InfoTextCM: Addressing Code-Mixed Data Retrieval Challenges via Text Classification

Aditya Sharma

Birla Institute of Technology and Sciences, Pilani

Abstract

Code-mixed IR in Bengali and English requires specialized methodologies, as conventional IR systems often face challenges due to the complexity of code-mixing, including language ambiguity, syntactic diversity, and vocabulary overlap. This paper evaluates several approaches for code-mixed retrieval, focusing on using advanced techniques such as Sentence-BERT. Specifically, we tackle the task of retrieving relevant documents for code-mixed Bengali-English queries from a large corpus. We explore various models, including traditional retrieval techniques like BM25 and PL2, and compare them against transformer-based approaches like Sentence-BERT. Additionally, we investigate the effectiveness of graph neural network-based models, leveraging semantic relationships to enhance retrieval performance. Our findings demonstrate that transformer-based models are able to handle the intricacies of code-mixed language, providing more accurate and context-aware retrieval results.

Keywords

Code-mixed, Information-retrieval, SBERT, GNN

1. Introduction

In recent years, multilingual and code-mixed material has become much more common on blogs, social media, and other digital platforms. In multilingual cultures, code-mixing—the practice of speakers switching between two or more languages during a single text or conversation—is particularly common. Bengali-English is one of the most common code-mixed language pairings in Bangladesh and India. It offers a distinct set of possibilities and problems for IR (IR) tasks in natural language processing (NLP). Conventional IR systems are primarily made for monolingual datasets; however, as code-mixed language becomes more common, new methods that can handle multilingual and mixed-lingual situations must be developed.

The work of IR becomes challenging in the case of Bengali-English code-mixing because of the morphological complexity of Bengali and the grammatical structural disparities between Bengali and English. English has a more rigorous word order and uses less inflection than Bengali, which is an inflectional language where the meaning of words frequently alters depending on their place in a sentence. When users mix these languages, the retrieval system must efficiently manage tokenization, parsing, and comprehension of both languages in real-time. Transliteration, in which Bengali words are transcribed using the English alphabet, presents additional difficulties since it makes it challenging to distinguish between the languages during processing. For instance, "tin" means "three" in Bengali, but "metal container" or "chemical element" in English. Furthermore, the Bengali word "tin" may also be transcribed as "teen," which again connotes "youngster" in English.

While much previous research has concentrated on creating language models and retrieval systems for specific languages, the code-mixed scenario calls for a blend of linguistics, machine learning, and natural language processing (NLP) methods. The challenges of dealing with code-mixed data have shown potential for techniques like language identification, transliteration normalization, and multilingual embeddings. Still, little is known about how these methods may be used in an IR system, particularly for low-resource languages like Bengali.

This paper aims to explore and propose solutions for the task of IR in code-mixed Bengali-English data. We investigate using a fine-tuned large language model, particularly BERT, to improve retrieval

Forum for Information Retrieval Evaluation, December 12-15, 2024, India

performance in this challenging context. By tackling the unique linguistic challenges posed by code-mixed data, we hope to contribute to developing more inclusive and effective IR systems for multilingual and code-mixed communities.

2. Task Description

The task is to create an information retrieval (IR) model capable of handling previously unseen queries effectively. To mathematically formulate the problem of IR, we can define the components involved in the retrieval process. Let's denote:

- D: a set of documents, where $D = d_1, d_2, \ldots, d_N$, N is the total number of documents in the corpus.
- Q: a set of queries, where $Q = q_1, q_2, \ldots, q_M$ and M is the total number of queries.
- $R(q, d)$: a relevance function that scores the relevance of document d with respect to query q.

