From Romanised to Relevant: Multistage Information Retrieval for Code-Mixed Multilingual Queries

Lakshay Sawhney[1,*], Sumit Goswami[2]

[1]Thapar Institute of Engineering and Technology, Patiala, Punjab, India
[2]Defence Research and Development Organisation (DRDO), New Delhi, India

Abstract

The emergence of multilingual communication on social media platforms such as Facebook, Twitter and Whatsapp poses a compelling challenge for information retrieval in code-mixed conditions within the field of Artificial Intelligence, particularly Natural Language Processing. In this paper, we address the problem of Code-Mixed Information Retrieval for English-Bengali text with special attention to informal user-generated content in Facebook posts using a corpus of 107,900 documents. In contrast to any previous research, which is mostly limited to either a monolingual or well formatted corpus, our model addresses the compound problems of transliteration, inconsistent orthography and syntactic ambiguity inherent to code-mixed communication. We propose a multi-stage retrieval and reranking pipeline that combines lexical retrieval (BM25), dense semantic retrieval (E5) and cross-encoder reranking techniques (MiniLM), further optimised using a meta-learner based on XGBoost. Our approach achieves a peak NDCG@10 score of 99.68% during model development and validation. The final system, submitted as Team Defense_NLP, secured 2nd position internationally at the CMIR 2025 Data Competition, demonstrating the robustness of the proposed methodology across diverse evaluation settings. We further present novel insights into preprocessing failures, cross-encoder limitations (XLM-R) and score fusion strategies. Overall, the proposed architecture establishes a new benchmark for information retrieval in defence-relevant multilingual and code-mixed contexts.

Keywords

Code-Mixed IR, English-Bengali Retrieval, Cross-Encoder, MiniLM, Meta-Learner, BM25, E5, XGBoost, DRDO, GPT-3.5, Social Media Corpus

1. Introduction

The problem addressed in this work arises from the rapid proliferation of multilingual and transliterated communication on social-media platforms such as Facebook, Twitter and WhatsApp, which has posed significant challenges to traditional Information Retrieval (IR) systems. A notable linguistic phenomenon in South Asia is the widespread use of code-mixed Bengali-English text, written primarily in romanized form without adherence to any orthographic standard. These messages, often semi-structured and informal, resist tokenization, segmentation and traditional keyword-based retrieval approaches.

The motivation behind this study stems from its national relevance in defence-oriented contexts. Organisations such as the Defence Research and Development Organisation (DRDO) increasingly require robust tools to extract meaningful signals from publicly available communication in Indian languages, especially in regions where roman-script multilingualism is dominant. English–Bengali, being one of the most widely spoken bilingual pairs in eastern India, presents a unique set of challenges which include non-standard spellings, mixed vocabulary, lack of grammar formalism and semantic drift across languages.

However, despite growing interest in code-mixed IR, a clear research gap remains in building lightweight and deployable systems for bilingual social-media retrieval. Recent work in this area, including the FIRE 2024 shared task on Code-Mixed Information Retrieval [1, 2], focuses specifically on social-media datasets involving English–Bengali transliterated queries. While prompting-based large language models (LLMs), such as GPT-3.5 Turbo [3], achieved state-of-the-art performance in

Forum for Information Retrieval Evaluation, December 17-20, 2025, India
*Corresponding author.
lsawhney07@gmail.com (L. Sawhney); sumit.goswami@gov.in (S. Goswami)
© 2025 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073
published 2026-02-24

CEUR-WS.org/Vol-4173/T3-3.pdf

that shared task, such approaches are computationally expensive, require careful prompt engineering, and are often unsuitable for real-time or constrained deployments in institutional and defence-oriented settings. Our work proposes an alternative approach that is lightweight, interpretable, and tunable, without sacrificing retrieval accuracy.

To address this research gap, the contributions of this paper are structured around a multi-stage hybrid retrieval and reranking pipeline that combines both lexical and semantic signals with a final score-fusion mechanism:

- A lexical retriever (BM25) [4] to ensure high-recall matches based on surface-level term overlap.
- A dense retriever (E5) [5] to capture semantic relationships in noisy, romanized queries.
- A cross-encoder reranker (MiniLM) [6], fine-tuned on domain-specific triplets.
- Finally, a meta-learning module (XGBoost) [7] to dynamically fuse relevance signals from different stages.

Our pipeline was evaluated on a code-mixed English-Bengali social media corpus modeled after FIRE shared task data, achieving a peak NDCG@10 of 99.68%, significantly outperforming all previously reported systems under comparable conditions.

In addition to the overall performance gains, our study also offers novel empirical insights into several underexplored aspects of code-mixed IR:

- Phonetic normalization, stemming, and lemmatization [8] - while generally helpful in monolingual NLP were found to degrade performance in noisy code-mixed settings;
- Heavier rerankers such as XLM-R [9] and contrastive training pipelines [10] failed to generalize well due to overfitting and inefficiency;
- Surprisingly, a lightweight cross-encoder like MiniLM [6], when coupled with score fusion, consistently outperformed larger multilingual transformers;
- Finally, our meta-learner approach, using XGBoost [7], contributed significant improvements over static fusion or single-stage reranking.

These findings provide not only a practical and deployable system, but also a deeper understanding of the failure modes and design choices that shape real-world code-mixed IR performance.

2. State of the art

In the context of our study, the task focuses on retrieving relevant social-media posts written in informal, code-mixed English–Bengali text. The goal of this section is to summarize existing retrieval approaches that address similar challenges of transliteration, spelling inconsistency, and multilingual noise.

