Analysing Code-Mixed Text in Programming Instruction Through Machine Learning for Feature Extraction

Myagmarsuren Orosoo[1], J Chandra Sekhar[2], Manikandan Rengarajan[3], Nyamsuren Tsendsuren[4], Dr. Adapa Gopi[5], Dr. Yousef A.Baker El-Ebiary[6], Dr. Prema S[7], Ahmed I. Taloba[8]

Mongolian National University of Education, Mongolia[1]
Professor in CSE, NRI Institute of Technology, Guntur, India[2]
Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology, Chennai, Tamil Nadu, India[3]
Lecturer, School of Mathematics and Natural Sciences, Department of Informatics, Mongolian National University of Education, Mongolia[4]
Associate Professor, Dept.of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Green Fields, Vaddeswaram, Guntur, Andhra Pradesh, India[5]
Faculty of Informatics and Computing, UniSZA University, Malaysia[6]
Department of English, Panimalar Engineering College, Chennai, India[7]
Department of Computer Science, College of Computer and Information Sciences, Jouf University, Saudi Arabia[8]
Information System Department, Faculty of Computers and Information, Assiut University, Assiut, Egypt[8]

***Abstract*—In programming education, code-mixed text using multiple languages or dialects simultaneously can significantly hinder learning outcomes due to misinterpretation and inadequate processing by traditional systems. For instance, students with bilingual or multilingual backgrounds may face difficulties with automated code reviews or multilingual coding tutorials if their code-mixed queries are not accurately understood. Motivated by these challenges, this paper proposes a Federated Bi-LSTM Model for Feature Extraction and Classification. This model leverages Bidirectional Long Short-Term Memory (Bi-LSTM) networks within a federated learning framework to effectively accommodate various code-switching methodologies and context-dependent linguistic elements while ensuring data security and privacy across distributed sources. The Federated Bi-LSTM Model demonstrates impressive performance, achieving 99.3% accuracy nearly 19% higher than traditional techniques such as Support Vector Machines (SVM), Multilayer Perceptron (MLP), and Random Forest (RF). This significant improvement underscores the model's capability to efficiently analyse code-mixed text and enhance programming instruction for multilingual learners. However, the model faces limitations in processing highly specialized code-mixed text and adapting to real-time applications. Future research should focus on optimizing the model for these challenges and exploring its applicability in broader domains of computer-assisted education. This model represents a substantial advancement in language-aware computing, offering a promising solution for the evolving needs of adaptive and inclusive programming education technologies. This advancement has the potential to transform language-sensitive computing, providing significant support for multilingual learners and setting a new standard for inclusive programming education.**

Keywords*—Code-mixed text; text processing; federated learning; bidirectional long short-term memory; programming education; real-time applications; computer-aided education

I. Introduction

Computer programming learning can be considered as a training process that helps students enhance their computational thinking abilities. Computational thinking is an essential skill that enables individuals to build and utilize the conceptual framework of computing. That is, computational thinking capacity encompasses a wide range of abilities, including logical reasoning, algorithmic choosing, and organized thinking, all of these can be applied to problem handling in a variety of learning situations or everyday life, not only in the academic arena of computational science. They provided a theoretical framework for computational analysis divided into five main groups: conditioned logic, algorithmic development, troubleshooting, simulations, and decentralized computation. These sections focus on the skills required for programming for computers, such as developing, designing, understanding, altering, and using programs [1] [2] [3]. This paradigm also exposed the social cooperative aspect of programming labour through distributed computation. These talents are the capabilities required for the generations to come to create knowledge and solve challenges in their online environments, implying that they represent a new education for all people in a digital age. Computational thinking abilities are essential not just for developers of computers, but for all prospective global citizens. This is because certain nations have recently added capacities for computational thinking in their school requirements [4] [5]. To accurately assess students' computational reasoning skills, a dependable and user-friendly tool for instructors and computers educators is required. From an educational standpoint, investigating learners' beliefs regarding their own skills in programming for computers, or self-esteem for computer programming, could offer an easy solution. It can happen that certain learners might reject the subject matter of programming in computers due to

dissatisfaction and their computational thought processes five categories could appear differently depending on the methods [6] [7].

India's linguistic diversity, shaped by a long history of foreign contact, results in widespread bilingualism and frequent code-mixing. English, an adopted language, is deeply embedded in the educational system, fostering comfort with bilingual communication among the populace. This phenomenon leads to the development of unique speech variants like Hinglish (English-Hindi) and Benglish (English-Bengali) [8]. Code-switching alternating between languages or dialects within a conversation is a common occurrence. Unlike code-mixing, which integrates words, phrases, and morphemes from different languages into a single sentence, code-switching involves shifting between different grammatical structures or components within the same interaction [11] [12]. However, the frequent use of code-mixed text poses significant challenges in educational settings, particularly in programming education. Traditional systems often misinterpret such text, leading to inadequate processing and hindering learning outcomes [9] [10]. This paper addresses these issues by proposing a Federated Bi-LSTM Model for Feature Extraction and Classification. The model utilizes Bidirectional Long Short-Term Memory (Bi-LSTM) networks within a federated learning framework to handle various code-switching methodologies and context-dependent linguistic elements while ensuring data security and privacy across distributed sources. Previous research has explored different aspects of sentiment analysis and language processing, yet handling code-mixed text in programming education remains an unresolved challenge. Traditional sentiment analysis systems struggle with complex linguistic phenomena such as sarcasm, humour, and cultural references, which are often overlooked. The proposed model represents a significant advancement in language-aware computing, offering a robust solution for accurately analysing code-mixed text and enhancing programming instruction for multilingual learners.

The mixing of codes is the use of grammar and vocabulary from various languages in the identical phrase. Code Mixing is a phenomenon of language that can happen in a multilingual environment where people communicate multiple languages at the same time [13] [14]. The issues in the Hindi-English code-mixed text were identified utilizing a PoS tag-annotated corpus [15]. To address the issue of shallow processing of Hindi-English code-mixed online information, an algorithm can recognize the script of the phrases, normalize them to their standardized forms, apply a PoS tag, and split them into pieces [16]. It is common in multilingual society and provides significant challenges to NLP applications such as sentiment analysis. The lack of a standard grammar for code-mixed sentences makes it difficult to determine composing language, which is necessary for sentiment evaluation utilizing based on rules and artificial intelligence techniques. In addition, when combining is up to each individual, there is no set mixing recommendations, which is a significant disadvantage. As an outcome, so as to analyse emotion on code-mixed information, novel programming representations must be created. Machine learning approaches play a vital role in characteristic extraction for processing code-mixed textual content, specifically in the context of computer programming training. By leveraging techniques including NLP and deep learning, those strategies' purpose to identify and extract relevant features from code-mixed textual content, which includes multiple languages. Feature extraction strategies might also encompass tokenization, component-of-speech tagging, syntactic parsing, and semantic evaluation to the particular traits of code-mixed textual content in computer programming. These extracted capabilities aid as inputs to machine learning methods for diverse responsibilities which include code recommendation, blunders detection, and automatic assessment of programming assignments. Effective characteristic extraction is critical for reinforcing the accuracy and efficiency of code-blended text processing structures in computer programming education, ultimately facilitating better learning practices and belongings for scholars in multilingual environments.

