SENTIMENT ANALYSIS IN CMTET

by

KANDUKURI KEERTHI 421159

DEVIREDDY SNEHALATHA 421133

ANITHA GUNTREDDI 421105

Under the guidance of

Mrs. SAMEERA

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY ANDHRA PRADESH

TADEPALLIGUDEM-534101, INDIA

MAY 2024

leaving second page

SENTIMENT ANALYSIS IN CMTET

Thesis submitted to
National Institute of Technology Andhra Pradesh
for the award of the degree

of

Bachelor of Technology

by

KANDUKURI KEERTHI 421159

DEVIREDDY SNEHALATHA 421133

ANITHA GUNTREDDI 421105

Under the guidance of

Mrs. SAMEERA

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY ANDHRA PRADESH

TADEPALLIGUDEM-534101, INDIA

MAY 2024

DECLARATION

This written submission is an accurate representation of our own thoughts and if there are any other ideas or words that have been borrowed from someone else, we have given them credit through proper citation. We assure you that we have not cheated in any way and all the information in this work is true as well as obtained legally. We acknowledge that failure to comply with these terms may result in severe consequences such as expulsion from college; moreover it could also lead to civil or criminal charges against us by those individuals whose intellectual property rights were violated due either not citing them correctly nor seeking permission where needed.

K KEERTHI	D SNEHALATHA	ANITHA G
421159	421133	421105
Date:	Date:	Date:

CERTIFICATE

It is certified that the work contained in the thesis titled **"Sentiment Analysis in CMTET"** by KANDUKURI KEERTHI, bearing Roll No: 421159 and DEVIREDDY SNEHALATHA, bearing Roll No: 421133, and ANITHA GUNTREDDI, bearing Roll No: 421105, has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

Signature
Mrs SAMEERA
DCSE
N.I.T. Andhra Pradesh
MAY 2024

ABSTRACT

In the present society, everyone communicate in more than one language, leading to Code-Mixed data. Sentiment analysis in Code-Mixed Telugu-English Text (cmtet) leads to unique challenges. The reason why code-mixed data is incomplete because language used is informal transliterations, and spelling mistakes.

This project involves carrying out sentiment analysis on the sentences of the Code-Mixed Text English Text (cmtet) dataset to determine if we can predict the sentiment of a code-mixed text well. We use different models such as Recurrent Neural Networks (Rnn), Bidirectional Long Short-Term Memory networks (Bilstm) and state-of-the-art models such as BERT and XLM-RoBERTa which are pre-trained on a large multilingual corpus and are capable of learning the context and meaning of the input code-mixed sentences during training.

Through our experiments we, and we posted our results of 78 per cent for the RNN model, 79 per cent for the BiLSTM model, 82 per cent for the BERT model and 82 per cent for the RoBERTa model, showing that using the most recent deep learning architectures and pretrained language models is effective for analysing sentiment in code-mixed text. Our results highlight the need to use complex methods to tackle colloquial text mixed from multiple languages, given the specific challenges that code-mixed text has compared with monolingual text. By applying a model that is able to capture sentiments in code-mixed text, we are taking a step forward to better understand natural languages, and provide a more nuanced analysis of posting behaviour in multilingual contexts

TABLE OF CONTENTS

Contents

List of Figures

1 Introduction

Mostly, data on social media and online blogging has seen a remarkable increase as far as volume is concern. The fact that people use more than one language when speaking different languages on daily basis and fewest language regulations online are among the reasons for code-mixed data generation. Using more than one language in one sentience or conversation which is Code-Mixing or Code-Switching creates this type of information. Primary knowledge such as emotions news and general information can be gotten by examining data that has been generated through mixing codes.

When dealing with Code-Mix social media data, sentimental analysis will assist in realizing the true feelings lying under these phrases or sentences which can be applied in many practical situations.Since the same will help me understand the clients opinion regarding different products like restaurants or movies for example.

