_-, \pi_{\text{gold}}) \tag{17}$$

which is large when the unperturbed distribution ranks documents better than its perturbed negative, and this is exactly the signal we wish to reinforce. Following the reference-free reward of SimPO (Meng et al., 2024), we place the margin inside a log-sigmoid and add a KL regularizer toward the reference distribution, both evaluated on the same input $x$:

$$\mathcal{L}_{\text{att-dpo}} = \underbrace{-\log \sigma\Big(\beta \cdot \Delta\text{aNDCG}\Big)}_{\text{preference term}} + \lambda_{\text{KL}} \underbrace{\text{KL}\big(\mathbf{p}_\theta(x) \,\|\, \mathbf{p}_{\text{ref}}(x)\big)}_{\text{anchor to att-sft}}. \tag{18}$$

The scale $\beta$ controls how hard the preference is enforced, and $\lambda_{\text{KL}}$ trades off improvement against drift. We place the KL term *outside* the log-sigmoid so that it acts as an explicit trust region on the document distribution, rather than being folded into the reward. Empirically this is essential for stability. The gradient flow of the objective is detailed in appendix A.1.

**Complete Training Pipeline.** Figure 1 summarizes the complete recipe: select $\mathcal{H}^\star$ via Eq.8, run att-sft with Eq.13, and finally run att-dpo with Eq.18. Head selection may optionally be repeated on the att-sft checkpoint, but we find that reusing the same $\mathcal{H}^\star$ throughout works well and simplifies the pipeline. The selection of $\mathcal{H}^\star$ is compatible with different scoring functions $F$, and we mainly apply $F_{\text{QR}}$ in the experiments. The pseudocode of the pipeline can be found in the appendix A.3.

# 4 Experiments

## 4.1 Experimental Setup

**Evaluation Datasets and Metrics.** As with previous works (Tran et al., 2025; Wang et al., 2026), we conducte our evaluation on the widely used listwise ranking benchmark BEIR (Thakur et al., 2021), selecting five subsets: NQ, HotpotQA, FiQA, trec-COVID and Quora. For the passage retrieval, we follow the setting of Tran et al. (2025) and use *granite-embedding-30m-english* to retrieve top-40 documents for each subset. Detailed statistics for the eval data are provided in the appendix B.1. For evaluation metrics, NDCG@10 serves as the primary metric.

**Training Datasets.** Our two training stages use different data sources. For the SFT stage, we use the training data released with RankZephyr (Pradeep et al., 2023b), which provides listwise supervision distilled from GPT-4 (Achiam et al., 2023). For the RL stage, we use the training data of RankGPT (Sun et al., 2023), which provides listwise supervision distilled from GPT-3.5. Both datasets come from MS MARCO, but they have different annotation sources, and the overlap between them is so low as to be negligible. They are both disjoint from the evaluation datasets above, which guarantees that the improvement achieved through training is not due to specific data. Detailed statistics for the training data are provided in the appendix B.2.

**Baselines.** We compare against two families of methods. The *training-free* family applies a fixed backbone without any parameter updates and differs only in how the ranking is produced: (i) **Retrieval**, which directly treats the retrieval order as the final order, regardless of the model. (ii) **Generative**, which decodes the ordered document identifiers in the RankGPT (Sun et al., 2023) style with untrained model; (iii) **ICR** (Chen et al., 2025), which ranks by query–document attention weights aggregated over all heads; (iv) **QR** (Zhang et al., 2025), which applies $F_{\text{QR}}$ to score heads,

Table 1: Main results (nDCG@10, %) on five BEIR datasets. **Bold** marks the best result in each column. Shaded rows are our proposed methods. *Avg.* is the macro-average over the five datasets.

| | Qwen3-4B | | | | | | Qwen3-8B | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Method | NQ | HotpotQA | FiQA | COVID | Quora | Avg. | NQ | HotpotQA | FiQA | COVID | Quora | Avg. |
| *No training (training-free)* | | | | | | | | | | | | |
| Retrieval | 51.6 | 62.9 | 36.9 | 63.1 | 86.7 | 60.2 | 51.6 | 62.9 | 36.9 | 63.1 | 86.7 | 60.2 |
| Generative | 61.4 | 71.9 | 39.5 | 70.3 | 76.4 | 63.9 | 62.1 | 73.8 | 42.8 | 75.5 | 75.1 | 65.9 |
| ICR | 59.8 | 74.5 | 40.1 | 74.3 | 79.5 | 65.6 | 60.7 | 74.6 | 41.2 | 76.7 | 80.7 | 66.8 |
| QR | 62.9 | 76.1 | 43.2 | 76.0 | 83.4 | 68.3 | 63.7 | 76.8 | 43.7 | 77.3 | 83.3 | 69.0 |
| CoRe | 62.6 | 76.3 | 43.7 | 76.8 | 82.9 | 68.5 | 63.7 | 76.9 | 43.9 | 78.5 | 84.3 | 69.5 |
| Freq-QR | 63.3 | 76.0 | 43.6 | 77.3 | 83.9 | 68.8 | 64.5 | 76.7 | 43.8 | 79.4 | 83.6 | 69.6 |
| Freq-CoRe | 62.8 | 76.5 | 43.8 | 77.0 | 83.4 | 68.7 | 63.8 | 76.8 | 43.9 | 79.0 | 84.6 | 69.6 |
| *SFT only* | | | | | | | | | | | | |
| Vanilla SFT | 69.0 | 74.5 | 46.2 | 83.6 | 83.7 | 71.4 | 69.1 | 76.2 | 46.0 | 83.3 | 84.8 | 71.9 |
| FIRST-style SFT | 68.8 | 75.6 | 44.4 | 85.1 | 70.1 | 68.8 | 69.6 | 76.7 | 46.1 | 85.7 | 72.6 | 70.1 |
| QRRanker | 67.7 | 76.3 | 45.2 | 81.4 | 87.3 | 71.6 | 68.5 | 76.9 | 45.4 | 82.7 | 86.3 | 72.0 |
| att-sft | 69.1 | 76.7 | **46.6** | 83.5 | **87.7** | 72.7 | 69.7 | 77.0 | 46.3 | 85.2 | **87.3** | 73.1 |
| *RL only* | | | | | | | | | | | | |
| HeadRank | 65.9 | 75.2 | 43.8 | 81.7 | 80.6 | 69.4 | 66.0 | 74.0 | 44.0 | 81.3 | 81.9 | 69.4 |
| att-dpo | 66.0 | 76.1 | 44.1 | 82.4 | 82.1 | 70.1 | 66.4 | 76.8 | 44.9 | 83.0 | 81.7 | 70.6 |
| *SFT + RL* | | | | | | | | | | | | |
| att-sft + att-dpo | **69.5** | **77.2** | 46.4 | **85.8** | 87.1 | **73.2** | **70.4** | **77.4** | **46.8** | **86.9** | 87.2 | **73.7** |

