PyTerrier-GenRank: The PyTerrier Plugin for Reranking with Large Language Models

Kaustubh D. Dhole
Department of Computer Science
Emory University
Atlanta, GA – USA
kdhole@emory.edu

Abstract

Using LLMs as rerankers requires experimenting with various hyperparameters, such as prompt formats, model choice, and reformulation strategies. We introduce **PyTerrier-GenRank**, a PyTerrier plugin to facilitate seamless reranking experiments with LLMs, supporting popular ranking strategies like pointwise and listwise prompting. Among 8B instruct-tuned zero-shot approaches, we find LLama-Spark outperforms others.

1 Introduction

Generative reranking methods have recently gained popularity for their ability to produce a more optimal ranking of documents by prompting large language models. These methods are also employed in Retrieval Augmented Generation (Dhole, 2024a) paradigms as they are effective for enhancing generation through the incorporation of retrieved information. Typically, a sparse retriever such as BM25 (Robertson et al., 1992) or a dense retriever like ColBERT (Khattab and Zaharia, 2020) is utilized to retrieve a small subset (ranging from 20 to 1000) from among millions or billions of text pieces. This subset is then forwarded to a second-stage generative reranker for reranking.

The increasing adoption of such reranking methods necessitates rigorous testing of various hyperparameters, including employing different prompts, evaluating performance with different model weights, and using it in conjunction with various retrieval, reranking, and reformulation (Dhole et al., 2024a,b) pipelines.

In that regard, we introduce – **PyTerrier-GenRank** – to encourage rapid experimentation of retrieval pipelines employing recently popular LLM-based reranking methods. The code includes PyTerrier (Macdonald and Tonellotto, 2020) wrappers over many helper functions of the RankLLM repository (Pradeep et al., 2023a,b). Our plugin can be quickly installed in any Python environment and is available at this link[1].

```python
import pyterrier as pt
from rerank import LLMReRanker

bm25 = pt.BatchRetrieve(dataset.
    get_index(), wmodel="BM25")

reranker = LLMReRanker("castorini/
    rank_vicuna_7b_v1")

pipeline = bm25 % 100
    >> pt.text.get_text(index, 'text')
    >> reranker

pipeline.search('Indian restaurants')
```

Listing 1: Retrieving using a listwise reranker

2 Generative Reranking

Large Language Models (LLMs) have been widely used for reranking tasks in various ways involving novel hyperparameters. Among the most popular ones have been ranking documents in a pointwise, pairwise and listwise fashion. MonoBERT (Nogueira et al., 2019) and MonoT5 (Nogueira et al., 2020) are among the earliest pointwise ranking methods which were trained to emit a single scalar relevance score for a query-document pair. In contrast, pairwise approaches (Qin et al., 2024) involved prompting models with a query and two documents to compare and rank them. `RankVicuna` (Pradeep et al., 2023a), `RankZephyr` (Pradeep et al., 2023b), and `RankGPT` (Sun et al., 2023; OpenAI, 2023) have explored listwise ranking strategies by prompting their respective models with a list of documents and generating a ranked list of document IDs. These paradigms have also been investigated in the context of recommender systems (Yue et al., 2023).

The number of hyperparameters involved in creating an optimal retriever, the choice of prompts,

[1] github.com/emory-irlab/pyterrier_genrank

the number of documents, the weight of the reformulation strategy, etc., all influence the final ranking set in various ways. Besides, the increasing number of LLMs released by various organizations further necessitates rigorous testing to evaluate performance across different settings. To ensure the validity and reproducibility of results, it is imperative that researchers and developers can easily replicate experiments across these novel sets of hyperparameters.

3 Python IR Toolkits

We choose PyTerrier (Macdonald and Tonellotto, 2020) due to its modularity and Python-friendly interface. It is built on the Terrier search engine and excels in modular, rapid experimentation for IR research, allowing users to easily integrate and evaluate various retrieval and re-ranking models. It has been employed in various studies to evaluate retrieval effectiveness (Dhole and Agichtein, 2024; Datta et al., 2024; Parry et al., 2024; Datta et al., 2022; Dhole, 2024b).

Training	Name	nDCG@10
0-shot	BM25	.480
0-shot	gpt-35-turbo$^{\beta}$ (OpenAI, 2023)	.660
0-shot	gpt-4-turbo-0409$^{\beta}$ (OpenAI, 2024a)	.701
0-shot	gpt-4o-mini (OpenAI, 2024b)	**.710**
0-shot	Llama-3-8B-Instruct (Meta, 2024a)	.572
0-shot	Llama-3.1-8B-Instruct (Meta, 2024b)	.594
0-shot	Llama-Spark (8B) (Arcee, 2024)	**.612**
Trained	Rank-Vicuna$^{\beta}$ (Pradeep et al., 2023a)	.672
Trained	Rank-Zephyr$^{\beta}$ (Pradeep et al., 2023b)	.711

Table 1: Comparing reranking performance of different models for TREC-DL 2019 (Craswell et al., 2020) using PyTerrier_Genrank

4 Generative Reranking Wrappers

Our plugin employs PyTerrier wrappers over reranking class functions modified from the Pyserini (Lin et al., 2021) based RankGPT and RankLLM repositories. We allow additional hyperparameters to be expressed from function calls.

