**BACHELOR OF SCIENCE IN COMPUTER SCIENCE AND ENGINEERING**

**Sadia Alam Nishita**
**190041105**
**Navid Hasin Alvee**
**190041109**
**Md. Shahnewaz Siddique**
**190041115**

**Department of Computer Science and Engineering**
Islamic University of Technology
June, 2024

**Sadia Alam Nishita**
**190041105**
**Navid Hasin Alvee**
**190041109**
**Md. Shahnewaz Siddique**
**190041115**

**Department of Computer Science and Engineering**
Islamic University of Technology
June, 2024

# Declaration of Candidate

This is to certify that the work presented in this thesis is the outcome of the analysis and experiments carried out by **Sadia Alam Nishita**, **Navid Hasin Alvee**, and **Md. Shahnewaz Siddique** under the supervision of **Dr. Md. Azam Hossain**, Associate Professor, Department of Computer Science and Engineering, Islamic University of Technology, Dhaka, Bangladesh. It is also declared that neither this thesis nor any part of it has been submitted anywhere else for any degree or diploma. Information derived from the published and unpublished work of others have been acknowledged in the text and a list of references is given.

**Dr. Md. Azam Hossain**
Associate Professor
Department of Computer Science and Engineering
Islamic University of Technology (IUT)
Date: June 04, 2024

**Sadia Alam Nishita**
Student ID: 190041105
Date: June 04, 2024

**Navid Hasin Alvee**
Student ID: 190041109
Date: June 04, 2024

**Md. Shahnewaz Siddique**
Student ID: 190041115
Date: June 04, 2024

# Contents

# List of Figures

# List of Tables

# Acknowledgement

We are profoundly grateful to our supervisors, **Dr. Md. Azam Hossain**, Associate Professor, Department of Computer Science and Engineering, Islamic University of Technology, and **Md Farhan Ishmam**, Lecturer, Department of Computer Science and Engineering, Islamic University of Technology. Their boundless patience, insightful critiques, and invaluable guidance have been instrumental in the successful completion of this thesis. Their expertise and unwavering encouragement have motivated us to persevere through challenges, and their motivational talks, often disguised as time management lectures, have been a constant source of inspiration. Their steadfast belief in our abilities kept us moving forward, even during moments of self-doubt.

We extend our heartfelt gratitude to our parents, whose love and support have been our anchor throughout this journey. Thank you for always being there, for patiently listening to our lengthy explanations of our research, and for your unwavering patience. Your belief in us has been a source of great strength and resilience.

To all of you, we extend our deepest appreciation and gratitude. Your support has been invaluable, and we are forever grateful for your encouragement along this journey.

# Abstract

Code-mixed data, blending two or more languages within sentences, offers valuable insights for low-resource languages like Bengali, which have limited annotated corpora. While sentiment analysis has been widely explored in various languages, code-mixed Bengali remains underrepresented, lacking a comprehensive benchmark dataset. To address this gap, we introduce BNSENTMIX, a sentiment analysis dataset for code-mixed Bengali-English, consisting of 20,000 samples annotated with four sentiment labels: positive, negative, neutral, and mixed. The data was sourced from e-commerce websites, YouTube, and Facebook, ensuring linguistic diversity and reflecting real-world, code-mixed scenarios. Our dataset captures a wide variety of user-generated content, providing robust coverage of both informal and formal language styles. We utilized a novel automated text filtering pipeline that employs fine-tuned pre-trained language models to detect and extract code-mixed samples, ensuring high-quality data. To evaluate the dataset, we applied 11 baseline approaches, ranging from traditional machine learning models to advanced transformer-based architectures. Our best model achieved an accuracy of 69.5% and an F1 score of 68.8%.

# Chapter 1

# Introduction

## 1.1 Overview

Code-mixing is a linguistic phenomenon where two or more languages are used interchangeably within the same sentence or conversation. It is distinct from code-switching, which typically involves switching between languages at sentence boundaries.[2] Code-mixing often occurs in multilingual societies, where speakers use multiple languages fluidly and unconsciously. In these settings, individuals draw upon different linguistic resources based on their cultural or linguistic backgrounds, their interlocutors, or the context of communication. Code-mixing has become increasingly prevalent in informal communication, especially on social media platforms and online forums, where speakers have the freedom to blend languages without formal linguistic constraints.[34]

Bengali-English code-mixing, in particular, has emerged as a common linguistic practice among Bengali-speaking communities, both in Bangladesh and the Indian state of West Bengal. Bengali is the seventh most spoken language globally, with over 265 million speakers, while English serves as a global lingua franca.[8] Given the historical and cultural influence of English in South Asia, it is not surprising that Bengali speakers frequently mix English into their conversations, especially in informal, digital spaces like social media. Platforms such as Facebook, YouTube, and messaging apps have become popular venues for Bengali-English code-mixed communication. Here, users effortlessly switch between Bengali and English, often incorporating English words or phrases into Bengali sentences to express modern concepts, technical terms, or simply for stylistic variation.[4]

The prevalence of Bengali-English code-mixing in these digital environments creates

unique challenges and opportunities for natural language processing (NLP), particularly in sentiment analysis. Sentiment analysis, which involves identifying and categorizing emotions, opinions, and attitudes expressed in text, is an essential task in NLP. It is widely applied in various industries, from customer service to political analysis, to gauge public opinion and emotions. While sentiment analysis is well-developed for monolingual texts in languages such as English and Chinese, it becomes far more complex when dealing with code-mixed text, where the syntax, semantics, and grammar of two languages are intertwined. Bengali-English code-mixed text, for instance, may follow Bengali grammar while incorporating English vocabulary, or vice versa, which complicates the interpretation of sentiment. [18]

Figure 1.1 shows examples of four different type of sentiment expressed through code-mixed Bengali-English language

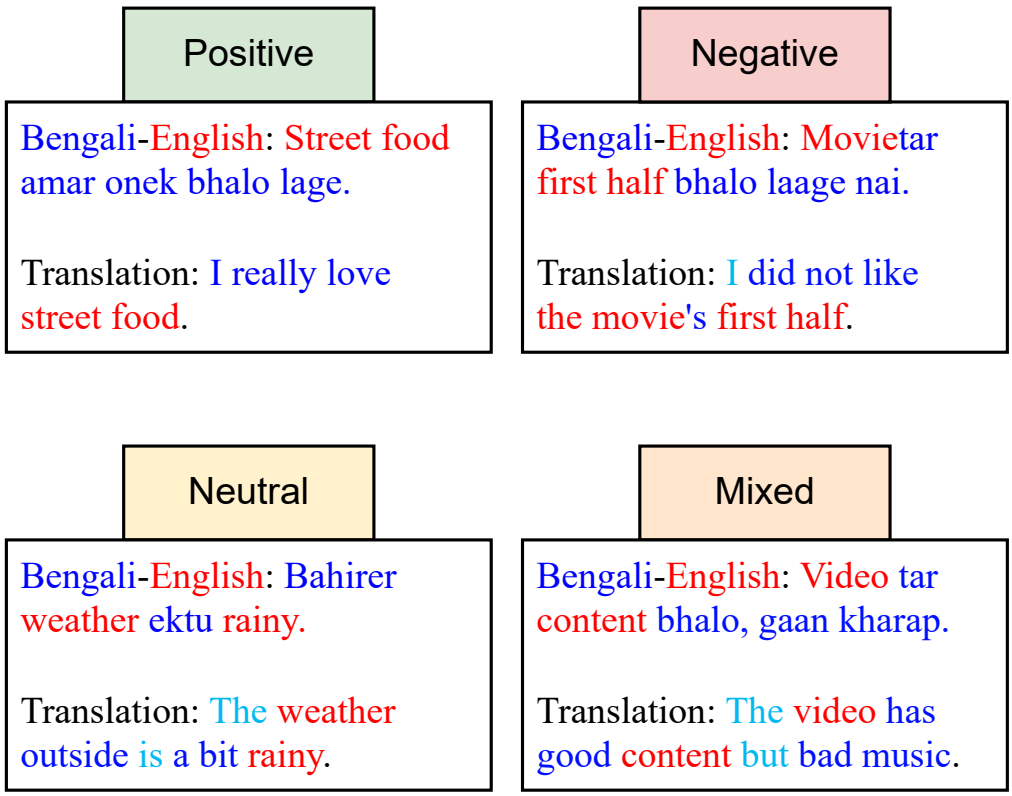



**Figure 1.1:** Examples of the four sentiment labels from our code-mixed Bengali-English dataset, and the corresponding English translations. Red represents English words, blue represents Bengali words written in English alphabets, and cyan represents implicit words in the code-mixed text.

In Bengali-English code-mixed sentences, four main types of sentiments are- positive,

negative, neutral, and mixed. These sentiments reflect various emotional tones in the text, ranging from favorable to unfavorable, or sometimes contains both.

In the case of Bengali-English code-mixing[2], the sentiment expressed in a single sentence may be influenced by the choice of language. For example, a user might switch to English to express a positive or neutral sentiment, while using Bengali to express a more emotional or negative sentiment. This interplay of languages within the same sentence adds another layer of complexity to sentiment analysis models, which are traditionally trained on monolingual data. Existing sentiment analysis models often struggle with code-mixed text due to the lack of labeled datasets that accurately reflect the structure and sentiment of mixed-language data. Current sentiment analysis systems may fail to account for the linguistic nuances of code-mixed communication, resulting in inaccurate sentiment detection.

To address these challenges, this research aims to develop a diverse Bengali-English code-mixed dataset designed specifically for sentiment analysis. The dataset will provide a resource for training machine learning models to better understand and process sentiment in Bengali-English code-mixed text. By providing high-quality labeled data, this work hopes to bridge the gap in resources for multilingual sentiment analysis, particularly for low-resource language pairs like Bengali-English.

## 1.2 Motivation and Scope

The increasing prevalence of code-mixed language, particularly in multilingual societies, has prompted a growing interest in natural language processing (NLP) tasks tailored to such data. Code-mixing refers to the practice of alternating between two or more languages within a single sentence or conversation, a phenomenon commonly observed on social media, e-commerce platforms, and other digital spaces.

As shown in Figure 1.2, code-mixing affects several key NLP tasks, including sentiment analysis, language identification, part-of-speech (POS) tagging, named entity recognition (NER), and abusive or offensive language detection. While each of these tasks is crucial for the development of robust NLP systems in multilingual environments, this thesis focuses specifically on sentiment analysis. Sentiment analysis, which involves the computational study of opinions, emotions, and attitudes expressed in text, plays an essential role in several real-world applications such as social media monitoring, customer support, market research, public opinion analysis, and healthcare. The complexities introduced by code-mixing, especially in languages like Bengali-English, make sentiment analysis a challenging yet vital area of study. Bengali-English

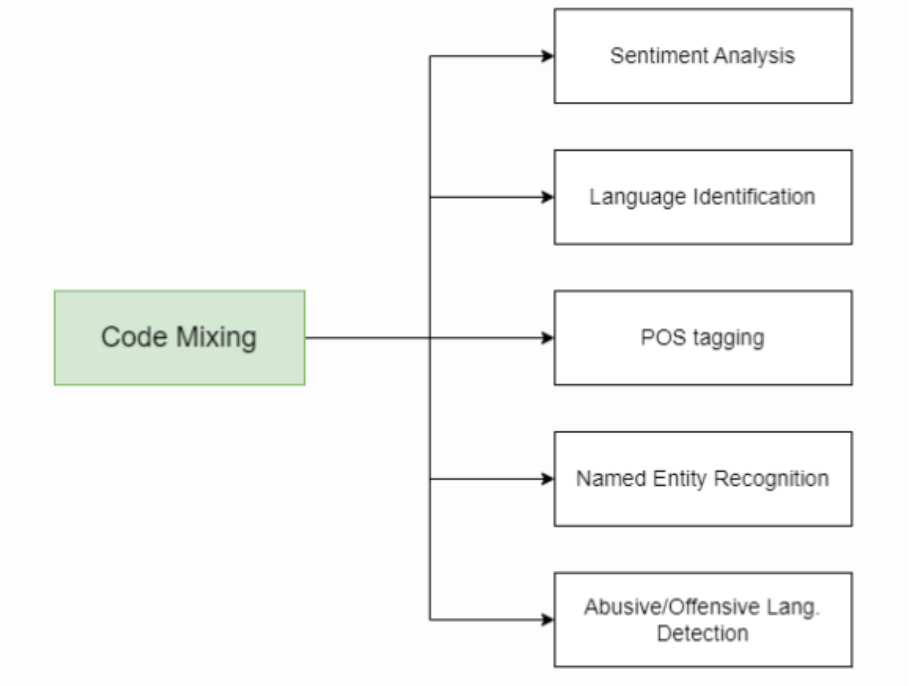



**Figure 1.2:** Different Domains of Code-Mixing in NLP Tasks

code-mixed text has become widespread, especially in user-generated content. However, despite Bengali being the seventh most spoken language globally, with over 250 million native speakers, research on Bengali-English code-mixed text remains limited, particularly in sentiment analysis.

Sentiment analysis, the computational study of opinions, emotions, and attitudes expressed in written language, is crucial for a range of applications, including but not limited to[39]:

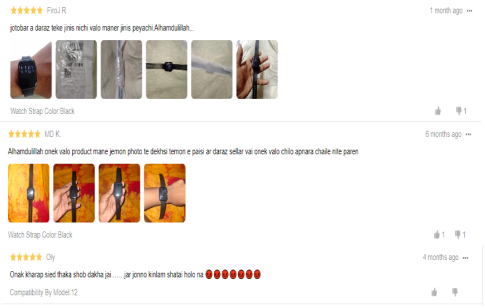
- **Social Media Monitoring:** Analyzing sentiment in social media posts allows businesses and organizations to gauge public reaction to events, products, and services.
- **Customer Support and Feedback:** Sentiment analysis aids in understanding customer emotions in real-time support interactions or product reviews, enabling businesses to enhance customer experience.
- **Market Research:** By analyzing public opinions in online forums, product reviews, and social media, companies can assess market trends and consumer preferences.
- **Government Policies and Public Opinion:** Governments and public institutions can monitor citizens' sentiment on policy decisions and important social issues, facilitating data-driven policy adjustments.

- **Healthcare:** Sentiment analysis is also used in healthcare, particularly in patient feedback, enabling healthcare providers to improve services based on emotional and sentiment cues.

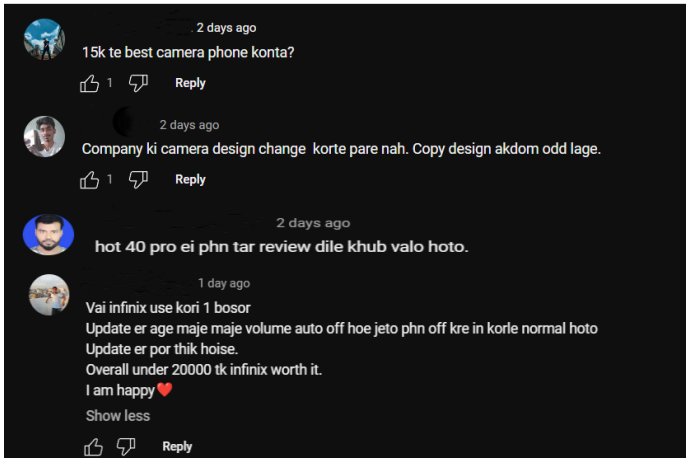
While significant advancements have been made in monolingual sentiment analysis, current models face challenges when dealing with code-mixed text due to the complex interactions between languages, especially within a single sentence or word. These challenges are amplified in Bengali-English code-mixed texts, which remain underrepresented in existing research efforts.