uses $\mathrm{TopK}_{\mathrm{avg}}(8)$ to select heads and ranks solely over selected heads; (v) **CoRe** (Tran et al., 2025), which applies $F_{\mathrm{Core}}$ to score heads, uses $\mathrm{TopK}_{\mathrm{avg}}(16)$ to select heads and ranks solely over selected heads; and (vi) **Freq-QR, Freq-CoRe**, the results of $\mathrm{Freq}(\mathrm{k}, \tau)$ applied to $F_{\mathrm{QR}}$ and $F_{\mathrm{Core}}$, respectively. The *trained* family additionally fine-tunes the backbone: (vii) **Vanilla SFT** (Pradeep et al., 2023b), standard listwise supervised fine-tuning without any attention-aware loss. (viii) **FIRST-style SFT** (Reddy et al., 2024), which conduct supervised fine-tuning and ranking in FIRST-style. (ix) **QRRanker** (Li et al., 2026), which trains attention heads selected by $F_{\mathrm{QR}}$ and $\mathrm{TopK}_{\mathrm{avg}}(8)$ during the SFT stage and has only one attention-related loss term. (x) **HeadRank** (Wang et al., 2026), which uses $F_{\mathrm{Core}}$ with a gating mechanism and $\mathrm{TopK}_{\mathrm{avg}}$ to select the head, and then performs RL training. (xi) **att-sft**,**att-dpo** and combined **att-sft+att-dpo**, our proposed recipe, which applies $F_{\mathrm{QR}}$ and $\mathrm{Freq}(k, \tau)$ to select heads and conduct training based on the selected heads. For fair comparison, all trained baselines use the same training data. The SFT phase uses data from RankZephyr, and the RL phase uses data from RankGPT.

**Implementation Details.** We instantiate our method on two open backbones, **Qwen3-4B** and **Qwen3-8B** (Yang et al., 2025). For $\mathrm{Freq}(k, \tau)$, $k$ is set to 8 and $\tau$ is set to 0.1. The head detection dataset $\mathcal{S}$ is consistent with Zhang et al. (2025) and contains only 100 nq examples, and this dataset is used for all head detection. All SFT training runs for 3 epochs, and the checkpoint with the best performance is reported. All RL training runs for up to 500 steps, with checkpoints saved every 100 steps, and the checkpoint with the best performance is reported. The temperature $\tau_a$ is set to 1.0. And $\lambda_1 = \lambda_2 = \lambda_3 = 1.0$ for att-sft. For att-dpo, $\beta$ is set to 0.5, $\lambda$ is set to 0.02 and $\sigma_p$ is set to 0.5. $\lambda_{\mathrm{KL}}$ is set to 0.03. We adopt the DeepSpeed ZeRO Stage 3 strategy and use 8 NVIDIA H20 GPUs. More details are presented in appendix B.3.

### 4.2 Main Results

Table 1 reports nDCG@10 on all five BEIR datasets for both backbones, which shows that: **(1) Frequency-based selection improves training-free ranking.** With the *same* scoring function and frozen backbone, our frequency criterion matches or exceeds average-based $\mathrm{TopK}_{\mathrm{avg}}$: on average, Freq-QR/Freq-CoRe improve over QR/CoRe by $+0.5$/$+0.2$ on Qwen3-4B and $+1.3$/$+0.1$ on Qwen3-8B. **(2) Each training stage beats its respective competitors.** Within each single-stage regime, our loss is the strongest. Our att-sft attains the best average among SFT-only methods on every backbone (72.7/73.1). The gain is telling: all use identical RankZephyr data and differ only in the loss, so it is directly attributable to the attention-aware InfoNCE and RankNet terms (Eq. equation 13), which supervise the head scores used at inference. The same applies to the RL phase, and the gain reflects the superiority of our training objective. **(3) The two stages are complementary**

**and their combination is best.** Starting from a weaker training-free initialization, att-dpo alone trails the SFT stage, so the two are complementary rather than substitutive. Composing them gives the best average on both backbones (73.2/73.7) and the best score, adding $+0.5$/$+0.6$ over att-sft. Overall, our contributions improve results and outperform baselines consistently and additively.

