AcuRank: Uncertainty-Aware Adaptive Computation for Listwise Reranking

{Soyoung Yoon♣*, Gyuwan Kim♠*}, Gyu-Hwung Cho♣, Seung-won Hwang♣
♣Seoul National University ♠University of California, Santa Barbara
*Equal contribution (author order is randomly determined via coin toss)

SEOUL NATIONAL UNIVERSITY UC SANTA BARBARA

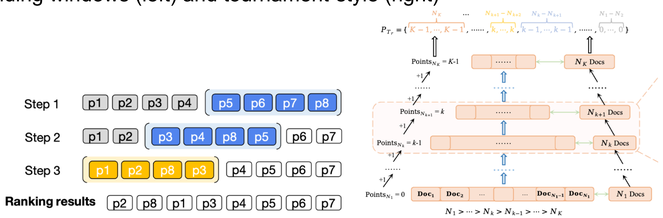
NEURAL INFORMATION PROCESSING SYSTEMS

Background and Motivation

- Modern information retrieval pipelines, such as RAG systems, use fast first-stage retrievers (e.g., BM25 or dense encoders) mainly for recall and speed, but their outputs are often noisy, making reranking crucial to obtain accurate and reliable document ordering.
- Recent work has explored neural rerankers that better capture document interactions, with LLM-based listwise rerankers achieving strong performance while incurring high computational cost due to input length limits that require multiple calls to cover all candidates.
- Fixed strategies like sliding windows or tournament reranking improve efficiency but lack adaptivity to query difficulty or uncertainty, motivating an uncertainty-aware framework that dynamically allocates computation.

Listwise Reranking

- Problem formulation
  - Given a query q and a set of document candidates $\mathscr{D} = \{D_1, \ldots, D_n\}$, the goal is to extract a top-k list $[D_{r_1} > \ldots > D_{r_k}]$
  - A listwise reranker $g(\mathscr{D}'; \mathscr{M})$, where $\mathscr{D}'$ is an ordered subset of $\mathscr{D}$ and $\mathscr{M}$ is an underlying reranking model (e.g., an instruction-tuned or zero-shot LLM), and the reranker returns a new ordering over the documents in $\mathscr{D}'$.
  - In practice, n is often large (e.g., 100 - 1000), making it infeasible to rerank all retrieved documents at once due to input sequence length constraints. Instead, g is applied to smaller subsets with $|\mathscr{D}'| = m \ll n$, and the final ranking is approximated by aggregating local reranking results across multiple batches.
- Fixed-computation baselines
  - Sliding windows (left) and tournament-style (right)



TrueSkill-based Relevance Modeling

- TrueSkill is a principled Bayesian ranking system originally developed for multiplayer games. We recast documents as players, and when one document is ranked above another, it is treated as having won that match.
- We model the latent relevance of each document D_i as a Gaussian variable $x_i \sim \mathcal{N}(\mu_i, \sigma_i^2 + \beta^2)$, where μ_i represents the estimated relevance, σ_i captures epistemic uncertainty, and β represents a global parameter for observation noise.

Ranking Uncertainty Estimation

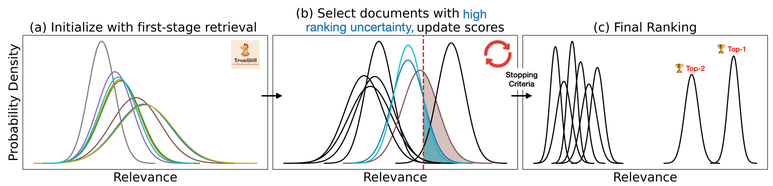
- The rank of D_i is the number of documents more relevant than D_i under their latent scores. Since each x_i is Gaussian, the exact probability that D_i has rank r can be computed via dynamic programming, but the computation becomes costly and numerically unstable as n grows, due to the need to compute dense integrals and multiply small probabilities.
- To improve scalability, we adopt a more efficient approximation. We define a threshold $t(r)$ such that the expected number of documents whose relevance exceeds $t(r)$ equals the target rank r. Based on the monotonicity of $t(r)$, we efficiently compute $t(r)$ via binary search. We then approximate the cumulative rank probability as $\mathbb{P}(r_i \leq r) \approx \mathbb{P}(x_i > t(r))$, which reflects the chance that x_i exceeds the estimated top-r threshold.