Figure 1.3 shows the examples of how people use code-mixed terms in social media and e-commerce sites very frequently. It is very common practice all over the world and specially in the subcontinent.


**(a)** Example from Facebook



**(b)** Example from Daraz




**(c)** Example from YouTube

**Figure 1.3:** Examples of use of Bengali-English Code-mixed sentences in social media and e-commerce websites (a) Facebook, (b) Daraz, and (c) YouTube.

Furthermore, the limited availability of large-scale, diverse datasets for Bengali-English code-mixed sentiment analysis hinders progress in this domain. Existing datasets, such as those focused on other Indic languages, are often small in scale, drawn from a single source, or not publicly accessible. This thesis aims to address these gaps by

introducing a large, diverse Bengali-English code-mixed dataset, comprising 20,000 samples labeled with four sentiment categories: positive, negative, neutral, and mixed. The dataset is curated from various sources, including social media platforms like Facebook, YouTube, and e-commerce websites, ensuring a broad linguistic representation of code-mixed text.

The primary motivation of this thesis is to enhance the understanding and development of NLP tools for low-resource languages, with a particular focus on Bengali-English code-mixing. By providing a comprehensive dataset and evaluating it using state-of-the-art models, the thesis aims to contribute to the broader research efforts in sentiment analysis for code-mixed languages. This work also introduces a novel pipeline for automated detection and filtering of code-mixed text, leveraging pre-trained language models to accurately identify and process such content. Ultimately, the goal is to facilitate the development of more inclusive and accurate sentiment analysis models that can handle the complexities of multilingual and code-mixed text, particularly in underrepresented language pairs.

## 1.3 Problem Statement

The problem addressed in this thesis revolves around the challenges associated with processing and analyzing code-mixed language data, specifically for the task of sentiment analysis. Code-mixing, the practice of alternating between two or more languages within a single sentence or discourse, has become increasingly common in digital communication. Social media platforms, e-commerce websites, and online forums frequently contain user-generated content in which Bengali and English are used interchangeably. The prevalence of such Bengali-English code-mixed text presents a significant challenge for natural language processing (NLP) tasks, particularly sentiment analysis.

Current sentiment analysis models are primarily trained on monolingual data and often struggle to handle the intricacies of code-mixed language. Code-Mixing[34] introduces unique linguistic structures and contexts that make it difficult for traditional models to accurately capture the sentiment expressed in a sentence. Moreover, the lack of publicly available, large-scale, and diverse datasets specifically tailored to Bengali-English code-mixed data further exacerbates this issue. Without such datasets, training reliable models capable of processing code-mixed text becomes a significant challenge.

Additionally, despite Bengali being one of the most spoken languages globally, it re-

mains underrepresented in NLP research, particularly in studies related to code-mixed language and sentiment analysis. The underdevelopment of NLP resources for low-resource languages like Bengali leaves a gap in addressing the linguistic diversity that exists in multilingual societies. This thesis seeks to address these challenges by creating a large, diverse dataset of Bengali-English code-mixed text for sentiment analysis and evaluating the performance of state-of-the-art models on this dataset. The goal is to bridge the gap in sentiment analysis for code-mixed text, providing resources and insights that can be applied to other low-resource, multilingual contexts as well.

## 1.4 Research Challenges

The study of code-mixed languages, particularly Bengali-English, introduces a number of research challenges that impede progress in natural language processing tasks like sentiment analysis. These challenges stem from the fact that most current NLP tools and models are designed primarily for monolingual data, and they often struggle to accurately process and interpret code-mixed text. The complexity increases when the languages involved, such as Bengali and English, have distinct linguistic structures and diverse vocabulary.

In the case of Bengali-English code-mixed text, several specific challenges need to be addressed to improve sentiment analysis models and other NLP applications. These include:

- The absence of adequately annotated datasets specific to Bengali-English code-mixed content.
- Limited availability of publicly accessible datasets for this language pair.
- Existing datasets tend to be small and derived from less varied sources, limiting their effectiveness.
- Models trained exclusively on monolingual data perform poorly when applied to code-mixed text.
- The performance of language filtering and processing tools is significantly lower for Bengali, making accurate text processing difficult.

Overcoming these challenges is crucial to advancing the ability of NLP models to effectively analyze sentiment in code-mixed text. This thesis aims to tackle these issues by building a large, diverse dataset, and evaluating its effectiveness in improving sentiment analysis performance for Bengali-English code-mixed data.

## 1.5 Contributions

This thesis addresses the significant challenges posed by Bengali-English code-mixed sentiment analysis by making several key contributions to the field. The specific contributions of this work are as follows:

- We present **BnSentMix**, a novel Bengali-English code-mixed dataset, comprising 20,000 samples annotated with 4 sentiment labels. This dataset, collected from diverse sources such as YouTube, Facebook, and e-commerce platforms, covers a wide variety of contexts and topics, providing a rich resource for sentiment analysis.
- To tackle the complexities of code-mixed text, as visualized in Fig. 1, we introduce an automated code-mixed text detection pipeline. This pipeline utilizes fine-tuned language models and achieves an impressive accuracy of 94.56%, demonstrating its effectiveness in identifying and processing code-mixed content.
- We establish 11 baselines using various approaches, including classical machine learning algorithms, neural network models, and transformer-based models. Among these, the pre-trained BERT model achieves the best performance, with an accuracy of 69.5% and an F1 score of 68.8%, setting a strong benchmark for future research in Bengali-English code-mixed sentiment analysis.

These contributions advance the understanding of sentiment analysis for code-mixed text and provide valuable resources and methodologies for future work in the field. By introducing a novel dataset and developing effective processing pipelines, this thesis lays the groundwork for improving the performance of NLP models on low-resource, multilingual data.

## 1.6 Organization

This thesis is structured to provide a comprehensive understanding of sentiment analysis for Bengali-English code-mixed text. The thesis begins with **Chapter 2**, which presents a detailed review of existing literature related to sentiment analysis, code-mixed language processing, and the challenges specific to Bengali-English code-mixing, highlighting gaps in current research. **Chapter 3** describes the creation and annotation of the novel Bengali-English code-mixed dataset used in this work, including the data collection process from various online platforms and the annotation scheme for sentiment labeling. **Chapter 4** explains the methodology and the development of a

novel automated code-mixed text detection pipeline using fine-tuned language models, designed to address the complexities of code-mixed data. In **Chapter 5**, we discuss the experimental setup and baseline models employed in the study, covering classical machine learning, neural networks, and transformer-based models, with particular attention to the model training and evaluation metrics. **Chapter 6** presents the experimental results, comparing the performance of various models on the dataset, and provides an in-depth analysis of the BERT model, which achieved the highest accuracy and F1 scores. Finally, **Chapter 7** concludes the thesis by summarizing the key findings and contributions, such as the dataset and the novel detection pipeline, while suggesting potential avenues for future research in code-mixed sentiment analysis.

# Chapter 2

# Related Works

## 2.1 Code-mixing

Code-mixed data, where speakers or writers alternate between two or more languages within a sentence or conversation, is a common phenomenon in multilingual societies. This linguistic behavior poses unique challenges for natural language processing (NLP) tasks due to the complexities introduced by language switching, differences in grammar, and the use of transliterations. Code-mixed data can be the source of several text classification tasks [35], with sentiment analysis [21] being one of the most popular ones, where the goal is to determine the sentiment expressed in a code-mixed sentence. The challenge arises from the varying sentiment polarity across languages in the same sentence, making it difficult for traditional monolingual sentiment analysis models to generalize effectively.

Other NLP tasks involving code-mixed data include hate speech detection [33], where the presence of abusive or harmful content needs to be identified across multiple languages. Translation [14] is another critical task, where code-mixed sentences are translated from one language to another, often requiring sophisticated handling of transliterated words and mixed grammar structures. Similarly, part of speech tagging [36] in code-mixed texts is more challenging than monolingual tagging because of the alternation between languages, which affects syntactic and grammatical rules.

Emotion classification [3] in code-mixed data is another area of interest, as different languages may express emotions differently, necessitating models that can understand the nuanced emotional cues in multilingual contexts. Language identification [23], the task of determining the language of a word or sentence, becomes more complex in a code-mixed setting where words from multiple languages coexist in a

single text. Finally, speech synthesis [32] in code-mixed contexts involves generating natural-sounding speech from text that seamlessly switches between languages, maintaining the fluency and pronunciation of both languages.

To address the challenges posed by code-mixed data, researchers have incorporated techniques like training data augmentation [15], [28], which helps expand the training set by creating synthetic examples that mimic code-mixed behavior. Additionally, code-mix word embeddings [25] have been developed to better represent code-mixed text by capturing the linguistic properties of both languages in a shared embedding space. These advancements help improve the performance of NLP models in processing and understanding code-mixed text, leading to better outcomes across a variety of tasks.

## 2.2 Sentiment Analysis

The significance of sentiment analysis has grown substantially with the rise of social media, where users frequently express opinions, emotions, and feedback. This surge in user-generated content has prompted extensive research, particularly on monolingual corpora. Early studies primarily focused on sentiment analysis in major languages such as English [16], [17], [37], where large corpora and well-developed language models contributed to significant advancements in this field. As sentiment analysis matured, researchers expanded their efforts to cover other languages, including Russian [30], German [9], and Norwegian [20], where sentiment classification tasks presented unique challenges due to linguistic differences and the lack of large-scale annotated datasets.

Furthermore, the field of sentiment analysis has made notable strides in Indian languages, where several studies have explored languages such as Hindi, Bengali, and Tamil [1], [27]. These studies aim to overcome the scarcity of resources available for low-resource languages and address the diverse linguistic characteristics inherent to Indian languages, such as complex syntax and rich morphology.

As the need for cross-lingual applications increased, multilingual sentiment analysis [11], [26] emerged as a key area of research. The rise of multilingual language models, such as BERT [12] and XLM-RoBERTa [10], has allowed for significant improvements in the performance of sentiment analysis across multiple languages by leveraging transfer learning and shared semantic representations. These models have revolutionized sentiment analysis, enabling models trained on one language to be effectively applied to other languages, particularly in low-resource settings.

The shift toward multilingual and cross-lingual sentiment analysis has opened up new possibilities, enabling the development of models that can process sentiment across diverse linguistic landscapes. This is particularly useful in multilingual societies where code-switching and language mixing are common, further demonstrating the importance of sentiment analysis in understanding public opinion in a globalized, interconnected world.

## 2.3 Datasets on Code-Mixed Sentiment Analysis

### 2.3.1 Malayalam-English

The paper "A Sentiment Analysis Dataset for Code-Mixed Malayalam-English" by Suryawanshi et al. presents a novel gold standard corpus for sentiment analysis of code-mixed Malayalam-English text collected from YouTube movie trailer comments [6]. The dataset contains 6,739 code-mixed comments annotated by voluntary annotators into positive, negative, neutral, mixed feeling, and non-Malayalam categories. The authors achieved a high Krippendorff's alpha of 0.890 using the nominal metric and 0.911 using the interval metric, indicating strong inter-annotator agreement. The corpus was created by downloading comments from YouTube movie trailers using the YouTube-comment-scraper tool. The authors filtered out non-code-mixed comments based on language identification at the comment level using the langdetect library. The comments were then preprocessed by removing emojis and sentences longer than 15 or shorter than 5 words. After cleaning, the authors obtained 6,738 sentences for Malayalam-English code-mixed post comments. The authors adopted the annotation approach used by Mohammad [24] and ensured that a minimum of three annotators annotated each sentence according to the following schema: positive state (the text suggests the speaker is in a positive state, such as happy, admiring, relaxed, or forgiving), negative state (the text suggests the speaker is in a negative state, such as sad, angry, anxious, or violent), mixed feelings (the text suggests the speaker is experiencing both positive and negative feelings), neutral state (there is no explicit or implicit indicator of the speaker's emotional state), and not in intended language (the sentence does not contain Malayalam). The authors used Krippendorff's alpha to measure inter-annotator agreement, a prominent method for assessing agreement between annotators. The high scores obtained—0.890 for the nominal metric and 0.911 for the interval metric—demonstrate a strong level of consistency among annotators in interpreting and applying the guidelines. Specifically, the nominal metric score indicates a high agreement in categorical decisions (e.g., positive, negative, or neu-

tral labels), while the interval metric score shows that annotators were also consistent in assessing the relative intensity within an ordered scale. These high scores suggest that the annotation instructions were clear and effectively guided annotators toward a shared understanding, enhancing the reliability of the results. The authors evaluated their dataset using various machine learning and deep learning models. Traditional machine learning algorithms such as Logistic Regression, Decision Trees, Random Forest, Multinomial Naive Bayes, and K-Nearest Neighbors were used with TF-IDF as input features. The deep learning models included Dynamic Meta-Embeddings (DME), Contextualized DME (CDME), 1D Dimensional Convolution (1DConv), and Bidirectional Encoder Representations for Transformers (BERT). The results show that all machine learning algorithms succeeded in classifying all classes except SVM, which failed to classify the non-Malayalam class correctly. The deep learning models using pre-trained embeddings performed well, with DME and CDME succeeding in identifying all classes, while BERT failed to identify the "Mixed Feeling" class. The authors note that the dataset can be used to create models specific for code-mixed data, as systems trained on monolingual data fail for code-mixed text due to the complexity of mixing at different levels [6]. The paper presents a valuable resource for sentiment analysis in code-mixed Malayalam-English text, essential for understanding viewers' emotional responses to movie trailers. The high inter-annotator agreement and benchmark results demonstrate the effectiveness of the dataset and its potential for future research in this area.

### 2.3.2 Hindi-English

The paper "Towards Sub-Word Level Compositions for Sentiment Analysis of Hindi-English Code Mixed Text" by Aditya Joshi, Ameya Prabhu, Manish Shrivastava, and Vasudeva Varma addresses the challenges of sentiment analysis in Hindi-English code-mixed text through a novel approach utilizing sub-word level representations [18]. They introduce a Long Short-Term Memory (LSTM) architecture termed Subword-LSTM, leveraging morphemes to better capture sentiment in code-mixed text. This sub-word level representation is obtained through 1-D convolutions on character inputs, enabling the model to process morpheme-like feature maps. The authors created a Hindi-English (Hi-En) code-mixed dataset from comments on popular Facebook pages, annotated with sentiment polarity. Notably, the dataset, with a size of 4981 comments, showcases spelling variations and non-standard constructions typical of social media text. Extensive experiments comparing Subword-LSTM with character-level and word-level models demonstrate its superior performance, achieving higher accuracy and outperforming existing systems by 18%. Dataset preparation involved

manual pre-processing to remove non-Roman script comments and annotators achieved substantial agreement in sentiment classification. The Subword-LSTM model's effectiveness lies in its ability to handle the noisy and variable nature of code-mixed text while capturing sentiment-related information from important morphemes. This research presents a significant advancement in sentiment analysis for Hindi-English code-mixed text, providing a valuable approach for multilingual and code-mixed language contexts. Moreover, the creation of an annotated dataset contributes to further exploration and development of sentiment analysis techniques for code-mixed languages, addressing a crucial gap in the field.