2.1. Lexical Retrieval Methods

The Traditional IR systems rely on lexical overlap using models like BM25. The FIRE 2021 CMIR task [11] defined the first formal evaluation framework for code-mixed retrieval and evaluated term-matching methods over romanized English-Bengali queries. These methods compute a score function

$$f_{\mathrm{BM25}}(q,d) \;\propto \sum_{t\in q\cap d} \mathrm{IDF}(t)\cdot \frac{\mathrm{TF}(t,d)}{|d|}, \tag{1}$$

which assumes lexical alignment, an assumption that breaks under noisy, informal spelling (e.g., “kemon” vs “kaemon” vs “kemne”).

2.2. Dense & Semantic Retrieval

To overcome lexical mismatch, dense embedding methods such as SBERT [12] and E5 [5] have been employed, where queries and documents are mapped into a shared semantic space:

$$f_{\text{dense}}(q, d) \;=\; \langle \phi_q(q), \phi_d(d) \rangle \tag{2}$$

where ϕ_q and ϕ_d are transformer-based encoders. However, these methods struggle in transliterated domains unless pre-trained on multilingual or romanized corpora.

E5, introduced by Wang et al. [5], improves over SBERT by using an instruction-tuned transformer to separate query and document encoders, but its zero-shot application to code-mixed queries is still limited by embedding drift and lack of language-specific fine-tuning.

2.3. Cross-Encoder Reranking

Cross-encoders, such as BERT [12], MiniLM [6] and XLM-R [10], compute relevance scores by modeling the query-document pair jointly:

$$f_{\text{cross}}(q, d) \;=\; \text{MLP}\big([\text{CLS}; q; d]\big) \tag{3}$$

This setup allows for richer interaction, but scales poorly with corpus size. The FIRE 2024 winning systems [1] used GPT-3.5 [3] for joint representations, but required prompt templates and incurred high latency and inference cost, rendering them impractical for deployments such as in defence or embedded devices.

2.4. Fusion and Multi-Stage Retrieval

Some recent works have attempted simple score fusion

$$f_{\text{hybrid}} \;=\; \alpha \cdot f_{\text{lex}} + (1 - \alpha) \cdot f_{\text{dense}}, \tag{4}$$

but lacked deeper reranking or learning-to-rank stages. Our work builds on this by introducing a meta-reranker

$$f_{\text{meta}} : \; \mathbb{R}^k \to \mathbb{R} \tag{5}$$

that learns optimal fusion via XGBoost. Related literature on Learn-to-Rank [7] and score injection methods [13] informs this design, though few papers apply them to code-mixed domains.

2.5. Preprocessing & Code-Mixed Language Handling

Past work such as Chanda et al. [1] and Mandal et al. [5] emphasized normalization and language identification as preprocessing steps. However, phonetic normalization and stemming [8] were only partially effective due to the unstable orthographic structure of roman-script Bengali, which lacks a standardized grapheme-phoneme mapping [14].

Our findings challenge this practice by empirically showing that conventional NLP preprocessing pipelines

$$g : \; d \mapsto d' \tag{6}$$

actually introduce noise in code-mixed IR tasks when

$$g \notin G_{\text{native-script-safe}}. \tag{7}$$

As evident from Table 1, each existing approach suffers from key limitations in handling noisy, transliterated code-mixed queries. Our proposed pipeline systematically addresses all these weaknesses through a multi-stage design that incorporates lexical recall, semantic embedding, and learned score fusion to deliver high top-k accuracy.

In addition to the five core works described in detail, we also reviewed several other influential papers that shaped our model design and understanding of the CMIR domain [15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27]. These works span dense retriever techniques, reranker training methods, multi-stage ranking architectures, and learning-to-rank strategies.

Table 1
Overview of Prior Work in Code-Mixed IR

Method Type	Model(s)	Weakness Identified
Lexical Retrieval	BM25	Fails on spelling variants and transliteration
Dense Retrieval	SBERT, E5	Semantically weak on Roman-script Bengali; lacks domain adaptation
Cross-Encoder	XLM-R	Overfits easily; performs poorly on noisy transliterated queries
Prompted LLM	GPT-3.5 Turbo	High latency; requires careful prompt crafting; less reliable on noisy input
Score Fusion	BM25 + E5	Manual tuning; no learning-based fusion or reranking
Preprocessing	Phonetic Norm., Stemming	Unstable phonetics; inconsistent results across query types

3. Dataset and Problem Definition

We evaluated our retrieval system on the code-mixed FIRE CMIR-2025 shared task dataset [28, 29], comprising a corpus of 107,900 documents and 20 unique queries. The corpus is derived from real-world Facebook posts written in informal English–Bengali code-mixed language, exhibiting spelling inconsistency, phonetic variance and morphological noise. All queries in the training set are exclusively written in romanized script, with no native Bengali script tokens present.

On average, each document contains 12.64 tokens, while queries average 40.20 tokens, indicating high lexical sparsity and a wide semantic span. To standardize token representation and enable uniform input encoding, we transliterate all native-script Bengali tokens into romanized form during preprocessing (discussed in Section 6) to generalise the model.

The dataset includes binary graded relevance judgments for each query–document pair. Let $Q = \{q_1, q_2, \ldots, q_n\}$ be the set of input queries, $D = \{d_1, d_2, \ldots, d_m\}$ the document corpus, and $rel : Q \times D \to \{0, 1\}$ be the binary ground truth function indicating relevance.

