NAVERLOO @ TREC DEEP LEARNING AND NEUCLIR 2023: AS EASY AS ZERO, ONE, TWO, THREE — CASCADING DUAL ENCODERS, MONO, DUO, AND LISTO FOR AD-HOC RETRIEVAL

Carlos Lassance[1], **Ronak Pradeep**[2], **Jimmy Lin**[2]
[1]Naver Labs Europe, [2] University of Waterloo

ABSTRACT

In this notebook, we outline the architecture and evaluation of our TREC 2023 submissions, which employ a sophisticated cascading multi-stage ranking framework comprising four distinct steps. Through experimentation across multiple configurations, we validate the efficacy of each stage within this hierarchy. Our findings demonstrate the high effectiveness of our pipeline, consistently outperforming median benchmarks and approaching the maximal aggregate scores. Notably, reproducibility is a key outcome of our methodology. Nevertheless, the reproducibility of the final component, termed "listo", is contingent upon interactions with the proprietary and inherently non-deterministic GPT$_4$, raising salient questions about its consistency and reliability in a research context.

1 Introduction

In this short notebook, we detail our TREC 2023 Deep Learning and NeuCLIR track submissions, based on a cascading retrieval with 4 steps. We refrain from additional fine-tuning, relying solely on off-the-shelf models. For an unvarnished understanding of the methods employed, we direct readers to the foundational articles of each system leveraged following its citation.

2 Methodology

In the following, we introduce the models we consider for both candidate generation as well as reranking and then detail our submitted runs

2.1 First Stage Retrieval (Dual Encoders)

We explore the impact of varying the initial retrieval stage within the context of the downstream effect on the mono rerankers and the subsequent pipeline, evaluating four core methods:

- Traditional Unsupervised Sparse Retrieval (with BM25)
- Learned Sparse Family (e.g., SPLADE)
- Learned Dense Family (e.g., AggRetriever)
- Brute Force (i.e., ensemble all the *good* first-stage methods that we could get our hands on)

Methods used for the DL Track

- BM25 with default Anserini parameters

- BM25 + doc2query–T5 [Nogueira and Lin, 2019, Ma et al., 2022] with default Anserini parameters
- SPLADE++ Self and Ensemble-Distil [Lassance and Clinchant, 2023a]
- AggRetriever [Lin et al., 2023]
- SLIM [Li et al., 2023] with SPLADE++ without retraining

Methods used for the NeuCLIR Track — Search in the original language corpora with translated queries (QT)

- BM25 with default Anserini parameters
- SPLADE-qt from NeuCLIR-22 [Lassance and Clinchant, 2023b]
- SPLADE-qt from NLE-MIRACL [Lassance, 2023]
- mContriever from NLE-MIRACL [Lassance, 2023]

Methods used for the NeuCLIR Track — Search in the English-translated corpora (DT)

- BM25 with default Anserini parameters
- RetroMAE reproduction on NLE-MIRACL [Lassance, 2023]
- SPLADE trained for NLE-MIRACL [Lassance, 2023]
- SPLADE++ SelfDistil [Lassance and Clinchant, 2023a]

2.2 Second Stage Ranker (*mono*)

For the second stage, we used a mix of pretrained cross encoders to rerank the top 1k passages retrieved from the first stage. In the mono framework, the cross-encoder model is presented with a query and a document, often concatenated with delimiters, and outputs a relevance score.

TREC DL mono rerankers: All mono rankers used for the TREC DL Track come from our TREC submissions from 2022 [Lassance and Clinchant, 2023a] and were trained in a consistent manner, varying only the pretrained model. The following methods form the *mono* stage that we tested and are accessible via the corresponding HuggingFace pointer:

- ALBERT-xxlarge: `naver/trecdl22-crossencoder-albert`
- DeBERTa-v2: `naver/trecdl22-crossencoder-debertav2`
- DeBERTa-v3: `naver/trecdl22-crossencoder-debertav3`
- ELECTRA-large: `naver/trecdl22-crossencoder-electra`
- RankT5-3B: `naver/trecdl22-crossencoder-rankT53b-repro`