Easily extract Code-Mixed Data from different online platforms like social media and blogs making use of APIs and various web-scraping tools. Although there is a large volume of data, conducting sentiment analysis on it is hard because it is informal.Therefore, this calls for strong preprocessing techniques that can enable us normalize this unstructured data

Over the course of this project, our primary goal is to be the first to advance the work in classification task applied to code-mixed text, with the objective of training and evaluating existing models on The Code-Mixed Telugu English Text (CMTET) dataset. The specific aims of this project include: To train sentiment analysis models using deep learning architectures like the RNN,BiLSTM and the transformer-based models like BERT and RoBERTa.To pre-process the dataset and conduct feature engineering.For fine-tuning the already trained language models to adapt them to the linguistic characteristics of the code-mixed text.Elicit comprehensive evaluation framework including metrics that will help to evaluate the models across the different sentiment classes and gauge robustness and generalisation.Conduct a comparative analysis of the strength and weakness of the elimination approach of handling code-mixed text and provide insight that will help improve future sentiment analysis work in multilingual and code-mixed text.Hopefully, this work will be the first step to harness sentiment properly using code-mixed and multilingual approaches and lead to the development of multilingual sentiment models that will be applicable to multilingual customer feedback to help businesses make more informed decisions, social media monitoring tools for the governments and social media-related things to help marketers have a deeper understanding of the factors driving their customers' purchase behaviours

2 Literature Review

Sentiment analysis in code-mixed text faces challenges due to combining multiple languages within the same text, prompting varied research approaches to enhance sentiment analysis accuracy.In this research paper the authors implemented mlp,naive bayes,logistic regression , svm.They have mainly data preprocessing steps and implemted traditional ml algorithms.

In our project, we conducted sentimental analysis in code-mixed text from CMTET dataset, leveraging advanced deep learning architectures and pre-trained language models such as BERT and XLM-RoBERTa. Our approach builds upon previous research by exploring models and techniques to accurately capturing sentiment in code-mixed text. We reported significant enhancements in sentiment analysis accuracy, achieving promising results across various models, including RNN, BiLSTM, BERT, and XLM-RoBERTa. This literature review offers insights into previous research efforts in sentiment analysis in code-mixed corpus and highlights the contributions and enhancements made in our project. By incorporating advanced techniques and leveraging models, we aim to address the unique challenges of sentiment analysis in code-mixed text and facilitate deeper understanding of user sentiments in multilingual environments.

3 Challenges in CMTET

The challenges posed by Code-Mixed Telugu-English Text (CMTET) are significant due to its unstructured nature, primarily driven by informal transliterations, informal language usage, and spelling and typing errors. Here's a summary of the challenges identified in the literature:

3.1 Informal Transliterations

- Variants in Long Vowels: Users transliterate long vowels in various ways, such as repeating the long vowel twice, indicating it once, or using a mixture of double and single vowels. - Variants in Double Consonants: same as long vowels, double consonants are represented in english inconsistently, with variations spelling observed. - Variants in Aspirated Consonants: Aspirated consonants, which require a burst of breath to pronounce, are transliterated differently, leading to multiple representations for the same syllable. - Variants in Homophones: Homophonic syllables are transliterated differently due to their identical pronunciation with different spellings, further complicating the text normalization process.

3.2 Informal Language

- Elongation: Users elongate certain words to express sentiments like excitement in an informal setting, leading to variations in spelling. - Shortening: Due to character limitations on platforms like Twitter, users tend to shorten words while retaining their phonetics, often by removing vowels or replacing single character in place of double consonants.

3.3 Spelling Mistakes

- Users frequently make spelling and typing errors in both English and Telugu text, leading to ungrammatical and error-filled content, as observed in Twitter conversations by Ritter et al. (2010).

These challenges result in high entropy in spellings within CMTET, making it difficult to process and analyze the text accurately. To address these challenges, the proposed solution involves an unsupervised data normalization technique tailored specifically for CMTET. This normalization technique aims to standardize the text and reduce variability caused by transliterations, informal language usage, and spelling errors, thereby improving the effectiveness of sentiment analysis and other natural language processing tasks on CMTET.

4 Data Set

The overview of dataset for this project consists of three main phases:

4.1 Data Collection

- The dataset includes data collected from Twitter users and regional movie review YouTube videos.
- users of Twitter are known to tweet about different topics such as movies and sports, while YouTube comments express emotions about movies or videos in mixed languages.
- Data collection was compiled from API of twitter and youtube .

4.2 Data Cleaning

- Unrelated content such as URLs and markup text were eliminated so as to ensure only the basic quality of the data is left behind using regex based pattern matching..
- Each sentence is divided to into sub words using nltk tokenizer.
- The sentences which are having length less than five, which were considered noisy and lacking information, were removed.