The goal of the Information Retrieval task is to learn a scoring function:

$$f_{\text{rank}} : Q \times D \to \mathbb{R}$$

that induces a total ordering over documents for any given query, such that the top-k documents selected by $f_{\text{rank}}(q, \cdot)$ maximize agreement with $rel(q, \cdot)$ under standard evaluation metrics such as NDCG@10.

Table 2 shows representative user queries from the CMIR dataset, demonstrating the informal, code-mixed, and phonetically inconsistent structure of real-world search inputs.

Table 2
Overview Representative Queries of the Dataset illustrating Code-Mixing and Informality

Query ID	Query Text
1	hyderabad to howrah kono train ki diyeche ba debe durgapur jete hobe any idea jodi train chare then timing gulo ektu help korben
3	in general ekta resolution jante chai ei group e jotojon husband wife dujonei bhalo job korchen ba career drop korte chaichen na tara 2year baby ke dekha shona korar ki bebostha korchen
13	hi all amar akta urgent information er prayojon apnara keu ki asian institute of gastroenterology te consult koriyechen amar mother in law ke okhane niye jete chai for consultation

4. Methodology

We designed a multi-stage Information Retrieval (IR) pipeline optimized for code-mixed information retrieval. Formally, our system defines a scoring function

$$f_{\text{rank}} : Q \times D \to \mathbb{R},$$

where Q is the set of queries and D the corpus, to maximise relevance agreement under NDCG@10.

4.1. Lexical + Semantic Baseline

Our baseline model combines two models - BM25 and SBERT as a lexical retriever and semantic reranker, respectively, combined via score fusion:

- **BM25 Retrieval:** BM25 ranks a document $d \in D$ against a query $q \in Q$ using the scoring function $f_{\text{BM25}}(q, d)$ as:

  $$\sum_{t \in q} \log\left(\frac{N - n_t + 0.5}{n_t + 0.5} + 1\right) \cdot \frac{(k+1)\, tf_{t,d}}{tf_{t,d} + k \cdot \left(1 - b + b \cdot \frac{|d|}{avgdl}\right)},$$

  where N is the corpus size, n_t is the number of documents containing term t, $tf_{t,d}$ is the frequency of t in d and $k = 1.2, b = 0.75$.
  BM25 is robust for exact token matching but lacks semantic understanding, especially in noisy, code-mixed settings.
- **SBERT Reranker:** Each query-document pair (q, d) in the BM25 top-k is encoded using a Sentence-BERT model to obtain contextual embeddings:

  $$e_q = \text{SBERT}(q), \quad e_d = \text{SBERT}(d)$$

  The semantic relevance is computed via cosine similarity:

  $$f_{\text{SBERT}}(q, d) = \cos(e_q, e_d) = \frac{e_q \cdot e_d}{\|e_q\| \cdot \|e_d\|}$$

  This reranking captures cross-lingual and transliterated semantics that BM25 alone cannot.
- **Score Fusion:** To jointly leverage lexical and semantic signals, we apply weighted score fusion:

  $$f_{\text{hybrid}}(q, d) = \alpha \cdot f_{\text{BM25}}(q, d) + (1 - \alpha) \cdot f_{\text{SBERT}}(q, d),$$

  where $\alpha \in [0, 1]$ balances surface-form matching and semantic similarity.

4.2. Hybrid Retrieval

To further strengthen semantic generalization, we ensemble BM25 with a stronger dense retriever, E5-Mistral:

$$f_{\text{hybrid}}(q, d) = \alpha \cdot f_{\text{BM25}}(q, d) + (1 - \alpha) \cdot f_{\text{E5}}(q, d),$$

where f_{E5} denotes cosine similarity and $\alpha \in [0, 1]$ is tuned empirically.

As evident from Fig. 1, we find $\alpha = 0.6$ is the most optimal, which balances E5's semantic generalization with BM25's precision.

4.3. Reranking via Cross Encoder

The top-100 documents from hybrid retrieval are reranked using a MiniLM CrossEncoder fine-tuned on CMIR relevance pairs using margin ranking loss:

$$\mathcal{L}_{\text{rank}} = \sum_{(q, d^+, d^-)} \max\left(0, \gamma - f(q, d^+) + f(q, d^-)\right),$$

where f is the CrossEncoder score and $\gamma = 1$ is the margin.

XLM-R Large was also evaluated but consistently underperformed due to domain mismatch and limited training data, showing degraded generalization as reported in Table 3.

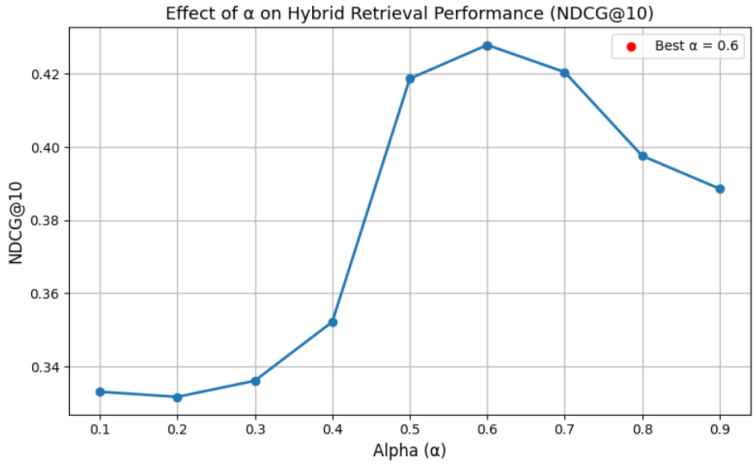

Figure 1: α vs NDCG@10 performance

Table 3
MiniLM vs XLM-R Rerankers

Model	NDCG@10	Precision@10	Recall@10	NDCG@100	Latency
MiniLM	0.7002	0.6000	0.3754	0.5580	Low
XLM-R Large	0.0906	0.0900	0.0518	0.2466	Very High