### 2.3.3 Tamil-English

The paper "Corpus Creation for Sentiment Analysis in Code-Mixed Tamil-English Text" by Chakravarthi et al. presents a significant contribution to the field of sentiment analysis for code-mixed languages [7]. The authors describe the creation of a gold standard corpus for sentiment analysis in Tamil-English code-switched text, addressing the lack of annotated data for low-resourced languages like Tamil. Sentiment analysis is a crucial task in natural language processing, particularly on social media platforms where comments and posts often contain code-mixed languages. Tamil, a Dravidian language spoken primarily in India and Sri Lanka, is one such language that lacks sufficient annotated data for sentiment analysis, which hinders the development of robust sentiment analysis models for Tamil-English code-mixed texts. The authors created a corpus of 15,744 comment posts from YouTube, which is the largest annotated dataset for sentiment analysis in Tamil-English code-switched text. The corpus was annotated with language and polarity tags by multiple annotators, ensuring high inter-annotator agreement. The annotation process involved identifying the language of each comment, whether it was Tamil or English, and then assigning a polarity label (positive, negative, or neutral) to each comment. The authors evaluated the performance of various machine learning models on the corpus, including Support Vector Machine (SVM), Decision Tree (DT), Random Forest (RF), Multinomial Naive Bayes (MNB), 1DConv-LSTM, BERT-Multilingual, DME, and CDME. The results demonstrate the effectiveness of these models in sentiment analysis for Tamil-English code-mixed texts, providing a benchmark for future research in this area. The paper presents a valuable resource for sentiment analysis in code-mixed Tamil-English text, which is essential for understanding public opinions and sentiments on various topics. The creation of this corpus and the evaluation of various machine learning models provide a foundation for further research in this area, enabling the development of more accurate and robust sentiment analysis models for

low-resourced languages like Tamil.

### 2.3.4 Bengali-English

#### Bengali-English Code-Mixed Corpus for Sentiment Analysis of Indian Languages

The paper "Preparing Bengali-English Code-Mixed Corpus for Sentiment Analysis of Indian Languages" by Soumil Mandal, Sainik Kumar Mahata, and Dipankar Das presents a comprehensive approach to creating a gold standard corpus for sentiment analysis of Bengali-English code-mixed data, crucial for addressing challenges in sentiment analysis within multilingual environments, especially Indian languages [22]. The authors collected 600 manually annotated code-mixed sentences from Twitter using a validated Bengali keyword list, filtering and cleaning them to enhance quality. The resulting corpus, with its substantial dataset size, is pivotal for robust sentiment analysis models. Their data collection process leveraged Twitter API and a hybrid system of rule-based and supervised models for language and sentiment tagging, reducing manual annotation efforts. The annotated corpus exhibits impressive inter-annotator agreement, quantified by Kappa values. Incorporating metrics like Code-Mixed Index (CMI) and Code-Mixed Factor (CF), the corpus provides insights into code-mixed and sentiment properties, alongside language and emotion aspects crucial for sentiment analysis. This contribution establishes a benchmark corpus for Bengali-English code-mixed sentiment analysis, facilitating the evaluation of sentiment analysis models. Additionally, the paper underscores the necessity of hybrid systems in multilingual sentiment analysis and suggests avenues for future research, including the development of advanced machine learning models capable of handling code-mixed data effectively and integrating additional linguistic features to enhance sentiment analysis accuracy. The creation of this corpus not only advances sentiment analysis capabilities but also contributes to the broader understanding of code-mixing phenomena and linguistic diversity in digital communication.

#### Event Detection in Bengali and Bengali-english Facebook Posts

The paper "Using Machine Learning to Detect Events on the Basis of Bengali and Banglish Facebook Posts" by Onan et al. presents a significant contribution to the field of event detection in Bengali and Banglish Facebook posts, addressing the crucial task of understanding public opinions and sentiments on various topics [13]. In the realm of natural language processing, event detection holds particular importance, especially on social media platforms where users actively share their thoughts and engage in dis-

cussions. Given the widespread usage of Bengali and Banglish languages, the authors emphasize the necessity for tailored approaches to accurately detect events within these linguistic contexts, which are often overlooked in existing literature. Employing a machine learning approach with a Bernoulli Naive Bayes classification model, the study effectively harnesses features extracted from Facebook posts to train a model capable of event detection with remarkable accuracy, achieving an impressive 90.41%. The comprehensive evaluation of the model's performance using various metrics, including accuracy, precision, recall, and F1 score, underscores its efficacy in accurately identifying events from Bengali and Banglish Facebook posts. The significance of the study lies not only in its successful application of machine learning techniques to address the challenges of event detection in low-resourced languages but also in its potential to serve as a benchmark for future research endeavors. By demonstrating the feasibility of utilizing machine learning in this context and highlighting the importance of developing robust event detection models tailored to Bengali and Banglish languages, the paper lays the groundwork for further advancements in the field. Moreover, the results offer valuable insights into the complexities of working with diverse linguistic data on social media platforms, paving the way for the development of more accurate and robust event detection models tailored to specific linguistic contexts.

### 2.3.5 Persian-English

The paper "Sentiment Analysis of Persian-English Code-mixed Texts" by Nazanin Sabri, Ali Edalat, and Behnam Bahrak presents a novel approach to sentiment analysis of code-mixed Persian-English text collected from Twitter [31]. The authors introduce a dataset of 3,640 tweets labeled with polarity values and use a combination of machine learning and deep learning models to automatically learn the polarity scores of these tweets. This study contributes to the growing body of research on sentiment analysis of multilingual and code-mixed texts, which has become increasingly important due to the widespread use of social media platforms and the need to understand user sentiment from a business and research perspective. The authors highlight the rapid growth of multilingual and code-mixed data on the internet, which has led to a need for code-mixed sentiment analysis systems. They note that previous studies have focused on monolingual sentiment detection systems, which are not effective for code-mixed texts due to the unstructured nature of social media content and the mixing of languages at different levels. The authors also discuss the challenges of sentiment analysis in code-mixed texts, including the difficulty of identifying the emotional energies of the text when words containing emotional energies are written in a

different language. The authors collect tweets using the Twitter API and label them with polarity values using a combination of automatic and manual methods. They use a dataset of Persian words from Wikipedia to identify non-Persian words in the tweets and then translate these words using Yandex and a dictionary-based approach. The authors then use pre-trained multilingual BERT embeddings to create vectorized representations of the tweets and employ an ensemble model consisting of three Bi-LSTM networks to learn the polarity scores. Here are the key results and discussion points from the paper: The authors report the following 10-fold cross-validation performance results for their ensemble model compared to baseline Naive Bayes and Random Forest models: Accuracy: 66.17%, F1 score: 63.66%. The authors' ensemble model outperforms the baseline models on all metrics. They found that the attention and pooling mechanisms in their models helped improve performance. The ensemble approach, which combines the outputs of three Bi-LSTM models using a weighted average, provides better results than each individual model alone, as each model compensates for the weaknesses of the others. The paper presents a valuable contribution to the field of sentiment analysis of code-mixed texts, particularly in the context of Persian-English code-mixed data. The authors demonstrate the effectiveness of their approach using a combination of machine learning and deep learning models and highlight the importance of considering the complexities of code-mixed texts in sentiment analysis systems. The dataset size of 3,640 tweets provides a significant resource for future research in this area, enabling further exploration and development of sentiment analysis techniques for Persian-English code-mixed texts.

### 2.3.6 Swiss-English

The paper "Multilingual Sentiment Analysis for a Swiss Gig" by Ela Pustulka-Hunt et al. presents a comprehensive study on the development of a multilingual sentiment analysis solution for a Swiss human resource company operating in the gig sector [26]. The authors examine the feasibility of using machine learning in this context, evaluating the performance of various sentiment assignment experiments on a dataset of 963 hand-annotated comments from workers and employers. The gig economy poses unique challenges in service quality assessment, particularly in understanding the sentiments of workers and employers. Sentiment analysis is crucial to provide relevant feedback and predict future participant behavior. The authors highlight the importance of accurate sentiment assignment in this scenario, where negative sentiment is harder to classify than positive sentiment. The study employs a hybrid approach combining machine learning and linguistic methods. The authors use a multilingual approach without stemming, which is more efficient and reliable in an industrial

scenario. They compare the performance of three sentiment assignment methods: a baseline using Twitter data, a hybrid solution from Semantria, and a tenfold cross-validation on the gig data. The results demonstrate the effectiveness of the hybrid approach, achieving an accuracy of 0.87, F1 score of 0.91, and Matthews correlation coefficient (MCC) of 0.65. The study shows that the hybrid approach outperforms the baseline and the hybrid solution from Semantria. The authors conclude that with more training data and some feature engineering, an industrial-strength solution to this problem should be possible. The results also highlight the importance of using standard machine learning software and avoiding language assignment or stemming, which can negatively impact prediction accuracy. The paper presents a valuable contribution to the field of multilingual sentiment analysis, particularly in the context of the gig economy. The study demonstrates the feasibility of using machine learning in this context and highlights the importance of accurate sentiment assignment. The results provide a benchmark for future research in this area, enabling the development of more robust sentiment analysis models for low-resourced languages.

Table 2.1 presents a comparison among BNSENTMIX and previously available code-mixed sentiment analysis datasets across various languages.

**Table 2.1:** Comparison of the number of samples, #SL: Sentiment Labels, #DS: Data Sources, filtering method, number of baselines, and PA: Public Availability of various code-mixed (with English) sentiment analysis datasets.

| Dataset | #Samples | #SL | #DS | Filtering | #Baselines | PA |
|---|---|---|---|---|---|---|
| Hindi [18] | 3.9k | 3 | 1 | Manual | 10 | ✓ |
| Bengali [22] | 5k | 3 | 1 | Manual | 5 | ✗ |
| Tamil [7] | 15.7k | 5 | 1 | `langdetect` | 10 | ✓ |
| Malayalam [6] | 6.7k | 5 | 1 | `langdetect` | 10 | ✓ |
| Persian [31] | 3.6k | 3 | 1 | Keywords search | 3 | ✓ |
| Swiss [26] | 963 | 3 | 1 | Manual | 7 | ✗ |
| **BNSENTMIX (Ours)** | **20k** | **4** | **3** | **mBERT** | **14** | ✓ |

# Chapter 3

# Proposed Methodology

The BnSentMix dataset has been meticulously curated from a diverse array of data sources to ensure it mirrors the realistic nature of code-mixed texts that are prevalent in digital communication spaces, particularly in online social media, forums, and messaging platforms. These platforms frequently feature conversations that switch between multiple languages, a phenomenon commonly referred to as "code-mixing." The primary goal of this dataset is to capture the complexity of these conversations and provide a valuable resource for sentiment analysis within such multilingual environments. By sourcing the data from different domains, including social media posts, user-generated comments, online chat platforms, and digital publications, the BnSentMix dataset presents a rich and varied collection of text that encapsulates the dynamic nature of everyday digital discourse.

A key aspect of our dataset is the labeling of sentiments, which we approached with a nuanced perspective. While the traditional sentiment categories—positive, negative, and neutral—are commonly used across many sentiment analysis tasks, we identified a need for an additional label: *mixed*. This *mixed* sentiment label reflects instances where both positive and negative sentiments are present within the same text, often within different parts of a sentence or across a conversation. For example, a sentence might begin with a compliment (positive sentiment) and end with criticism (negative sentiment), making it difficult to classify the overall sentiment as purely positive or negative.

The inclusion of the *mixed* sentiment is crucial because such sentiment combinations are not uncommon in real-world text data. In fact, we observed that a significant portion of the data consists of sentences or utterances that cannot be cleanly classified into the three traditional categories. Without this label, those instances would ei-

ther be misclassified or entirely overlooked, leading to inaccurate sentiment analysis models. Therefore, the addition of the *mixed* sentiment enhances the dataset's ability to reflect real-world scenarios more accurately and provides researchers with a more comprehensive tool for analyzing sentiment in complex, code-mixed environments.

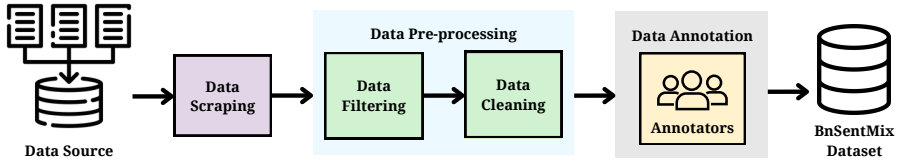



**Figure 3.1:** Dataset Preparation pipeline

Figure. 3.1 gives an overview of the dataset preparation process. The key points are-

- **Data Source:** The dataset was sourced from user-generated content on popular platforms such as Facebook, YouTube, and e-commerce websites. These sources provide a broad spectrum of topics and linguistic contexts.
- **Data Scraping:** Data was scraped from the selected sources using various tools. The `YouTube API`[1] was used to extract comments, `Facepager`[2] was employed for scraping Facebook comments, and Selenium was utilized for extracting reviews from e-commerce platforms.
- **Data Pre-processing:** This phase consisted of two key steps:
  - *Data Filtering:* A fine-tuned pre-trained language model was used to filter out non-code-mixed samples, ensuring that the dataset contained predominantly Bengali-English code-mixed data.
  - *Data Cleaning:* The data was cleaned by removing URLs, special characters, non-ASCII symbols, and extra whitespaces. Typographical and grammatical errors were left intact to maintain real-world authenticity.
- **Data Annotation:** The filtered and cleaned data was then annotated for sentiment labels (positive, negative, neutral, and mixed). Each sample was labeled by two independent annotators, and a third annotator resolved any disagreements.
- **Final Dataset:** After pre-processing and annotation, the final dataset comprised 20,000 code-mixed samples, making it ready for sentiment analysis model training and evaluation.

[1] `https://developers.google.com/youtube/v3/docs/comments`
[2] `https://github.com/strohne/Facepager`

## 3.1 Data Sourcing

The data collection process for the BNSENTMIX dataset was designed with the objective of obtaining extensive, diverse, and representative samples of user-generated content from various platforms that are known for high levels of user engagement and rich linguistic diversity. Specifically, we targeted platforms such as YouTube, Facebook, and e-commerce websites, all of which are prolific sources of natural language text where users freely express their opinions, emotions, and sentiments in an informal, often code-mixed manner.