The objective is to retrieve the most relevant documents for each query in Q from the corpus D. For each query $q_i \in Q$, we want to rank the documents in D based on their relevance scores. This can be formulated as:

$$R_i(d_j) = R(q_i, d_j) \tag{1}$$

The documents are then ranked based on the computed relevance scores:

$$Rank(q_i) = sort(R_i(d_1), R_i(d_2), \ldots, R_i(d_N)) \tag{2}$$

where the sorting is performed in descending order of relevance scores. For a given threshold T, we can define a retrieval function that selects documents above this threshold:

$$Retrieved(q_i) = d_j \in DR_i(d_j)T \tag{3}$$

This configuration mimics real-world IR problems when models have to find the most pertinent articles from sizable corpora in response to user queries that the model hasn't seen in training.

3. Related Work

Code-mixed language processing has attracted a lot of attention since multilingual interactions are becoming more and more common in casual communication, especially on social media. Transformers, including BERT[1], have shown promise in various tasks involving code-mixed languages. For example, a study by [2] focused on transformer-based language identification for Malayalam-English code-mixed text, demonstrating the model's efficacy in code-mixed IR tasks. [3] examines the performance and behaviour of BERT in ranking tasks, specifically focusing on MS MARCO passage reranking and TREC Web Track ad hoc document ranking. BERT distributes attention between query-document tokens and favours semantic matches between paraphrase tokens, distinguishing itself from other neural rankers. The authors concluded that BERT demonstrates strong effectiveness in question-answering focused passage ranking tasks (MS MARCO) but is less effective in ad hoc document ranking (TREC).

BERT and its variants have been extensively used for handling code-mixed text. For instance, [4] utilized a multilingual BERT for sentiment analysis of COVID-19 vaccination-related code-mixed Bangla text from social media. Another study introduced Tri-Distil-BERT [5], a multilingual model pre-trained on Bangla, English, and Hindi, specifically fine-tuned for code-mixed text. Handling code-mixed text poses unique challenges, such as language identification, transliteration, and context switching. [6] introduced a character-level inclusive transformer architecture to improve IR in low-resource code-mixed languages, demonstrating the need for specialized models. SetBERT [7] is a specialized version of the BERT model, fine-tuned to improve query embeddings for set operations and Boolean logic queries like

Intersection (AND), Difference (NOT), and Union (OR). Traditional and neural retrieval methods often struggle with these types of queries, but SetBERT addresses this gap by using an innovative inversed contrastive loss to identify negative sentences. Combined with fine-tuning on a dataset generated via prompt GPT, this approach significantly enhances retrieval performance. [8] proposes a Graph Embedding-based Ranking Model for Product Search (GEPS) that uses graph embedding techniques to incorporate graph-structured data into neural retrieval models. The approach helps overcome the long-tail problem of click-through data and incorporates external heterogeneous information to improve search results. [9]addresses the high computational complexity of re-ranking in image retrieval and suggests a Graph Neural Network (GNN) method to increase effectiveness. Retrieving high-quality samples and upgrading features make up the re-ranking process. Creating a k-nearest neighbour graph is the first step, and message dissemination inside the graph is the second.

4. Background

4.1. Traditional Information Retrieval models

Term Frequency-Inverse Document Frequency (TF-IDF) is a statistical measure that assesses the significance of a term in a document relative to a corpus. It consists of two components: Term Frequency (TF), which quantifies how often a term appears in a document, and Inverse Document Frequency (IDF), which reflects the term's rarity across the corpus. A higher TF-IDF score indicates greater relevance of the term to the document. BM25 (Best Matching 25) [10] is a probabilistic retrieval model that enhances TF-IDF by incorporating term saturation and document length normalization. It utilizes parameters to adjust the influence of term frequency and document length, improving the effectiveness of document ranking. PL2 (Poisson Model) [11] is a probabilistic retrieval model based on the Poisson distribution, which assesses the likelihood of observing a term in a document given its frequency across the collection. This approach provides a statistical basis for scoring document relevance. InL2 (Independence Language Model) [12] builds on the independence language modelling approach, estimating the probability of generating a query from a document's language model. It employs smoothing techniques to address the presence of terms not found in the document. Hiemstra_LM [13] extends language modelling in IR, focusing on the relationship between query terms and document relevance within a probabilistic framework. It emphasizes query likelihood and employs a variant of Dirichlet smoothing to refine the document's language model based on overall collection statistics.