4.3 Data Annotation

- The dataset adopts a three-class classification for sentiment analysis: positive, negative, and neutral.
- Each sentence was annotated for sentiment and word-level language tags (English, Telugu, Named Entities, and Universal).
- Sentences containing only English or only Telugu words were removed.
- Annotation was performed by five Telugu native speakers proficient in English, using a Telegram Bot API for efficient annotation.

4.4 Dataset Overview

The dataset comprises a total of 34,588 sentences. - Sentiment distribution: positive sentences are 11061, negatives are 12583, and neutral sentences are 10944. - The dataset is available to facilitate further research and development in sentiment analysis for CMTET.

5 Data Normalisation

A method for normalizing unsupervised data normalization that could be applied to CMTET. Our first step involved elongating words if necessary then separating out Telugu and English languages using tags from the database so that they could each be handled separately. After this was done we ran an English spell check on all of the words using a set similarity measure against an English dictionary wordlist. With regard specifically towards Telugu there were two stages required in order for them to become consistently transformed from one representation system into another by means of correcting any mistakes made either during transliterating between different alphabets and/or actual misspellings themselves being present within what should have been otherwise correct text passages; more detailed explanations follow below

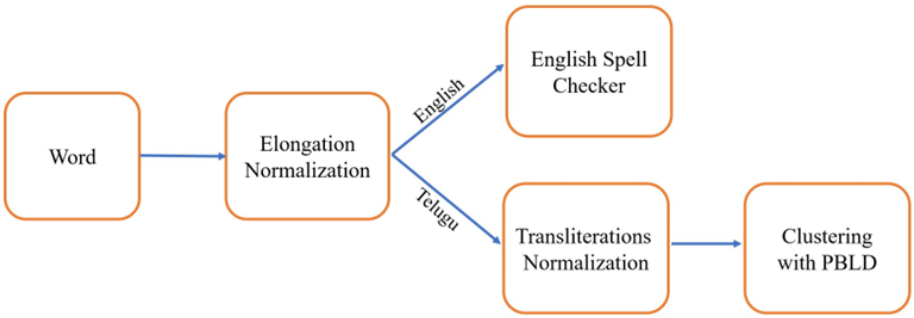


Figure 1: Data normalization

5.1 Elongation Normalization

To resolve the elongation problem, we change all characters to lowercase and limit the number of consecutive characters to two. For example, we convert hiiiiiiiiiii to hii, dooor to door, nachind-hiiii to nachindhii. If any errors such as hii or nachindhi still exist after these steps then they are considered spelling mistakes which will be corrected in subsequent normalization procedures.

5.2 Normalizing English Words

To fix errors in spelling and typing, we used a spell-check driven by dictionary with minimum edit distance as the comparision score between two words. SymSpell library is used to do this in a fast way.

5.3 Normalizing Telugu Words

our goal is to form clusters of words that share the same meaning, capturing transliteration variations and misspellings for each word in telugu senences . We have proposed a two step normalization process for achieving this.

6 Models

6.1 RNN

RNN is a one of the neural network that deals with the sequential data. RNNs, comprised of feedforward networks, works close to human brain.simply we can say that this networks predicts the sequential that traditional networks cannot do. All inputs and outputs of a standard neural network are independent of each other, but in some situations, such as when predicting the next word in a phrase, the previous word is needed and the previous word must be remembered. As a result, RNNs using hidden layers were created to solve this problem.A key element of an RNN is the hidden state, which retains essential details about the sequence.

RNNs have memory to store all computational information. We do the same for all inputs or hidden layers, producing the same output, so we use the same settings for each input.

