Building an Efficient Multilingual Non-Profit IR System for the Islamic Domain Leveraging Multiprocessing Design in Rust

@EMNLP 2024 Industry Track

Vera Pavlova and Mohammed Makhlouf
rttl labs
rttl.ai

Motivation:

Setting:

- Non-profit
- Multilingual
- Domain-specific

Challenges:

➔ Resource-constrained devices and limited budget

➔ Heavyweight MLLM

➔ Domain-specific data is more scarce in different languages

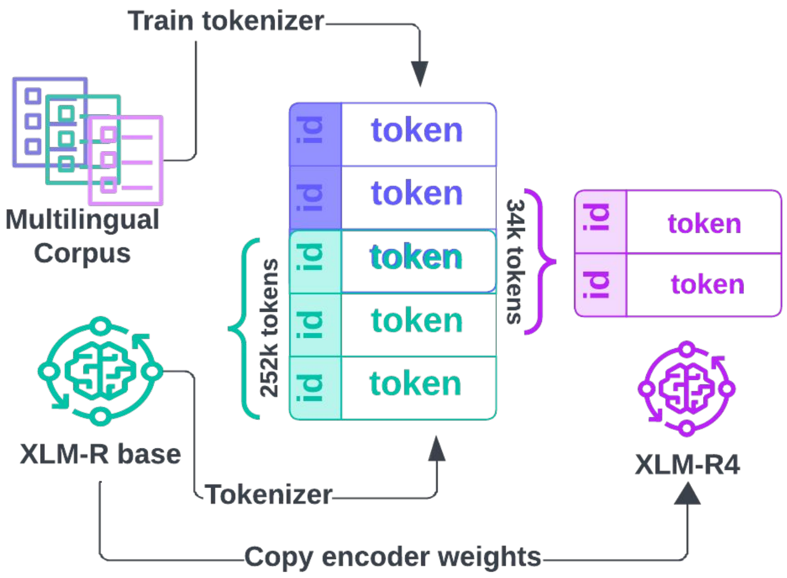
Solutions:

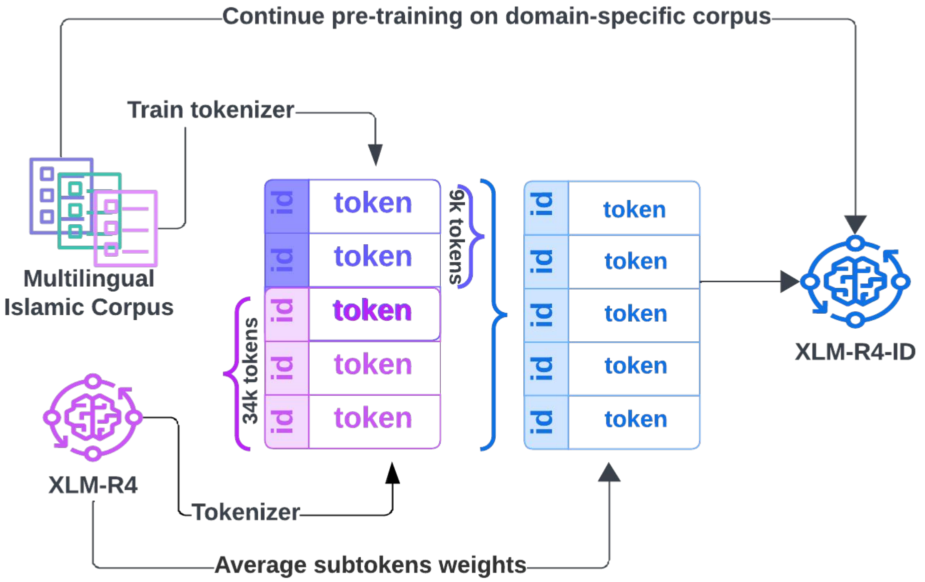
✓ CPU-based semantic search leveraging multiprocessing capabilities of Rust language

✓ Language reduction

✓ Continued pre-training with domain-specific vocabulary

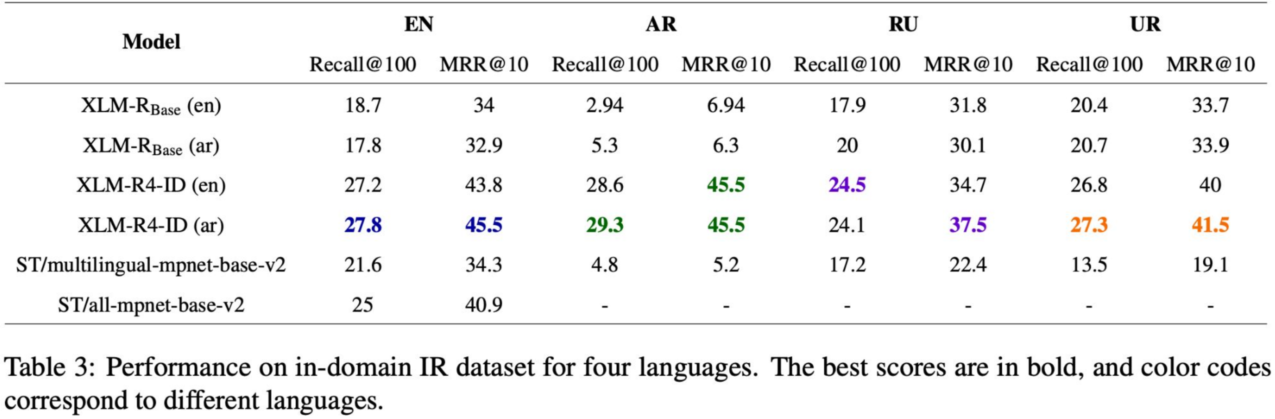
Language Reduction and Domain Adaptation:





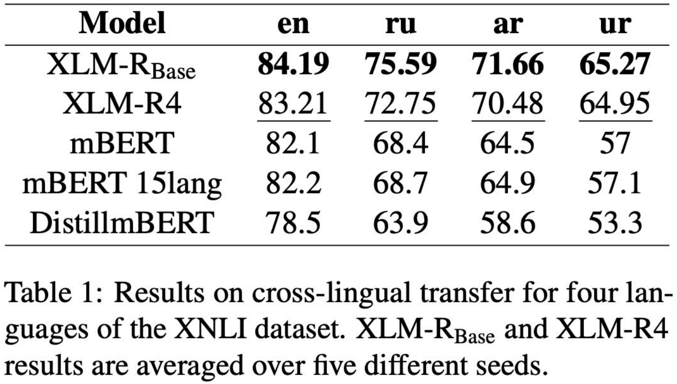
Domain adaptation of Language models: Gururangan et al. (2020), Beltagy et al. (2019), Lee et al. (2019)

Model Performance and Model Size comparison:



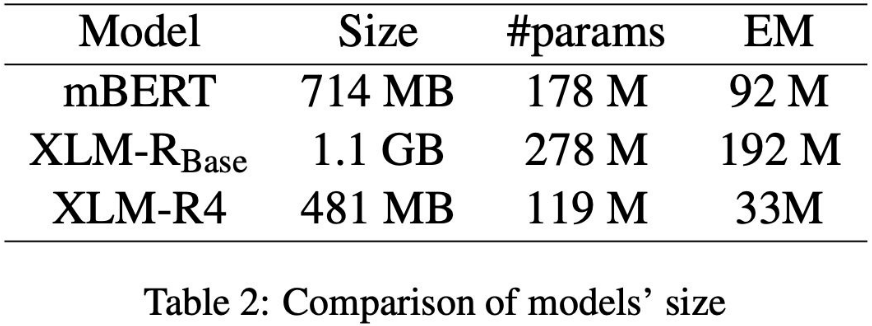
Model	EN		AR		RU		UR	
	Recall@100	MRR@10	Recall@100	MRR@10	Recall@100	MRR@10	Recall@100	MRR@10
XLM-R$_{\text{Base}}$ (en)	18.7	34	2.94	6.94	17.9	31.8	20.4	33.7
XLM-R$_{\text{Base}}$ (ar)	17.8	32.9	5.3	6.3	20	30.1	20.7	33.9
XLM-R4-ID (en)	27.2	43.8	28.6	**45.5**	**24.5**	34.7	26.8	40
XLM-R4-ID (ar)	**27.8**	**45.5**	**29.3**	**45.5**	24.1	**37.5**	**27.3**	**41.5**
ST/multilingual-mpnet-base-v2	21.6	34.3	4.8	5.2	17.2	22.4	13.5	19.1
ST/all-mpnet-base-v2	25	40.9	-	-	-	-	-	-

Table 3: Performance on in-domain IR dataset for four languages. The best scores are in bold, and color codes correspond to different languages.



Model	en	ru	ar	ur
XLM-R$_{\text{Base}}$	**84.19**	**75.59**	**71.66**	**65.27**
XLM-R4	83.21	72.75	70.48	64.95
mBERT	82.1	68.4	64.5	57
mBERT 15lang	82.2	68.7	64.9	57.1
DistillmBERT	78.5	63.9	58.6	53.3