TREC NeuuCLIR mono rerankers:

- Unicamp's MonoT5 MT [Jeronymo et al., 2023]: Most effective reranker in NeuCLIR 2022, evaluated in the translated queries setting (QT) and can be accessed with the pointer `unicamp-dl/mt5-13b-mmarco-100k`
- RankT5-MSMARCO: evaluated in the translated documents setting (DT), can be accessed with the pointer `naver/trecdl22-crossencoder-rankT53b-repro`
- RankT5-MIRACL: The most effective reranker in MIRACL [Lassance, 2023], it is the UniCAMP MonoT5 finetuned on the MIRACL dataset, and it is not available online yet
- DeBERTa-v3: `naver/trecdl22-crossencoder-debertav3`
- MonoT5 [Nogueira et al., 2020, Pradeep et al., 2021]: `castorini/monot5-3b-msmarco-10k`

2.3 Third Stage (*duo*)

For the third stage, we use three models within the *duo* paradigm. To recall, the duo framework receives a query and two documents and returns a score indicating which document is preferred. We run the models in this stage over all possible pairs of the top 50 retrieved by the previous stage and aggregate the scores. Finally, we ensemble the results. Note that costs here are quadratical with the number of candidates, growing quickly. We use an ensemble of the following three duo rerankers:

- DuoT5 [Pradeep et al., 2020]:
- PRP-FlanT5 [Qin et al., 2023]:
- PRP-FlanUL2 [Qin et al., 2023]:

This stage of reranking is only employed in the TREC Deep Learning 2023 Track.

2.4 Fourth Stage (*listo*)

Finally, for the final stage, we employ the listwise reranker RankGPT [Sun et al., 2023] with the large language model GPT_4, to rerank the top 30 results from the preceding phase. This paradigm of zero-shot reranking, dubbed "prompt decoders", has shown increasing adoption and effectiveness in recent months [Pradeep et al., 2023]. Note that the costs here grow with the number of tokens, so each added document would increase the costs (i.e., even removing only 20 documents, from the duo stage, helps in keeping costs reasonable).

The GPT_4 model, theorized to use Mixture-Of-Experts models, comes with the issue of non-determinism and subsequently, questions some of the reproducibility of our results [Pradeep et al., 2023]. Recent work like RankVicuna have distilled open-source models to help deal with these concerns and we hope to evaluate their effectiveness on these test collections.

Note that our method is applied over the translated corpora (DT) in the NeuCLIR Track as the GPT family of models is predominantly trained in the English language. We hope to explore the effects of using the original without translation in future work.

2.5 Ensembling

We also applied ensembling at each of the stages, to improve the effectiveness, locally as well as the subsequent global (downstream) boost. We utilized the `ranx` framework [Bassani and Romelli, 2022] to generate all our ensembles, using average normalized score over the ensembles, unless explicitly noted. The normalized score uses the min and max values of the query so that, for each model, the highest score is 1 and the lowest is 0.

3 TREC DL 2023 Track

For the TREC DL Track, we submit runs for both the passage and document tasks. Results for the passage task are found in Table 3 and for the document task in Table 3.

Passage ranking task: The first thing we notice on the passage ranking task is the decrease in the effectiveness of first-stage dual encoders. While in TREC DL19 and 20 they were mostly capable of standing their ground against the stronger/more expensive steps (with a 0.7+ nDCG@10, while the most effective rerankers reaching 0.8 nDCG@10), in TREC DL23 they are far below subsequent stages. This drop and widening of the gaps is unlikely due to their training confined to MSMARCO v1, as this limitation applies to other methods as well. The reasons behind this phenomenon warrant further investigation.

We also confirm that using first-stage retrievers improves over BM25, even in post-reranking comparisons. Moreover, while ensembling all retrievers yields improvements, the marginal gains shrink in comparison to integrating a single effective trained sparse method like SPLADE.