The problem statement arising from the reviewed research turns around effectively processing and understanding code-mixed text throughout various languages in different contexts. Code-mixed text poses challenges for sentiment evaluation, classification, and proficiency due to linguistic versions, cultural references, and casual language usage. Existing techniques struggle with imbalanced datasets, lack of complete lexicons, and limitations in generalizability to various linguistic contexts. Pre-processing strategies may not absolutely capture the intricacies of code-mixed languages, leading to errors in class and sentiment evaluation [19] [23]. These challenges prevent accurate extraction of functions and category of code-mixed text, impacting responsibilities along with sentiment evaluation and language identity. Therefore, there may be a need for superior models which could effectively pre-process code-mixed textual content, extract applicable capabilities, and accurately classify it, overcoming the limitations of current methods to facilitate better understanding and evaluation of code-blended data.

This paper aims to address the challenges of processing code-mixed text in programming education, particularly for multilingual learners. It introduces a Federated Bi-LSTM Model for Feature Extraction and Classification, leveraging Bidirectional Long Short-Term Memory (Bi-LSTM) networks within a federated learning framework. The model is designed to handle various code-switching methodologies and context-dependent linguistic elements while ensuring data security and privacy. The paper emphasizes the model's high accuracy and significant improvement over traditional techniques, highlighting its potential to enhance programming instruction for multilingual learners. Additionally, it discusses limitations in processing specialized code-mixed text and adapting to real-time applications, proposing future research directions to optimize the model and explore its broader applicability in computer-assisted education.

The key contributions of the study are given below:

- This work presents a novel Federated Bi-LSTM Model designed specifically for teaching computer programming through code-mixed text processing. In a federated learning framework, this model incorporates Bidirectional Long Short-Term Memory (Bi-LSTM) neural networks in a novel way, guaranteeing that

feature extraction and classification tasks are carried out effectively while adhering to strict data privacy standards.

- This study addresses several issues related to Code-Mixed Text Processing, such as contextual language aspects, different code-switching methods, and adaptability to different programming languages and learning environments. The suggested paradigm provides a complete way to handle code-mixed text by overcoming these obstacles.
- Federated Bi-LSTM outperforms SVM, MLP, and RF with an accuracy of 99.3%. This demonstrates how well it works to improve code-mixed text feature extraction and classification, boosting the dependability of teaching resources for programming.
- Through its accommodation of varied linguistic origins, the paradigm fosters inclusion and increases accessibility and equity in programming education.

This study's rest of the section is organized as follows. Section II includes the previous research on the code-mixed data processing. Problematic statement discussed in Section III. Section IV discussed the proposed method and the outcome of findings. Finally, Section V provides the conclusion of the paper.

II. Related Works

Gasiorek and Dragojevic [17] discussed about the participants viewed English-based web materials from fake groups which had either no mixing codes, Hawaiian terms without parenthetical interpretations, or Hawaiian words with English interpretations. Compared with inadequate mixing codes, code-mixing without glosses disturbed manufacturing, making members feel less accepted in the workplace. Code-mixing with gloss did not affect communication for individuals from Hawai'i, where it is widespread, but it did for others. No changes in being accepted were seen among mixing of codes with gloss and no combination of codes configurations. The findings suggest that mixing of codes in textual structured resources may result in expenses and advantages, and that these effects are dependent on the audience's familiarity with mixing of codes as a method and the structure of code-mixing. It may additionally limit the generalizability of the findings to other linguistic or cultural contexts. The results of code-mixing on workplace popularity and dispatch, doubtlessly overlooking other crucial elements that could have an impact on these outcomes, which include individual language ability levels. The study's reliance on self-pronounced perceptions of contributors can also introduce bias or inaccuracies within the statistics accumulated. The study's scope is limited to examining the effects of code-mixing with and without glosses, potentially overlooking other versions in code-mixing practices that would also affect verbal exchange and popularity inside the place of workplace.

Kodirekka and Srinagesh [18] discussed the Sentiment extraction from English-Telugu code-mixed data. The programming language used to access data from the API provided by Twitter must be a mixed Telugu and English code. That data consists of phonetically typed, noisy, lexicon-borrowed, code-mixed, unstructured text, and misspelled words. The identification of languages and emotional labels for classes are applied to every tweet in the collection of tweets as the first phase. The work of data standardization comes in second, and classification, the last phase can be accomplished in three ways: lexicon, machine learning, and deep learning. As part of the lexicon-based strategy, assign an appropriate language identifier to each and every tweet. Transliterate the roman script into Telugu words when the language used in the tag is in that language. Sentiment extraction from Telugu words is done using TeluguSentiWordNet, and sentiment extraction from English tokens is done using English SentiWordNets. This work proposes and applies, using data that has been normalized, an aspect-based sentiment analysis method. Sentiment scores are extracted using deep learning and machine learning approaches, and the outcomes are contrasted with earlier research. Its reliance on sentiment lexicons won't effectively capture the complexities of sentiment in code-blended textual content. These lexicons may not cover all viable versions and contexts of sentiment expressions in Telugu and English, main to inaccuracies or biases in sentiment extraction. Method of transliterating Roman script into Telugu words can also introduce errors, mainly for code-mixed text containing slang, abbreviations, or unconventional spellings. A factor-based totally sentiment evaluation method may forget about positive elements applicable to code-blended text, inclusive of cultural references or linguistic conventions unique to Telugu-English mixing. The evaluation with earlier studies may not absolutely interpretation for differences in dataset composition, preprocessing techniques, or evaluation metrics, restricting the validity of the contrasted consequences.

Madasamy and Padannayil [19] discovered that challenging to extract relevant information from large amounts of text. Social media has presently offered numerous options for scholars and professionals to do proper studies in this field of study. A large amount of the written material on social media platforms is in English or a code-mixed language of the region. In a nation that is bilingual like India, mixing of codes is common in social media debates. Multilingual customers usually utilize Roman script, an easier means of communication, rather than the local script when publishing comments on social media, and they commonly blend it with English in their native language. Stylish and syntactic inconsistencies present substantial obstacles in analyzing code-mixed text employing traditional methods. This research uses the ICON-2015 and ICON-2016 NLP tool challenge data sets to describe the novel word embedding via letter base encoding as characteristics for POS tagging code-mixed text in Indian languages. The suggested word embedding characteristics are contextually added, and the algorithm was trained using the widely used Support Vector Machine (SVM) classifier. Its dependence on the ICON-2015 and ICON-2016 NLP device assignment datasets, which won't completely represent the range and complexity of code-mixed text located in social media systems. These datasets may not capture the extensive range of linguistic versions, cultural references, and informal language utilized in real-international social media conversations. Word embedding techniques for part-of-speech (POS) tagging code-blended text in Indian languages may not

generalize well to different language pairs or code-mixing situations outdoor of the Indian context. Use of Support Vector Machine (SVM) classifier might also constraint its overall performance in comparison to greater advanced device studying or deep getting to know strategies, in particular while dealing with the stylistic and syntactic inconsistencies essential in code-mixed text. POS classification might also overlook other critical elements of code-mixed text processing, consisting of sentiment analysis.