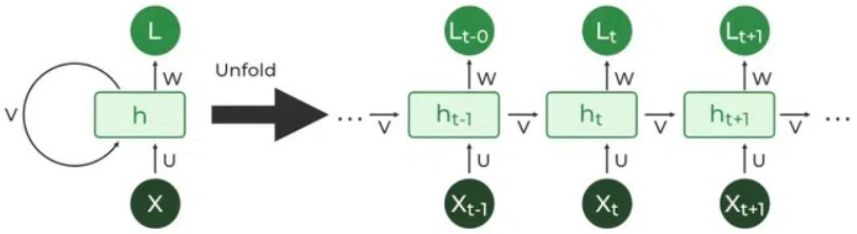


Figure 2: Stacked RNN

where,

ht - current state , ht-1 - previous state ’ xt - input state , W,U,V - parameters

Optionally, bidirectional RNN layers can be incorporated, allowing the model to consider both forward and backward context information. An attention mechanism may also be added to dynamically weigh the relevance of every token in the sentence. Subsequently, output of RNN layers is passed through fully connected layers for non-linear transformations and dimensionality reduction. Finally, a softmax output layer computes probabilities for each sentiment class. Throughout training, model parameters are updated via backpropagation and gradient descent methods such as Adam or RMSProp, while validation and fine-tuning on separate datasets help optimize performance and prevent overfitting. This architecture offers increased representational power, hierarchical feature extraction, and regularization benefits, making it a robust choice for sentiment analysis tasks.

6.2 BiLSTM

In the sentiment analysis in Code-Mixed Telugu-English Text (CMTET), Bidirectional Long Short-Term Memory networks (BiLSTMs) are essential in interpreting the complex sentiment expressions found in the code-mixed data. BiLSTMs utilize their bidirectional architecture to capture the contextual dependencies within the CMTET sequences. Through bidirectional processing, BiLSTMs excel at recognizing sentiment indicators that span multiple languages and dialects in the CMTET dataset. This bidirectional approach allows them to understand the subtleties of sentiment expression, regardless of language directionality or the intricacy of code-mixed sequences. Additionally, fine-tuning pretrained BiLSTM models on CMTET data enables these models to adapt to the specific linguistic features and sentiment patterns present in the code-mixed text. Consequently, the incorporation of BiLSTMs in CMTET sentiment analysis enhances the accuracy and depth of understanding of sentiment dynamics in multilingual and code-mixed environments, leading to improved outcomes in real-world sentiment analysis applications.

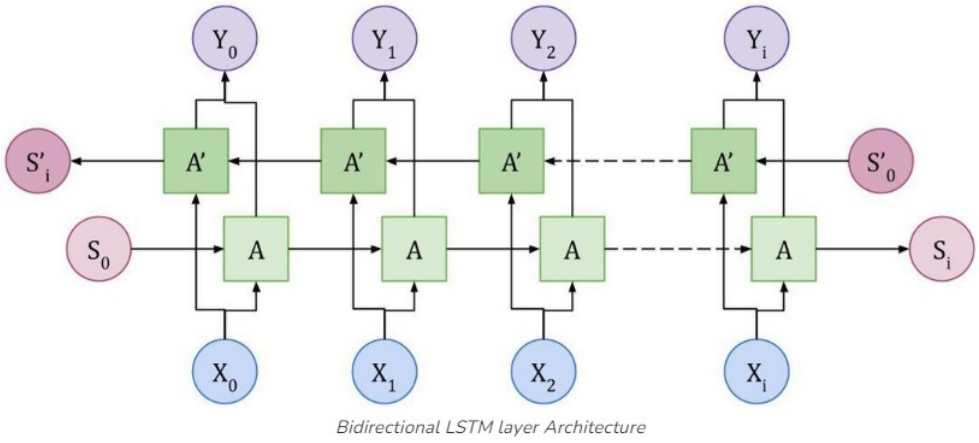


Figure 3: The BiLSTM used for Text Classification

where
X_i is the input token, Y_i is the output token, and A and A' are LSTM nodes.
The final output of Y_i is the combination of A and A' LSTM nodes.

6.3 Pretrained Models

BERT, RoBERTa, and XLM-RoBERTa are pretrained on large corpora of text data using self-supervised learning objectives, such as masked language modeling and next sentence prediction. This pretraining enables them to learn rich contextual representations of words and sentences, which can used for downstream tasks like information retrieval that process is called fine tuning.