5 Sanity Check

As a sanity check, we re-evaluate some previously published rerankers like RankVicuna, RankGPT, using this framework. Those marked with superscript $^{\beta}$ have already been previously tested with the Pyserini framework in their respective papers. We also evaluate the zero-shot retrieval effectiveness of newly published instruction-tuned models like LLama-Spark.

We find that among the 8B zero-shot approaches, LLama-Spark is the most effective as a reranker. The complete results are shown in Table 1.

Ethics Statement

Large Language Models are embedded within a broader socio-technical framework (Dhole, 2023; Dhole et al., 2023). When deploying them for tasks such as ranking and recommendations, it's crucial to assess their long-term impacts, including the potential for creating echo chambers and the risk of generating biased suggestions.

References

Arcee. 2024. Llama spark model card.

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*.

Suchana Datta, Debasis Ganguly, Sean MacAvaney, and Derek Greene. 2024. A deep learning approach for selective relevance feedback. In *European Conference on Information Retrieval*, pages 189–204. Springer.

Suchana Datta, Sean MacAvaney, Debasis Ganguly, and Derek Greene. 2022. A 'pointwise-query, listwise-document' based query performance prediction approach. In *SIGIR*, pages 2148–2153. ACM.

Kaustubh Dhole. 2023. Large language models as SocioTechnical systems. In *Proceedings of the Big Picture Workshop*, pages 66–79, Singapore. Association for Computational Linguistics.

Kaustubh Dhole. 2024a. KAUCUS - knowledgeable user simulators for training large language models. In *Proceedings of the 1st Workshop on Simulating Conversational Intelligence in Chat (SCI-CHAT 2024)*, pages 53–65, St. Julians, Malta. Association for Computational Linguistics.

Kaustubh Dhole. 2024b. Kaucus-knowledgeable user simulators for training large language models. In *Proceedings of the 1st Workshop on Simulating Conversational Intelligence in Chat (SCI-CHAT 2024)*, pages 53–65.

Kaustubh Dhole, Shivam Bajaj, Ramraj Chandradevan, and Eugene Agichtein. 2024a. QueryExplorer: An interactive query generation assistant for search and exploration. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 3: System Demonstrations)*, pages 107–115, Mexico City, Mexico. Association for Computational Linguistics.

Kaustubh Dhole, Varun Gangal, Sebastian Gehrmann, Aadesh Gupta, Zhenhao Li, Saad Mahamood, Abinaya Mahadiran, Simon Mille, Ashish Shrivastava, Samson Tan, et al. 2023. Nl-augmenter: A framework for task-sensitive natural language augmentation. *Northern European Journal of Language Technology*, 9(1).

Kaustubh D Dhole and Eugene Agichtein. 2024. Genqrensemble: Zero-shot llm ensemble prompting for generative query reformulation. In *European Conference on Information Retrieval*, pages 326–335. Springer.

Kaustubh D Dhole, Ramraj Chandradevan, and Eugene Agichtein. 2024b. Generative query reformulation using ensemble prompting, document fusion, and relevance feedback. *arXiv preprint arXiv:2405.17658*.

Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 39–48.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2356–2362.

Craig Macdonald and Nicola Tonellotto. 2020. Declarative experimentation ininformation retrieval using pyterrier. In *Proceedings of ICTIR 2020*.

Meta. 2024a. Llama 3 model card.

Meta. 2024b. Llama 3.1 model card.

Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. 2020. Document ranking with a pre-trained sequence-to-sequence model. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 708–718, Online. Association for Computational Linguistics.

Rodrigo Nogueira, Wei Yang, Kyunghyun Cho, and Jimmy Lin. 2019. Multi-stage document ranking with bert. *arXiv preprint arXiv:1910.14424*.

OpenAI. 2023. Gpt-3.5.

OpenAI. 2024a. GPT-4-turbo-0409.

OpenAI. 2024b. GPT-4o mini.

Andrew Parry, Sean MacAvaney, and Debasis Ganguly. 2024. Top-down partitioning for efficient list-wise ranking. *arXiv preprint arXiv:2405.14589*.

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*.

Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Le Yan, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, Xuanhui Wang, and Michael Bendersky. 2024. Large language models are effective text rankers with pairwise ranking prompting. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1504–1518, Mexico City, Mexico. Association for Computational Linguistics.

Stephen E Robertson, Steve Walker, Micheline Hancock-Beaulieu, Aarron Gull, and Marianna Lau. 1992. Okapi at trec. In *Text retrieval conference*, pages 21–30.

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. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14918–14937.

Zhenrui Yue, Sara Rabhi, Gabriel de Souza Pereira Moreira, Dong Wang, and Even Oldridge. 2023. Llamarec: Two-stage recommendation using large language models for ranking. *arXiv preprint arXiv:2311.02089*.