Tareq et al. [20] addressed that online review comments regularly mix languages, use foreign characters, and do not follow traditional grammar conventions. A shortage of identified code-mixed data in a low-resource languages such as Bangla complicates computerized sentiment analysis. To solve this, this study collected online feedback on various goods and created an augmented Bangla-English code mix database. On these sentiments collections we also evaluate several additional models from the available research. To increase cross-lingual contextual awareness, this study describes an inexpensive but successful data augmentation strategy that can be used using current word embedding methods and does not require an additional database. The outcomes from the experiments indicate that using our data augmentation method to train word-embedded algorithms may assist the model's accuracy in gathering the cross-lingual connections for code-mixed sentences. This can enhance the overall efficacy of current machine learning models in both supervised learning and zero-shot cross-lingual flexibility. The dataset used in this study may not completely establish the range of code-mixed languages. The effectiveness of the proposed data augmentation strategy may additionally range throughout one-of-a-kind languages, and its generalizability to different low-aid languages beyond Bangla can be limited. Sentiment evaluation cannot recollect the whole variety of factors that may affect model overall performance, inclusive of domain-precise language variations in sentiment analysis. Word embedding strategies for go-lingual contextual focus may also inattention to different approaches or strategies that could enhance the model's accuracy. The evaluation of the model efficacy may be restricted through the provision and representativeness of the accumulated online feedback data.

Srinivasan and Subalalitha [21]investigates the issue of disparity between classes in sentimental analysis, highlighting its significance. There has been limited research on sentimental analysis with an unequal class label distribution. The paper also discusses another facet of the issue, which incorporates the idea of "Code Mixing." Code mixed data includes text that alternates among two or more languages. Unequal class distribution is a frequently seen phenomenon in code-mixed data. Existing research has primarily focused on sentiment analysis in monolingual data, rather than code-mixed data. This work tackles all of these challenges and proposes a strategy for analysing sentiments for a class imbalanced code-mixed dataset using a method of sampling along with Levenshtein distance measures. Specific method of sampling combined with Levenshtein distance measures to deal with class imbalance in code-mixed datasets for sentiment evaluation. This method may additionally offer insights into mitigating class disparity, its effectiveness can be sensitive to the characteristics of the dataset and the languages involved inside the code mixing. The proposed method's applicability to specific languages or code-mixing eventualities is probably limited, as sampling techniques and distance measures to yield various outcomes across linguistic contexts. The assessment of sentiment analysis performance using this approach won't absolutely capture the challenges inherent in real-world code-combined facts.

Jain, Jindal, and Jain [22] offered a solution for code-mixed Hindi-English social media writing that includes language recognition, detection, and rectification of both non-word and real-word problems that happen concurrently. Each recognized languages have unique obstacles and difficulties. Errors are recognized separately in every language, and a suggested list of incorrect terms is generated. Following that, a fuzzy graph between distinct phrases from the suggested lists is created utilizing various semantic relationships in Hindi WordNet. The context-embedded data and fuzzy graph-based similarity measurements are utilized to identify the correct term. Several studies are carried out on various social media databases sourced from the social media platform, YouTube, Twitter, feedback, blog posts, and Messenger. This method may be restrained by means of accuracy of Hindi WordNet, probably leading to inaccuracies in the identification of accurate phrases, especially for much less common or specialised vocabulary. The effectiveness of the solution may range throughout different social media platforms and textual content sorts, as the linguistic patterns and code-mixing phenomena can range notably between platforms and user communities. It can be stimulated with the aid of the best and representativeness of the social media databases used for evaluation, as biases or inconsistencies of data to affect the generalizability of the findings. The solution's scalability to other language pairs or code-mixing eventualities past Hindi-English can be restricted, as the proposed technique might not fully generalize to different linguistic contexts.

Shanmugavadivel et al. [23] suggested a method that uses a code-mixed collection of Tamil and English languages. To address the group's imbalances issue, resample is used, and the effect is evaluated. Initial processing incoming textual information can help code-mixed data categorization by eliminating extraneous content. The purpose of this study is to investigate the influence of pre-processing on Tamil code-mixed data using a variety of methods for pre-processing such as symbol elimination, repeating text elimination, and spelling and grammar, sign, and numerical removal. The pre-processed text is used to train classical deep learning, machine learning, transfer learning, and hybrid deep learning models, and the precision of each model is evaluated before and following preprocessing. Conventional machine learning methods rely on several weighing methods for choosing features. The primary goal this study's findings are to create hybrid deep learning algorithms that combine CNN with LSTM and CNN with Bi- LSTM in order to automatically gather both global and local characteristics from code-mixed data for sentiment evaluation and then categorize the Tamil code-mixed information to identify favourable, adverse, mixed feelings, and unknown state. The effectiveness of hybrid deep learning frameworks has been assessed through assessing them to modern approaches that included various conventional methods. The dependence

on pre-processing techniques for dealing with code-mixed data, which won't completely capture the intricacies of Tamil and English languages. The effectiveness of CNN with LSTM and CNN with Bi-LSTM models can be stimulated through elements including hyperparameter tuning and dataset length. The assessment of the models' effectiveness may also be restrained by means of the choice of metrics used, possibly overlooking positive distinctions in sentiment analysis. The assessment of modern strategies might not absolutely capture the improvements in the subject, as more recent strategies. The generalizability of the findings may be confined if the dataset used isn't always representative of actual global code-mixed statistics throughout specific domain names or contexts.

Nelatoori and Kommanti [24] present a Co-Attentive Multi-task Learning framework based on transfer learning for understanding Hindi-English (Hinglish) texts with limited resources. The collaborative goals of rationale/span identification and hazardous comment categorization form a strong multifaceted learning purpose. This task interaction element is intended to take advantage of the bidirectional awareness provided by the categorization and span forecasting activities. The model's combination function of loss is derived from the separate loss coefficients of these two jobs. Though an English harmful span identification dataset exists, there is currently no Hinglish code-mixed text database. As a result, they created a set of data that includes harmful span classifications for Hinglish code-mixed text. Harmful span classifications in Hinglish code-combined text. The synthetic generation of facts won't fully seize the range and complexity present in actual-world situations, probably excluding the version's ability to generalize effectively. The absence of a longtime Hinglish code-combined text database increases worries approximately the version's adaptability to diverse linguistic variations and contexts within Hinglish. The effectiveness of difference in a constrained-aid placing will also be restrained via the supply and representativeness of the pre-skilled models, probably affecting the version's overall performance on unique responsibilities. The selected multi-mission approach may introduce dependencies between tasks, and the version's performance can be sensitive to the stability between cause/span identification and risky remark categorization goals.

Above studies highlights the challenges and solutions in code-mixed textual content processing across different domains. One look at explores the impact of code-mixing on administrative centre dynamics, revealing how glosses have an effect on communication and popularity. Another discusses sentiment extraction from English-Telugu code-mixed data, providing lexicon-based totally and deep learning methods. Focus on word embedding for POS tagging in Indian code-mixed text is seen in another study. Introducing a Bangla-English code-mixed sentiment dataset and a statistics augmentation technique is the point of interest of a distinct examine. Addressing sentiment evaluation in imbalanced code-mixed datasets is another thing explored. A method for correcting mistakes in Hindi-English code-blended social media content material is proposed in a single examine. Pre-processing and deep learning models for Tamil-English code-combined sentiment evaluation is offered in every study. A Co-Attentive Multi-Undertaking Learning framework for Hindi-English textual content know-how is advanced. That research makes a contribution insight into the complexities of code-mixed text processing and suggests diverse methodologies to cope with those demanding situations, even though with capacity boundaries in generalizability and effectiveness.