# 5 ANALYSIS

In this section, we will examine each optimization strategy individually, analyze the impact of relevant factors. The experiment was primarily conducted on the NQ, FIQA, and HotpotQA datasets.

## 5.1 HEAD SELECTION

Table 2: Head-selection study (nDCG@10, %), using the $F_{\mathrm{QR}}$ on the training-free backbone. ***#H*** is the number of selected heads (adaptive for $\mathrm{Freq}$).

| Qwen3-4B | | | | | | Qwen3-8B | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Selection | #H | NQ | HotpotQA | FiQA | Avg. | Selection | #H | NQ | HotpotQA | FiQA | Avg. |
| $\mathrm{TopK}_{\mathrm{avg}}(8)$ | 8 | 62.9 | 76.1 | 43.2 | 60.7 | $\mathrm{TopK}_{\mathrm{avg}}(8)$ | 8 | 63.7 | 76.8 | 43.7 | 61.4 |
| $\mathrm{TopK}_{\mathrm{avg}}(16)$ | 16 | 62.4 | 76.2 | 43.1 | 60.6 | $\mathrm{TopK}_{\mathrm{avg}}(16)$ | 16 | 63.4 | 76.4 | 43.7 | 61.2 |
| $\mathrm{TopK}_{\mathrm{avg}}(21)$ | 21 | 62.2 | 76.0 | 42.5 | 60.2 | $\mathrm{TopK}_{\mathrm{avg}}(22)$ | 22 | 63.8 | 76.5 | 43.4 | 61.2 |
| $\mathrm{TopK}_{\mathrm{avg}}(32)$ | 32 | 62.0 | 75.8 | 42.5 | 60.1 | $\mathrm{TopK}_{\mathrm{avg}}(32)$ | 32 | 63.6 | 76.1 | 43.6 | 61.1 |
| $\mathrm{Freq}(8, 0.1)$ | 21 | **63.3** | 76.0 | 43.6 | **61.0** | $\mathrm{Freq}(8, 0.1)$ | 22 | **64.5** | 76.7 | 43.8 | **61.7** |
| $\mathrm{Freq}(8, 0.05)$ | 48 | 62.4 | 75.8 | **43.7** | 60.6 | $\mathrm{Freq}(8, 0.05)$ | 39 | 63.2 | 76.5 | 43.4 | 61.0 |
| $\mathrm{Freq}(8, 0.15)$ | 11 | 62.7 | **76.3** | 43.5 | 60.8 | $\mathrm{Freq}(8, 0.15)$ | 9 | 63.2 | **76.9** | **44.3** | 61.5 |
| $\mathrm{Freq}(16, 0.1)$ | 40 | 62.2 | 76.1 | 42.7 | 60.3 | $\mathrm{Freq}(16, 0.1)$ | 40 | 63.3 | 76.7 | 43.4 | 61.1 |

In the table 1, we have demonstrated that our proposed frequency-based selection method $\mathrm{Freq}(k, \tau)$ outperforms previous $\mathrm{TopK}_{\mathrm{avg}}(k)$. Here, we further validate this finding and discuss the impact of hyperparameters in table 2. Our default $\mathrm{Freq}(8, 0.1)$ attains the best average on both backbones, surpassing the best $\mathrm{TopK}_{\mathrm{avg}}$ configuration. The most direct comparison is against $\mathrm{TopK}_{\mathrm{avg}}(21/22)$, which selects the *same number* of heads yet trails by $0.7$/$0.4$ points, isolating the benefit of the selection *rule* rather than the head budget. Two hyperparameter trends further support our design. First, $\mathrm{TopK}_{\mathrm{avg}}$ degrades monotonically as $k$ grows ($60.7 \rightarrow 60.1$ on 4B), because a fixed larger budget forces in heads that help on average yet hurt on most queries, which is exactly the outlier-sensitivity our frequency criterion avoids. Second, $\mathrm{Freq}$ is robust to $\tau$: over $\tau \in \{0.05, 0.1, 0.15\}$ the selected set adapts in size (from $48$ down to 9–11 heads) while the average stays within $0.4$ point and remains competitive with or above $\mathrm{TopK}_{\mathrm{avg}}$. We therefore adopt $\mathrm{Freq}(8, 0.1)$ throughout.

## 5.2 SFT PHASE

Table 3: Ablation of att-sft (nDCG@10, %). † is our default configuration.