4.2. Sentence-BERT

SBERT [14] uses a Siamese neural network made up of two BERT [1] sub-models with identical configurations, parameters, and weights to evaluate semantic similarity on a large scale efficiently. Also, the updating of parameters occurs simultaneously in both sub-models. SBERT, like BERT, also trains pairwise, but it doesn't compare every pair. Instead, it applies a triplet loss function to compare an anchor baseline input with a positive (true) and negative (false) input. To achieve the goal, the spatial distance between the anchor and negative input must be increased as much as possible, while the spatial distance between the anchor and positive input must be reduced. The triple loss function can be written as:

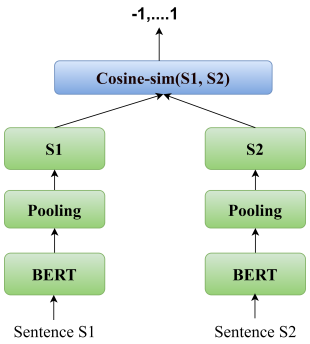

Figure 1: Sentence-BERT to compute similarity score

$$L(A, P, N) = max(\|f(A) - f(P)\|^2 - \|f(A) - f(N)\|^2 + \alpha, 0) \tag{4}$$

where α indicates the buffer(margin) used to increase the gap between similar and different triplets., $f(A), f(P), f(N)$ are embeddings for anchor, positive and negative sentences.

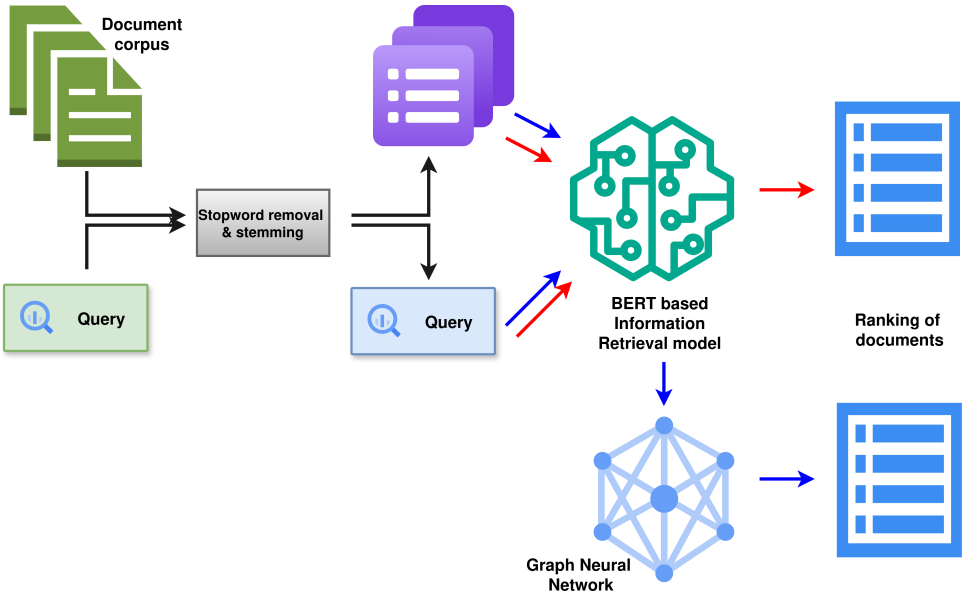

Figure 2: Framework of code-mixed information retrieval using SBERT and Graph Neural Networks. The black arrows show (→) the preprocessing steps common to both workflows. Red arrows (→) represent ranking using a fine-tuned SBERT model, and blue arrows (→) represent first ranking documents using a fine-tuned SBERT model and then re-ranking it using graph neural networks.