III. Proposed Federated Bi-LSTM Model for Feature Extraction and Classification in Code-Mixed Text Processing

The data collection system for SentMix-3L, a Bangla-English-Hindi code-combined dataset, involves several steps. Firstly, a wide variety of text information is combined from computer programming, boards, and different online resources in which code-mixed conversations are regularly occurring. The collected data undergoes preliminary preprocessing to take away noise, consisting of inappropriate content material or reproduction entries. Subsequently, the textual content is tokenized to interrupt it down into words or tokens, considering the linguistic distinctions of all 3 languages involved. After tokenization, the sentences are labelled primarily based on language to ensure proper handling of code-mixed segments. This step is crucial for retaining the integrity of the code-mixed dataset and allowing accurate evaluation. Following sentence classification, the data is partitioned into suitable training, validation, and test sets for model improvement and assessment. The Federated Bi-LSTM Based Feature Extraction and Classification process then makes use of a decentralized method to model training, where Bi-LSTM architectures are deployed at multiple nodes, maintaining nearby data. The models are learned domestically through the use of federated learning techniques, changing version updates even maintaining data privateness. Model aggregation is done to mix the local models' parameters, creating an international version capable of extracting functions and classifying code-mixed text accurately. This methodology guarantees effective data preprocessing, language-aware tokenization, and decentralized model training for strong feature extraction and classification in code-mixed textual content processing. Fig. 1 provides the block illustration of the proposed federated Bi-LSTM model.

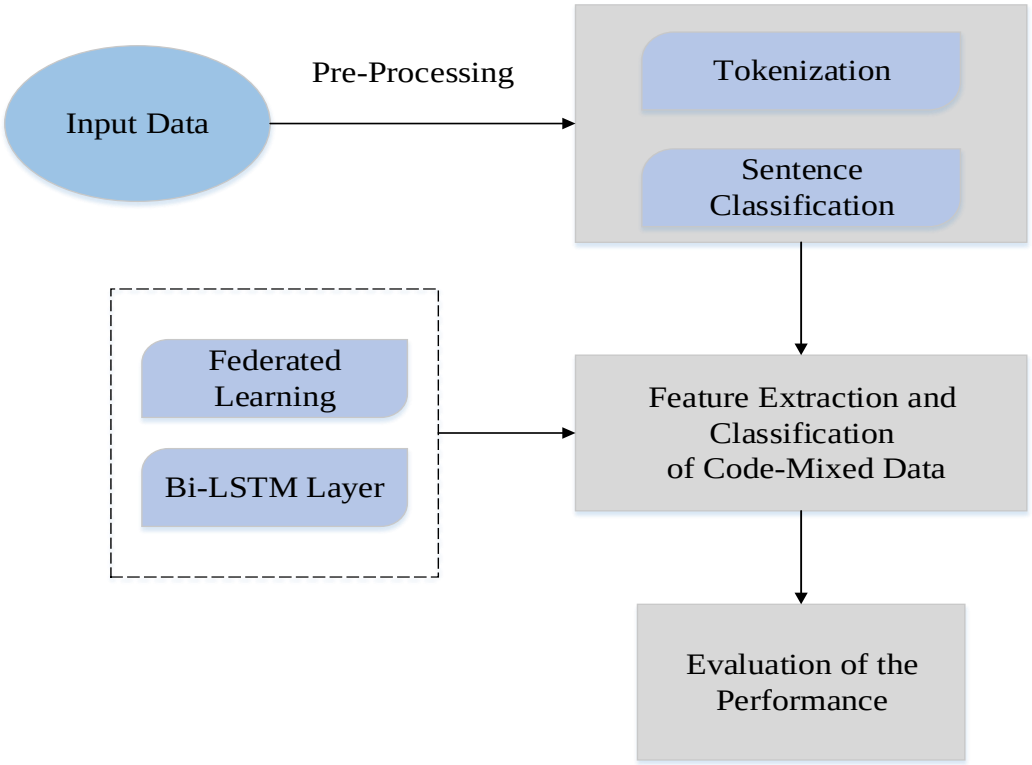

Fig. 1. Block illustration of the proposed federated Bi-LSTM model.

A. *Data Collection*

The data collection process for SentMix-3L, a Bangla-English-Hindi code-combined dataset for sentiment evaluation. This dataset became especially intended to address the need for resources in code-blended sentiment evaluation throughout multiple languages to gather data, researchers employed various techniques to make sure variety and representativeness. This procedure involved significant manual looking and scraping strategies to acquire a variety of code-mixed textual content samples encompassing Bangla, English, and Hindi languages. The researchers collaborated with language experts and native speakers proficient in Bangla, English, and Hindi to verify the authenticity and accuracy of the gathered data. Stringent quality control measures had been implemented to filter out noise and inappropriate content, ensuring that the dataset comprised significant times appropriate for sentiment analysis duties. The very last dataset, consisting of 1,007 instances, underwent thorough validation to confirm its suitability as a level for sentiment analysis inside the context of code-blended textual content processing across three languages. The data collection technique for SentMix-3L involved a mixture of manual curation, expert validation, and high-quality assurance techniques to create a treasured useful resource for advancing research in code-combined sentiment evaluation [25]. Table I gives the sample of SentMix-3L dataset.

TABLE I. SAMPLE OF SENTMIX-3L DATASET

	English	Hindi	Bangla	Other	All
Standard Deviation	2.94	5.81	8.39	9.70	19.19
Average	5.96	15.03	31.91	35.98	88.87
Types	1073	1474	8167	9092	19686
Tokens	5998	1474	32133	36232	89494

B. *Data Pre-Processing using Tokenization and Sentence Classification*

Data pre-processing is an essential phase in improving and extracting relevant findings from the information. It removes shortcomings and discrepancies in data. Inaccuracies in data can lead to incorrect or incomplete data, affecting its precision. During the phase of preprocessing, execute these procedures:

1) Tokenizing: To improve the quality of the data, unnecessary content, such as Hyperlinks and html text, is deleted utilizing regex-based pattern recognition after data acquisition. The sentences were tokenized into words using NLTK Tokenizer. The sentences with fewer than five phrases were eliminated due to their noise and lack of data. To implement the NLTK tokenizer for Bangla and Hindi, modify the result to adjust for differences in punctuation between English, Hindi, and Bangla.

2) Sentence classification: The database is separated into three scripts: English-Hindi, English-Bangla, and English (monolingual). The sentences that are not written in English-

Hindi or English-Bangla script will be removed in the initial phase.

Sentence Classification Algorithm
Input: *Sentence*
Codes: αrefers the English script, β refers the English-Hindi script, γ refers the English-Bangla script
Output:*α,β,γ (sentence classification)*
Stage 1: Read the input
Stage 2:Calculate the length of input
Stage 3: Split the input into $v_1, v_2, \dots \dots v_n$
Stage 4: Each terms refers $v_i = 1$ *to n*
if v_i*=English*
Increase the value of count
else, Stop
End for

Stage5: if value of the count =words extracted from input α, orβ, or γ
Stage 6: End

The sentence classification system distinguishes between code-mixed (β, γ) and non-code-mixed (α) phrases. The system receives phrases as input and divides them into words ($v_1, v_2, \dots \dots v_n$). The linguistic detector checks every single word in the sentence to determine if it is English or not. Code mixed phrases are those that include non-English terms. Using the sentence classification technique, 600 sentences were identified as (α) and eliminated from the sample.