| | Qwen3-4B | | | | Qwen3-8B | | | |
|---|---|---|---|---|---|---|---|---|
| Setting | NQ | HotpotQA | FiQA | Avg. | NQ | HotpotQA | FiQA | Avg. |
| *(i) Head set used to compute attention scores* | | | | | | | | |
| $\mathrm{random}(8)$ | 66.6 | 76.3 | 44.8 | 62.6 | 68.0 | 76.6 | 44.8 | 63.1 |
| $\mathrm{random}(21)$ | 67.0 | 76.6 | 44.2 | 62.6 | 68.3 | 77.0 | 45.9 | 63.7 |
| all (1152) | 66.9 | 76.6 | 43.9 | 62.5 | 68.0 | 76.7 | 44.8 | 63.2 |
| $\mathcal{H}_{\mathrm{avg}}(8)$ | 66.2 | 76.3 | 45.5 | 62.7 | 68.2 | **77.4** | 46.0 | 63.9 |
| $\mathcal{H}^{\star}$ † | **69.1** | 76.7 | **46.6** | **64.1** | **69.7** | 77.0 | **46.3** | **64.3** |
| *(ii) Loss-term combination* | | | | | | | | |
| $\lambda_1{=}\lambda_2{=}1$ (no RankNet) | 67.9 | **76.8** | 44.1 | 62.9 | 68.6 | 77.1 | 45.1 | 63.6 |
| $\lambda_1{=}\lambda_3{=}1$ (no InfoNCE) | 66.3 | 75.8 | 44.1 | 62.1 | 67.0 | 75.5 | 44.6 | 62.4 |
| $\lambda_2{=}\lambda_3{=}1$ (no NTP) | 68.8 | 76.4 | 45.4 | 63.5 | 68.7 | 77.2 | 44.6 | 63.5 |
| $\lambda_1{=}\lambda_2{=}\lambda_3{=}1$ † | **69.1** | 76.7 | **46.6** | **64.1** | **69.7** | 77.0 | **46.3** | **64.3** |
| *(iii) RankNet pair weighting $w_{ij}$* | | | | | | | | |
| ndcg | 68.5 | 76.7 | 46.1 | 63.8 | 68.7 | 77.0 | 45.8 | 63.8 |
| FIRST † | **69.1** | 76.7 | **46.6** | **64.1** | **69.7** | 77.0 | **46.3** | **64.3** |

We ablate three design choices of att-sft: (1) which head set supplies the attention scores used in the loss, (2) which loss terms are combined, and (3) how the RankNet pair weights $w_{ij}$ are computed.

Table 3 reports all three studies. The last row of each block, $\mathcal{H}^\star$ with the full loss ($\lambda_1{=}\lambda_2{=}\lambda_3{=}1$) and FIRST-style weighting, is our default configuration and serves as the shared reference. All three factors, on both backbones, support our default. **(1)** Supervising our selected set $\mathcal{H}^\star$ is decisive, beating random subsets, all heads, and $\mathcal{H}_{\text{avg}}(8)$ by 1.4–1.7 on average. Using *all* heads is no better than a small random subset, showing the gain comes from concentrating supervision on the selected heads rather than from more heads. **(2)** Every loss term helps and the full combination is best. Dropping InfoNCE hurts most ($-2.0$/$-1.9$). **(3)** The FIRST-style $\mathrm{first}$ weighting edges out $\mathrm{ndcg}$ ($+0.3$/$+0.5$) by emphasizing every pair touching a top document, but using the $\mathrm{ndcg}$ weights also yields good results.

### 5.3 RL PHASE

In the main experiment, we demonstrated the performance of the proposed RL enhancement method on the backbone model. In this section, to more fairly highlight the superiority of the RL algorithm, we run all methods under an identical protocol: starting from the *same* listwise-SFT Qwen3-4B checkpoint, we train for 400 steps with the same RankGPT data and hyperparameters, and compare against two baselines, the RL method HeadRank, and Continued SFT that simply keeps supervised fine-tuning on the RankGPT data. Figure 2 plots nDCG@10 of attention-based ranking on NQ against training steps. Our att-dpo is clearly the strongest RL algorithm under this controlled setup. Two conclusions follow. First, since all methods share the initialization and data, the advantage comes purely from the *objective*: optimizing the attention document distribution with an approxNDCG preference (Eq.18) is better aligned with attention ranking than HeadRank's gated token-level RL. Second, the near-flat Continued-SFT curve shows that standard SFT have limited effectiveness in attention ranking.

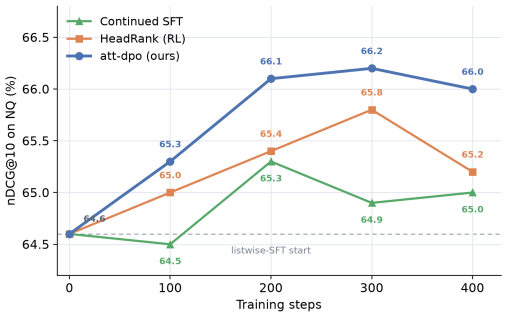



Figure 2: RL-phase *attention-based ranking* curves on Qwen3-4B, starting from a shared listwise-SFT checkpoint (step 0, dashed line).

### 5.4 ADDITIONAL EXPERIMENTS

In this section, we discuss the impact of several additional relevant factors on the overall results. Due to space limitations, we summarize three further studies here and defer full results to the appendix. **(a) Model architecture.** Repeating our recipe on Llama-3.1-8B (Grattafiori et al., 2024) yields the same trends as on the Qwen3 backbones, confirming that frequency-based selection and the att-sft/att-dpo training are not specific to a single model family (Appendix C.1). **(b) Re-selecting heads each epoch.** Re-running head selection after every training epoch does *not* outperform fixing $\mathcal{H}^\star$ once before training. The selected set is stable enough that periodic re-selection only adds cost without accuracy gains, so we keep a single fixed set (Appendix C.2). **(c) Inference efficiency.** Because attention ranking needs no autoregressive decoding and can early-exit once the attention scores at the selected heads are read off, it is substantially faster than generative reranking at the same quality (Appendix C.3).