AcuRank Framework

Algorithm 1 AcuRank: Uncertainty-Aware Adaptive Computation for Listwise Reranking

Input: Query q, retrieved documents $\mathcal{D} = \{D_1, \ldots, D_n\}$, listwise reranker g, target rank cutoff k
Output: Ranked list $[D_{r_1} > \cdots > D_{r_n}]$, with top-$k$ used downstream

```
Initialize TrueSkill-based relevance scores (μ_i, σ_i) for all D_i ∈ D
repeat
  Select candidate documents C ⊂ D with high ranking uncertainty
  Partition C into ordered groups {B_1, ..., B_b}
  Apply listwise reranker g to each B_j and update TrueSkill scores accordingly
until |C| is small, top-k converges, or computational budget is exhausted
```



Experimental Setup

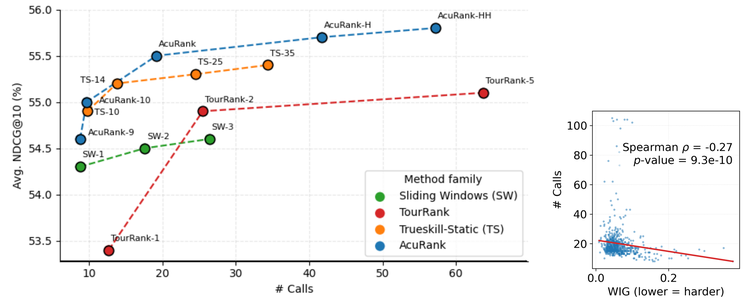
- Datasets
  - TREC-DL: DL19, DL20, DL21, DL22, DL23, and DL-Hard
  - BEIR: TREC-COVID, NFCorpus, Signal-1M, News, Robust04, Touché, DBPedia, and SciFact
- Evaluation metrics
  - Ranking accuracy: Normalized Discounted Cumulative Gain (NDCG@10)
  - Efficiency: the number of of reranker calls per query
- Retrievers: BM25 top-100/1000, SPLADE++ED top-100, Contriever top-100
- Rerankers ($m = 20$): RankZephyr, RankGPT (*gpt-4.1-mini*), RankVicuna-7B, Llama-3.3-70B-Instruct

Experimental Setup (cont')

- AcuRank configurations
  - Initialize TrueSkill scores as the mean μ_i to the raw first-stage retrieval score and the standard deviation to $\sigma_i = \mu_i/3$.
  - We select documents whose rank probability $s_i = \mathbb{P}(x_i > t(k))$ falls within the range $(\epsilon, 1-\epsilon)$ with $\epsilon = 0.01$ and $k = 10$ as default
  - If the number of uncertain documents exceeds the reranker capacity $m = 20$, we divide them into equally sized groups using sequential partitioning.
  - We terminate reranking when the number of uncertain documents falls below $\tau = 10$, or the reranker call budget is exhausted.

Experimental Results

- AcuRank consistently lies along the Pareto frontier, achieving stronger accuracy at a given budget or using fewer calls to reach the same target compared to fixed-computation baselines.



- AcuRank maintains strong performance across varying retrieval settings with different first-stage retrievers and reranker model.
- Each component meaningfully contributes to the effectiveness and the default configuration offers a strong balance between accuracy and efficiency.

Init	Partitioning	Stopping Criterion	TREC	BEIR	All	# Calls
✓	-	-	**59.1**	**52.8**	**55.5**	19.7
×	-	-	59.0	51.7	54.8	**13.4**
✓	Random	-	58.8	52.7	55.3	22.6
✓	-	Top-k stability	58.8	52.4	55.2	22.7

Conclusion

- We propose **AcuRank**, a novel listwise reranking framework that performs adaptive computation guided by uncertainty-aware relevance modeling. Our approach maintains probabilistic relevance estimates using TrueSkill and selectively allocates reranking effort to documents with high ranking uncertainty.
- By focusing computation on ambiguous candidates, AcuRank consistently outperforms fixed-computation baselines and achieve better accuracy-efficiency trade-offs across various retrieval scenarios (with varying reranker models and first-stage retrieval sizes) on different benchmark datasets (TREC-DL and BEIR).