SBERT incorporates a pooling operation to the output of BERT models, which involves computing the mean of all the output vectors. This process results in deriving fixed-sized embeddings that are semantically meaningful for each sentence. The embeddings produced exhibit the property whereby sentences with similar meanings are situated in close proximity to one another within the vector space. Following pooling, we get two embeddings that correspond to both sentences. These two embeddings are then concatenated and trained using the softmax loss function. After training, when the model makes a prediction, it compares both embeddings using cosine similarity, resulting in a similarity score for this pair of sentences.

4.3. Graph Neural Networks

Graph Neural Networks (GNNs) are a class of neural networks designed to operate on graph-structured data, effectively capturing the relationships and dependencies among nodes. The core of a GNN involves two key processes: message passing and node aggregation. For a given node v in a graph $G = (V, E)$, the propagation of information can be expressed as:

$$h_v^k = \sigma\left(W^{(k)} \sum_{u \in N(v)} h_u^{(k-1)} + b^{(k)}\right) \tag{5}$$

Here, $h_v^{(k)}$ represents the feature vector of node v at layer k, $W^{(k)}$ is the learnable weight matrix, $b^{(k)}$ is a bias term, and $N(v)$ denotes the set of neighboring nodes of v. The aggregation function typically sums the features of neighbouring nodes, allowing each node to update its representation based on its local neighbourhood. The final node representations can then be transformed through a readout function, enabling downstream tasks such as node classification, link prediction, or graph classification. This framework allows GNNs to learn complex patterns and relationships within graph data, making them powerful tools in various applications like social networks, molecular chemistry, and recommendation systems.

5. Proposed approaches

5.1. Dataset

The dataset comprises 107,900 multilingual documents in TREC format, structured akin to HTML tags, in both Bengali and English [15]. Every document has a distinct identifier assigned to it. A train set of 20 queries/topics, also in code-mixed English and Bengali, are provided to help train the models. Additionally, a document with human annotations providing relevance feedback is provided, which indicates the relevance or non-relevance of documents related to a particular query.

Number of documents	107900
Average document length	12.58
Number of tokens	1358402
Number of unique terms	81414
Number of postings	1203258
Number of queries	20

Table 1
Statistics of dataset

5.2. Effect of different preprocessing steps

5.2.1. Stopword removal

Taking motivation from [16], we explored the effect of stopword removal in relevant document extraction from a corpus. [16] classifies stopword removal methods into corpus-based and non-corpus-based. While the corpus-based techniques are applied to the current corpus to extract a set of stopwords specific to the current corpus, the non-corpus-based techniques may be part of some pre-generated libraries or sources and are not dependent on the document collection at hand. Authors proposed that corpus-based stopword removal techniques outperform non-corpus-based stopword removal techniques. On a similar note, we performed experiments with stopword removal corpus made available by NLTK library (both English and Bengali words), terrier platform (English only), SMART library (English only), and Custom stopwords library prepared by [16] using tf-idf approach.

5.2.2. Stemming

After adding stemmer to the best-performing stopword removal method in the previous section, we used "pt.TerrierStemmer.porter" English stemmer along with best-performing "custom-corpus" to perform stemming on English tokens (words) only. We further evaluated the performance of base models by preprocessing both the query and documents, which marginally improved the performance of these models. The results are compiled in Table 9.

5.3. Reranking traditional models with pre-trained models

In the second phase of experiments, we aimed to enhance the rankings by leveraging transformer-based pre-trained models, specifically focusing on Sentence-BERT (SBERT) [14]. Few models that we compared are Mixed-DistilBERT[17], IndicSBERT [18], LaBSE [19], hkunlp-Instructor [20].

Initially, we captured the document rankings from the top-performing base model and conducted further re-ranking using different SBERT-based models. For our subsequent re-ranking experiments, we chose the rankings obtained from Hiemstra-LM as the foundation. We compared various SBERT models to assess their performance in code-mixed IR. The outcomes of these comparisons are summarized in a table 10.