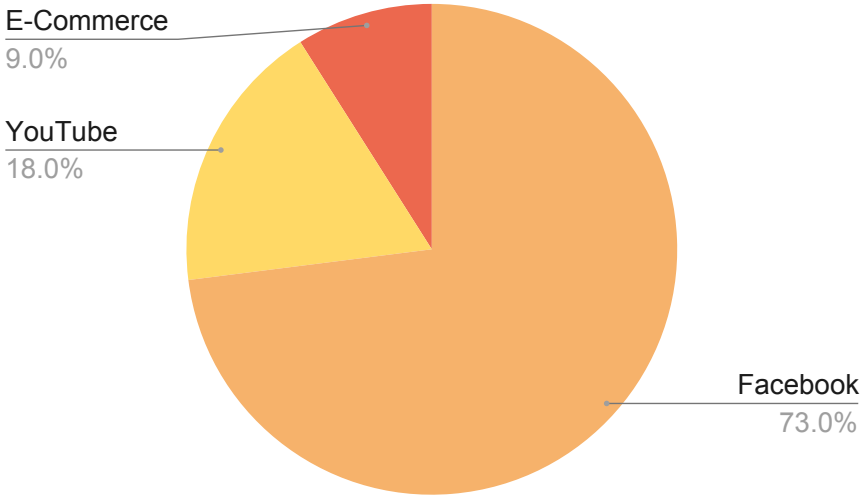



**Figure 3.2:** Percentage of data source from different social media and e-commerce sites

YouTube comments were selected due to the platform's massive global audience and the variety of topics discussed in the comment sections of videos, ranging from entertainment and tutorials to news and personal vlogs. The YouTube API was utilized to scrape these comments, enabling us to gather large volumes of publicly available user interactions with minimal interference. By tapping into the YouTube API, we were able to systematically retrieve comments across a wide range of videos, ensuring a broad spectrum of user sentiment and expression.

For Facebook, which is another major platform where users engage in discussions and share content in an informal setting, we employed `Facepager`[3] to extract comments. `Facepager` allowed us to efficiently collect data from public posts, pages, and groups, where users often discuss a variety of topics and interact in code-mixed languages. This tool enabled us to access Facebook's public-facing data without breaching user

[3] `https://github.com/strohne/Facepager`

privacy, focusing only on content that was available for public consumption. By extracting comments from different types of posts, we were able to ensure diversity in our dataset, reflecting the many ways in which people communicate on this platform.

In addition to social media platforms, we turned our attention to e-commerce websites, where users frequently leave product reviews. These reviews are a valuable source of sentiment-rich data, as they often include positive, negative, or mixed opinions about products. To collect this data, we employed `Selenium`[4], a powerful browser automation tool that mimics human browsing behavior. `Selenium` allowed us to scrape product reviews from e-commerce sites by simulating a real user's interaction with the website, effectively bypassing some of the restrictions that might otherwise limit data access. This approach proved essential for collecting large volumes of reviews, which are often found in code-mixed languages, particularly in regions where multilingualism is common.

By combining data from these three platforms—YouTube, Facebook, and e-commerce websites—we amassed over 3 million samples of user-generated content. Each source was chosen not only for its volume of available data but also for its relevance to the type of sentiment analysis we aimed to conduct. These platforms represent key digital spaces where everyday users express a wide range of sentiments in code-mixed text, making them ideal for building a dataset that accurately reflects the linguistic and emotional complexity of real-world communication. The composition of these data sources is illustrated in **Figure 3.2**, providing a visual representation of the diverse origins of our dataset.

This large and varied collection of user-generated content formed the foundation for our subsequent analysis, offering a rich and comprehensive set of examples for training and testing models designed to handle the complexities of code-mixed text and nuanced sentiment expressions.

## 3.2 Data Cleaning

We implemented a comprehensive data cleaning process to refine the dataset and remove noise that could potentially affect the quality of sentiment analysis. First, we discarded any samples that contained four words or less. This threshold was chosen because shorter samples often lack meaningful content for sentiment classification, and may consist of phrases or abbreviations that are difficult to interpret in isolation.

[4] `https://selenium-python.readthedocs.io/`

Additionally, we excluded samples containing external URLs, which are typically not useful for sentiment analysis and can introduce unnecessary noise into the dataset.

Next, we performed a series of preprocessing steps aimed at removing extraneous characters and formatting inconsistencies. Redundant whitespaces, which often appear due to human typing errors or improper formatting, were removed to ensure uniformity in text structure. Special characters, such as currency symbols or mathematical operators, were also stripped from the text, as they do not contribute to sentiment. Similarly, non-ASCII characters, including emojis and emoticons, were excluded from the dataset to maintain linguistic consistency. These characters, while expressive, often lack standard sentiment interpretations in code-mixed text, making them less useful for our analysis.

To further clean the text, consecutive sequences of punctuation symbols—such as multiple exclamation marks or ellipses—were reduced to single instances. This step was necessary to avoid over-representation of certain punctuations, which could skew the sentiment analysis results. For example, "!!!" was simplified to "!" to maintain uniformity while preserving the emphasis intended by the user.

English words within the dataset were converted to lowercase unless they appeared at the beginning of a sentence, where they retain their capitalization to preserve grammatical integrity. Downcasing is a common practice in natural language processing as it reduces the number of unique tokens in the text, thereby simplifying the analysis.

Despite the rigorous cleaning procedures, we made a conscious decision not to correct any typing or grammatical errors present in the dataset. This decision was based on the need to train models that can handle real-world, noisy data. In practice, user-generated content, especially in informal digital environments, is often rife with such errors. By retaining these errors, we ensured that the trained model would be robust and adaptable to practical scenarios, where perfect grammar and spelling are rare.

## 3.3 Data Filtering

We construct a novel Bengali-English code-mix detection dataset and fine-tune pre-trained language models to automatically filter code-mixed Bengali-English. Detecting these texts can pose significant challenges: (i) rule-based methods struggle with intra-word switching, which occurs when a single word is formed from multiple languages, (ii) romanized Bengali or English samples may be incorrectly classified as code-mixed text by automated methods due to the shared character set, and (iii) sam-

ples from a third language, such as Hindi, often bypass the filtering process as they may not be recognized by models trained on Bengali-English texts. To overcome these challenges, our approach leverages pre-trained language models, which excel in handling complex linguistic structures and patterns. We fine-tune these models specifically for the task of detecting code-mixed text in a Bengali-English context, significantly improving the accuracy of the filtering process. The data filtering pipeline is detailed in Algorithm 1 'Detect Code-mixed Bengali', which demonstrates the step-by-step method for detecting and categorizing code-mixed Bengali texts.

**Algorithm 1** Detect Code-mixed Bengali

**Require:** $S \leftarrow$ List of sentences
**Require:** $model \leftarrow$ Pre-trained mBERT model
**Require:** $tokenizer \leftarrow$ Pre-trained mBERT tokenizer
**Ensure:** $pred \leftarrow$ Predicted class label (0 or 1)

$b\_count \leftarrow 0$
$w\_count \leftarrow 0$
**for** each $sent$ in $S$ **do**
  $words \leftarrow$ split($sent$)
  **for** each $w$ in $words$ **do**
    $w \leftarrow$ preprocess($w$)
    **if** $w$ is empty **then**
      continue
    **end if**
    $w\_count \leftarrow w\_count + 1$
    $inputs \leftarrow$ tokenize($w$)
    $outputs \leftarrow$ model($inputs$)
    $pred\_class \leftarrow$ argmax($outputs$)
    **if** $pred\_class == 1$ **then**
      $b\_count \leftarrow b\_count + 1$
    **end if**
  **end for**
**end for**
**if** $w\_count < 4$ **then**
  **return** 0
**end if**
$b\_percent \leftarrow b\_count / w\_count$
**if** $b\_percent \geq 0.3$ **then**
  **return** 1
**else**
  **return** 0
**end if**

### 3.3.1 Key Components of the Algorithm

The algorithm for detecting code-mixed Bengali-English text consists of several key components aimed at analyzing each sentence, identifying language mixing, and classifying text as code-mixed or monolingual.

First, the algorithm initializes inputs, receiving a list of sentences $S$, a pre-trained language model (specifically mBERT), and its corresponding tokenizer. It utilizes counters $b\_count$ and $w\_count$ to track the number of detected Bengali code-mixed words and total words, respectively.

Next, for each sentence in $S$, the algorithm tokenizes the words and processes each word individually. Basic preprocessing is applied to each word, which may involve removing special characters.

Following this, each word is tokenized using the mBERT tokenizer, and the resulting tokenized input is passed through the model. The model's output undergoes classification via the `argmax` function, yielding a label that indicates whether the word is Bengali-English code-mixed or monolingual.

After processing all words in a sentence, the algorithm computes the proportion of code-mixed words. If this proportion exceeds a predefined threshold (30% in this case), the sentence is classified as code-mixed; otherwise, it is classified as monolingual.

Finally, the algorithm includes a mechanism for handling short sentences. Specifically, sentences containing fewer than four words are directly classified as monolingual to mitigate the risk of misclassification due to insufficient data.

This approach leverages the mBERT model's multilingual capabilities to effectively distinguish code-mixed content based on intra-word or character-level patterns, thus addressing challenges such as intra-word switching and character set ambiguity.

### 3.3.2 Code-mix Detection Dataset

To create a robust dataset for pre-training, we sourced and combined three key datasets to ensure comprehensive coverage of both Bengali and English text. The first dataset we utilized was Google's Dakshina dataset [29], which is a rich collection of text in South Asian languages, including many Bengali-English code-mixed sentences. These sentences were particularly instrumental for training our model, as they reflect common patterns of code-mixing found in real-world conversations. The Dakshina dataset

helped us capture a wide variety of linguistic structures in code-mixed texts, providing a strong foundation for model fine-tuning.

The second dataset was sourced from Kaggle, specifically an English word frequency dataset[5]. This dataset provided a comprehensive range of English words, ensuring that the model could effectively handle pure English samples as well as those mixed with Bengali. By including a broad spectrum of English words, we aimed to enhance the model's capability to differentiate between purely English sentences and code-mixed text.

For the third data source, we turned to the dataset curated by Mandal et al. [23], which offered additional samples of Bengali-English code-mixed text. This dataset was particularly useful for balancing the overall composition of our training data, ensuring that our model was exposed to a wide range of code-mixed examples. By integrating these diverse sources, we curated a comprehensive dataset of 100,000 words, ensuring a balanced mix of Bengali, English, and code-mixed text.

To maintain the linguistic purity of the code-mixed Bengali-English dataset, we applied a rigorous filtering process to exclude sentences containing words that did not belong to either the Bengali or English languages. For example, sentences that included words from Hindi or other third languages were removed to prevent the model from being exposed to irrelevant linguistic patterns. This careful curation ensured that the resulting dataset was specifically tailored for the task of detecting and analyzing Bengali-English code-mixed text, enhancing the model's performance in this specialized area.

### 3.3.3 Code-mix Detection Results

We evaluate three pre-trained models – the multilingual models, mBERT [12] and XLM-RoBERTa [10], and the Bengali-English specific model BanglishBERT [5]. Table 3.1 showcases the performance metrics, including accuracy and F1 score, for code-mixed Bengali-English detection. Our results indicate that mBERT significantly outperformed the other models, exhibiting a higher accuracy of 94.56% and an F1 score of 94.03%, compared to 90.56% accuracy for BanglishBERT and 89.60% for XLM-RoBERTa. The substantial performance boost provided by mBERT can be attributed to its pre-trained multilingual capabilities, which seem to excel in capturing the intricate nuances of code-mixed texts. This finding suggests that mBERT's strong capacity for multilingual understanding, combined with its robust context-handling abilities, al-

[5]https://www.kaggle.com/datasets/rtatman/english-word-frequency

lows it to effectively distinguish between monolingual and code-mixed sentences in Bengali-English contexts.

**Table 3.1:** Comparison of the accuracy and F1 score of the code-mixed Bengali-English detection methods.

| Model | Acc(%) | F1 Score(%) |
|---|---|---|
| XLM-RoBERTa | 89.60 | 89.85 |
| BanglishBERT | 90.56 | 89.61 |
| **mBERT** | **94.56** | **94.03** |

### 3.3.4 Filtering Challenges

The task of filtering and processing Bangla-English code-mixed sentences presents significant challenges due to the limitations of commonly used language detection tools. These tools often struggle with detecting code-mixed content accurately, especially when Bangla is written in the Roman script or when transliterations occur. Below is an overview of the challenges associated with various tools used for language detection in this context.

- **Langdetect[6]: Detects Bengali Only in Its Native Script**
  Langdetect is a widely used language detection tool that relies on n-gram models to classify text into different languages. However, its major limitation is that it can only detect Bengali in its native script. In code-mixed sentences, where Bengali is often written in Romanized form (i.e., "Banglish"), Langdetect struggles to correctly identify the Bengali portion, leading to misclassifications and inaccuracies in detecting language boundaries.
- **Polyglot[7]: Transliterates Any Word to Bengali, Regardless of Its Origin**
  Polyglot is a multilingual processing tool that can detect languages and perform transliteration. However, it has a serious flaw in that it transliterates any word to Bengali, regardless of the word's origin. This means that even English words, loanwords, or even words from other languages may be wrongly transliterated into Bengali script, thus misclassifying the language of the word. This behavior can lead to incorrect assumptions about which words belong to which language, making it unsuitable for accurate code-mixed sentence processing.
- **Langid[8]: Detects Bengali Only in Its Native Script**

[6] https://pypi.org/project/langdetect/
[7] https://pypi.org/project/polyglot/14.11/
[8] https://github.com/saffsd/langid.py

Similar to Langdetect, Langid also detects Bengali text only when it is written in the native Bangla script. It does not handle Romanized Bangla well, which is often used in code-mixed texts. As a result, Langid is limited in its ability to distinguish between Bangla and English in such contexts, leading to potential misclassification and loss of information.

- **Fasttext-Langid[9]: Detects Bengali Only in Its Native Script**
  Fasttext-langid, a variant of Fasttext designed for language identification, also suffers from the same issue of detecting Bengali only in its native script. Although Fasttext is known for its high-speed processing and scalability, its inability to handle Romanized Bangla makes it less effective for real-world code-mixed data, where both languages coexist in an informal, unstructured manner.
- **BnB Phonetic Parser[10]: Rule-Based Phonetic Parser, Unmaintained**
  The BnB phonetic parser is a rule-based tool designed to handle phonetic transliterations of Bangla and English. However, this tool is outdated and unmaintained, which limits its usefulness in modern applications. Moreover, its rule-based nature is not flexible enough to handle the complexities of code-mixed sentences, where multiple languages intermingle with little regard to strict phonetic rules.

### 3.3.5 Filtering Pipeline

Following the pipeline illustrated in Figure. 3.3, our approach first applies a rigorous cleaning process to the code-mixed Bengali-English data, removing noise and irrelevant content such as special characters, emojis, and external links. This cleaned data is then passed through a data filtering classifier trained using the mBERT model, which we demonstrated to be the most effective in code-mix detection. The classifier filters out non-code-mixed data and ensures that the remaining dataset retains only high-quality, linguistically pure code-mixed samples. Manual verification is performed on the filtered data to ensure accuracy and to address any edge cases that the model may have missed, such as partial code-mixing or misclassified samples.

To maintain linguistic purity, sentences containing words that do not belong to either Bengali or English are excluded. For example, sentences with Hindi or other third-language words are discarded to prevent any contamination of the code-mixed data. Additionally, we enforce strict exclusion of any samples that contain hate speech or

[9] `https://pypi.org/project/fasttext-langdetect/`
[10] `https://github.com/porimol/bnbphoneticparser`

offensive content to ensure that the final dataset is appropriate for further analysis and model training.

We also filter out sentences with insufficient levels of code-mixing. Specifically, we define a sentence as code-mixed Bengali-English if it contains at least 30% English words. Any sentence with a smaller proportion of code-mixing is discarded, as it does not meet our threshold for meaningful linguistic blending. After applying these rigorous filtering criteria, we retain a total of 21,587 high-quality code-mixed samples.