The incremental benefits of subsequent stages—uno, duo, and listo—are evident. RankGPT seems to be so effective, that it does not seem to have any benefits from ensembling with the results from

Description	Run Name	nDCG@10	MAP	MRR	P@10	Recall@100
	First Stage (Dual Encoders)					
BM25	N/A	0.2569	0.0781	0.3872	0.1610	0.2319
DocT5	N/A	0.3151	0.1084	0.5323	0.2183	0.3007
AggRetriever	`agg-cocondenser`	0.4562	0.1776	0.6562	0.3561	0.3714
BM25 + SPLADE Ensemble	`bm25_splades`	0.4590	0.1886	0.6586	0.3415	0.4326
SLIM unrefined	`slim-pp-0shot-uw`	0.4762	0.1773	0.6732	0.3634	0.3835
SPLADE++ ED	`splade_pp_ensemble_distil`	0.4730	0.1924	0.6832	0.3549	0.4137
SPLADE++ SD	`splade_pp_self_distil`	0.4768	0.1960	0.6941	0.3671	0.4139
All First Stage Ensemble	`fs`	0.5045	0.2116	0.7142	0.3841	0.4528
	Second Stage (mono)					
ALBERT-xxlarge over `fs`	N/A	0.5599	0.2629	0.8158	0.4305	0.5115
RankT5-3B over `fs`	N/A	0.5799	0.2721	0.8266	0.4671	0.5191
Deberta-v3 over `fs`	N/A	0.5799	0.2687	0.7814	0.4659	0.5246
ELECTRA over `fs`	N/A	0.5865	0.2702	0.8198	0.4598	0.5198
DeBERTa-v2 over `fs`	N/A	0.5890	0.2640	0.8040	0.4780	0.5064
Ensemble uno over BM25	`bm25_RR`	0.5377	0.2070	0.7364	0.4098	0.3740
Ensemble uno over BM25 + SPLADE	`bm25_splades_RR`	0.5891	0.2811	0.7904	0.4561	0.5331
Ensemble uno over `fs`	`fs_RR`	0.5972	0.2844	0.7960	0.4659	0.5433
	Third Stage (duo)					
FLANT5-3B over `fs_RR`	N/A	0.5689	-	0.7276	-	0.5433
FLANT5-UL over `fs_RR`	N/A	0.5943	-	0.7698	-	0.5433
DuoT5-3B over `fs_RR`	N/A	0.6107	-	0.8426	-	0.5433
Ensemble duo	`fs_RR_duo`	0.6584	0.3130	0.8713	0.5476	0.5433
	Fourth Stage (listo)					
`fs_RR_duo` + RankGPT	`frgpt4`	0.6899	0.3300	0.9029	0.5780	0.5433
RankGPT	`rgpt4`	0.6994	0.3382	0.8835	0.5927	0.5433
	`median`	0.5329	0.2159	0.7803	0.4085	-
	`max`	0.7892	0.3839	0.9939	0.7000	-

Table 1: Results on the Passage Ranking Task of the TREC 2023 Deep Learning Track.

its prior stage, in fact, if anything, we note a general decrease in terms of the effectiveness according to nDCG@10 and MAP.

One interesting question that is raised is how to either reduce the gaps or improve the listwise ranker, which in this case is a model whose size, training data, and training methodology are unknown. Compared to the median and max runs, our results mostly align with what we saw last year, with our most effective runs approaching the max setting.

Run Name	nDCG@10	MAP	MRR	P@10	Recall@100
`bm25_splades`	0.5322	0.2976	0.7905	0.3963	0.5702
`bm25_RR`	0.6061	0.3131	0.8588	0.4634	0.5156
`bm_splade_RR`	0.6355	0.3793	0.8732	0.4915	0.6452
`frgpt4`	0.7226	0.4189	0.9489	0.5829	0.6569
`median`	0.6123	0.3291	0.8637	0.4634	-
`max`	0.7510	0.4414	0.9878	0.6098	-

Table 2: Results on the Document Ranking Task of the TREC 2023 Deep Learning Track.