Model name	MAP	Recip rank	NDCG	P@5	P@10
TF_IDF	0.186	0.667	0.412	0.35	0.28
BM25	0.191	0.677	0.417	0.37	0.26
PL2	0.181	0.657	0.403	0.34	0.25
INL2	0.196	0.672	0.422	0.4	0.28
Himestra_LM	0.25	0.85	0.49	0.51	0.35

Table 2
NLTK English corpus only

Model name	MAP	Recip rank	NDCG	P@5	P@10
TF_IDF	0.2	0.722	0.428	0.4	0.295
BM25	0.2	0.718	0.43	0.39	0.29
PL2	0.193	0.713	0.417	0.38	0.29
INL2	0.213	0.723	0.439	0.42	0.32
Himestra_LM	0.257	0.81	0.5	0.51	0.36

Table 3
NLTK Bengali corpus only

Model name	MAP	Recip rank	NDCG	P@5	P@10
TF_IDF	0.213	0.762	0.45	0.43	0.31
BM25	0.215	0.787	0.455	0.42	0.3
PL2	0.207	0.753	0.439	0.42	0.27
INL2	0.224	0.792	0.463	0.46	0.31
Himestra_LM	0.268	0.857	0.52	0.50	0.38

Table 4
NLTK English + Bengali corpus both

Model name	MAP	Recip rank	NDCG	P@5	P@10
TF_IDF	0.192	0.676	0.426	0.35	0.28
BM25	0.193	0.673	0.429	0.35	0.29
PL2	0.188	0.677	0.422	0.33	0.28
INL2	0.204	0.703	0.439	0.38	0.29
Himestra_LM	0.267	0.842	0.516	0.53	0.38

Table 5
SMART English corpus only

Model name	MAP	Recip rank	NDCG	P@5	P@10
TF_IDF	0.191	0.688	0.419	0.35	0.27
BM25	0.193	0.695	0.423	0.35	0.27
PL2	0.188	0.695	0.415	0.33	0.25
INL2	0.202	0.718	0.432	0.39	0.28
Himestra_LM	0.261	0.833	0.505	0.51	0.37

Table 6
Terrier English corpus only

Model name	MAP	Recip rank	NDCG	P@5	P@10
TF_IDF	0.232	0.747	0.494	0.47	0.32
BM25	0.233	0.75	0.495	0.48	0.32
PL2	0.225	0.746	0.487	0.46	0.31
INL2	0.24	0.756	0.502	0.49	0.34
Himestra_LM	**0.28**	**0.875**	**0.554**	**0.55**	**0.4**

Table 7
Custom corpus - English + Bengali both

Model name	MAP	Recip rank	NDCG	P@5	P@10
TF_IDF	0.24	0.746	0.506	0.46	0.35
BM25	0.241	0.746	0.507	0.47	0.35
PL2	0.23	0.702	0.494	0.44	0.34
INL2	0.248	0.758	0.514	0.5	0.37
Himestra_LM	**0.285**	**0.898**	**0.564**	**0.55**	**0.4**

Table 8
Custom corpus + Stemmer

Model name	MAP	Recip rank	NDCG	P@5	P@10
TF_IDF	0.241	0.75	0.506	0.47	0.35
BM25	0.242	0.75	0.507	0.48	0.36
PL2	0.231	0.71	0.494	0.45	0.35
INL2	0.249	0.764	0.514	0.5	0.37
Himestra_LM	**0.287**	**0.898**	**0.565**	**0.55**	**0.42**

Table 9
Preprocessing both query and documents

Model name	MAP	Recip rank	NDCG	P@5	P@10
Mixed DistilBERT	0.101	0.373	0.355	0.1	0.125
LaBSE	**0.214**	0.458	**0.452**	**0.35**	**0.25**
Instruct-LM	0.118	0.232	0.351	0.1	0.15
Indic-Sbert multilingual	0.142	**0.633**	0.405	0.25	0.175