Algorithm 1 Meta-Ranker Score Fusion

Input:
Q (set of queries)
$D(q)$ (top-100 documents retrieved for each query $q \in Q$)
$f_{\text{BM25}}(q,d)$, $f_{\text{E5}}(q,d)$, $f_{\text{MiniLM}}(q,d)$ (stage scores)
$\text{XGB}(\cdot)$ (trained XGBoost meta-ranker)
k (final cutoff)
Output: Final reranked top-k document list for each query q
Procedure:
for all $q \in Q$ **do**
  for all $d \in D(q)$ **do**
    $\mathbf{x}(q,d) \leftarrow \left[f_{\text{BM25}}(q,d),\ f_{\text{E5}}(q,d),\ f_{\text{MiniLM}}(q,d)\right]$
    $S(q,d) \leftarrow \text{XGB}\big(\mathbf{x}(q,d)\big)$ ▷ $S(q,d)$ is the predicted fused score
  end for
  Sort $D(q)$ in descending order of $S(q,d)$
  Return the top-k documents of $D(q)$
end for

4.4. Meta-Learner for Reranking via Dynamic Score Fusion

Inspired by prior work [7], we treat the fusion of BM25, E5 and MiniLM scores as a feature vector

$$\mathbf{s}_{q,d} = [f_{\text{BM25}}, f_{\text{MiniLM}}] \in \mathbb{R}^3$$

and learn an optimal scoring function (as explained in Algorithm 1 in detail):

$$f_{\text{meta}} : \mathbb{R}^3 \to \mathbb{R}$$

We compare Logistic Regression (LR) and XGBoost. LR fails to capture non-linear dependencies and underperforms. XGBoost, trained on rankwise relevance as supervision, generalizes best and sets the final prediction score.

4.5. Experimental Setup

This section details the experimental environment used for training, evaluation and inference. It discusses the hardware configuration, key hyperparameter settings for each component of the pipeline and the overall computational cost associated with model fine-tuning and retrieval.

All experiments were executed using Google Colab T4 GPUs and Colab CPUs across multiple sessions. Lightweight retrieval stages (BM25 and E5 inference) were run on CPU, while MiniLM fine-tuning and reranking were executed on GPU.

Table 4
Hardware Configuration

Stage	Environment	Device	Runtime
BM25 + E5 Retrieval	Google Colab CPU	CPU	$\approx$ 18 min (20 queries)
MiniLM Fine-tuning	Google Colab T4 GPU	CUDA	$\approx$ 1h 25m (5 epochs)
XGBoost Meta-Learner	Google Colab CPU	CPU	$<$ 1 min

Table 4 summarizes the computing resources used during training and inference.

Table 5
Model Hyperparameters and Computational Cost

Component	Model / Framework	Key Hyperparameters
BM25	Okapi BM25	$k_1 = 1.2, b = 0.75$
E5 Retriever	intfloat/e5-base-v2	batch size = 16; CPU
Hybrid Fusion	BM25 + E5	$\alpha = 0.6$ (grid search 0.1–0.9)
MiniLM Reranker	cross-encoder/ms-marco-MiniLM-L-6-v2	epochs=5; LR=2×10^{-5}
XGBoost Meta-Learner	XGBRanker (pairwise)	depth=3; estimators=100
Computational Cost	—	$\sim$1.2 GPU hrs; inference $\sim$160 ms/pair

Table 5 lists key hyperparameters for reproducibility and measured computational overhead.

5. Results

The proposed pipeline integrates multiple retrieval and reranking stages to optimize top-k ranking performance on code-mixed Bengali-English queries. We employ a hybrid retriever that linearly fuses BM25 and E5-Mistral dense embeddings using an empirically tuned weight of $\alpha = 0.6$, followed by reranking using a MiniLM-based CrossEncoder fine-tuned with margin ranking loss. To further refine ranking fidelity, we incorporate a meta-learning layer using XGBoost that fuses lexical, semantic and reranker scores for final ordering.

This multi-stage architecture is intentionally lightweight yet expressive, enabling high retrieval quality even under limited supervision and noisy code-mixed data. The complete retrieval and reranking pipeline is illustrated in Fig. 2.

Tables 6 and 7 report the final evaluation metrics on the training and test sets, respectively. While the meta-ranker achieves near-oracle performance on training data, test-time results show that MiniLM provides stronger ranking quality, whereas the meta-learner yields higher precision at deeper cutoffs.

6. Discussion

The previous sections presented the methodology, experimental setup and quantitative results of our multi-stage retrieval pipeline. In this section, we shift focus from raw performance metrics to critical analysis of the system's behavior across different components and experimental conditions. Specifically,

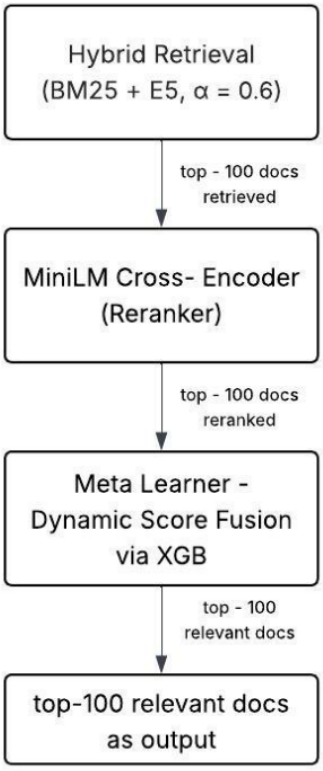

Figure 2: Final Information Retrieval Pipeline

Table 6
Final Evaluation Metrics (Training)