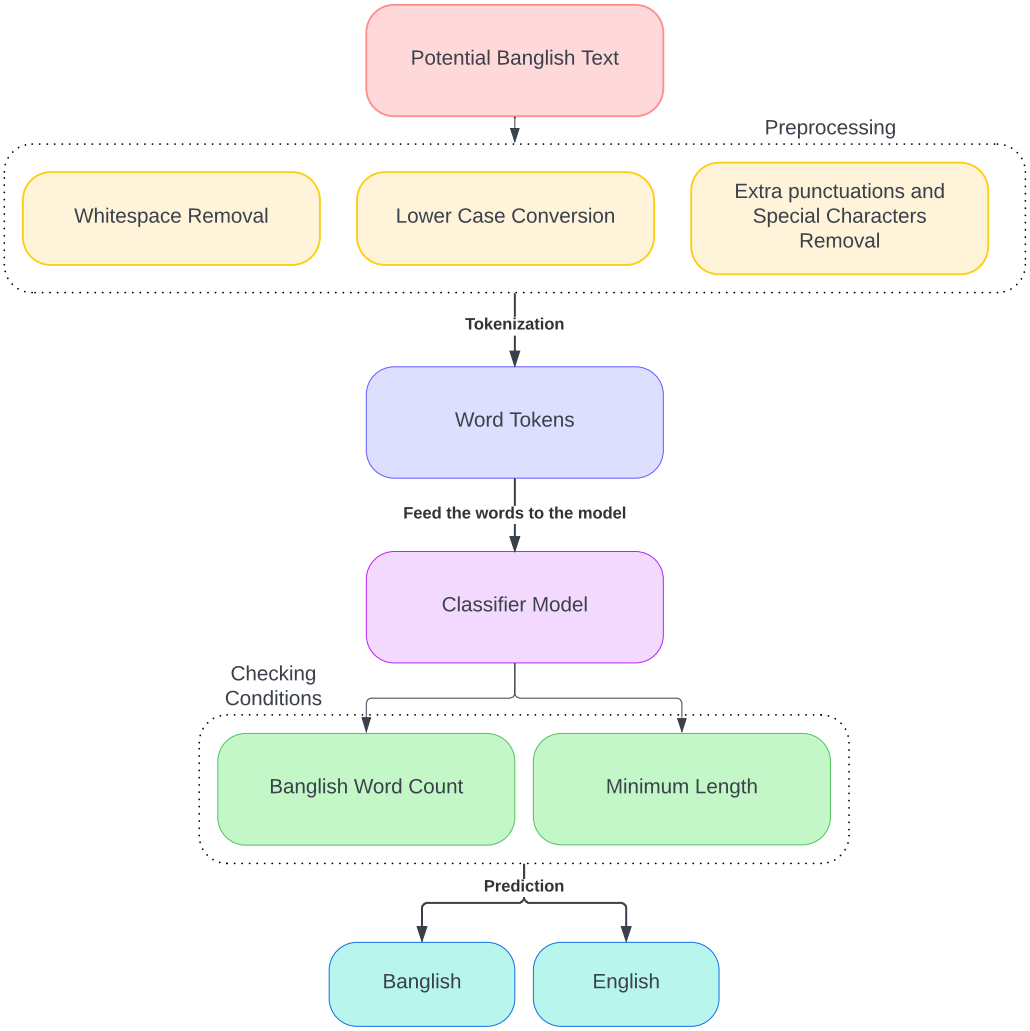



**Figure 3.3:** Data Filtering Pipeline

## 3.4 Data Annotation

The data annotation process for the BNSENTMIX dataset was designed with meticulous attention to accuracy and consistency, ensuring that the subjective task of sentiment labeling reflects a reliable and generalizable understanding of the content. Each sample in the dataset underwent a rigorous double-annotation process, wherein two distinct annotators independently labeled the sentiment of each sample. This dual-layered annotation approach was employed to mitigate individual biases and ensure that the sentiment labels capture a broad spectrum of interpretation, especially in the context of code-mixed texts where multiple nuances in meaning often emerge. Given that sentiment can be interpreted in varying ways, particularly in informal, conversational text, this redundancy in annotation was critical in achieving a high level of accuracy.


**Figure 3.4:** Data Annotation Procedure

The dual annotation process sometimes resulted in discrepancies between the two annotators' sentiment labels. In such cases, we incorporated a third annotator, whose role was to act as a mediator and break the tie, thereby establishing the final sentiment label for the sample. This tie-breaking process was crucial for maintaining the reliability of the dataset, ensuring that the sentiment labels represent a considered consensus rather than arbitrary decision-making. By incorporating this third layer of review, the dataset was protected from potentially errant individual judgments that could skew the overall quality of the sentiment classification. This structured approach to annotation highlights the depth of the process and the lengths taken to ensure quality control in sentiment tagging.

To carry out the annotation at scale, we recruited a total of 64 annotators. The recruitment process was stringent, as we needed individuals who not only possessed linguistic proficiency but were also familiar with the kinds of code-mixed and often informal texts commonly found in digital spaces, particularly on social media platforms. The annotators were carefully selected, ensuring that all of them had at least a high-school education (equivalent to Grade 12), which we considered as a baseline for understanding both English and Bengali. Additionally, familiarity with digital platforms like Facebook, YouTube, and e-commerce websites was a key requirement. This

familiarity was essential because the textual content in our dataset was derived from these platforms, and a contextual understanding of the type of language typically used in these settings (i.e., code-mixed, colloquial, and informal) was necessary for accurate sentiment annotation.

Monetary compensation for the annotators was provided on an hourly basis, which was designed to reflect the time-intensive and cognitively demanding nature of the task. Since sentiment annotation requires close attention to detail and nuanced judgment, it was crucial to ensure that the annotators were adequately compensated for their efforts to maintain motivation and consistency in their work. The hourly payment system also allowed for flexibility, encouraging annotators to take the time needed to carefully assess each sample without feeling rushed to meet per-sample quotas, which could compromise the quality of the labeling.

To ensure consistency across the annotators and to measure the reliability of their annotations, we implemented a re-annotation procedure on a subset of the dataset. Each annotator was asked to re-label the same set of 250 samples, selected at random, to assess inter-annotator agreement. This agreement was quantified using Cohen's Kappa statistic, which measures the level of agreement between two annotators beyond what would be expected by chance. Our calculation yielded a Cohen's Kappa score of $\kappa = 0.86$, indicating substantial agreement between the annotators. A Kappa score of 0.86 is considered excellent in most contexts, underscoring the reliability and consistency of the sentiment annotations across the dataset. This score is particularly noteworthy in light of the inherent challenges in sentiment analysis for code-mixed data, where the blending of languages and cultural context can often introduce ambiguity.

The high level of inter-annotator agreement also suggests that the annotators were generally in sync in their interpretation of sentiment, further validating the robustness of our annotation guidelines and training process. Achieving such a high Kappa score is significant given the subjective nature of sentiment labeling, especially when dealing with informal, user-generated content that can sometimes contain sarcasm, irony, or mixed emotions. The substantial agreement demonstrated by the Kappa score thus enhances the credibility of the BnSentMix dataset and assures researchers and developers that the sentiment labels can be trusted for subsequent machine learning tasks.

In summary, the data annotation process involved careful planning and execution, with multiple layers of quality control to ensure the consistency and reliability of the sentiment labels. The dual-annotation process, the use of a third annotator for re-

solving conflicts, the rigorous selection and training of annotators, the implementation of re-annotation procedures, and the calculation of inter-annotator agreement all contribute to a highly robust dataset. This dataset serves as a reliable foundation for developing and evaluating models in code-mixed sentiment analysis, a task that is inherently complex due to the linguistic and contextual intricacies present in the data. The structured and comprehensive nature of the annotation process ensures that the resulting dataset is not only large in scale but also rich in quality and consistency.

## 3.5 Dataset Statistics

After a rigorous evaluation process, 20,000 samples were finalized from the 21,587 samples, ensuring high-quality annotations for subsequent analysis and model training.

An overview of the key statistics of the annotated dataset is shown in Table 3.2. The dataset was split into [70 : 15 : 15] for training, validation, and test sets, resulting in 14,000, 3,000, and 3,000 samples respectively.

Given the importance of accuracy in sentiment annotation, a meticulous validation process was implemented. This process aimed to ensure the reliability and consistency of the annotated data, address potential discrepancies, and uphold a high standard of quality throughout the dataset.

**Table 3.2:** Key statistics of the annotated dataset.

| **Statistic** | **Value** |
|---|---|
| Total samples | 20,000 |
| Training samples | 14,000 |
| Validation samples | 3,000 |
| Test samples | 3,000 |

### 3.5.1 Validation Sampling

To ensure the integrity and reliability of our annotations, we employed a random sampling technique for validation. Specifically, from each set of 250 annotated samples, 30 samples were randomly selected for further validation. This process was designed to ensure that a diverse range of data was reviewed, reducing the potential for bias and ensuring that the validation process was representative of the dataset as a whole. By randomizing the selection, we aimed to create a validation process that captured a

wide array of linguistic expressions, sentiments, and contexts, enhancing the overall robustness of the validation phase.

### 3.5.2 Independent Re-Annotation

For each of the selected 30 samples, two additional independent annotators were assigned to re-annotate the data. This step was pivotal in cross-checking the initial annotations, as it introduced an external perspective to identify any inconsistencies or discrepancies that might have been overlooked in the initial labeling process. The independent re-annotation process was not only about ensuring the accuracy of the sentiment labels but also about reinforcing the robustness of our annotation guidelines. Having two separate annotators re-label the same samples provided an extra layer of verification and allowed for a clear measurement of inter-annotator agreement.

### 3.5.3 Comparison and Agreement Analysis

Once the independent re-annotations were completed, the next step involved comparing these annotations with the original ones. To establish the accuracy and reliability of the annotations, we set a threshold where annotations with at least 80% similarity were deemed acceptable. The similarity metric was based on comparing the labels assigned to each sample by the independent annotators and the original annotator. If both independent annotations were in agreement with each other and matched the original annotation with at least 80

## 3.6 Inter-Annotator Agreement

The consistency and reliability of our annotation process were further evaluated through inter-annotator agreement analysis. Each annotator followed the same set of rigorous standards during the labeling process, which led to a high degree of uniformity in the sentiment categorization. The level of agreement among annotators is a key indicator of how well the guidelines were understood and followed. To quantify this agreement, we used Krippendorff's alpha ($\alpha$), a widely accepted statistical measure that accounts for variations such as missing data, differing sample sizes, and the number of annotators. This metric is especially suited for datasets labeled by multiple annotators, as it provides a robust measure of agreement even when not all annotators label every sample.

We computed Krippendorff's alpha using the `nltk` library, achieving a score of 89.7%

for the nominal metric (which is applicable for categorical data) and 92.5% for the interval metric (which applies when considering the magnitude of disagreements). These results indicate a substantial level of agreement, confirming that the annotators consistently interpreted the sentiment labels and applied the guidelines correctly. The high alpha score also highlights the reliability of our annotation process, as such values typically signify strong consensus among raters.

## 3.7 Finalization of Annotated Data

Following the validation and agreement analysis, the final step was to finalize the annotations that met the established criteria. Out of the original 21,587 samples, 20,000 were accepted as high-quality, well-annotated data points. This meticulous validation process ensured that only the most accurate and reliable samples were included in the final dataset. By rigorously filtering and verifying the annotations, we have created a robust dataset that forms a strong foundation for subsequent sentiment analysis tasks. The final dataset not only reflects a diverse range of sentiments but also upholds a high standard of annotation quality, making it suitable for training and evaluating sentiment analysis models.

## 3.8 Methodology and Experimental Setup

### 3.8.1 Baseline Models

We evaluate 11 baseline models, ranging from traditional machine learning techniques to state-of-the-art deep learning architectures. These models include both recurrent neural network variants and transformer-based pre-trained language models, which have shown great promise in the field of natural language processing. The models used in our experiments are detailed in Table 4.2.

The pre-trained transformer models, such as BERT and its variants, were fine-tuned on our specific dataset. While these models achieved impressive performance, BERT emerged as the best-performing model with 69.5% test accuracy and 68.8% test F1 score, highlighting its ability to capture the complex linguistic patterns in the code-mixed Bengali-English text. These results provide a strong baseline against which other models can be evaluated.

### 3.8.2 Further Pre-trained Transformer Models

To improve the performance of our transformer-based models, we introduce a further pre-training phase, leveraging Masked Language Modeling (MLM) loss [12]. This phase involves pre-training the models on an additional corpus of unlabeled code-mixed data scraped from various sources. By pre-training the models on this additional data, we aim to improve their understanding of the linguistic features of code-mixed text, particularly in the context of Bengali-English mixtures.

The MLM loss function, which is commonly used in BERT and similar transformer architectures, can be defined as:

$$L_{\text{MLM}} = -\frac{1}{N}\sum_{i=1}^{N}\log P(x_i \mid \mathbf{x}_{\text{mask}}) \tag{3.1}$$

where: - $x_i$ is the token at position $i$ in the sequence that is masked, - $\mathbf{x}_{\text{mask}}$ represents the input sequence with dynamically masked tokens, - $N$ is the total number of masked tokens.

During this phase, we ensure that samples from the test and validation sets are excluded from the pre-training corpus to avoid data leakage and ensure a fair evaluation of the models.

These further pre-trained models are referred to as Code-Mixed Bengali (CMB) transformer models. The naming convention for these models follows the format $x$-CMB, where $x$ refers to the specific pre-trained transformer variant used. These CMB models are expected to capture more nuanced patterns of code-mixing, making them more suitable for the task at hand.

### 3.8.3 Evaluation Metrics

In this work, we utilize two widely adopted metrics for evaluating the performance of the text classification models: classification accuracy and F1-score. Both of these metrics are essential for understanding how well the model predicts the class labels, especially in imbalanced datasets.

**Classification Accuracy** is the proportion of correct predictions (both true positives and true negatives) out of all the predictions made. It is given by the formula:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.2}$$

where $TP$, $TN$, $FP$, and $FN$ stand for true positives, true negatives, false positives, and false negatives, respectively. Although accuracy is a simple and intuitive measure, it can be misleading when dealing with imbalanced classes, where the model might predict the majority class predominantly, resulting in high accuracy but poor performance in predicting the minority class.

**F1-score** is the harmonic mean of precision and recall, and it is particularly useful when the class distribution is imbalanced. The formula for F1-score is:

$$\text{F1-score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3.3}$$

where Precision is the proportion of positive predictions that are actually correct ($\frac{TP}{TP+FP}$), and Recall is the proportion of actual positives that are correctly predicted ($\frac{TP}{TP+FN}$). The F1-score is an important metric as it balances both precision and recall, offering a single metric to evaluate the classifier's performance when the class distribution is skewed.

### 3.8.4 Implementation Details

The models used in this study were trained on NVIDIA Tesla P100 GPUs with 16GB of memory, ensuring that the computational requirements for training large-scale language models were adequately met. We followed the Huggingface implementation of pre-trained language models [38], leveraging their well-optimized pipelines for fine-tuning on the specific task of text classification.