Table 10
Re-ranking using pre-trained SBERT

Model name	MAP	Recip rank	NDCG	P@5	P@10
Mixed DistilBERT	**0.23**	**0.63**	**0.47**	**0.5**	**0.325**
LaBSE	0.0.05	0.04	0.24	0.1	0.05
Instruct-LM	0.19	0.63	0.44	0.35	0.275
Indic-Sbert multilingual	0.115	0.35	0.37	0.3	0.275

Table 11
Ranking using fine-tuned SBERT

5.4. Ranking using fine-tuned SBERT models

In addition to reordering the results generated by a base IR model, we also experimented with conducting retrieval using SBERT models exclusively. To do this, we fine-tuned the SBERT models on the code-mixed corpus. We utilized the Information_Retrieval_Evaluator function from the SBERT library to enable the model to learn suitable weights for enhancing the retrieval results. The outcomes of this process have been consolidated in a table 11. We noticed that the performance of some models declined after fine-tuning. This could be due to the initial language in which the model was pre-trained. Fine-

tuning a model pre-trained in a different language with a code-mixed Bengali-English corpus caused further confusion, leading to a decline in performance.

5.5. Rerank results from fine-tuned SBERT using GNN

To further improve the ranking results obtained from finely-tuned Sentence-BERT (SBERT) models, we explored an innovative approach by integrating graph neural networks (GNN). Initially, we selected the most promising results from our best-performing SBERT model as a baseline. These selected results were then fed into a GNN model, aiming to leverage its capability to understand and process data in a graph-structured form. This approach was motivated by the potential of GNNs to capture complex relationships and interactions within the data, which we believed could significantly enhance the ranking accuracy. An overview of the proposed approach has been compiled in figure 2.

6. Experiments and results

The findings from the impact of different preprocessing and stemming techniques have been summarized in tables 2 to table 9. Tables 10 and table 11 display the outcomes of re-ranking the base model results using SBERT and conducting independent ranking using an SBERT-based IR model. The performance of the GNN-based model in re-ranking SBERT results was notably poor. Despite our high expectations for this methodology, the outcomes were disappointing. The integration of the GNN model did not yield the anticipated improvement in the ranking results. Instead, the performance was suboptimal, falling short of our initial projections. This experience has led us to reflect on the potential mismatches between the problem at hand and the model's suitability, or possibly the need for further tuning and optimization of the GNN parameters. Moving forward, we intend to conduct a thorough analysis to understand the underlying reasons for this performance gap and explore alternative strategies or adjustments to the GNN model that could potentially rectify these issues and enhance the overall effectiveness of our approach.

7. Conclusion

Code-mixed IR in Bengali and English entails extracting helpful information from a dataset whose content combines both languages. This is especially prevalent in multilingual cultures like India and Bangladesh, where people often transition between languages, sometimes within the same phrase.

This paper explored the performance of various techniques for code-mixed Bengali-English IR, including traditional models, pre-trained transformer-based Sentence-BERT (SBERT), and graph neural networks (GNN). While transformer-based models showed strong retrieval performance, the GNN models did not meet expectations. This opens up avenues for future research, where we aim to investigate and address the limitations GNN models face in handling the complexities of code-mixed languages, intending to improve their effectiveness in IR tasks.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] J. Devlin, M. Chang, K. Lee, K. Toutanova, BERT: pre-training of deep bidirectional transformers for language understanding, CoRR abs/1810.04805 (2018).

[2] S. Thara, P. Poornachandran, Transformer based language identification for malayalam-english code-mixed text, IEEE Access 9 (2021) 118837–118850. doi:10.1109/ACCESS.2021.3104106.

[3] Y. Qiao, C. Xiong, Z. Liu, Z. Liu, Understanding the behaviors of bert in ranking, 2019. URL: https://arxiv.org/abs/1904.07531. arXiv:1904.07531.