C. Federated Bi-LSTM based Feature Extraction and Classification in Code-Mixed Text Processing

Federated Bi-LSTM (Bidirectional Long Short-Term Memory) is a novel approach in code-mixed text processing that addresses the challenging situations modelled by using code-mixed textual content, in which multiple languages are mixed in the same sentence. This approach utilizes federated learning data of, a decentralized method to educate Bi-LSTM models on data chosen from exceptional sources without centralizing the data. The Bi-LSTM neural structure is employed for feature extraction from code-mixed textual content. Bi-LSTM networks are especially powerful in capturing long-range dependencies and contextual data in sequential information. By processing text bidirectionally, these networks can effectively capture the features present in code-mixed language. The trained Bi-LSTM models are then used for classification on code-mixed textual content, along with language identification, and sentiment analysis. The federated learning framework assurances privacy and data security by means of allowing models to train locally on distributed data resources without sharing raw data. Fig. 2 shows the architecture diagram of Bi-LSTM.

Federated learning is a decentralized learning approach that incorporates data from various sources using privacy-protected technology to generate a worldwide framework. During training of models, individuals are able to share necessary details regarding the model (variables, framework, range, etc.) using various methods (e.g., simple text, data encryption, sound), but local data used for training is not shared. The exchange of information protects local user data and reduces the risk of information leakage. Federated learning models can be distributed and implemented by multiple users [26]. The training procedure can be divided into three parts:

Step 1- Initialization of Task: The central server sets the learning the desired level, variables, and output components, then sends a global model (v_g^0) to the device.

Step 2-Training and Update of the Training Model:The global framework v_g^t has been obtained from the server where it is stored, and $t$denotes the present learning process. The device i learns the model's data on its own and modifies parameters such as v_i^t. The purpose of the training process is to discover the most suitable localized version by reducing the loss functions ($l(v_i^t)$ and uploading it to a server using the latest local version variable (v_i^a). It is given in Eq. (1).

$$v_i^a = arg_{v_i^t}\, minimum(l(v_i^t)) \tag{1}$$

Step 3-Accumulation and update of the Global Model: In Eq. (2), the server collects the parameters of the model from participating components, combines them as well upgrades the global model (v_g^{t+1}).

$$l(v_i^t) = {}^1\!/_n \sum_{i=1}^{n} l(v_i^t) \tag{2}$$

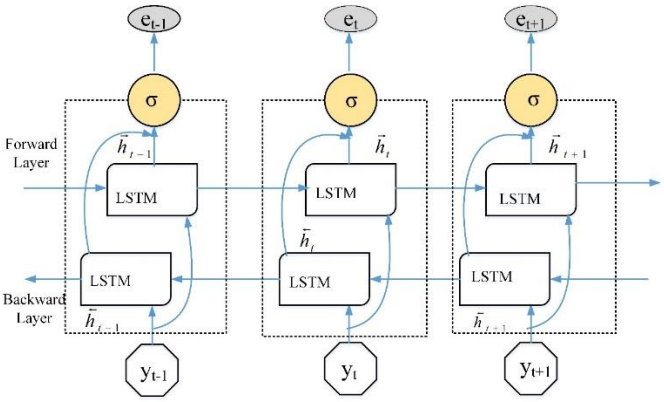

Fig. 2. Architecture diagram of Bi-LSTM.

The Bi-LSTM network captures features of input before and following processing. The present position of the Bi-LSTM secret layer is q_t. The forward-looking hidden layer state ($\overrightarrow{h_t}$) and the backwards hidden layer state ($\overleftarrow{h_t}$) can both find t. t is composed of two parts. The current state of the forward hidden layer. The currently active inpute_t and the forward-looking hidden layer state are used to determine the value of $t+1$ at time $\overrightarrow{h_{t-1}}$. The backwards layers that are hidden values include. The present input and backwards hidden layer state ($\overleftarrow{h_{t+1}}$) determine t at time $t+1$. The computation method is as follows:$v_i = (1,2,3, \dots \dots 6)$ represents the amount of weight transferred from a single layer of cells to subsequent. It is given in Eq. (3), (4) and (5).

$$\overrightarrow{h_t} = f(v_1 e_t + v_2 \overrightarrow{h_{t-1}}) \tag{3}$$

$$\overleftarrow{h_t} = f(v_3 e_t + v_5 \overleftarrow{h_{t+1}}) \tag{4}$$

$$q_t = g(v_4 \overrightarrow{h_t} + v_6 \overleftarrow{h_t}) \tag{5}$$

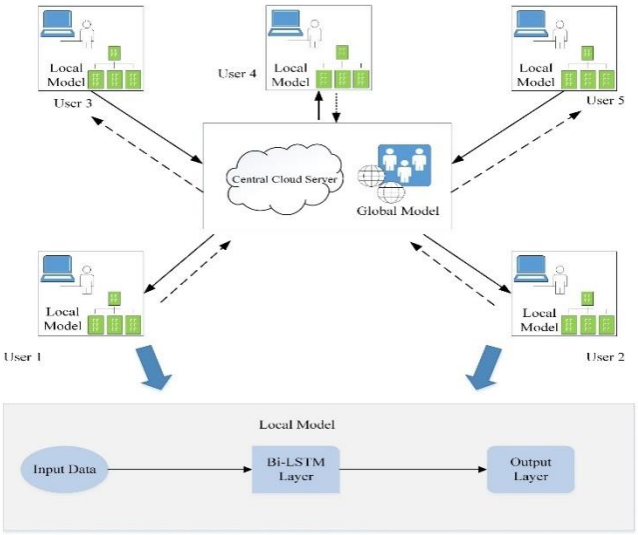

Fig. 3. Federated Bi-LSTM framework.

Federated Bi-LSTM in Code-Mixed Text Processing involves many steps. First, code-mixed textual content from numerous resources is collected and pre-processed to ensure consistency. Then, a federated learning framework is recognized, wherein multiple nodes preserve their information regionally to ensure privateness. Each node trains a Bi-LSTM neural network regionally to extract features from the code-mixed textual content. The models are trained in the usage of approaches to replace parameters. After local training, model updates are aggregated using federated averaging to create a global model. This global model is then evaluated on a separate validation to assess its overall performance in classification tasks. The technique can be iterated to refine the models in addition, incorporating feedback and updates from the federated learning knowledge of the framework. This method ensures accurate classification at the same time as keeping data privacy and security across disbursed data sources. Fig. 3 shows the framework for federated Bi-LSTM Model in code-mixed text processing.

IV. Results and Discussion

This study enables the automatic extraction of relevant features from code-mixed textual content, permitting better understanding and analysis of programming concepts conveyed in mixed-language contexts. Through strategies along with federated Bi-LSTM, effectively capture the difficult relations between different programming languages and human languages. General outcomes imply that those machine learning strategies can appropriately identify and extract code snippets, programming key phrases, and syntactic structures embedded within code-mixed textual content. They contribute to enhancing the learning enjoy by students by offering automated support in understanding multilingual programming environments, selling inclusivity, and catering to diverse linguistic backgrounds. However, challenges remain in adapting these strategies to precise programming languages, managing with versions in code-mixed textual content structures, and making sure of robustness across special academic contexts.

A. Training and Testing Accuracy

Fig. 4 presents the training and testing accuracy of a federated Bi-LSTM of version at specific epochs for the duration of the training manner. Each row corresponds to a selected epoch, indicating the wide variety of iterations through the training dataset. The training accuracy measures the model's performance at the data it changed into educated on, while the testing accuracy evaluates its overall performance on unseen statistics, providing an estimate of how properly the model generalizes to new times. Because the training progresses, both training and testing accuracies may additionally begin rather low. This is expected because the version learns to become aware of styles and relationships within the data. As training continues, the model's overall performance typically improves, pondered in increasing accuracy positions for each training and testing datasets. The fluctuations in accuracy ratings among epochs may also imply versions inside the model's potential to generalize. The conjunction of training and testing accuracies closer to better values to later epochs, suggests that the variety is becoming more robust and effective in both studying from and generalizing to new data, resulting in progressed universal overall performance.