For the optimizer, we used the Adam optimizer [19], which is popular for training deep learning models due to its adaptive learning rate properties. The models were trained with a batch size of 32, which was found to balance performance and computational efficiency. Most hyperparameters were left at their default values, which were chosen based on extensive empirical validation in prior works.

Logistic Regression, RNN, and LSTM models were trained using a learning rate of $1E{-}5$, while the BERT-family models employed a smaller learning rate of $1.5E{-}6$. This distinction is based on the observation that large pre-trained models like BERT tend to require smaller learning rates for fine-tuning to avoid overfitting.

The training time for each epoch varied between 8 and 13 minutes, depending on the model and the hardware configuration.

# Chapter 4

# Results and Discussion

In this section, we present the results obtained from our extensive experimentation with various machine learning, neural network, and transformer-based models. The performance of these models is evaluated on the Bengali-English code-mixed dataset introduced in this thesis. We compare the performance of traditional machine learning models, recurrent neural networks, and transformer-based models using accuracy, precision, recall, and F1 score.

## 4.1 Dataset Analysis

### 4.1.1 Dataset Statistics

Table 4.1 presents the key statistics of the `BNSENTMIX` dataset, which offers a detailed understanding of its structure and diversity. These statistics give insights into the distribution of sentence lengths, word counts, and vocabulary richness in the dataset, all of which are important for effective model training in sentiment analysis.

**Table 4.1** shows the overall characteristics of our dataset. The **mean character length** of 62.77 suggests that, on average, sentences in the dataset are of moderate length, providing a balance between brevity and complexity. The **maximum character length** of 1985 indicates the presence of some very long sentences, which likely introduce additional syntactic complexity that models must learn to handle. In contrast, the **minimum character length** of 14 points to much shorter sentences, often lacking depth, which may pose challenges for sentiment classification due to the limited contextual information.

In terms of word count, the **mean word count** of 11.65 words per sentence aligns

**Table 4.1:** Key statistics of the BNSENTMIX dataset.

| Statistic | Value |
|---|---|
| Mean Character Length | 62.77 |
| Max Character Length | 1985 |
| Min Character Length | 14 |
| Mean Word Count | 11.65 |
| Max Word Count | 368 |
| Min Word Count | 4 |
| Unique Word Count | 37734 |
| Unique Sentence Count | 20000 |

with typical short social media posts or comments, indicating that much of the dataset reflects conversational or informal writing styles. However, there is considerable variation, with the **maximum word count** reaching 368, indicative of longer, more detailed user reviews or comments, while the **minimum word count** of 4 reflects very brief sentences, which are common in online platforms and may affect the model's ability to accurately predict sentiment due to limited context.

The dataset exhibits substantial **vocabulary diversity**, as reflected in the **37,734 unique words**, a significant number for a code-mixed dataset. This diversity is critical for training models that can generalize well, especially in multilingual settings where users switch between languages. The high number of unique words suggests that the dataset captures a wide range of expressions and linguistic nuances, which is crucial for effectively handling code-mixed text. Figure 4.1 shows the length distribution of positive labeled sentences of our dataset. We can see that the sentences are mostly scattered around 5 to 15 words which we think is ideal.

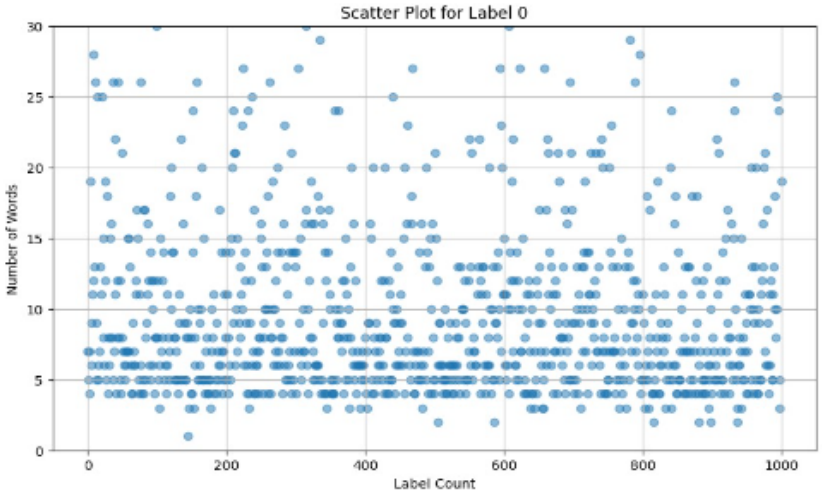



**Figure 4.1:** Analysis of Word count of positive labeled Sentences

Figure 4.2 shows the distribution word count per sentence for negative labeled sentences. The distribution is similar to positive labeled sentences here.

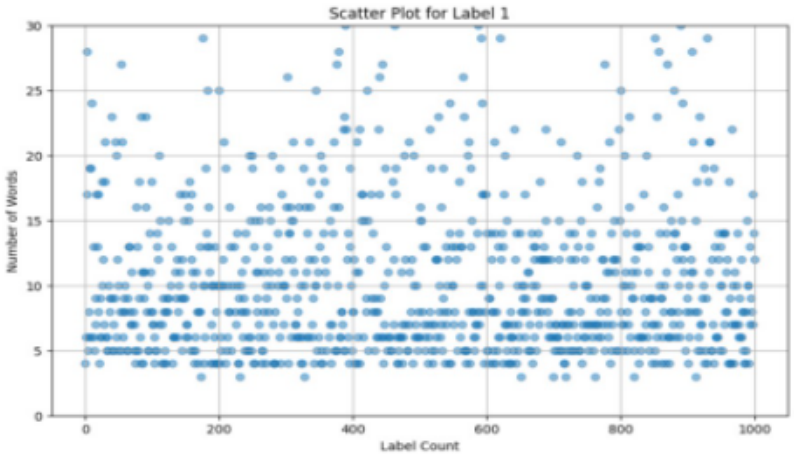



**Figure 4.2:** Analysis of Word count of negative labeled Sentences

Figure 4.3 shows the distribution word count per sentence for neutral labeled sentences. We can see almost all the sentences here have around 5 to 10 words.

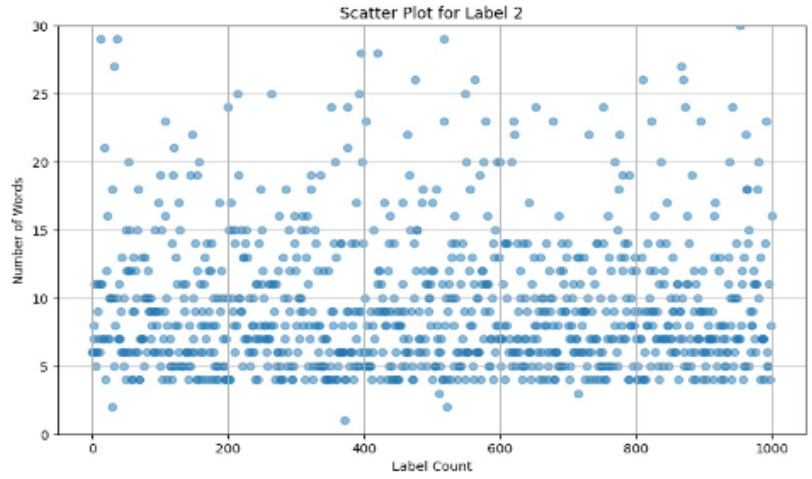



**Figure 4.3:** Analysis of Word count of Neutral labeled Sentences

Figure 4.4 shows the distribution word count per sentence for mixed labeled sentences. We can see the word count distribution is scattered all over the place here ranging from 5 words per sentence to 30 words per sentence. It is because, people tend to use more words in order to express there mixed emotions.

Furthermore, the dataset includes **20,000 unique sentences**, indicating that no sentences are repeated, which enhances the dataset's ability to expose models to a wide variety of sentence structures and contexts. This diversity in sentence composition is


**Figure 4.4:** Analysis of Word count of Mixed labeled Sentences

crucial for training models that need to perform well across different contexts, tones, and sentiment expressions in real-world applications.

Overall, the dataset's combination of short and long sentences, varied word counts, and rich vocabulary makes it an ideal resource for training sentiment analysis models that can handle the complexities of code-mixed Bengali-English text. Compared to other sentiment analysis datasets, `BNSENTMIX` offers a larger and more diverse range of linguistic data, making it a valuable resource for developing robust models capable of performing well on real-world multilingual and code-mixed content.

### 4.1.2 Sentiment Label Distribution

Figure 4.5 illustrates the distribution of sentiment labels in the `BNSENTMIX` dataset. The dataset consists of four sentiment labels: Positive, Negative, Neutral, and Mixed, with the distribution as follows:

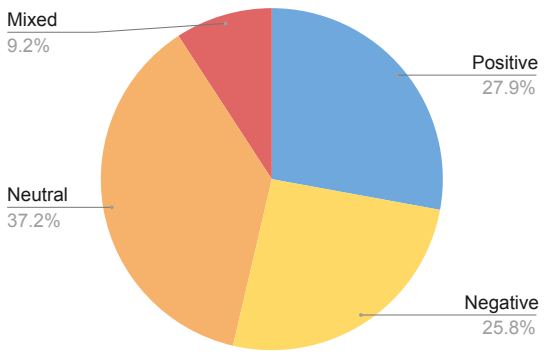



**Figure 4.5:** Distribution of Sentiment Labels in the `BNSENTMIX` dataset

The **Neutral** sentiment accounts for the largest portion of the dataset, representing

37.2% of the total data. This is followed by **Positive** (27.9%) and **Negative** (25.8%) labels, which are fairly balanced, indicating a well-represented range of sentiments. However, the **Mixed** sentiment is significantly smaller, making up only 9.2% of the dataset.

The imbalanced representation of the Mixed sentiment label may present challenges for models in accurately learning and predicting this class, as it has far fewer examples compared to the other categories. On the other hand, the relatively balanced distribution between Positive and Negative labels, along with the dominance of Neutral labels, reflects a wide range of sentiment expressions typical of online platforms and provides a robust foundation for training models on sentiment analysis tasks.

## 4.2 Performance Evaluation

Table 4.2 provides a comparative analysis of different models on the Bengali-English code-mixed sentiment analysis task, assessed across four key metrics: accuracy, precision, recall, and F1 score on both the validation and test sets. The models are categorized into five types: traditional machine learning models (Logistic Regression, Random Forest, SVM), recurrent neural network (RNN) variants, multilingual language models (XLM-RoBERTa, mBERT), Bangla-specific language models (BanglaBERT, BanglishBERT), and English language models (DistilBERT, BERT). Among the machine learning models, SVM demonstrates the best performance, though it falls short when compared to the more advanced neural and language models. Recurrent neural networks, particularly RNN, perform poorly with both accuracy and F1 scores, while LSTM shows moderate improvement in performance. Multilingual models, especially mBERT, show strong results, with mBERT achieving the highest accuracy and F1 scores in the test set, demonstrating its strength in handling code-mixed language. The Bangla language models, BanglaBERT and BanglishBERT, also perform well, reflecting their ability to capture language-specific nuances. English language models like BERT stand out as the top performers, achieving the highest F1 score, indicating their strong capacity to manage the intricacies of code-mixed text in sentiment analysis. Overall, BERT and mBERT emerge as the leading models, while SVM shows the best results among the machine learning approaches.

### 4.2.1 Machine Learning Models

The machine learning models, including Logistic Regression, Random Forest, and SVM, offer a simple yet effective baseline for sentiment analysis in Bengali-English

**Table 4.2:** Performance of the proposed baselines based on accuracy, precision, recall, and F1 score.

| Model | Validation | | | | Test | | | |
|---|---|---|---|---|---|---|---|---|
| | **Acc** | **Precision** | **Recall** | **F1** | **Acc** | **Precision** | **Recall** | **F1** |
| **Machine Learning Models** | | | | | | | | |
| Logistic Regression | 0.668 | 0.656 | 0.668 | 0.662 | 0.667 | 0.614 | 0.667 | 0.639 |
| Random Forest | 0.672 | 0.661 | 0.672 | 0.666 | 0.648 | 0.635 | 0.648 | 0.641 |
| SVM | 0.694 | 0.676 | 0.694 | 0.685 | 0.660 | 0.637 | 0.660 | 0.648 |
| **Recurrent Neural Network Variants** | | | | | | | | |
| RNN | 0.406 | 0.308 | 0.406 | 0.350 | 0.401 | 0.352 | 0.401 | 0.375 |
| LSTM | 0.678 | 0.670 | 0.678 | 0.674 | 0.670 | 0.657 | 0.670 | 0.663 |
| **Multilingual Language Models** | | | | | | | | |
| XLM-RoBERTa | 0.726 | 0.709 | 0.726 | 0.717 | 0.698 | 0.642 | 0.698 | 0.669 |
| mBERT | 0.726 | 0.713 | 0.726 | 0.719 | 0.694 | 0.675 | 0.694 | 0.684 |
| **Bangla Language Models** | | | | | | | | |
| BanglaBERT | 0.721 | 0.668 | 0.721 | 0.693 | 0.698 | 0.642 | 0.698 | 0.669 |
| BanglishBERT | 0.694 | 0.715 | 0.694 | 0.704 | 0.686 | 0.653 | 0.686 | 0.669 |
| **English Language Models** | | | | | | | | |
| DistilBERT | 0.701 | 0.694 | 0.701 | 0.697 | 0.672 | 0.665 | 0.672 | 0.668 |
| BERT | 0.727 | 0.710 | 0.724 | 0.717 | 0.695 | 0.683 | 0.694 | 0.688 |

code-mixed data. While these models achieve relatively good accuracy, they are outperformed by more complex neural network models. SVM shows competitive performance with an accuracy of 66%, closely rivaling larger transformer-based models such as BanglishBERT.

### 4.2.2 Recurrent Neural Networks

The recurrent neural network (RNN) underperformed significantly compared to other baselines, with an accuracy of 40%. This can be attributed to its difficulty in capturing long-term dependencies in the code-mixed text. On the other hand, Long Short-Term Memory (LSTM) models performed notably better, achieving an accuracy of 67%. The improvement can be linked to LSTM's ability to handle long-term dependencies, which are critical for understanding the structure of code-mixed sentences.

### 4.2.3 Transformer-Based Models

Transformer-based models demonstrated superior performance, with BERT-CMB achieving the highest accuracy of 72.7% and F1 score of 68.8%. This success can be attributed to BERT's capability of handling the linguistic intricacies of English tokens present in the code-mixed data, resulting in better overall performance. Interestingly, the performance of multilingual models like mBERT and XLM-RoBERTa was slightly lower than BERT-CMB, which may be due to the lower proportion of Bengali data in the multilingual pre-training corpora.

Further pre-training of these models on code-mixed data, as described in the method-

ology, led to slight performance gains for all models, with BERT-CMB still being the top performer. This reinforces the importance of fine-tuning models on task-specific data for code-mixed text.

### 4.2.4 Training Loss Analysis

Figure 4.6 illustrates the training loss across 15 epochs for the baselines. We observe that all models converge before reaching the $15^{th}$ epoch. The only exception is the LSTM model which shows a slight indication of being benefited by additional training epochs. Excluding DistilBERT, the other BERT family models converged relatively faster in the earlier epochs. For most models, training for 5-8 epochs is appropriate to prevent overfitting.