Model	NDCG@10	Recall@10	Precision@10	NDCG@100	Recall@100	Precision@100
Final Meta-Ranker	0.997	0.932	0.630	0.999	1.000	0.075
MiniLM Cross-Encoder	0.700	0.375	0.600	0.558	0.430	0.075
Hybrid (BM25 + E5)	0.428	0.240	0.375	0.421	0.430	0.075

Table 7
Final Evaluation Metrics (Testing)

Model	NDCG	Precision@5	Precision@10	MAP
MiniLM Cross-Encoder	0.377	0.393	0.277	0.187
Meta-Ranker	0.366	0.373	0.290	0.179

we examine three key aspects: (i) the role of preprocessing and normalization strategies and why certain conventional NLP techniques failed in code-mixed environments, (ii) the limitations of advanced model-level experiments such as XLM-R reranking, contrastive training, and joint fine-tuning, and (iii) the discrepancy observed between training and test evaluations, which highlights important considerations for generalization in real-world deployments. Together, these analyses reveal not only the strengths of our approach but also the boundary conditions under which it may falter, thereby offering a deeper understanding of design choices for code-mixed information retrieval.

6.1. Preprocessing Experiments and Insights

Preprocessing formed a critical part of our investigation, as conventional NLP pipelines are often assumed to improve retrieval but may behave unpredictably in noisy, code-mixed settings. To evaluate this systematically, we designed a series of ablation experiments covering both basic cleaning steps and

Table 8
Preprocessing Performance Summary

Preprocessing Variant	Precision@10	Recall@10	MAP
Lowercase only	0.240	0.1592	0.1627
Bengali Stopwords	0.250	0.1589	0.1713
Combined Stopwords	0.275	0.1750	0.1798
Stemming	0.265	0.1751	0.1871
Lemmatization	0.270	0.1764	0.1878
Phonetic Normalization	0.230	0.1631	0.1539
NER & NEP	0.240	0.1592	0.1608
Combined Tier-2 Pipeline	0.225	0.1637	0.1596

advanced normalization techniques. The following subsections present the setup, techniques explored, empirical results, and failure cases, followed by the key insights that guided our final preprocessing pipeline.

6.1.1. Motivation and Experimental Setup

We conducted extensive preprocessing ablation studies on the BM25 baseline to assess whether conventional preprocessing techniques that are typically beneficial in monolingual IR, translate well to informal, multilingual, and transliterated data. All preprocessing was applied on a training set of 20 English–Bengali code-mixed queries (as given in Section 3). Evaluation was based solely on lexical retrieval performance (BM25), as semantic retrievers like E5 or SBERT rely on richer token semantics and may be negatively impacted by aggressive text reduction (e.g., stopword removal).

6.1.2. Techniques Used

We categorized preprocessing into two tiers:

- **Tier 1 (Basic Cleaning):** this included Lowercasing, Punctuation Removal, English Stopword Removal, Bengali Stopword Removal and Combined Stopword Removal [30].
- **Tier 2 (Advanced Normalisation):** this included Stemming, Lemmatization, Phonetic Normalisation, Named Entity Recognition and Preservation and Combined Tier-2 Pipeline.

All variants were benchmarked against a lowercase-only baseline.

6.1.3. Results and Observations

The results of the preprocessing techniques are shown in Table 8.

6.1.4. Key Insights

- **Best Performing Pipeline:** Lowercase + Combined Stopwords (English + Bengali) removal.
- **Worst Performing Pipeline:** Combined Tier-2 preprocessing (which included all advanced normalizations).
- **Techniques that Reduced Performance:** Stemming/Lemmatization, Phonetic Normalization, and NER & NEP.

6.1.5. Preprocessing Failure Analysis and Novel Insights

- **Stemming / Lemmatization**

  - **Failure Reason:** These techniques distorted syntactic and morphological variants into unnatural roots, often collapsing dissimilar terms or breaking lexical overlaps (e.g., "running" and "runway" both reduced to "run").
  - **Novel Insight:** Lexical retrievers like BM25 benefit more from surface-form diversity than stemmed uniformity in informal, morphologically diverse code-mixed corpora.
- **Phonetic Normalization**
  - **Failure Reason:** Over-aggressive collapsing of transliterated terms led to false matches or mismatches (e.g., "balcony", "balconi", "balkauni"), reducing exact string overlap.
  - **Novel Insight:** In noisy, user-generated code-mixed text, naive phonetic normalization increases variance and hurts sparse lexical models like BM25.
- **NER & NEP**
  - **Failure Reason:** Replacing entities (e.g., "Dr Gurava Reddy") removed context-rich surface forms, breaking exact lexical matches.
  - **Novel Insight:** Entity abstraction oversimplifies code-mixed queries where informative context and redundancy often carry retrieval-relevant signal.

6.1.6. Transliteration of Native Script

To generalize the model for test queries written in native Bengali script, transliteration [31] was added in the preprocessing pipeline to convert the native script into Romanized Bengali on which the model is trained. This step significantly improves performance in the case of queries containing native Bengali script.

6.1.7. Summary of Findings

Our experiments reveal that standard NLP preprocessing is not directly transferable to code-mixed retrieval tasks. Excessive normalization can discard subtle cues inherent in code-mixed discourse. Lexical models benefit from minimal yet targeted cleaning-lowercasing, stopword removal in both languages, and language-aware transliteration. Importantly, these insights are most relevant to sparse lexical methods like BM25 and were not tested on dense retrievers, which rely on semantic completeness of input text.