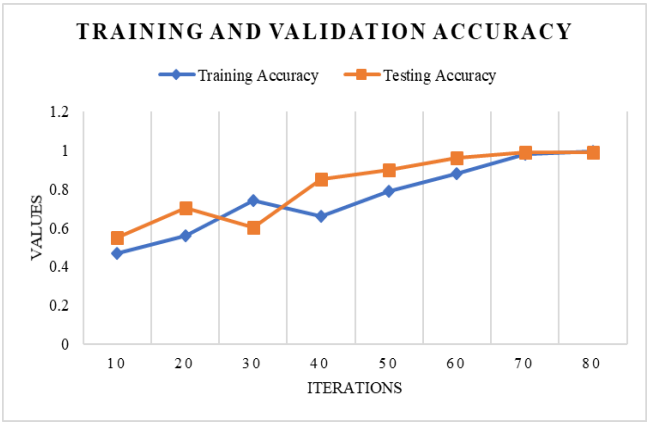

Fig. 4. Training and validation accuracy of federated Bi-LSTM.

B. Training and Testing Loss

Fig. 5 illustrates the training and testing values at various epochs all through the training process of a system mastering version. Loss, in this context, refers to a measurement of the type's predictions align with the real goal values. Lower loss values indicate higher alignment and, consequently, higher model overall performance. At some point in the early epochs, both training and testing loss values may additionally start incredibly high. This is expected because the version's parameters are initialized randomly, and it has no longer learned to appropriately expect the goal values. As training progresses, the version adjusts its parameters via optimization techniques like gradient descent to limit the loss characteristic. Throughout the training manner, the system's performance typically improves, leading to reducing loss values for each the training and testing datasets. Lower training loss suggests that the version is effectively from the learning data, even as lower trying out loss suggests that the model is generalizing nicely to unseen data.

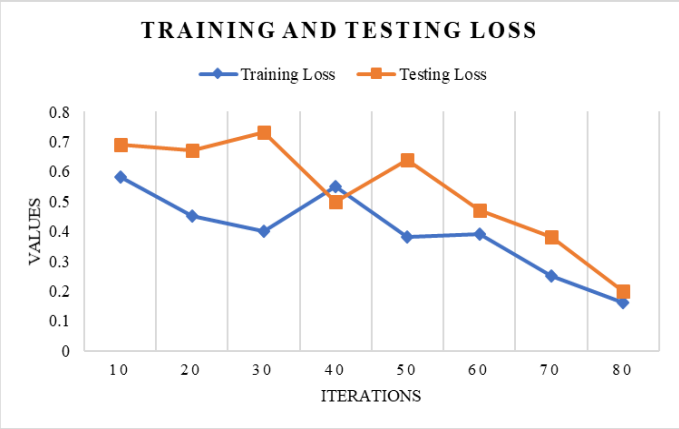

Fig. 5. Training and validation loss of federated Bi-LSTM.

C. Evaluation of Performance

Table II and Fig. 6 offer the performance metrics, inclusive of accuracy, precision, recall, and F1 score, for distinct device machine learning strategies: Support Vector Machine (SVM), Multi-Layer Perceptron (MLP), Random Forest (RF), and the Proposed Federated Bi-LSTM model. Accuracy measures the overall correctness of the model's predictions, at the same time as precision displays the proportion of efficaciously expected fine instances among all instances predicted as positive. Recall suggests the percentage of successfully expected high-quality instances amongst all real superb times, even as the F1 score is the harmonic mean of precision and don't forget, providing a balanced degree of a model's performance. Comparatively, the proposed Federated Bi-LSTM version outperforms the traditional machine learning methods, reaching considerably better accuracy, precision, remember, and F1 rating. This demonstrates the effectiveness of leveraging Bi-LSTM networks within a federated getting to know framework for code-combined textual content category duties. The advanced performance of the Federated Bi-LSTM model underscores its potential for appropriately classifying code-mixed text processing retaining data privacy and protection throughout distributed data sources.

TABLE II. COMPARISON OF PERFORMANCE METRICS

Methods	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)
SVM[27]	76.98	74.98	73.61	74.05
MLP[27]	80.22	78.08	78.8	78.31
RF[27]	77.65	75.81	75.81	75.67
Proposed Federated Bi-LSTM	99.3	99	98.9	99.5

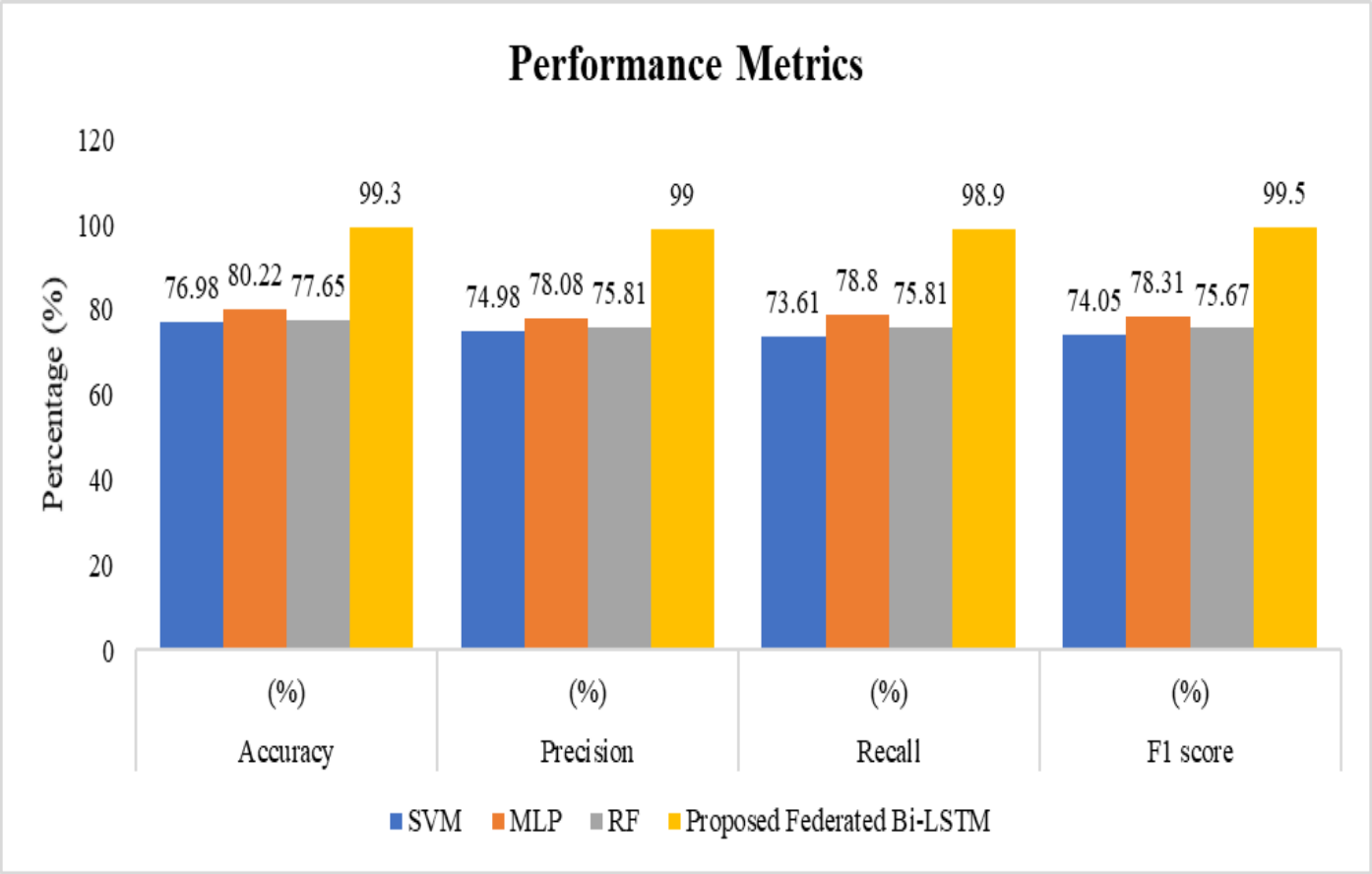

Fig. 6. Comparison of performance metrics.