Document ranking task: For document ranking, we adapt our passage task strategies, employing MaxP aggregation to transition from passage to document ranking. Our methods uphold their effectiveness, still following the incremental benefits from each subsequent stage of the pipeline. Our most effective run is pretty close to the max setting, which we partly attribute to the lower participation volume of the track, as evident from prior years.

4 TREC NeuCLIR 2023 Track

In the NeuCLIR track, our methods were tested across each language and were merged for the multilingual retrieval task. Although the NeuCLIR findings are preliminary, they generally correspond to our predicted hierarchy of effectiveness, with RankGPT integration yielding improvements, albeit less pronounced than those observed in TREC DL.

Run Name	nDCG*	nDCG@20	MAP	RBP	Recall@1K
Dual Encoders					
A1PND_SpladeMiraclMonoqt	0.3718	0.3915	0.1775	0.3012	0.5454
A1PND_mContrieverqt	0.3875	0.3786	0.2131	0.2495	0.5314
A1PNS_bm25qt	0.4570	0.4529	0.2482	0.3281	0.6668
A1PNL_spladeqt	0.5187	0.5595	0.3045	0.4069	0.7023
A1NETSP_BM25s	0.5459	0.5499	0.3338	0.3915	0.7070
AETS_bm25dt	0.5572	0.5818	0.3505	0.4028	0.7046
AETL_spladedt	0.5752	0.5813	0.3214	0.4267	0.7792
AETD_RetroMAEReprodt	0.5753	0.5895	0.3437	0.3966	0.7317
AETD_SpladeMiraclENdt	0.5815	0.5865	0.3448	0.4047	0.7175
A1NETHP_BM25sSplades	0.6336	0.6174	0.4000	0.4344	0.8090
A1NETHP_EverythingRun	0.6463	0.6395	0.4095	0.4473	0.8239
Mono					
A_RERANKBM25s	0.6140	0.6999	0.4337	0.5035	0.7070
A_RERANKBM25sSplades	0.6851	0.7260	0.4773	0.5273	0.8090
A_RERANKEverythingRun	0.6910	0.7148	0.4780	0.5191	0.8239
Listo					
A_frgpt4	0.7015	0.7316	0.4946	0.5319	0.8239
A_rgpt4	0.7020	0.7339	0.4970	0.5343	0.8239
Median	0.5719	-	0.3478	-	-
Max	0.7693	-	0.5800	-	-

Table 3: Results on the Persian Single-Language News Retrieval Task of the TREC 2023 NeuCLIR Track.

Run Name	nDCG*	nDCG@20	MAP	RBP	Recall@1K
Dual Encoders					
A1NETHR_BM25sSplades	0.6481	0.5286	0.3753	0.4094	0.8907
A1NETHR_EverythingRun	0.6626	0.5362	0.3792	0.4182	0.9216
A1NETSR_BM25s	0.6057	0.4735	0.3336	0.3690	0.8560
A1RND_SpladeMiraclMonoqt	0.4540	0.3381	0.1966	0.2672	0.6921
A1RND_mContrieverqt	0.5406	0.4484	0.2611	0.3431	0.7494
A1RNL_spladeqt	0.5175	0.4076	0.2633	0.3143	0.7348
A1RNS_bm25qt	0.5629	0.4393	0.2973	0.3423	0.8075
AETD_RetroMAEReprodt	0.5776	0.4738	0.2996	0.3646	0.7841
AETD_SpladeMiraclENdt	0.5968	0.4786	0.3111	0.3702	0.8404
AETL_spladedt	0.6094	0.5002	0.3324	0.3889	0.8359
AETS_bm25dt	0.6001	0.4732	0.3231	0.3667	0.8481
Mono					
A_RERANKBM25s	0.6887	0.5990	0.4343	0.4665	0.8560
A_RERANKBM25sSplades	0.7083	0.6057	0.4425	0.4707	0.8907
A_RERANKEverythingRun	0.7185	0.6041	0.4465	0.4699	0.9216
Listo					
A_frgpt4	0.7167	0.6204	0.4471	0.4860	0.9216
A_rgpt4	0.7182	0.6288	0.4506	0.4945	0.9216
Median	0.6022	-	0.3169	-	-
Max	0.8007	-	0.5535	-	-

Table 4: Results on the Russian Single-Language News Retrieval Task of the TREC 2023 NeuCLIR Track.