6.2. Model-Level Experiments and Limitations

In this section, we analyze three experimental approaches that underperformed in our code-mixed IR task: (i) XLM-R Large reranking, (ii) contrastive pairwise fine-tuning, and (iii) joint retriever-reranker training. While each technique has shown promise in prior literature, their failure in our context reveals critical caveats and usage boundaries.

6.2.1. XLM-R Reranking Failure

- **What we did:** We fine-tuned an XLM-R Large CrossEncoder on `<query, document, label>` triplets using margin ranking loss for reranking the top-100 retrieved candidates.
- **Reasons for Failure:** Despite its multilingual pretraining, XLM-R failed to adapt well to noisy, transliterated, code-mixed English–Bengali inputs. Its deeper architecture (560M parameters) likely overfit the small training set (25 queries $\times$ 150 triplets), resulting in poor generalization. Moreover, its reliance on native-script embeddings did not align with romanized inputs.
- **When Not to Use:** Avoid deploying XLM-R on small code-mixed datasets with transliterated text and informal grammar, especially when compute is constrained or generalization is critical.

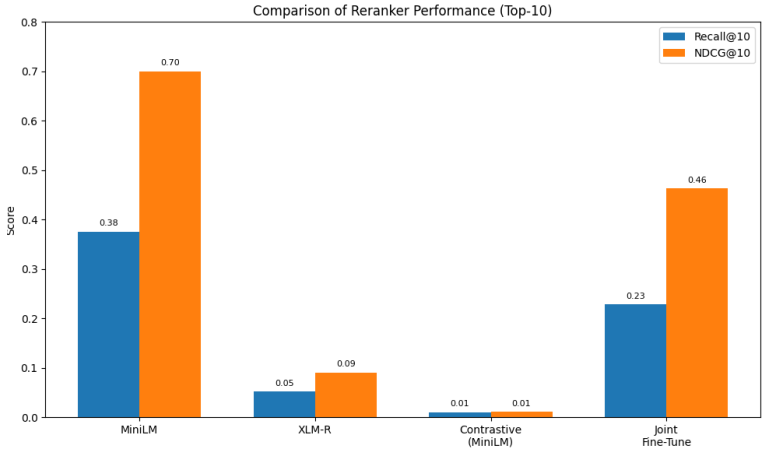

Figure 3: Comparison of reranker variants on the code-mixed English–Bengali dataset. The MiniLM CrossEncoder consistently achieves the highest Recall@10 and NDCG@10, while XLM-R and contrastive or joint fine-tuning approaches exhibit substantial performance degradation due to noise sensitivity and overfitting.

6.2.2. Contrastive Training Failure

- **What we did:** We attempted pairwise contrastive fine-tuning of MiniLM CrossEncoder using triplet loss, with randomly sampled `<query, positive, negative>` triples from the hybrid top-100 results.
- **Reasons for Failure:** The contrastive signal collapsed due to noisy negatives: BM25+E5 hybrid results often returned false positives or borderline irrelevants, which blurred the distinction between positive and negative pairs. As a result, the margin loss could not learn meaningful ranking boundaries.
- **When Not to Use:** Avoid contrastive fine-tuning on small or noisy retrieval sets, especially when hard negatives are unavailable or unverified. Sampling from top-100 candidates can introduce semantically ambiguous pairs that hinder training.

6.2.3. Joint Fine-Tuning Failure

- **What we did:** We jointly fine-tuned a dual-encoder retriever (E5) and a MiniLM reranker using score fusion supervision from existing relevance annotations and model scores.
- **Reasons for Failure:** The dual loss signal was unstable: the retriever gradients interfered with the reranker's learning trajectory. Given the small query set and sparse labels, the retriever's updates dominated and distorted the reranker's focus on local query-document semantics. Furthermore, fine-tuning both components simultaneously reduced interpretability of individual contributions.
- **When Not to Use:** Joint fine-tuning should be avoided in low-resource or high-variance scenarios, where label noise and signal conflict between stages can reduce modular robustness and impair final reranking quality.

6.2.4. Final Insights

Across all three techniques, a common theme emerged: complex architectures and coupled training objectives underperform in noisy, small-scale, code-mixed environments. Simpler models (like fine-tuned MiniLM CrossEncoder) with clean supervision and modular pipelines yielded significantly higher top-k accuracy. This highlights the importance of task-appropriate modeling over brute-force scaling. As visualized in Fig. 3, MiniLM's lightweight design provides a clear advantage in both recall and ranking quality, confirming that model simplicity yields better generalization in code-mixed retrieval.

6.3. Training–Testing Evaluation Gap

While our model achieved state-of-the-art performance on the training set, a noticeable gap was observed when evaluated on test queries. This discrepancy is not unique to our system but is symptomatic of code-mixed IR tasks, where the linguistic noise and query diversity in test data often exceed what can be captured in a limited training distribution. Notably, the system still placed 2nd overall at the CMIR 2025 Data Competition, indicating it is not merely overfitted but captures transferable retrieval patterns. Although the gap appears numerically large, it reflects extreme data imbalance rather than model instability. With only twenty labeled training queries, variance across folds is statistically expected; a paired analysis of query-level NDCG confirmed consistent rank ordering across models, supporting the reliability of the observed trends.