D. Error Metrics Comparison

Table III and Fig. 7 offers the Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) metrics for both Support Vector Machine (SVM) and the proposed Federated Bi-LSTM model, these metrics are generally used to assess the performance of regression models by measuring the distinction among anticipated and real values. Comparing the two models, the Federated Bi-LSTM version demonstrates appreciably lower RMSE and MAE values as compared to SVM. This shows that the Federated Bi-LSTM model provides greater correct predictions with smaller errors. The superior overall performance of the Federated Bi-LSTM model shows its effectiveness in capturing complicated styles and relationships in the records, in particular within the context of code-mixed text processing. The decrease errors values show the assistance of the Federated Bi-LSTM model signify its capability to enhance predictive accuracy and improve effects in regression responsibilities, showcasing its suitability for numerous packages wherein particular predictions are essential.

TABLE III. COMPARISON OF ERROR METRICS

Metrics	SVM[28]	Proposed Federated Bi-LSTM
RMSE	0.2971	0.14
MAE	0.2304	0.076

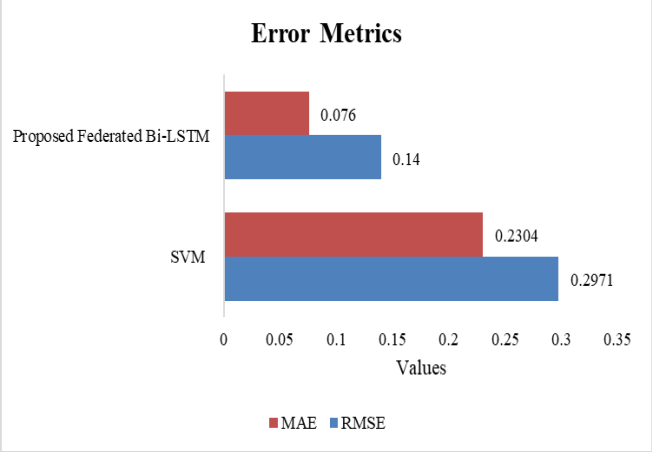

Fig. 7. Comparison of error metrics.

E. Discussion

The results of the evaluation between Support Vector Machine and the proposed Federated Bi-LSTM version, as indicated by Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) metrics, highlight the superiority of the Federated Bi-LSTM model in regression requirements. With an extensively lower RMSE and MAE, the Federated Bi-LSTM model demonstrates its capacity to offer more accurate predictions with smaller errors compared to SVM [27]. This suggests that the Federated Bi-LSTM model is skilful at capturing the intricate patterns and relationships inside the data, in particular in complicated contexts which includes code-mixed textual content processing. The decrease error values done by the Federated Bi-LSTM version represent its capability to establish predictive accuracy and reliability, underscoring its suitability for regression obligations throughout various domain names.

The rate of overall performance among SVM and the proposed Federated Bi-LSTM version underscores the importance of leveraging advanced deep learning architectures, particularly in situations concerning complicated data structures like code-mixed text. The Federated Bi-LSTM model's advanced potential to capture patterns and dependencies in the facts is vital for attaining more accurate and reliable regression predictions. This suggests that the adoption of modern deep learning techniques, blended with federated learning frameworks, can notably enhance the effectiveness of regression models, paving the manner for advanced results in various applications in which unique predictions are crucial.

V. CONCLUSION AND FUTURE WORK

The development of the SentMix-3L dataset and implementation of the Federated Bi-LSTM model are some of the huge strides toward conducting code-mixed text processing and sentiment analysis using multiple languages. With careful methods in collecting data and robust pre-processing methods used, SentMix-3L itself builds a useful tool for any researcher trying to find sentiment evaluation in code-blended text. The Federated Bi-LSTM model with the use of federated learning principles in Bi-LSTM neural networks gives better performance in accurate feature extraction and classification of code-mixed texts keeping the data private and secure. The results of such research and the potential use of machine learning techniques, in general, help learn through federated learning with Bi-LSTM models for applications in the task of solving complex linguistic phenomena and a better understanding of different multilingual contexts. However, utilizing these techniques now effectively across specific programming languages and generalizing robustness across the diversity of educational settings presents its challenges. For instance, integration into specific programming languages with unique syntaxes often presents a challenge to the system, such as in cases of real-time processing, something that is required of JavaScript for web development or Python for data science. At the same time, the effectiveness of the model could depend on the kind of different educational settings characterized by levels of digital infrastructure and language proficiency differences of students.

Work in the future on this model would aim to work in a better and fine-tuned way for some specific languages and to handle the variations in code-mixed structures of the text. On the other hand, enlarging data to add more diversity in examples of language and real-life utilization could enhance the applicability and reliability of the model. Moreover, the error metrics of the proposed model and that of traditional models, for instance in Support Vector Machine, compare very well, further asserting the supremacy of this approach. Hence, lower error values for the Federated Bi-LSTM model are generally indicative of their potential to improve the predictive accuracy and possibly refine the outcome of various applications that heavily rely on very accurate predictions. For instance, this model could be especially useful when analysing interactions on social media sites, with real-time and accurate sentiment analysis being crucial for cases in point, or in educational tools

designed to support bilingual students. The union of innovative data-collection strategies with advanced machine learning techniques and rigorous evaluation methodologies paves the way for new lines of research on the processing of code-mixed text and sentiment analysis. These changes will impact drastically in making language learning more efficient, inclusive, and relevant for educational and real-life contexts across diverse linguistic backgrounds.

REFERENCES

[1] M.-J. Tsai, C.-Y. Wang, and P.-F. Hsu, "Developing the Computer Programming Self-Efficacy Scale for Computer Literacy Education," Journal of Educational Computing Research, vol. 56, no. 8, pp. 1345–1360, Jan. 2019, doi: 10.1177/0735633117746747.

[2] L. Zhao, X. Liu, C. Wang, and Y.-S. Su, "Effect of different mind mapping approaches on primary school students' computational thinking skills during visual programming learning," Computers & Education, vol. 181, p. 104445, May 2022, doi: 10.1016/j.compedu.2022.104445.

[3] C. Kazimoglu, "Enhancing Confidence in Using Computational Thinking Skills via Playing a Serious Game: A Case Study to Increase Motivation in Learning Computer Programming," IEEE Access, vol. 8, pp. 221831–221851, 2020, doi: 10.1109/ACCESS.2020.3043278.