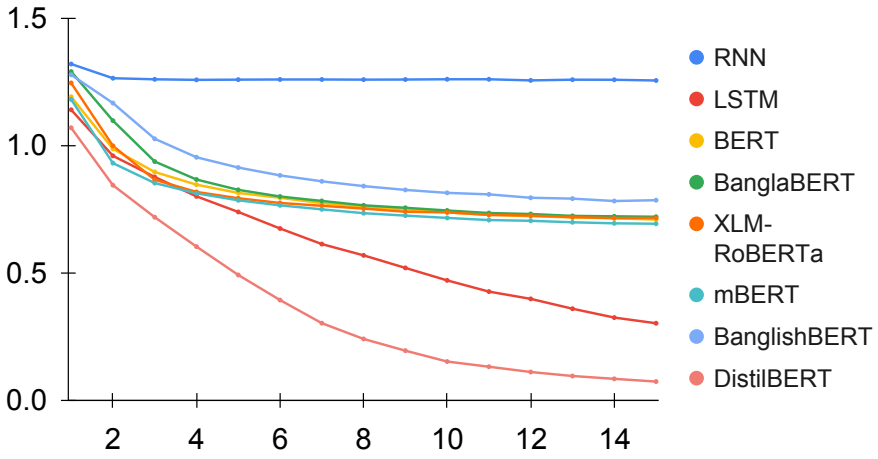



**Figure 4.6:** Training loss across 15 epochs for the baseline models.

The RNN model starts with the highest training loss ( 1.2) and shows minimal reduction, indicating poor learning efficiency. The LSTM model decreases steadily from 1.1 to below 0.4, reflecting strong learning and convergence. Both BERT and Bengali models reduce from 1.1 and 1.0 to 0.6, indicating efficient learning. The XLM model starts at 1.2 and decreases to 0.7, showing moderate efficiency. The mBERT model follows a similar trend to BERT and Bengali, decreasing from 1.1 to 0.6. The Banglish model shows a moderate decrease from 1.2 to 0.7 with slower convergence. The Distil model demonstrates the most significant reduction from 1.0 to below 0.3, indicating highly efficient learning, and exhibits the most efficient training, while the RNN model shows the slightest improvement. LSTM, BERT, Bangla, and mBERT models show robust learning, and Banglish and XLM models show moderate improvements.

The RNN model starts with the highest training loss ( 1.2) and exhibits minimal reduction, indicating poor learning efficiency and convergence. The LSTM model shows a

steady decrease from 1.1 to below 0.4, reflecting strong learning capability and effective convergence. The BERT model displays significant initial reduction in training loss from 1.1 to 0.6, indicating strong learning potential and efficient training. Similarly, the Bangla model follows a trend close to BERT, starting at 1.0 and reducing to 0.6, indicating effective learning.

The XLM model starts higher at 1.2, decreasing steadily to 0.7, showing moderate learning efficiency. The mBERT model decreases from 1.1 to 0.6, similar to BERT and Bangla models, indicating efficient learning. The Benglish model shows a moderate decrease from 1.2 to 0.7, with a slower convergence rate. The Distil model shows the most significant reduction from 1.0 to below 0.3, indicating highly efficient learning and rapid convergence.

Overall, the Distil model demonstrates the most efficient training process, while the RNN model shows the least improvement. LSTM, BERT, Bangla, and mBERT models exhibit strong and consistent reductions in training loss, reflecting their effective learning capabilities. Benglish and XLM models show moderate improvements with relatively slower convergence rates.

### 4.2.5 Accuracy vs F1-score

Figure 4.7 presents the accuracy and F1 scores of various machine learning and transformer-based models on the BnSentMix dataset for sentiment analysis. Traditional machine learning models like Logistic Regression, Random Forest, and SVM demonstrate solid performance, with SVM achieving the highest accuracy and F1 score among these models. In contrast, the Recurrent Neural Network (RNN) underperforms significantly, likely due to its inability to handle long-term dependencies effectively in code-mixed text. Long Short-Term Memory (LSTM) networks show notable improvement over RNNs, performing similarly to Random Forest and SVM, highlighting the importance of capturing long-term dependencies in sentiment classification tasks.

The transformer-based models, particularly BERT and its variants (BanglaBERT, BanglishBERT, and mBERT), outperform all traditional models, demonstrating superior performance in handling the linguistic complexity of code-mixed Bengali-English text. Among them, BERT-CMB (pre-trained on code-mixed Bengali-English data) achieves the highest accuracy and F1 score, showing its effectiveness in understanding both languages in a code-mixed context. Although DistilBERT, a lightweight version of BERT, lags slightly behind, it still performs better than traditional models. XLM-RoBERTa and mBERT, both multilingual models, also show competitive results but

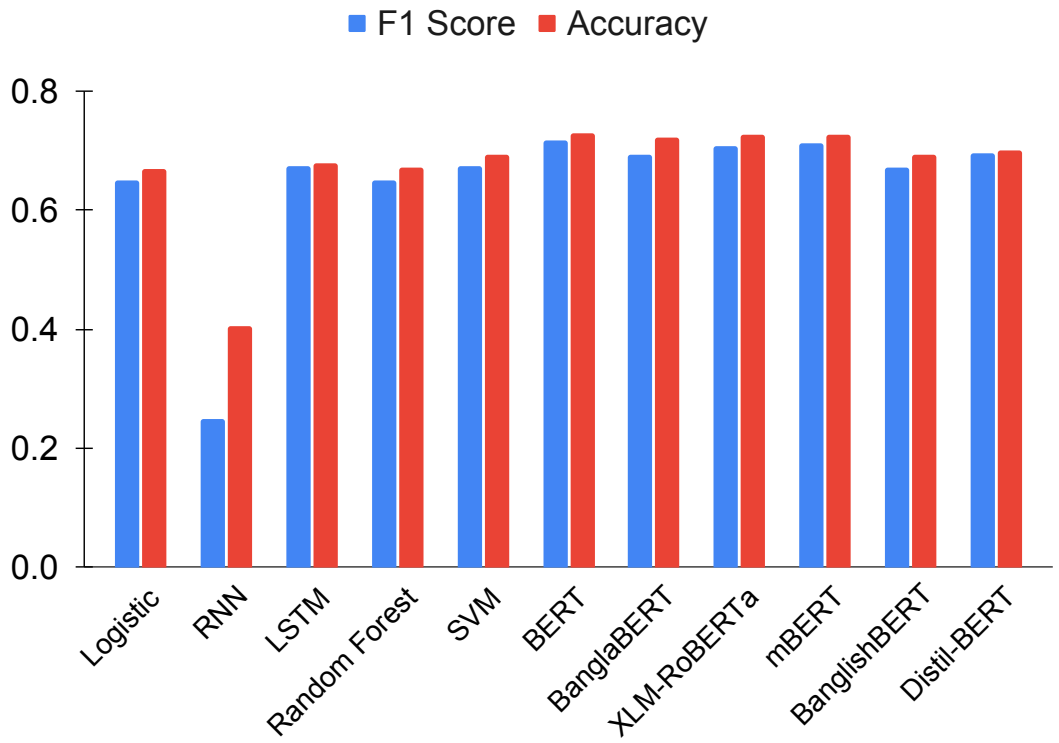



**Figure 4.7:** Accuracy vs. F1-Score of Baseline Models

fall short of BERT-CMB, likely due to their limited exposure to Bengali-English code-mixed text during pre-training. Overall, the results highlight that transformer-based models, especially those pre-trained on domain-specific or code-mixed data, are more effective for sentiment analysis in a Bengali-English code-mixed environment.

### 4.2.6 Hyperparameter Tuning

The hyperparameters play a crucial role in optimizing the performance of deep learning models. We analyze the role of hyperparameters for the best-performing BERT-CMB model and optimize them to gain superior performance.

#### Fine-tuning Epochs

Figure 4.8 shows the effect of fine-tuning epochs on the performance metrics. Both metrics increase up to the $6^{th}$ and stabilize onwards, suggesting keeping the number of fine-tuning epochs between 5 to 7.

Accuracy starts at 0.55, rapidly increases to 0.70 by epoch 3, and stabilizes around 0.72 from epoch 6 onwards. The F1 score starts at 0.50, rises sharply to 0.65 by epoch

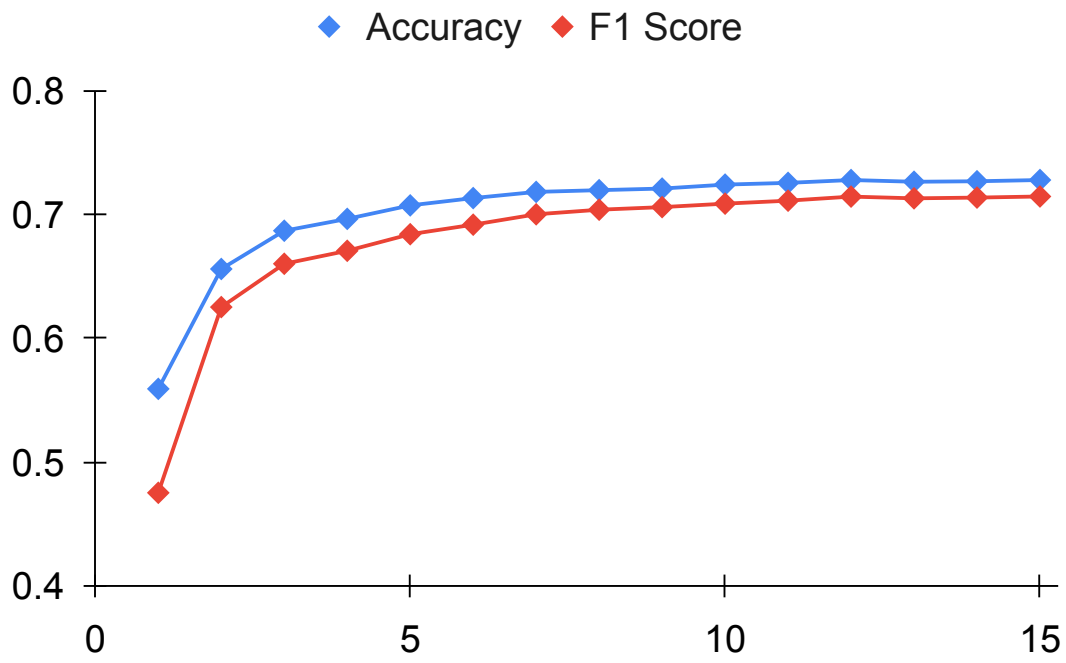



**Figure 4.8:** Epoch-wise training accuracy and F1 score curves of the best performing BERT-CMB model.

3, and stabilizes around 0.68 from epoch 6 onwards.

Both accuracy and F1 score improve significantly in the early epochs, indicating rapid learning. They stabilize around epoch 6, suggesting diminishing returns from further training. The slight difference between accuracy and F1 score indicates room for improvement in balancing precision and recall.

### 4.2.7 Learning Rate vs. Accuracy Analysis

Figure 4.9 illustrates the relationship between the learning rate and the accuracy for the BERT-CMB model. While tuning the hyperparameters, we observed the highest accuracy in the validation set at the learning rate of $1.25E{-}6$ which is slightly lower than the default learning rate of $1.5E{-}6$. The curve shows no particular advantage for higher or lower rates.

Accuracy starts at 0.722 for a learning rate of $1.20 \times 10^{-6}$, peaks at 0.730 for learning rates $1.25 \times 10^{-6}$ and $1.45 \times 10^{-6}$, and shows significant variation with other learning rates.

The highest accuracy is achieved at learning rates $1.25 \times 10^{-6}$ and $1.45 \times 10^{-6}$, suggesting these are optimal. The model's accuracy is highly sensitive to learning rate adjust-

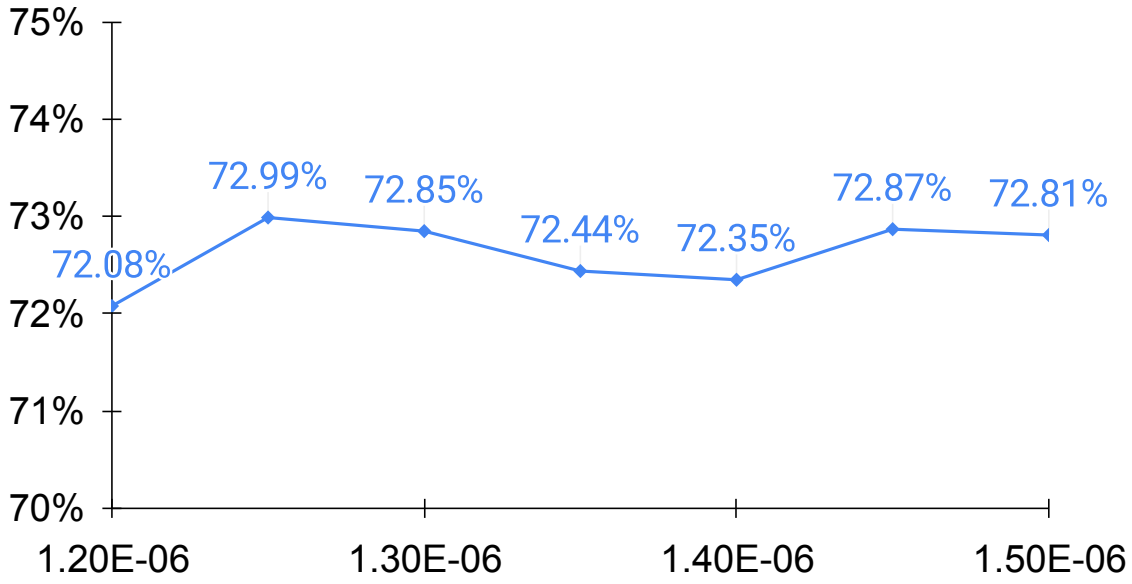



**Figure 4.9:** The effect of learning rate on the validation accuracy of the best performing BERT-CMB model.

ments, highlighting the importance of fine-tuning this hyperparameter. Too low or too high learning rates result in suboptimal performance, underscoring the need for careful selection.

### 4.2.8 Overall Summary

The analysis provides valuable insights into the training performance and hyperparameter tuning of different models. The Distil model shows exceptional training efficiency, while RNN lags behind. Accuracy and F1 score analyses suggest that significant performance gains are achieved within the first few epochs, with stabilization thereafter. The learning rate analysis underscores the critical role of selecting an appropriate learning rate for optimal model performance. These insights are crucial for selecting the most suitable model and hyperparameters for specific tasks, ensuring efficient resource utilization in model training.

The results indicate that while traditional machine learning models such as Logistic Regression, SVM, and Random Forest offer reasonable performance, deep learning models, particularly those based on the BERT architecture, significantly outperform them. Among the BERT variants, the standard BERT model showed the highest performance, closely followed by XLM-RoBERTa and mBERT. This highlights transformer-based models' superior ability to understand and classify text within our dataset.

These findings suggest that for tasks similar to the one studied, employing transformer-based models like BERT can substantially improve accuracy and F1 score, making them the preferred choice for achieving optimal performance.