6.3.1 BERT

(104 languages, 12-layer, 768-hidden, 12-heads, 110M parameters)
BERT is a transformer model pre-trained on large multilingual data corpora in a self-supervised environment fashion. This means that it is only pre-trained on raw text and has not been tagged by humans. in some way (which is why so much publicly available data is available), through automated processes. Create inputs and labels from these texts. More precisely, he was previously trained for two purposes. Masked Language Modeling (MLM): When you extract a sentence, the model randomly masks 15 percent of it. The input word must be run through the model to run the entire masked sentence and make predictions. Disguised words. This is different from the more commonly used traditional recurrent neural networks (RNNs). Either look at the words one by one, or internally use an autoregressive model like GPT. Disguise future tokens. This allows the model to learn bidirectional representations of sentences. Next Sentence Prediction (NSP): The model combines two masked sentences as input. Preliminary preparation. Sometimes they correspond to sentences that were originally side by side. Text, sometimes not. The model must then predict whether the two sentences follow each other or not.

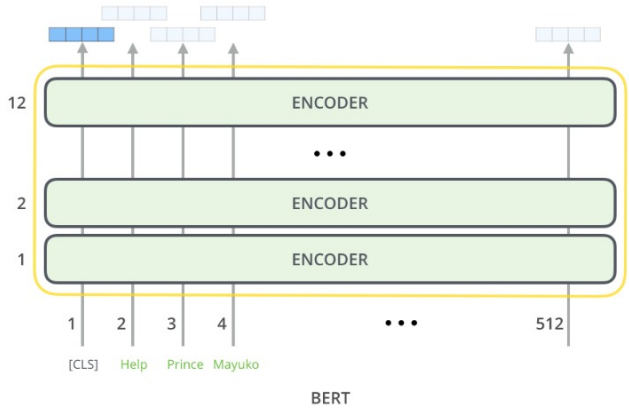


Figure 4: Sentiment Classification using BERT

Thus, the model learns an internal representation of the language in the training set. It can then be used to extract useful features for downstream tasks. If you have a labeled dataset You can train a standard classifier using the features provided by the proposal. BERT model as input.

6.3.2 XLM-RoBERTa

(Robustly optimized BERT approach): RoBERTa's refinements over BERT make it particularly well-suited for sentiment analysis in CMTET. With optimizations in training objectives, hyperparameters, and data preprocessing, RoBERTa enhances the model's ability to capture subtle linguistic nuances and variations present in Code-Mixed Text. By removing the next sentence prediction task during pretraining and leveraging larger datasets, RoBERTa achieves improved robustness and performance across various NLP tasks, including sentiment analysis. Fine-tuning RoBERTa on your CMTET sentiment analysis dataset can lead to enhanced accuracy and generalization, ensuring reliable sentiment classification even in the presence of informal language variations and transliterations.

XLM-RoBERTa is a multilingual demonstrate prepared on 100 diverse dialects. Not at all like a few XLM multilingual models, it does not require lang tensors to get it which dialect is utilized, and ought to be able to decide the rectify dialect from the input ids. Employments RoBERTa traps on the XLM approach, but does not utilize the interpretation dialect modeling objective. It as it were employments veiled dialect modeling on sentences coming from one language.XLM-RoBERTa is an expansion of XLM that utilizes the RoBERTa engineering, a variation of the Transformer demonstrate, for pre-training. It is based on the RoBERTa engineering, an optimized variation of BERT (Bidirectional Encoder Representations from Transformers). XLM-RoBERTa builds upon the qualities of XLM whereas consolidating progressions from RoBERTa to accomplish indeed way better execution in multilingual dialect understanding and downstream NLP tasks. The key thought behind XLM-RoBERTa is to use the pre-training technique of RoBERTa, which includes large-scale pre-training and broad hyperparameter tuning, to make a more effective and successful cross-lingual dialect show.

7 Methodology

7.1 RNN Implementation

The embedding layer transforms input tokens (words) into dense vectors of a fixed size (embeddingdim). The vocabsize denotes the size of the vocabulary, while the maxlen specifies the maximum length of input sequences.
In terms of recurrent layers, the model architecture incorporates two SimpleRNN layers.
The first SimpleRNN layer, consisting of 512 units, is set to return sequences to capture temporal information within the input sequence.
The second SimpleRNN layer, comprising 256 units, aggregates the sequence outputs to capture higher-level representations.
Dropout Layer: To regularize the model and address overfitting, this layer with a dropout rate of 0.2 is included, randomly setting a fraction of input units to zero during training.
Dense Layer: The final dense layer, with softmax activation, produces output predictions for sentiment classes (positive, negative, neutral) based on the learned representations.