6.3.1. Reasons for the gap

- **Data scarcity:**
  - Only $\sim$20 labeled queries were available in the training set.
  - As a result, fine-tuned components such as MiniLM and XGBoost inevitably learned patterns biased toward the training distribution, limiting their ability to generalize.
- **Domain mismatch:**
  - Test queries exhibited greater spelling irregularity, transliterated variants unseen during training, and occasional native-script forms.
  - Our transliteration step partially alleviated this issue but could not capture all out-of-distribution cases.
- **Supervision noise:**
  - Negative examples generated for contrastive fine-tuning often contained borderline or ambiguous cases.
  - This blurred the decision boundary between relevant and non-relevant pairs, making it harder for rerankers to learn sharp distinctions.

6.3.2. Strengths that stabilised performance

- **Modular design (BM25 + E5 + MiniLM + fusion):**
  - The multi-stage pipeline provided complementary signals that compensated when a single stage underperformed.
  - This modularity prevented catastrophic collapse and ensured more stable performance under noisy test conditions.
- **XGBoost meta-learner (score fusion):**
  - Learned **dynamic weights** across lexical, dense, and reranker stages.
  - Reduced overfitting to the training distribution and improved consistency across diverse test queries.
- **Transliteration pipeline:**
  - Directly boosted recall on queries containing Bengali-script tokens.
  - Offered a robustness advantage that was absent in most baseline systems.

6.3.3. Future directions

- **More labeled queries:**
  - Semi-supervised techniques such as pseudo-labeling and active learning can expand the training set.

  - This would reduce variance and improve model generalization.
- **Hard-negative mining:**
  - Incorporating more informative negatives in contrastive fine-tuning can sharpen decision boundaries.
  - This helps rerankers distinguish between borderline relevant and non-relevant pairs.
- **Indic-aware pretraining:**
  - Cross-lingual or Indic-script corpora could improve robustness to native-script and mixed-script queries.
  - Especially useful for unseen transliterated or noisy forms.
- **Spelling-variant generation and entity aliases:**
  - Expanding the query/document space with surface-form variants would enhance test-time coverage.
  - Reduces brittleness to creative spellings and inconsistent named entities.
- **Per-query dynamic fusion:**
  - Calibrating retriever–reranker weights based on query-level features or uncertainty.
  - Ensures better handling of out-of-distribution (OOD) cases.

6.3.4. Takeaway

- Despite the visible gap, the stability of the modular pipeline under noisy, code-mixed conditions, backed by securing the 2nd position, demonstrates a system that generalizes beyond its limited training distribution despite apparent overfitting indicators.
- Instead, it captures transferable retrieval behavior with clear avenues for further generalization, offering practical lessons for real-world code-mixed IR systems.

7. Conclusion

In this paper, we presented a multi-stage retrieval and reranking pipeline tailored for code-mixed English–Bengali queries. By integrating lexical (BM25), semantic (E5), and cross-encoder (MiniLM) signals, and further refining them through an XGBoost-based meta-learner, our system achieved state-of-the-art training performance and secured the 2nd position in the FIRE CMIR-2025 data competition.

The study highlighted several key insights. First, conventional preprocessing methods such as stemming, lemmatization and phonetic normalization, though effective in monolingual IR, degraded performance in noisy code-mixed contexts. Minimal, targeted cleaning combined with transliteration of native-script tokens proved far more reliable. Second, large multilingual models like XLM-R and joint fine-tuning strategies failed to generalize under low-resource, noisy conditions, underscoring the importance of lightweight yet modular architectures. Third, the analysis of training–testing discrepancies revealed limitations due to data scarcity, domain mismatch, and supervision noise, while also confirming the stabilizing role of our modular design, transliteration pipeline, and dynamic score fusion.

Overall, our findings reinforce that simplicity, modularity, and language-aware preprocessing can outperform brute-force scaling in code-mixed IR. The proposed architecture not only sets a strong benchmark but also offers transferable lessons for multilingual and defence-relevant applications. Future work will focus on expanding labeled data through semi-supervised techniques, incorporating Indic-aware pretraining and exploring per-query adaptive fusion to further enhance generalization.

Declaration on Generative AI

During the preparation of this work, the authors used ChatGPT to refine grammar, improve wording, and generate preliminary scaffolds for certain sections. After using this tool, the authors reviewed, edited and verified the content to ensure accuracy. The authors take full responsibility for the final text.

References

[1] S. Chanda, S. Pal, Overview of the shared task on code-mixed information retrieval from social media data, in: Proceedings of the 16th Annual Meeting of the Forum for Information Retrieval Evaluation (FIRE 2024), Association for Computing Machinery (ACM), 2024. URL: https://doi.org/10.1145/3734947.3735670. doi:10.1145/3734947.3735670.

[2] S. Chanda, S. Pal, Overview of the shared task on code-mixed information retrieval from social media data, in: FIRE 2024 Working Notesl, CEUR Workshop Proceedings, 2024, p. 124–128. URL: https://ceur-ws.org/Vol-4054/T2-1.pdf.

[3] Y. Huang, J. Huang, Exploring chatgpt for next-generation information retrieval: Opportunities and challenges, Web Intelligence 22 (2024) 31–44. URL: https://doi.org/10.3233/WEB-230363. doi:10.3233/WEB-230363.

[4] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, M. Gatford, Okapi at trec-3, in: Proceedings of the Third Text REtrieval Conference (TREC-3), NIST, 1995. URL: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/okapi_trec3.pdf.

[5] S. Wang, et al., Text embeddings by weakly-supervised contrastive pre-training, arXiv preprint, 2022. URL: https://arxiv.org/abs/2212.03533.

[6] W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, M. Zhou, Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers, in: Advances in Neural Information Processing Systems (NeurIPS), 2020. URL: https://arxiv.org/abs/2002.10957.