## 6 CONCLUCION

We revisited two neglected questions in attention-based listwise ranking: which heads to use and how to train them. Our frequency-based criterion selects an outlier-robust, adaptive head set that beats average-based top-$k$, and our two-stage recipe, which consists of *att-sft* (InfoNCE and weighted-RankNet losses on the head-aggregated scores) and *att-dpo* (a preference objective on the attention document distribution that removes the train–inference mismatch of DPO), improves results consistently and additively across five BEIR datasets and three backbones, with the combination best overall while preserving attention ranking's efficiency. Future work includes characterizing head-set transfer to new domains and longer contexts, richer preference pairs for att-dpo, and extending the recipe to retrieval and larger models.

### AI USE STATEMENT

In this work, we used generative AI tools to assist in the writing of proofs and to provide critical ingredients for proving the mathematical claims in the method section. We have not used generative AI tools to generate synthetic datasets, to propose or refine hypotheses, or to design research methodology or experiments, and the remaining required-disclosure tasks (e.g., developing theoretical models, implementing methods, cleaning or reformatting datasets, qualitative or thematic data analysis) are not applicable to this work. Additionally, we used generative AI tools for recommended-disclosure tasks, namely to draft parts of the paper and to aid or polish the writing. We did not use generative AI tools for retrieval and discovery (e.g., finding related work) or for research ideation and execution. We have reviewed all AI-assisted work: every AI-assisted proof step and mathematical claim was checked and verified by the authors, and all AI-drafted or AI-polished text was read and edited by the authors for correctness and accuracy. We take responsibility for the final content of this work, including all text, claims, and artifacts produced with the aid of generative AI.

### ETHICS STATEMENT

This work studies attention-based document ranking and conducts all experiments on publicly available benchmark datasets (BEIR subsets) using publicly released language models. It does not involve human subjects, the collection or release of new data, or personally identifiable or sensitive information. We do not foresee harmful applications, discrimination, fairness, privacy, security, or legal-compliance concerns arising directly from our methods or findings. We therefore believe this work raises no issues that conflict with the ICLR Code of Ethics.

### REPRODUCIBILITY STATEMENT

We have taken several steps to ensure our results are reproducible. The complete method, including the frequency-based head-selection criterion and the att-sft and att-dpo objectives, is specified with full equations in Section 3, and the end-to-end procedure is given as pseudocode in Algorithm 1. The experimental setup— evaluation datasets, metric, training data, baselines, backbones, and hyperparameters—is described in Section 4.1. Dataset statistics and the exact processing used to build the reranking inputs are reported in Appendix B.1, and additional results (different backbones, per-backbone ablations, and inference-efficiency measurements) are provided in Appendices C through C.3. All experiments use publicly available datasets (BEIR subsets) and publicly released models (Qwen3-4B/8B and Llama3-8B). We will release our code as anonymized supplementary material to facilitate exact reproduction.

## REFERENCES

OpenAI Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mo Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Benjamin Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Sim'on Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Raphael Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Lukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik

Kim, Hendrik Kirchner, Jamie Ryan Kiros, Matthew Knight, Daniel Kokotajlo, Lukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Adeola Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel P. Mossing, Tong Mu, Mira Murati, Oleg Murk, David M'ely, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Ouyang Long, Cullen O'Keefe, Jakub W. Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alexandre Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Pondé de Oliveira Pinto, Michael Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario D. Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin D. Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas A. Tezak, Madeleine Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cer'on Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll L. Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. Gpt-4 technical report. 2023. URL https://api.semanticscholar.org/CorpusID:257532815.

Sebastian Bruch, Shuguang Han, Mike Bendersky, and Marc Najork. A stochastic treatment of learning to rank scoring functions. In *Proceedings of the 13th ACM International Conference on Web Search and Data Mining (WSDM 2020)*, pp. 61–69, 2020.

Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*, pp. 89–96, 2005.

Shijie Chen, Bernal Jimenez Gutierrez, and Yu Su. Attention in large language models yields efficient zero-shot re-rankers. In *International Conference on Learning Representations*, volume 2025, pp. 73047–73069, 2025.

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

Jing Jin, Yuhan Song, Wen Luo, and Houfeng Wang. Attention weights as an indicator: Analyzing and improving document utilization in retrieval-augmented generation. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 27034–27052, 2026.

Yuqing Li, Jiangnan Li, Mo Yu, Guoxuan Ding, Yanyu Chen, Zheng Lin, Wei Zhang, and Jie Zhou. Query-focused and memory-aware reranker for long context processing. *arXiv preprint arXiv:2602.12192*, 2026.

Yuping Lin, Zitao Li, Yue Xing, Pengfei He, Yingqian Cui, Yaliang Li, Bolin Ding, Jingren Zhou, and Jiliang Tang. Retrieval heads are dynamic. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15710–15729, 2026.

Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. *Advances in Neural Information Processing Systems*, 37:124198–124235, 2024.

Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? *Advances in neural information processing systems*, 32, 2019.

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

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

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

Revanth Gangi Reddy, JaeHyeok Doo, Yifei Xu, Md Arafat Sultan, Deevya Swain, Avirup Sil, and Heng Ji. First: Faster improved listwise reranking with single token decoding. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 8642–8652, 2024.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. Is chatgpt good at search? investigating large language models as re-ranking agents. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 14918–14937, 2023.

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

Linh Tran, Yulong Li, Radu Florian, and Wei Sun. Contrastive retrieval heads improve attention-based re-ranking. *arXiv preprint arXiv:2510.02219*, 2025.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pp. 5797–5808, 2019.

Juyuan Wang, Chenxing Wang, Yuchen Fang, Huiyun Hu, Junwu Du, Aolin Li, Shunlin Rong, Haijun Wu, Jin Xu, Ligang Liu, et al. Headrank: Decoding-free passage reranking via preference-aligned attention heads. *arXiv preprint arXiv:2604.17237*, 2026.