Run Name	nDCG*	nDCG@20	MAP	RBP	Recall@1K
Dual Encoders					
A1CNS_bm25qt	0.2208	0.2066	0.0790	0.1511	0.3863
A1CND_SpladeMiraclMonoqt	0.3439	0.2850	0.1609	0.1926	0.5389
A1CNL_spladeqt	0.4116	0.3565	0.2058	0.2412	0.6557
A1CND_mContrieverqt	0.4217	0.3436	0.1771	0.2458	0.7089
A1NETSC_BM25s	0.4386	0.3183	0.1840	0.2368	0.7806
AETD_RetroMAEReprodt	0.4736	0.4017	0.2238	0.2818	0.7413
AETD_SpladeMiraclENdt	0.5112	0.4385	0.2534	0.3205	0.7631
AETL_spladedt	0.5143	0.4623	0.2586	0.3305	0.7968
AETS_bm25dt	0.5163	0.4404	0.2673	0.3216	0.7976
A1NETHC_BM25sSplades	0.5512	0.4731	0.2908	0.3297	0.8507
A1NETHC_EverythingRun	0.5718	0.4823	0.3000	0.3405	0.8715
Mono					
A_RERANKBM25s	0.5990	0.5869	0.3714	0.4201	0.7806
A_RERANKBM25sSplades	0.6344	0.5996	0.3883	0.4305	0.8507
A_RERANKEverythingRun	0.6409	0.5989	0.3907	0.4295	0.8715
Listo					
A_frgpt4	0.6611	0.6302	0.4196	0.4499	0.8715
A_rgpt4	0.6715	0.6393	0.4331	0.4586	0.8715
Median	0.5035	-	0.2427	-	-
Max	0.7422	-	0.5142	-	-

Table 5: Results on the Chinese Single-Language News Retrieval Task of the TREC 2023 NeuCLIR Track.

Run Name	nDCG*	nDCG@20	MAP	RBP	Recall@1K
Dual Encoders					
A_BM25s	0.5184	0.3926	0.2298	0.3185	0.7375
A_BM25sSplades	0.6112	0.4994	0.3136	0.3959	0.8061
A_EverythingRun	0.6290	0.5175	0.3223	0.4079	0.8248
Mono					
A_RERANKBM25s	0.6670	0.5922	0.3872	0.4685	0.8015
A_RERANKBM25sSplades	0.7021	0.6026	0.4040	0.4741	0.8605
A_RERANKEverythingRun	0.7063	0.6013	0.4053	0.4733	0.8664
Listo					
A_frgpt4	0.7102	0.6299	0.4109	0.4952	0.8664
A_rgpt4	0.7159	0.6366	0.4208	0.5034	0.8664
Median	0.5618	-	0.2689	-	-
Max	0.8644	-	0.6547	-	-

Table 6: Results on the Multilingual News Retrieval Task of the TREC 2023 NeuCLIR Track.

In the context of NeuCLIR, incorporating SPLADE with BM25 during the first retrieval stage elevates the retrieved list. However, the addition of additional retrieval methods yields increasingly diminished improvements. Models fine-tuned with the MIRACL dataset fell short of anticipated levels of effectiveness, particularly in Chinese due to discrepancies between traditional and simplified characters in the corpus and queries. Despite this, our methods demonstrated commendable effectiveness when benchmarked against median and maximal submissions. Yet, our Multilingual Information Retrieval (MLIR) strategy, which involved amalgamating outputs from document-translated networks, proved ineffective, suggesting the need for more effective aggregation strategies for multiple language sources.