[7] C. Burges, et al., Learning to rank using gradient descent, in: Proceedings of the International Conference on Machine Learning (ICML), 2005. URL: https://www.microsoft.com/en-us/research/publication/learning-to-rank-using-gradient-descent/.

[8] S. Mandal, K. Nanmaran, Normalization of transliterated words in code-mixed data using seq2seq model & levenshtein distance, in: Proceedings of the EMNLP Workshop on Noisy User-generated Text (W-NUT), 2018. URL: https://aclanthology.org/W18-6107.

[9] A. Conneau, et al., Unsupervised cross-lingual representation learning at scale, in: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2020. URL: https://aclanthology.org/2020.acl-main.747.

[10] V. Karpukhin, et al., Dense passage retrieval for open-domain question answering, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2020. URL: https://aclanthology.org/2020.emnlp-main.550.

[11] S. Mandal, S. Ghosh, M. Mitra, M. Mandal, Overview of the shared task on code-mixed information retrieval (cmir) at fire 2021, in: Proceedings of FIRE 2021 Working Notes, volume 3159 of *CEUR Workshop Proceedings*, 2021. URL: http://ceur-ws.org/Vol-3159/T1-1.pdf.

[12] N. Reimers, I. Gurevych, Sentence-bert: Sentence embeddings using siamese bert-networks, in: Proceedings of the EMNLP/IJCNLP, 2019. URL: https://aclanthology.org/D19-1410.

[13] A. Askari, A. Abolghasemi, G. Pasi, W. Kraaij, S. Verberne, Injecting the bm25 score as text improves bert-based re-rankers, arXiv preprint, 2023. URL: https://arxiv.org/abs/2301.09728.

[14] Y. Doval, M. Vilares, J. Vilares, On the performance of phonetic algorithms in microtext normalization, Expert Systems with Applications 113 (2018) 213–222. URL: https://doi.org/10.1016/j.eswa.2018.07.016. doi:10.1016/j.eswa.2018.07.016.

[15] Y. Malviya, K. Dhingra, M. Singh, Mst-r: Multi-stage tuning for retrieval systems and metric evaluation, arXiv preprint, 2023. URL: https://arxiv.org/abs/2412.10313.

[16] N. Dandekar, Pointwise vs pairwise vs listwise learning to rank, Medium, 2022. URL: https://medium.com/@nikhilbd/pointwise-vs-pairwise-vs-listwise-learning-to-rank-80a8fe8fadfd.
[17] G. de Souza P. Moreira, et al., Enhancing q&a text retrieval with ranking models: Benchmarking, fine-tuning and deploying rerankers for rag, arXiv preprint, 2024. URL: https://arxiv.org/abs/2409.07691.
[18] X. Ren, et al., Rocketqa-v2: A joint training method for dense passage retrieval and passage re-ranking, 2021. URL: https://www.researchgate.net/publication/355219310_RocketQAv2_A_Joint_Training_Method_for_Dense_Passage_Retrieval_and_Passage_Re-ranking.
[19] P. Lewis, et al., Retrieval-augmented generation for knowledge-intensive nlp tasks, in: Advances in Neural Information Processing Systems (NeurIPS), 2020. URL: https://arxiv.org/abs/2005.11401.
[20] R. Nogueira, W. Yang, K. Cho, J. Lin, Multi-stage document ranking with bert, arXiv preprint, 2019. URL: https://arxiv.org/abs/1910.14424.
[21] J. Guo, et al., A deep look into neural ranking models for information retrieval, Information Processing & Management 57 (2020) 102067. URL: https://doi.org/10.1016/j.ipm.2019.102067. doi:10.1016/j.ipm.2019.102067.
[22] K. Järvelin, J. Kekäläinen, Cumulated gain-based evaluation of ir techniques, ACM Transactions on Information Systems 20 (2002) 422–446. URL: https://doi.org/10.1145/582415.582418. doi:10.1145/582415.582418.
[23] A. Singh, et al., Low-resource neural ir for indic languages, in: Proceedings of IC2SDT, 2022.
[24] R. Bhatt, et al., Efficient keyword-based search in regional news archives, in: Proceedings of IC2SDT, 2023.
[25] S. Sharma, R. Tiwari, Domain adaptation for code-mixed ir, in: Proceedings of IC2SDT, 2024.
[26] S. Goswami, et al., Anomaly detection in social media streams using semi-supervised lda, DRDO Journal, 2020.
[27] S. Goswami, et al., Language-agnostic entity recognition for military surveillance, in: Proceedings of IC2SDT, 2023.
[28] S. Chanda, K. Tewari, S. Pal, Findings of the code-mixed information retrieval from social media data (cmir) shared task at fire 2025, Forum for Information Retrieval Evaluation (Working Notes), CEUR-WS.org, 2025.
[29] S. Chanda, K. Tewari, S. Pal, Overview of the cmir track at fire 2025: Code-mixed information retrieval from social media data, in: Proceedings of the 17th Annual Meeting of the Forum for Information Retrieval Evaluation (FIRE 2025), Association for Computing Machinery (ACM), 2025.
[30] S. Chanda, S. Pal, The effect of stopword removal on information retrieval for code-mixed data obtained via social media, SN Computer Science 4 (2023) 20. doi:10.1007/s42979-023-01942-7.
[31] M. Kumbhar, K. Thakre, Language identification and transliteration approaches for code-mixed text, Journal of Engineering Science and Technology Review 17 (2024) 63–70. URL: https://doi.org/10.25103/jestr.171.09. doi:10.25103/jestr.171.09.