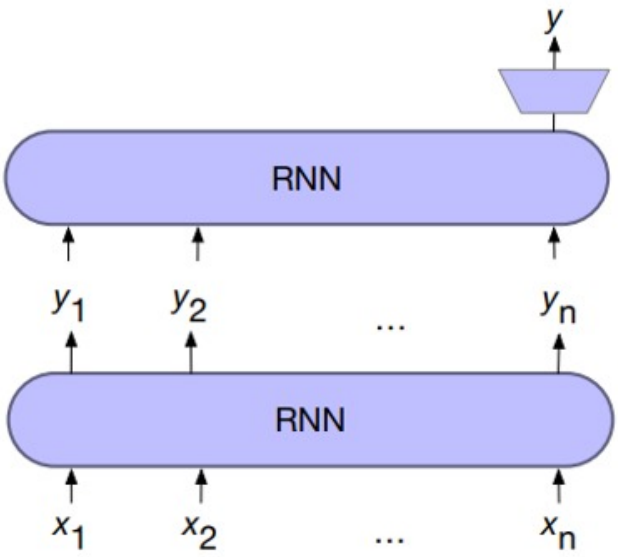


Figure 5: RNN implementation

Training Process:
Moving on to the training process, data preprocessing involves tokenization and padding to ensure consistent input sequence lengths.
Using the fit() function, the model trains the training dataset which was compiled with categorical cross-entropy loss function, the Adam optimizer with a learning rate of 0.0001, and accuracy as

the evaluation metric; subsequently, model is evaluated, with batch sizes and epochs determined through experimentation. Validation is conducted on a separate dataset to monitor model performance, preventing overfitting and ensuring generalization.

7.2 BiLSTM Implementation

The Bidirectional Long Short-Term Memory (BiLSTM) model is employed for sentiment analysis in code-mixed text. The architecture of the model comprises the given layers:

Embedding Layer:

The embedding layer converts input tokens (words) into dense vectors of a fixed size (embeddingdim). The input dimension (vocabsize) represents the size of the vocabulary, and the input length (maxlen) specifies the maximum length of input sequences.LSTM Layers:

Two Bidirectional LSTM layers are utilized in the model architecture.first Bidirectional LSTM layer with 512 units returns sequences to capture temporal information across the input sequence in both forward and backward directions.second Bidirectional LSTM layer with 256 units aggregates the bidirectional sequence outputs to capture higher-level representations.

Dropout Layer: A dropout layer with a dropout rate of 0.3 is inserted to regularize the model and mitigate overfitting by randomly setting a fraction of input units to zero during training.

Dense Layer: The final dense layer with softmax activation generates the output predictions for sentiment classes (positive, negative, neutral) based on the learned representations.

Model Parameters:

Vocabulary Size (vocabsize): 5000

Embedding Dimension (embeddingdim): 300

Maximum Input Length (maxlen): 25

Training Process:

Data Preprocessing: The training data is preprocessed, including tokenization and padding to ensure uniform input sequence lengths.

Model Compilation: Using the fit() function, the model trains the training dataset which was compiled with categorical cross-entropy loss function, the Adam optimizer with a learning rate of 0.0001, and accuracy as the evaluation metric; subsequently, 23 models are compiled, with batch sizes and epochs determined through experimentation.

Validation:Model performance is monitored on a validation dataset to prevent overfitting and ensure generalization.

7.3 BERT Implementation

Implementation of BERT-based Multilingual Model **Model Initialization**

The implementation utilizes the BERT (Bidirectional Encoder Representations from Transformers) architecture for sentiment analysis in code-mixed text. The BERT-based multilingual model