[4] M. R. Khan, S. Nawsad Rahmatullah, M. F. Islam, A. R. M. Kamal, M. A. Hossain, Sentiment analysis of covid-19 vaccination in bangla language with code-mixed text from social media, in: 2022 12th International Conference on Electrical and Computer Engineering (ICECE), 2022, pp. 76–79. doi:10.1109/ICECE57408.2022.10088478.

[5] M. N. Raihan, D. Goswami, A. Mahmud, Mixed-distil-bert: Code-mixed language modeling for bangla, english, and hindi, 2024. URL: https://arxiv.org/abs/2309.10272. arXiv:2309.10272.

[6] R. S. Bhowmick, I. Ganguli, J. Sil, Character-level inclusive transformer architecture for information gain in low resource code-mixed language, Neural Computing and Applications (2022). URL: https://api.semanticscholar.org/CorpusID:247376379.

[7] Q. Mai, S. Gauch, D. Adams, Setbert: Enhancing retrieval performance for boolean logic and set operation queries, 2024. URL: https://arxiv.org/abs/2406.17282. arXiv:2406.17282.

[8] Y. Zhang, D. Wang, Y. Zhang, Neural ir meets graph embedding: A ranking model for product search, 2019. URL: https://arxiv.org/abs/1901.08286. arXiv:1901.08286.

[9] X. Zhang, M. Jiang, Z. Zheng, X. Tan, E. Ding, Y. Yang, Understanding image retrieval re-ranking: A graph neural network perspective, 2020. URL: https://arxiv.org/abs/2012.07620. arXiv:2012.07620.

[10] S. Robertson, H. Zaragoza, The probabilistic relevance framework: Bm25 and beyond, Found. Trends Inf. Retr. 3 (2009) 333–389. URL: https://doi.org/10.1561/1500000019. doi:10.1561/1500000019.

[11] A. Singhal, C. Buckley, M. Mitra, Pivoted document length normalization, in: Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '96, Association for Computing Machinery, New York, NY, USA, 1996, p. 21–29. URL: https://doi.org/10.1145/243199.243206. doi:10.1145/243199.243206.

[12] G. Amati, C. J. Van Rijsbergen, Probabilistic models of information retrieval based on measuring the divergence from randomness, ACM Transactions on Information Systems (TOIS) 20 (2002) 357–389.

[13] D. Hiemstra, Using language models for information retrieval (2001).

[14] N. Reimers, I. Gurevych, Sentence-bert: Sentence embeddings using siamese bert-networks, CoRR abs/1908.10084 (2019). URL: http://arxiv.org/abs/1908.10084. arXiv:1908.10084.

[15] S. Chanda, S. Pal, Overview of the shared task on code-mixed information retrieval from social media data, in: Forum of Information Retrieval and Evaluation FIRE-2024, 2024.

[16] S. Chanda, S. Pal, The effect of stopword removal on information retrieval for code-mixed data obtained via social media, SN Comput. Sci. 4 (2023) 494. URL: https://doi.org/10.1007/s42979-023-01942-7. doi:10.1007/S42979-023-01942-7.

[17] M. N. Raihan, D. Goswami, A. Mahmud, Mixed-distil-bert: Code-mixed language modeling for bangla, english, and hindi, arXiv preprint arXiv:2309.10272 (2023).

[18] S. Deode, J. Gadre, A. Kajale, A. Joshi, R. Joshi, L3cube-indicsbert: A simple approach for learning cross-lingual sentence representations using multilingual bert, 2023. arXiv:2304.11434.

[19] F. Feng, Y. Yang, D. Cer, N. Arivazhagan, W. Wang, Language-agnostic bert sentence embedding, 2020. arXiv:2007.01852.

[20] H. Su, W. Shi, J. Kasai, Y. Wang, Y. Hu, M. Ostendorf, W. tau Yih, N. A. Smith, L. Zettlemoyer, T. Yu, One embedder, any task: Instruction-finetuned text embeddings, 2023. URL: https://arxiv.org/abs/2212.09741. arXiv:2212.09741.