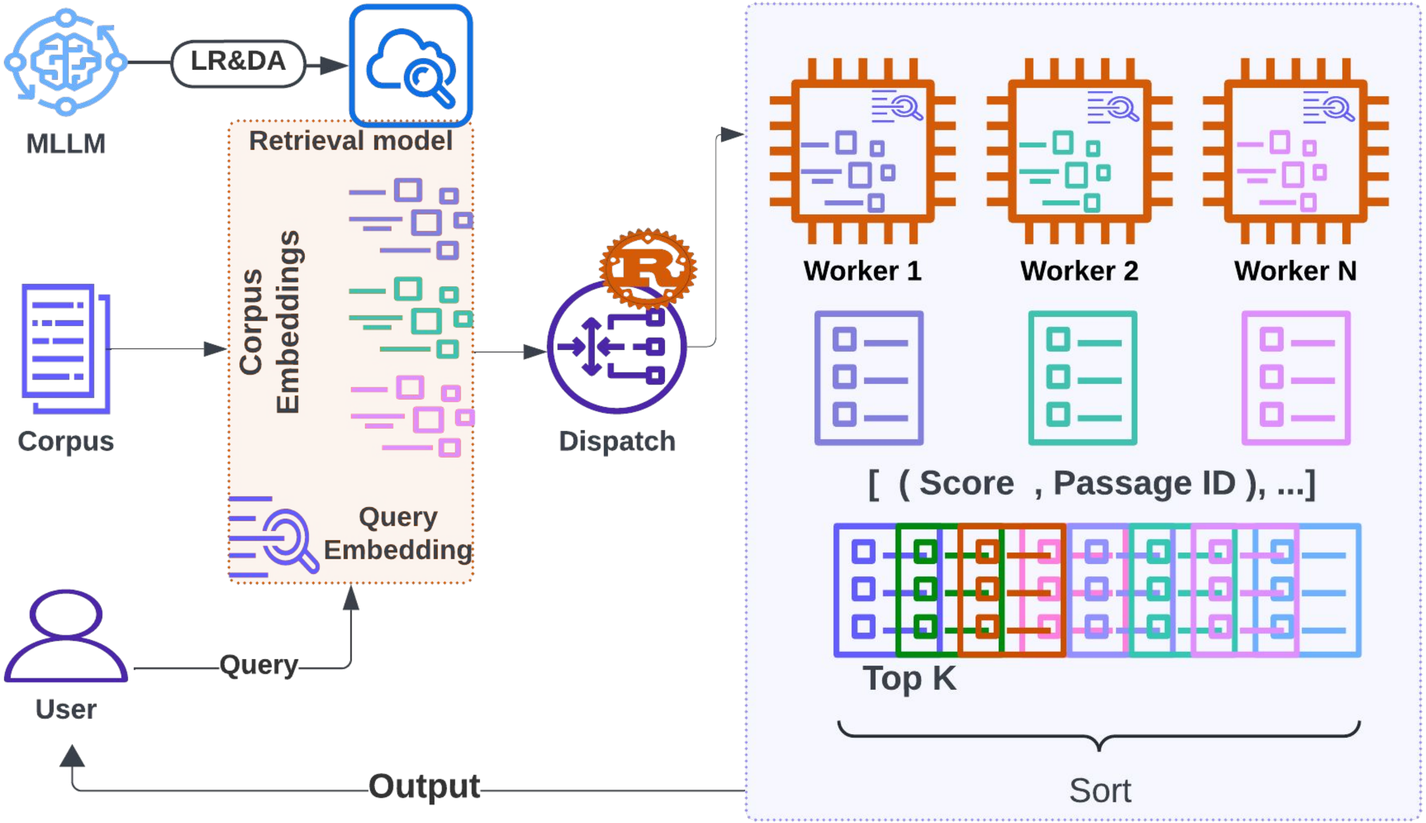
Table 1: Results on cross-lingual transfer for four languages of the XNLI dataset. XLM-R$_{\text{Base}}$ and XLM-R4 results are averaged over five different seeds.



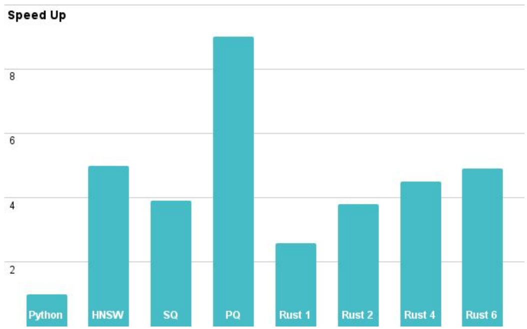
Model	Size	#params	EM
mBERT	714 MB	178 M	92 M
XLM-R$_{\text{Base}}$	1.1 GB	278 M	192 M
XLM-R4	481 MB	119 M	33M

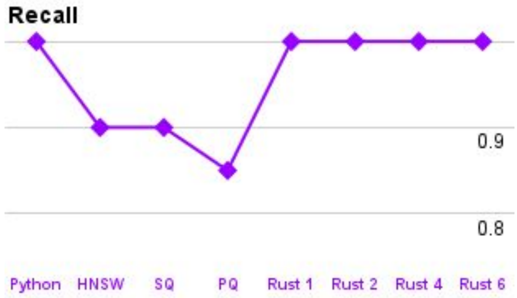
Table 2: Comparison of models' size

CPU-based Semantic Search with Multiprocessing Capabilities of Rust language:



Comparison of SUTs:





SUT	**Python**	**HNSW**	**SQ**	**PQ**	**Rust**	**Rust**	**Rust**	**Rust**
	(e.s.)		**(e.s.)**	**(e.s.)**	**1 w. (e.s.)**	**2 w. (e.s.)**	**4 w. (e.s.)**	**6 w. (e.s.)**
Speedup	1x	5x	3.9x	9x	2.6x	3.8x	4.5x	4.9x
Recall	100%	90%	90%	85%	100%	100%	100%	100%

Table 4: Comparisons of SUTs for the speedup of retrieval against baseline and percentage of baseline Recall (e.s stands for exact search and w. for worker).