is initialized with the following configurations: **Model: bert**
Pre-trained Checkpoint: 'bert-base-multilingual-cased'
Number of Labels: 3 (to accommodate the three sentiment classes: positive, negative, neutral)
Training Arguments
The training process is configured with the following parameters:
Reprocess Input Data: True (to reprocess input data for BERT compatibility)
Overwrite Output Directory: True (to overwrite the output directory if it already exists)
Number of Training Epochs: 3 (determines the number of times the model sees the entire training dataset)
Class Weights: [0.85714286, 1.57070857, 0.83564669] (class weights for handling class imbalance)
Training Process
Tokenization and Input Encoding:
- The input text data is tokenized using the BERT tokenizer, which breaks down the text into subword tokens.
- Tokenized sequences are encoded into numerical representations suitable for model input, incorporating special tokens such as [CLS] (classification) and [SEP] (separator).
Initialization with Pre-trained Weights:
- The BERT-based multilingual model is initialized with pre-trained weights from the 'bert-base-multilingual-cased' checkpoint.
- These pre-trained weights encode knowledge about language structures, semantics, and sentiment cues learned from vast amounts of text data across multiple languages.
Fine-tuning and Gradient Descent:
- During training, the model's parameters are fine-tuned using backpropagation and gradient descent.
- The categorical cross-entropy loss function measures the discrepancy between predicted probabilities and ground truth labels.
- The Adam optimizer adjusts model weights iteratively to minimize the loss and improve prediction accuracy.
Learning Multilingual Contextual Representations:
- BERT utilizes a multi-layer bidirectional transformer architecture to encode contextual representations of input sequences.
- Through self-attention mechanisms and transformer blocks, BERT captures contextual information from both left and right contexts, enabling it to understand the semantics and sentiment of code-mixed text across multiple languages.
Updates to Model Parameters:
- Model parameters, including attention weights and feed-forward network weights, are updated based on gradients computed from the loss function.
- The learning rate controls the size of parameter updates, influencing training convergence speed and stability.
Epochs and Batch Training:

- The training process consists of multiple epochs, with each epoch iterating over the entire training dataset.
- The dataset is divided into batches, and model parameters are updated after processing each batch.
Handling Class Imbalance:
- Class weights are applied during training to address class imbalance, ensuring that the model effectively learns from minority classes and produces balanced predictions.
Through this iterative process, the BERT-based multilingual model learns to understand the contextual features of code-mixed sentences and predicts sentiment labels accurately across multiple languages.

7.4 XLM-RoBERTa Implementation

Importing Necessary Libraries: The transformers library is imported to access the RoBERTa tokenizer and model. The simpletransformers library is imported to instantiate and train the ClassificationModel. Model Initialization: The XLMRobertaTokenizer is used to tokenize the input text and prepare it for model input. The ClassificationModel from simpletransformers is instantiated with the 'xlmroberta' architecture and the 'xlm-roberta-base' pre-trained checkpoint. The number of labels is specified as 3 to accommodate the three sentiment classes (positive, negative, neutral). **Training Arguments:** Training arguments are defined, including the number of training epochs, learning rate, and other parameters. The 'overwriteoutputdir' parameter is set to True to overwrite the output directory if it already exists. During the training process, the ClassificationModel utilizing the RoBERTa architecture acquires the ability to predict sentiment labels by utilizing the comprehensive contextual representations learned from the pre-trained XLM-RoBERTa language model. Here is a detailed explanation of what occurs during training:

Tokenization and Input Encoding

- The input text is tokenized into subword tokens by the XLM-RoBERTa tokenizer and then encoded into numerical representations suitable for model input.
- To ensure consistent input size across samples, the tokenized sequences are either padded or truncated to a fixed length.

Initialization with Pre-trained Weights

- The RoBERTa model is initialized by the ClassificationModel using the pre-trained weights obtained from the 'xlm-roberta-base' checkpoint.
- These pre-trained weights encapsulate knowledge about the structure, syntax, semantics, and sentiment cues of the language, acquired from extensive amounts of text data.

Fine-tuning and Gradient Descent

- During training, the model's parameters are fine-tuned through the utilization of backpropagation and gradient descent.
- The loss function, which is categorical cross-entropy, measures the disparity between the predicted probabilities and the actual ground truth labels.
- The optimizer, specifically Adam, iteratively adjusts the model's weights to minimize the loss

and enhance the accuracy of predictions.

Learning Contextual Representations

- As the model processes each token within the input sequence, it generates contextual representations that capture the semantic meaning and sentiment information of the entire sentence.
- By employing multiple layers of self-attention mechanisms and transformer blocks, RoBERTa dynamically focuses on different segments of the input sequence to extract pertinent features for sentiment analysis.

Model Parameter Updates

- While training, the model's parameters, such as attention weights and feed-forward network weights, get adjusted according to the gradients derived from the loss function.
- The learning rate plays a crucial role in determining the magnitude of parameter updates, impacting the speed of convergence and training stability.