### 4.2.9 Error Analysis

The performance of baseline models on our Bengali-English code-mixed dataset revealed several key challenges tied to the linguistic complexity of code-mixed text. Frequent switching between Bengali and English within sentences disrupted the models' ability to capture consistent syntactic and semantic patterns, especially for simpler models like Logistic Regression and Random Forest. Informal language, slang, and transliterated words prevalent in social media posts posed further difficulties, often leading to misclassification of positive or negative sentiments as neutral. Additionally, mixed sentiment sentences, where both positive and negative emotions coexist, were frequently misclassified, as traditional models struggle to detect nuanced shifts in sentiment within a single sentence. BERT-based models showed relatively better performance but still faced issues in handling mixed sentiments.

Another significant challenge stemmed from vocabulary limitations and cultural references unique to Bengali, where models, especially non-BERT ones, struggled with named entity recognition. This often led to incorrect sentiment classifications when proper nouns carried implicit sentiment. Furthermore, the imbalanced distribution of sentiment labels in the dataset, with fewer "neutral" and "mixed" examples, contributed to biased model predictions towards the more frequent "positive" and "negative" labels. Addressing these issues may require more advanced techniques, such as domain-specific fine-tuning of transformer models or data augmentation to improve the handling of underrepresented sentiment categories. Overall, these findings underscore the need for specialized approaches to effectively capture the nuances of sentiment in Bengali-English code-mixed text.

# Chapter 5

# Conclusion

## 5.1 Summary

In this work, we presented a new dataset for sentiment analysis of Bengali-English code-mixed text, designed to address the growing prevalence of multilingual communication in digital spaces. The dataset comprises 20,000 samples annotated across four sentiment labels: positive, negative, neutral, and mixed, offering a diverse and realistic representation of the sentiments expressed in code-mixed language. To ensure high-quality code-mixed data, we developed an automated filtering pipeline using pre-trained language models, enabling us to efficiently extract relevant samples from larger corpora with a high degree of accuracy. This pipeline is particularly useful in settings where manually labeled code-mixed data is scarce, as it provides a scalable approach for dataset creation that can be adapted to other language pairs.

The dataset also includes samples collected from various real-world platforms such as social media, online forums, and e-commerce reviews, reflecting a wide range of linguistic patterns, conversational contexts, and topic domains. This diversity makes the dataset an excellent benchmark for training machine learning models on complex, mixed-language sentiment analysis tasks, where the boundaries between languages are often fluid and unpredictable. By capturing linguistic and stylistic variations inherent to Bengali-English communication, we aim to contribute a resource that helps researchers and practitioners understand and model the unique challenges posed by code-mixed language.

To validate the utility of our dataset, we provided baseline evaluations using a combination of traditional machine learning and state-of-the-art transformer-based models. Among the models tested, we observed that transformer-based architectures, partic-

ularly fine-tuned BERT variants, performed the best, achieving an accuracy of 69.5% and an F1 score of 68.8%. However, simpler models like Logistic Regression and Random Forest also produced valuable insights into the complexity of the task, despite their lower performance. These baseline results serve as a starting point for future research and set a benchmark for evaluating new approaches to code-mixed sentiment analysis.

This research not only addresses the lack of high-quality, annotated datasets for Bengali-English code-mixed text but also opens up avenues for exploring various challenges in natural language processing, such as handling mixed sentiment within sentences, managing transliteration and spelling variations, and understanding culturally specific references. Given the limited availability of resources for low-resource language pairs, we believe this dataset will significantly aid in the development of more accurate and culturally nuanced sentiment analysis models.

In conclusion, our Bengali-English code-mixed sentiment analysis dataset, along with the baseline results and proposed methodologies, provides a strong foundation for advancing NLP in multilingual and code-mixed contexts. This work paves the way for further improvements in language models' ability to process code-mixed text, ultimately contributing to more inclusive, accurate, and culturally aware sentiment analysis systems. Future work may include expanding this dataset with more examples across sentiment categories, fine-tuning larger language models specifically for Bengali-English data, and exploring advanced techniques in transfer learning and data augmentation to further enhance model performance.

## 5.2 Future Work

Looking ahead, there are several avenues for extending this research:

- Developing a more generalized sentiment analysis model capable of handling both monolingual Bengali and code-mixed Bengali-English text. This would allow for a broader application of the model in real-world settings where the type of text (monolingual or code-mixed) varies unpredictably.
- Creating a comprehensive language identification model that can accurately detect not only Bengali-English code-mixed data but also other multilingual scenarios. Such a model could enhance the pre-processing pipeline by dynamically adapting to mixed-language inputs beyond Bengali-English, enabling greater versatility for future research in multilingual NLP.

- Expanding the dataset by incorporating more samples from various online sources, including e-commerce, social media, and news comments, to improve dataset diversity and model robustness. A larger, more varied dataset would help capture a wider range of linguistic and cultural nuances in Bengali-English communication.
- Exploring transfer learning and domain adaptation techniques to enhance model performance on low-resource code-mixed data. Leveraging large, pre-trained language models with fine-tuning on code-mixed datasets could significantly improve sentiment classification accuracy, especially for rare or complex linguistic patterns in Bengali-English code-mixed text.
- Investigating semi-supervised or unsupervised learning approaches to reduce the dependency on labeled data. Since labeled code-mixed data is often limited, semi-supervised techniques could help generate additional labeled examples, improving model training efficiency and expanding dataset size without intensive manual annotation.
- Developing techniques for detecting and handling implicit sentiment in code-mixed text, where sentiment may not be explicitly expressed but is inferred from context. This could involve using sentiment-specific embeddings or emotion-based word embeddings to enhance the model's understanding of nuanced sentiment expressions common in informal, code-mixed text.
- Conducting in-depth error analysis on sentiment misclassification to identify specific linguistic challenges within code-mixed text, such as handling transliteration, code-switching at clause boundaries, and colloquial expressions. Insights from such analysis could guide the development of targeted improvements to the model architecture and training data.

These future directions aim to enhance the robustness, scalability, and applicability of sentiment analysis models for code-mixed languages, ultimately advancing the field of NLP for low-resource languages like Bengali. Additionally, they provide a foundation for cross-lingual insights, improving the adaptability of models to diverse linguistic settings.

# References


[1] S. Agrawal and A. Awekar, "No more beating about the bush: A step towards idiom handling for indian language NLP," in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan: European Language Resources Association (ELRA), May 2018.

[2] G. I. Ahmad, J. Singla, and N. Nikita, "Review on sentiment analysis of indian languages with a special focus on code mixed indian languages," in *2019 international conference on automation, computational and technology management (ICACTM)*, IEEE, 2019, pp. 352–356.

[3] I. Ameer, G. Sidorov, H. Gomez-Adorno, and R. M. A. Nawab, "Multi-label emotion classification on code-mixed text: Data and methods," *IEEE Access*, vol. 10, pp. 8779–8789, 2022.

[4] U. Barman, A. Das, J. Wagner, and J. Foster, "Code mixing: A challenge for language identification in the language of social media," in *Proceedings of the first workshop on computational approaches to code switching*, 2014, pp. 13–23.

[5] A. Bhattacharjee, T. Hasan, W. Ahmad, *et al.*, "BanglaBERT: Language model pretraining and benchmarks for low-resource language understanding evaluation in Bangla," in *Findings of the Association for Computational Linguistics: NAACL 2022*, M. Carpuat, M.-C. de Marneffe, and I. V. Meza Ruiz, Eds., Seattle, United States: Association for Computational Linguistics, Jul. 2022, pp. 1318–1327.

[6] B. R. Chakravarthi, N. Jose, S. Suryawanshi, E. Sherly, and J. P. McCrae, "A sentiment analysis dataset for code-mixed Malayalam-English," English, in *Proceedings of the 1st Joint Workshop on Spoken Language Technologies for Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL)*, D. Beermann, L. Besacier, S. Sakti, and C. Soria, Eds., Marseille, France: European Language Resources association, May 2020, pp. 177–184.

[7] B. R. Chakravarthi, V. Muralidaran, R. Priyadharshini, and J. P. McCrae, "Corpus creation for sentiment analysis in code-mixed Tamil-English text," English, in *Proceedings of the 1st Joint Workshop on Spoken Language Technologies for*

*Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL)*, D. Beermann, L. Besacier, S. Sakti, and C. Soria, Eds., Marseille, France: European Language Resources association, May 2020, pp. 202–210.

[8] A. Chanda, D. Das, and C. Mazumdar, "Unraveling the english-bengali code-mixing phenomenon," in *Proceedings of the second workshop on computational approaches to code switching*, 2016, pp. 80–89.

[9] M. Cieliebak, J. M. Deriu, D. Egger, and F. Uzdilli, "A twitter corpus and benchmark resources for german sentiment analysis," in *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*, Valencia, Spain: Association for Computational Linguistics, Apr. 2017, pp. 45–51.

[10] A. Conneau, K. Khandelwal, N. Goyal, *et al.*, "Unsupervised cross-lingual representation learning at scale," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, Eds., Online: Association for Computational Linguistics, Jul. 2020, pp. 8440–8451.

[11] K. Dashtipour, S. Poria, A. Hussain, *et al.*, "Multilingual sentiment analysis: State of the art and independent comparison of techniques," *Cognitive computation*, vol. 8, pp. 757–771, 2016.

[12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, J. Burstein, C. Doran, and T. Solorio, Eds., Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186.

[13] N. Dey, M. S. Rahman, M. S. Mredula, A. S. Hosen, and I.-H. Ra, "Using machine learning to detect events on the basis of bengali and banglish facebook posts," *Electronics*, vol. 10, no. 19, p. 2367, 2021.

[14] D. Gautam, P. Kodali, K. Gupta, A. Goel, M. Shrivastava, and P. Kumaraguru, "Comet: Towards code-mixed translation using parallel monolingual sentences," in *Proceedings of the Fifth Workshop on Computational Approaches to Linguistic Code-Switching*, 2021, pp. 47–55.

[15] A. Gupta, A. Vavre, and S. Sarawagi, "Training data augmentation for code-mixed translation," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021, pp. 5760–5766.

[16] M. Hu and B. Liu, "Mining and summarizing customer reviews," in *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '04, New York, NY, USA: Association for Computing Machinery, 2004, pp. 168–177.

[17] Q. Jiang, L. Chen, R. Xu, X. Ao, and M. Yang, "A challenge dataset and effective models for aspect-based sentiment analysis," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 6279–6284.

[18] A. Joshi, A. Prabhu, M. Shrivastava, and V. Varma, "Towards sub-word level compositions for sentiment analysis of Hindi-English code mixed text," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, Y. Matsumoto and R. Prasad, Eds., Osaka, Japan: The COLING 2016 Organizing Committee, Dec. 2016, pp. 2482–2491.

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[20] S. Mæhlum, J. Barnes, L. Øvrelid, and E. Velldal, "Annotating evaluative sentences for sentiment analysis: A dataset for norwegian," in *Proceedings of the 22nd Nordic Conference on Computational Linguistics*, Turku, Finland: Linkoping University Electronic Press, Sep. 2019, pp. 121–130.

[21] N. H. Mahadzir *et al.*, "Sentiment analysis of code-mixed text: A review," *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, vol. 12, no. 3, pp. 2469–2478, 2021.

[22] S. Mandal, S. K. Mahata, and D. Das, "Preparing bengali-english code-mixed corpus for sentiment analysis of indian languages," *ArXiv*, vol. abs/1803.04000, 2018.

[23] S. Mandal and A. K. Singh, "Language identification in code-mixed data using multichannel neural networks and context capture," in *Proceedings of the 2018 EMNLP Workshop W-NUT: The 4th Workshop on Noisy User-generated Text*, W. Xu, A. Ritter, T. Baldwin, and A. Rahimi, Eds., Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 116–120.

[24] S. M. Mohammad, "A practical guide to sentiment annotation: Challenges and solutions," in *WASSA@NAACL-HLT*, 2016.

[25] A. Pratapa, M. Choudhury, and S. Sitaram, "Word embeddings for code-mixed language processing," in *Proceedings of the 2018 conference on empirical methods in natural language processing*, 2018, pp. 3067–3072.

[26] E. Pustulka-Hunt, T. Hanne, E. Blumer, and M. Frieder, "Multilingual sentiment analysis for a swiss gig," in *2018 6th International Symposium on Computational and Business Intelligence (ISCBI)*, IEEE, 2018, pp. 94–98.

[27] P. Rani, S. Suryawanshi, K. Goswami, B. R. Chakravarthi, T. Fransen, and J. P. McCrae, "A comparative study of different state-of-the-art hate speech detection methods in Hindi-English code-mixed data," English, in *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, R. Kumar, A. K. Ojha, B. Lahiri, *et al.*, Eds., Marseille, France: European Language Resources Association (ELRA), May 2020, pp. 42–48.

[28] M. S. Z. Rizvi, A. Srinivasan, T. Ganu, M. Choudhury, and S. Sitaram, "Gcm: A toolkit for generating synthetic code-mixed text," in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, 2021, pp. 205–211.

[29] B. Roark, L. Wolf-Sonkin, C. Kirov, *et al.*, "Processing South Asian languages written in the Latin script: The Dakshina dataset," in *Proceedings of The 12th Language Resources and Evaluation Conference (LREC)*, 2020, pp. 2413–2423.

[30] A. Rogers, A. Romanov, A. Rumshisky, S. Volkova, M. Gronas, and A. Gribov, "Rusentiment: An enriched sentiment analysis dataset for social media in russian," in *Proceedings of the 27th International Conference on Computational Linguistics*, Santa Fe, New Mexico, USA: Association for Computational Linguistics, Aug. 2018, pp. 755–763.

[31] N. Sabri, A. Edalat, and B. Bahrak, "Sentiment analysis of persian-english code-mixed texts," in *2021 26th International Computer Conference, Computer Society of Iran (CSICC)*, IEEE, 2021, pp. 1–4.

[32] S. Sitaram and A. W. Black, "Speech synthesis of code-mixed text," in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, 2016, pp. 3422–3428.

[33] K. Sreelakshmi, B. Premjith, and K. P. Soman, "Detection of hate speech text in hindi-english code-mixed data," *Procedia Computer Science*, vol. 171, pp. 737–744, 2020.

[34] S. Thara and P. Poornachandran, "Code-mixing: A brief survey," in *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2018, pp. 2382–2388.

[35] S. Thara and P. Poornachandran, "Code-mixing: A brief survey," in *2018 International conference on advances in computing, communications and informatics (ICACCI)*, IEEE, 2018, pp. 2382–2388.

[36] Y. Vyas, S. Gella, J. Sharma, K. Bali, and M. Choudhury, "Pos tagging of english-hindi code-mixed social media content," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 974–979.

[37] J. Wiebe, T. Wilson, and C. Cardie, "Annotating expressions of opinions and emotions in language," *Language Resources and Evaluation*, vol. 39, no. 2, pp. 165–210, May 2005.

[38] T. Wolf, L. Debut, V. Sanh, *et al.*, "Huggingface's transformers: State-of-the-art natural language processing," *arXiv preprint arXiv:1910.03771*, 2019.

[39] W. Yang *et al.*, "Functions and application of code-mixing and code-switching in a second/foreign language classroom," *Frontiers in Educational Research*, vol. 3, no. 4, pp. 38–40, 2020.