Wenhao Wu, Yizhong Wang, Guangxuan Xiao, Hao Peng, and Yao Fu. Retrieval head mechanistically explains long-context factuality. In *International Conference on Learning Representations*, volume 2025, pp. 62143–62156, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Wuwei Zhang, Fangcong Yin, Howard Yen, Danqi Chen, and Xi Ye. Query-focused retrieval heads improve long-context reasoning and re-ranking. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 23802–23816, 2025.

# A Additional Explanation of the Method

## A.1 Gradient Flow of our proposed objectives

**Gradient of the att-sft objective.** The overall att-sft loss is a weighted sum,

$$\mathcal{L}_{\text{att-sft}} \;=\; \lambda_1 \mathcal{L}_{\text{ntp}} + \lambda_2 \mathcal{L}_{\text{infonce}} + \lambda_3 \mathcal{L}_{\text{ranknet}}, \tag{19}$$

with $\lambda_1, \lambda_2, \lambda_3 \geq 0$. Here $\mathcal{L}_{\text{ntp}}$ keeps the model competitive as a generative ranker, whereas the two attention-side terms shape the heads in $\mathcal{H}^\star$. A key implementation point is that $s_i$ in Eq.10 and 11 is assembled through Eq.9 *only* from heads in $\mathcal{H}^\star$. Consequently, the gradients of $\mathcal{L}_{\text{infonce}}$

**Algorithm 1** Attention ranking training recipe.

**Require:** Base LLM $\pi_0$, dev set $\mathcal{S}$, thresholds $k, \tau$

1: **head selection:** On the dev set $\mathcal{S}$, use $\mathrm{Freq}(\mathrm{k}, \tau)$(Eq.8) to select attention heads and form $\mathcal{H}^\star$; the default scoring function is $F_{\mathrm{QR}}$.
2: **att-sft:** Fine-tune $\pi_0$ with Eq.13 to obtain $\pi_{\mathrm{sft}}$.
3: **att-dpo:** initialize $\pi_\theta \leftarrow \pi_{\mathrm{sft}}$ and $\pi_{\mathrm{ref}} \leftarrow \pi_{\mathrm{sft}}$; optimize Eq.18 using the perturbation-based pair $(\mathbf{p}_+, \mathbf{p}_-)$.
4: **return** final policy $\pi_\theta$.

and $\mathcal{L}_{\mathrm{ranknet}}$ reach the parameters of the query/key projections of the selected heads and leave the remaining heads essentially untouched. Empirically this locality is important: it sharpens the ranking-relevant heads while preserving the model's language-modeling ability, which the global $\mathcal{L}_{\mathrm{ntp}}$ term continues to maintain. Since the two attention-related loss terms are computed during the prefill stage, att-sft adds negligible overhead over plain SFT.

**Gradient of the att-dpo objective.** Writing $g \coloneqq \beta\,\Delta\mathrm{aNDCG}$, the gradient of the preference term is

$$\nabla_\theta\big[-\log\sigma(g)\big] = -\big(1-\sigma(g)\big)\,\beta\,\nabla_\theta\,\mathrm{approxNDCG}(\mathbf{p}_+), \tag{20}$$

where we used that $\mathbf{p}_-$ is a (stop-gradient) perturbation of $\mathbf{p}_+$ so that only the positive branch carries gradient. Two consequences follow. First, the factor $1-\sigma(g)$ automatically *down-weights already-satisfied pairs* (those with a large margin $g$) and focuses learning on hard pairs, mirroring the adaptive weighting of DPO. Second, all gradient signal flows through $\mathrm{approxNDCG}(\mathbf{p}_+)$ and hence, via Eqs.15 and3, through the selected heads only—so, as in att-sft, the update is localized to $\mathcal{H}^\star$ while the KL term keeps the distribution close to the att-sft reference.

### A.2 APPROXNDCG

We score a candidate distribution with the differentiable approxNDCG (Bruch et al., 2020), which admits gradients through the (otherwise non-differentiable) sort. Given the distribution $\mathbf{p}$ and gold relevances $r_i$, we form a soft rank and the corresponding metric,

$$\widehat{\mathrm{rank}}(i;\mathbf{p}) \;=\; 1+\sum_{j\neq i}\sigma\!\left(\frac{p_j-p_i}{\alpha}\right), \quad \mathrm{approxNDCG}(\mathbf{p}) = \frac{1}{Z}\sum_i \frac{2^{r_i}-1}{\log_2\!\big(1+\widehat{\mathrm{rank}}(i;\mathbf{p})\big)}, \tag{21}$$

where $Z$ is the ideal DCG (so that $\mathrm{approxNDCG}\in[0,1]$) and $\alpha>0$ is a temperature: as $\alpha\to 0$ the soft rank recovers the true integer rank, while larger $\alpha$ smooths it for stable gradients. We then define the preference margin

### A.3 PSEUDOCODE

Algorithm 1 summarizes the complete recipe.