[4] Y. Li et al., "Computational Thinking Is More about Thinking than Computing," Journal for STEM Educ Res, vol. 3, no. 1, pp. 1–18, Apr. 2020, doi: 10.1007/s41979-020-00030-2.

[5] M.-J. Tsai, J.-C. Liang, and C.-Y. Hsu, "The Computational Thinking Scale for Computer Literacy Education," Journal of Educational Computing Research, vol. 59, no. 4, pp. 579–602, Jul. 2021, doi: 10.1177/0735633120972356.

[6] L.-L. Ung, J. Labadin, and F. S. Mohamad, "Computational thinking for teachers: Development of a localised E-learning system," Computers & Education, vol. 177, p. 104379, Feb. 2022, doi: 10.1016/j.compedu.2021.104379.

[7] A. Threekunprapa and P. Yasri, "Unplugged Coding Using Flowblocks for Promoting Computational Thinking and Programming among Secondary School Students," International Journal of Instruction, vol. 13, no. 3, pp. 207–222, Jul. 2020.

[8] G. I. Winata, A. F. Aji, Z.-X. Yong, and T. Solorio, "The Decades Progress on Code-Switching Research in NLP: A Systematic Survey on Trends and Challenges." arXiv, May 24, 2023. doi: 10.48550/arXiv.2212.09660.

[9] H. Sahib, W. Hanafiah, M. Aswad, A. H. Yassi, and F. Mashhadi, "Syntactic Configuration of Code-Switching between Indonesian and English: Another Perspective on Code-Switching Phenomena," Education Research International, vol. 2021, p. e3402485, Dec. 2021, doi: 10.1155/2021/3402485.

[10] M. C. P. Couto, M. G. Romeli, and K. Bellamy, "Code-switching at the interface between language, culture, and cognition," Lapurdum, 2021, Accessed: Feb. 09, 2024. [Online]. Available: https://shs.hal.science/halshs-03280922

[11] A. Jamatia, S. D. Swamy, B. Gambäck, A. Das, and S. Debbarma, "Deep Learning Based Sentiment Analysis in a Code-Mixed English-Hindi and English-Bengali Social Media Corpus," Int. J. Artif. Intell. Tools, vol. 29, no. 05, p. 2050014, Aug. 2020, doi: 10.1142/S0218213020500141.

[12] J. Jamali, M. Rasool, and H. Batool, "Code-Switching By Multilingual Pakistanis On Twitter: A Qualitative Analysis," 2022, Accessed: Feb. 09, 2024. [Online]. Available: http://dspace.khazar.org/handle/20.500.12323/6004

[13] A. S. Abubakar, "Code Switching and Code-Mixing (CS-CM) in Multilingual Teacher-Talk: Pedagogic Functions and Educational Implications," 2022.

[14] H. Rochayati and N. Gailea, "English-Indonesian Code Switching and Code Mixing on Students' Bulletin Board," Journal of English Language Teaching and Cultural Studies, vol. 4, no. 2, Art. no. 2, Dec. 2021, doi: 10.48181/jelts.v4i2.13662.

[15] E. Sippola, "Multilingualism and the structure of code-mixing," in The Routledge Handbook of Pidgin and Creole Languages, Routledge, 2020.

[16] G. I. Ahmad, S. Talwani, and J. Singla, "Adapting Machine Learning And Deep Learning Approach Towards Language Identification And Sentiment Analysis Of Code-Mixed Urdu-English And Hindi-English Social Media Text," Webology (ISSN: 1735-188X), vol. 19, no. 4, 2022.

[17] J. Gasiorek and M. Dragojevic, "Effects of written code-mixing on processing fluency and perceptions of organizational inclusiveness," Communication Monographs, vol. 90, no. 3, pp. 393–413, Jul. 2023, doi: 10.1080/03637751.2023.2202749.

[18] A. Kodirekka and A. Srinagesh, "Preprocessing of Aspect-based English Telugu Code Mixed Sentiment Analysis," Journal of Information Technology Management, vol. 15, no. Special Issue: Digital Twin Enabled Neural Networks Architecture Management for Sustainable Computing, pp. 150–163, Mar. 2023, doi: 10.22059/jitm.2023.91573.

[19] A. K. Madasamy and S. K. Padannayil, "Transfer learning based code-mixed part-of-speech tagging using character level representations for Indian languages," J Ambient Intell Human Comput, vol. 14, no. 6, pp. 7207–7218, Jun. 2023, doi: 10.1007/s12652-021-03573-3.

[20] M. Tareq, Md. F. Islam, S. Deb, S. Rahman, and A. A. Mahmud, "Data-Augmentation for Bangla-English Code-Mixed Sentiment Analysis: Enhancing Cross Linguistic Contextual Understanding," IEEE Access, vol. 11, pp. 51657–51671, 2023, doi: 10.1109/ACCESS.2023.3277787.

[21] R. Srinivasan and C. N. Subalalitha, "Sentimental analysis from imbalanced code-mixed data using machine learning approaches," Distrib Parallel Databases, vol. 41, no. 1, pp. 37–52, Jun. 2023, doi: 10.1007/s10619-021-07331-4.

[22] M. Jain, R. Jindal, and A. Jain, "Code-mixed Hindi-English text correction using fuzzy graph and word embedding," Expert Systems, vol. n/a, no. n/a, p. e13328, 2023, doi: 10.1111/exsy.13328.

[23] K. Shanmugavadivel et al., "An analysis of machine learning models for sentiment analysis of Tamil code-mixed data," Computer Speech & Language, vol. 76, p. 101407, Nov. 2022, doi: 10.1016/j.csl.2022.101407.

[24] K. B. Nelatoori and H. B. Kommanti, "Toxic comment classification and rationale extraction in code-mixed text leveraging co-attentive multi-task learning," Lang Resources & Evaluation, Jan. 2024, doi: 10.1007/s10579-023-09708-6.

[25] M. N. Raihan, D. Goswami, A. Mahmud, A. Anastasopoulos, and M. Zampieri, "SentMix-3L: A Bangla-English-Hindi Code-Mixed Dataset for Sentiment Analysis." arXiv, Nov. 29, 2023. Accessed: Feb. 09, 2024. [Online]. Available: http://arxiv.org/abs/2310.18023

[26] X. Zhou, J. Feng, J. Wang, and J. Pan, "Privacy-preserving household load forecasting based on non-intrusive load monitoring: A federated deep learning approach," PeerJ Computer Science, vol. 8, p. e1049, Aug. 2022, doi: 10.7717/peerj-cs.1049.

[27] Language Technologies Research Centre IIIT Hyderabad, Telangana, India, K. S. S. Varma, P. Sathineni, Language Technologies Research Centre IIIT Hyderabad, Telangana, India, R. Mamidi, and Language Technologies Research Centre IIIT Hyderabad, Telangana, India, "Sentiment Analysis in Code-Mixed Telugu-English Text with Unsupervised Data Normalization," in Proceedings of the Conference Recent Advances in Natural Language Processing - Deep Learning for Natural Language Processing Methods and Applications, INCOMA Ltd. Shoumen, BULGARIA, 2021, pp. 753–760. doi: 10.26615/978-954-452-072-4_086.

[28] U. Sandamini et al., "A Singlish Supported Post Recommendation Approach for Social Media:," in Proceedings of the 14th International Conference on Agents and Artificial Intelligence, Online Streaming, --- Select a Country ---: SCITEPRESS - Science and Technology Publications, 2022, pp. 412–419. doi: 10.5220/0010829700003116.