5 Conclusion

Our empirical findings affirm the viability of the multi-stage retrieval pipeline, first-stage → mono → duo → listo, for both passage and document ranking tasks. Despite the robustness of the methodology, the use of a non-deterministic, closed-source model like RankGPT in the final stage introduces a level of unpredictability that tempers our enthusiasm. Such pipelines still seem effective, even in the case of cross-lingual retrieval NeuCLIR Shared Task, albeit the task was often reduced to English-based, mono-lingual retrieval in some parts of the framework. Our emphasis on optimizing the first stage was justified, even within a complex pipeline, demonstrated by their effect on downstream results. However, this leads to several unresolved queries:

1. How do we reduce the gaps between different stages of the pipeline?
2. Can we build a more effective and deterministic *listo* method than RankGPT, preferably open-sourced?
3. Is it time to introduce some computation limits to allow for fair comparisons across different pipelines?
4. Can we ever build a model as effective as with just a single-retrieval stage?

We hope to explore some of these questions and more in future work.

References

[Bassani and Romelli, 2022] Bassani, E. and Romelli, L. (2022). ranx. fuse: A python library for metasearch. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 4808–4812.

[Jeronymo et al., 2023] Jeronymo, V., Lotufo, R., and Nogueira, R. (2023). Neuralmind-unicamp at 2022 trec neuclir: Large boring rerankers for cross-lingual retrieval. *arXiv preprint arXiv:2303.16145*.

[Lassance, 2023] Lassance, C. (2023). Extending english ir methods to multi-lingual ir. *arXiv preprint arXiv:2302.14723*.

[Lassance and Clinchant, 2023a] Lassance, C. and Clinchant, S. (2023a). Naver labs europe (splade)@ trec deep learning 2022. *arXiv preprint arXiv:2302.12574*.

[Lassance and Clinchant, 2023b] Lassance, C. and Clinchant, S. (2023b). Naver labs europe (splade)@ trec neuclir 2022. *arXiv preprint arXiv:2303.11171*.

[Li et al., 2023] Li, M., Lin, S.-C., Ma, X., and Lin, J. (2023). Slim: Sparsified late interaction for multi-vector retrieval with inverted indexes. *arXiv preprint arXiv:2302.06587*.

[Lin et al., 2023] Lin, S.-C., Li, M., and Lin, J. (2023). Aggretriever: A simple approach to aggregate textual representations for robust dense passage retrieval. *Transactions of the Association for Computational Linguistics*, 11:436–452.

[Ma et al., 2022] Ma, X., Pradeep, R., Nogueira, R., and Lin, J. (2022). Document expansion baselines and learned sparse lexical representations for ms marco v1 and v2. *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[Nogueira et al., 2020] Nogueira, R., Jiang, Z., and Lin, J. (2020). Document ranking with a pretrained sequence-to-sequence model. *arXiv preprint arXiv:2003.06713*.

[Nogueira and Lin, 2019] Nogueira, R. and Lin, J. (2019). From doc2query to docttttquery.

[Pradeep et al., 2020] Pradeep, R., Ma, X., Zhang, X., Cui, H., Xu, R., Nogueira, R., and Lin, J. (2020). H2oloo at trec 2020: When all you got is a hammer... deep learning, health misinformation, and precision medicine. *Corpus*, 5(d3):d2.

[Pradeep et al., 2021] Pradeep, R., Nogueira, R., and Lin, J. J. (2021). The Expando-Mono-Duo design pattern for text ranking with pretrained sequence-to-sequence models. *arXiv:2101.05667*.

[Pradeep et al., 2023] Pradeep, R., Sharifymoghaddam, S., and Lin, J. (2023). RankVicuna: Zero-shot listwise document reranking with open-source large language models. *arXiv:2309.15088*.

[Qin et al., 2023] Qin, Z., Jagerman, R., Hui, K., Zhuang, H., Wu, J., Shen, J., Liu, T., Liu, J., Metzler, D., Wang, X., et al. (2023). Large language models are effective text rankers with pairwise ranking prompting. *arXiv preprint arXiv:2306.17563*.

[Sun et al., 2023] Sun, W., Yan, L., Ma, X., Ren, P., Yin, D., and Ren, Z. (2023). Is ChatGPT good at search? Investigating large language models as re-ranking agent. *arXiv:2304.09542*.