## B EXPERIMENTAL DETAILS

### B.1 STATISTICS OF EVAL DATA

Table 4 summarizes the five BEIR evaluation datasets. All statistics are computed on the first-stage retrieval output that is fed to the reranker: for every query we concatenate its retrieved candidate passages and measure the total length in tokens using the Qwen3-4B tokenizer. Each query is reranked over a fixed pool of 100 retrieved candidates. The datasets span a wide range of scales, from $\sim$10,000 queries (Quora) down to only 50 (TREC-COVID), and, more importantly for attention-based ranking, they differ by more than an order of magnitude in concatenated candidate length: from a mean of $\sim$1,300 tokens on Quora (short, near-duplicate questions) to $\sim$21,900 tokens on FiQA (long financial passages). This spread in context length underlies the per-dataset behavior discussed in

§4.2: longer, information-dense inputs (FiQA, NQ, TREC-COVID) leave more headroom for head selection and attention-aware training, whereas the very short Quora is close to saturation for all attention-based methods.

Table 4: Statistics of the five BEIR evaluation datasets. *#Queries* is the number of evaluation queries and *#Docs* the number of retrieved candidates reranked per query. Token-length columns report the distribution of the concatenated candidate passages per query (Qwen3-4B tokenizer): minimum, mean, median, 75th/90th percentile, and maximum.

| Dataset | #Queries | #Docs | Min | Mean | Median | P75 | P90 | Max |
|---|---|---|---|---|---|---|---|---|
| NQ | 3,452 | 100 | 2,535 | 11,404 | 11,114 | 13,166 | 15,161 | 32,195 |
| HotpotQA | 7,405 | 100 | 2,369 | 7,435 | 7,336 | 8,287 | 9,224 | 16,720 |
| FiQA | 648 | 100 | 10,968 | 21,884 | 21,656 | 24,749 | 27,488 | 34,758 |
| TREC-COVID | 50 | 100 | 3,817 | 14,969 | 14,198 | 18,470 | 26,026 | 33,481 |
| Quora | 10,000 | 100 | 753 | 1,253 | 1,208 | 1,356 | 1,541 | 2,879 |

### B.2 Statistics of Training Data

Table 5 summaries the training data we used. In actual training, only the SFT data undergoes a full training cycle. During the RL phase, training is based solely on the number of steps, using only a portion of the data.

Table 5: Statistics of the training data. *#Queries* is the number of evaluation queries and *#Docs* the number of retrieved candidates reranked per query. *Source* is their identifier in huggingface.

| Dataset | Stage | #Queries | #Docs | Source |
|---|---|---|---|---|
| RankZephyr | SFT | 39713 | 2-20 | castorini/rank_zephyr_training_data |
| RankGPT | RL | 86909 | 20 | liuqi6777/RankGPT-msmarco-100k-clean |

### B.3 Hyperparameters

For attention-based ranking, we use the same prompt template as in previous works (Chen et al., 2025; Zhang et al., 2025; Tran et al., 2025), which is shown in figure 3. For generative rankers, we use prompt template as Sun et al. (2023), which is shown in figure 4.

```
{prompt_prefix} Here are some paragraphs:

[1] {Title 1 (if available)}
{Paragraph text 1}

[2] {Title 2 (if available)}
{Paragraph text 2}

...

Please find information that is relevant to the following query in
the paragraphs above.

Query:  {query}{prompt_suffix}
```

Figure 3: Prompt used for attention-based ranking.

For att-sft, we fine-tune both Qwen3-4B and Qwen3-8B for 3 epochs with the AdamW optimizer (fused implementation, $\beta$ and $\epsilon$ at their PyTorch defaults), using a peak learning rate of 5e-6 under

```
{prompt_prefix} You are RankLLM, an intelligent assistant that can
rank passages based on their relevancy to the query.
I will provide you with 20 passages, each indicated by a numerical
identifier [].
Rank the passages based on their relevance to the search query:
query.

[1] {Title 1 (if available)}
{Paragraph text 1}

[2] {Title 2 (if available)}
{Paragraph text 2}

...

Rank the 20 passages above based on their relevance to the search
query.  All the passages should be included and listed using
identifiers, in descending order of relevance.  The output format
should be [] > [], e.g., [2] > [1], Only respond with the ranking
results, do not say any word or explain.  {prompt_suffix}
```

Figure 4: Prompt used for generative rankers.

a cosine schedule with a 5% linear warmup, no weight decay, and gradient clipping at a maximum norm of 1.0. The per-device training batch size is fixed at 1. To keep the effective batch size comparable across model scales, we set gradient accumulation to 8 for the 4B model and 16 for the 8B model. All training is performed in bfloat16 with TF32 matmul enabled, and memory is further reduced via gradient checkpointing. Multi-GPU training is orchestrated through DeepSpeed ZeRO stage 3 with find_unused_parameters disabled and four data-loading workers per device. Unless otherwise noted, all experiments use a fixed random seed of 42 to ensure reproducibility.

For att-dpo, we also conduct optimization with AdamW optimizer (fused implementation), using a peak learning rate of 5e-7 under a cosine schedule with a 10% linear warmup, no weight decay, and gradient clipping at a maximum norm of 1.0. Following common DPO practice, we use a smaller learning rate than the SFT stage to prevent the policy from drifting too far from the reference. The per-device training batch size is 1 with 8 gradient accumulation steps. Training is performed in bfloat16 with TF32 matmul enabled. Memory is further reduced through non-reentrant gradient checkpointing. Multi-GPU training is orchestrated with DeepSpeed ZeRO stage 3, find_unused_parameters disabled, and two data-loading workers per device. All experiments use a fixed random seed of 42 to ensure reproducibility.

# C Correlation Study

## C.1 Other Model Architectures

Table 6 repeats the full comparison on Llama3-8B over NQ/HotpotQA/FiQA. The trends match those on the Qwen3 backbones: frequency-based selection is on par with or above average-based selection in the training-free regime, att-sft is the best single-stage method (63.6 avg), att-dpo beats the RL baseline HeadRank (62.0 vs. 60.9), and the combined att-sft + att-dpo is best overall (64.5 avg), with the top score on HotpotQA and FiQA. (On NQ the untrained Llama3 favors Vanilla SFT slightly, but our combination still wins the average by 1.8 points.) This confirms that the recipe is not specific to a single model family.

## C.2 Re-selecting Heads Each Epoch

Table 7 compares our default of fixing $\mathcal{H}^\star$ once before att-sft against re-running the frequency selection after each training epoch (*dynamic-e1/e2/e3*). Fixing the head set is best on both backbones (64.1/64.3 avg); every dynamic variant is 0.5–1.4 points lower and none recovers the fixed-set ac-

Table 6: Results on **Llama3.1-8B** (nDCG@10, %). **Bold** marks the best per column; *Avg.* is over the three datasets.

| Method | NQ | HotpotQA | FiQA | Avg. |
|---|---|---|---|---|
| *No training* | | | | |
| Retrieval | 51.6 | 62.9 | 36.9 | 50.5 |
| Generative | 51.6 | 63.0 | 36.9 | 50.5 |
| ICR | 50.3 | 72.6 | 30.2 | 51.0 |
| QR | 56.2 | 74.8 | 39.9 | 57.0 |
| CoRe | 57.0 | 74.8 | 40.2 | 57.3 |
| Freq-QR | 56.7 | 75.2 | 40.0 | 57.3 |
| Freq-CoRe | 57.1 | 75.0 | 40.1 | 57.4 |
| *SFT only* | | | | |
| Vanilla SFT | **70.0** | 74.5 | 43.6 | 62.7 |
| FIRST-style SFT | 68.3 | 74.8 | 44.0 | 62.4 |
| QRRanker | 69.0 | 75.3 | 44.7 | 63.0 |
| att-sft | 69.3 | 76.7 | 44.8 | 63.6 |
| *RL only* | | | | |
| HeadRank | 63.8 | 75.1 | 43.9 | 60.9 |
| att-dpo | 65.5 | 75.7 | 44.9 | 62.0 |
| *SFT + RL* | | | | |
| att-sft + att-dpo | 69.6 | **77.3** | **46.6** | **64.5** |

curacy, while also incurring the cost of an extra selection pass per epoch. The selected set is thus stable enough that periodic re-selection is unnecessary, and we keep a single fixed $\mathcal{H}^\star$ throughout.

Table 7: Fixed vs. per-epoch re-selected head sets for att-sft (nDCG@10, %). *dynamic-en* re-runs frequency selection after epoch $n$. **Bold** marks the best per column.

| | **Qwen3-4B** | | | | **Qwen3-8B** | | | |
|---|---|---|---|---|---|---|---|---|
| Setting | NQ | HotpotQA | FiQA | Avg. | NQ | HotpotQA | FiQA | Avg. |
| Fixed $\mathcal{H}^\star$ (ours) | **69.1** | 76.7 | **46.6** | **64.1** | **69.7** | 77.0 | **46.3** | **64.3** |
| dynamic-e1 | 68.3 | 76.8 | 44.9 | 63.3 | 68.4 | 76.8 | 44.4 | 63.2 |
| dynamic-e2 | 67.1 | 76.7 | 45.1 | 63.0 | 68.3 | 76.9 | 44.8 | 63.3 |
| dynamic-e3 | 67.5 | **76.9** | 44.3 | 62.9 | 68.7 | 77.0 | 45.1 | 63.6 |

### C.3 Inference Latency

Table 8 reports per-query latency (median p50 and tail p95, in ms) and compute (TFLOPs per query) for three inference modes on the same hardware. Attention-based ranking is dramatically faster than generative reranking: on Qwen3-4B it cuts median latency from 30,003 ms to 2,099 ms—a $14.3\times$ speedup—and on Qwen3-8B from 34,224 ms to 3,019 ms ($11.3\times$). Two properties drive this. First, attention ranking requires *no autoregressive decoding*: a single forward pass produces all document scores, removing the $O(N)$ token generation that dominates generative latency. Second, it can *early-exit*: once the attention maps at the selected heads $\mathcal{H}^\star$ (which lie in the lower-to-middle layers) are available, the remaining layers need not be computed, further reducing both latency and FLOPs. As a result attention ranking is also $1.4$–$1.5\times$ faster than the single-token FIRST-style decoding while using the fewest FLOPs of all three modes, and its p95 tail stays low, which matters for serving. Together with the accuracy results above, this shows our recipe improves attention ranking without sacrificing—indeed while preserving—its core efficiency advantage.

Table 8: Inference cost per query on the same hardware. *p50*/*p95* are median/95th-percentile latency (ms, lower is better); *TFLOPs* is compute per query. Attention ranking avoids decoding and early-exits at the selected heads.

| | **Qwen3-4B** | | | **Qwen3-8B** | | |
|---|---|---|---|---|---|---|
| Method | p50 (ms) | p95 (ms) | TFLOPs | p50 (ms) | p95 (ms) | TFLOPs |
| Generative | 30,003 | 31,066 | 180.14 | 34,224 | 35,701 | 359.82 |
| FIRST-style | 3,103 | 4,007 | 171.05 | 4,383 | 5,606 | 339.11 |
| Attention (ours) | **2,099** | **2,991** | **170.79** | **3,019** | **4,168** | **293.29** |