Training with Epochs and Batches

- The training procedure involves multiple epochs, with each epoch going through the entire training dataset.
- Typically, the dataset is segmented into batches, and model parameters are modified after processing each batch.
- The number of training epochs (numtrainepochs) specifies how many times the model goes through the complete training dataset.

Dropout and Regularization

- Dropout regularization is implemented during training to prevent overfitting by randomly deactivating a portion of neuron activations.
- The dropout rate (0.3 in this instance) determines the percentage of neurons to be deactivated at each training step.

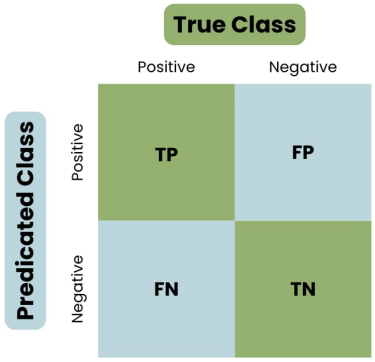
Hyperparameters

Number of Training Epochs: 3

Learning Rate: 1e-5

8 Evaluation and Metrics

8.1 Metrics used



tp = true positive , fp = false positive , tn = true negative , fn = false negative

Accuracy

Accuracy represents the number of correctly classified data instances over the total number of data instances.

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

precision

Precision is the fraction of instances that the model classifies as positive that are indeed correct.

$$Precision = \frac{TP}{TP + FP}$$

Recall

Recall is a metric that measures how often a machine learning model correctly identifies positive instances (true positives) from all the actual positive samples in the dataset.

$$Recall = \frac{TP}{TP + FN}$$

f1-score

F1-score is a metric which takes into account both precision and recall and is defined as follows:

$$F1\ Score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

8.2 RNN results

Precision	recall	f1-score
0.77 2	0.79	0.78
0.78	0.74	0.76
0.78	0.80	0.89
ACCURACY = 0.78		

Table 1: RNN results

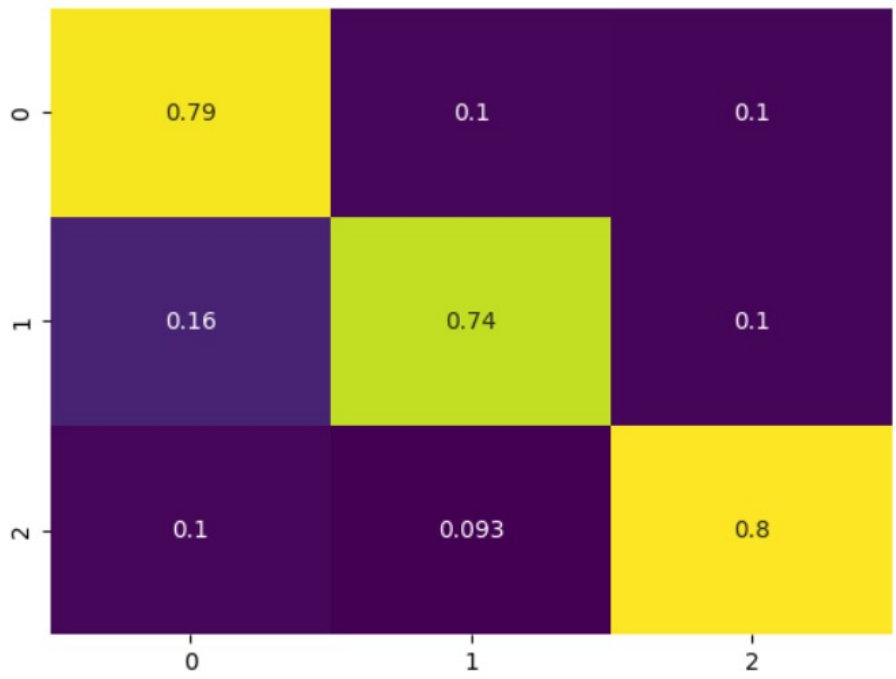


Figure 6: confusion matrix of RNN

8.3 BiLSTM results

Precision	recall	f1-score
0.78 2	0.84	0.81
0.79	0.76	0.77
0.84	0.80	0.82
ACCURACY = 0.80		

Table 2: BiLSTM results

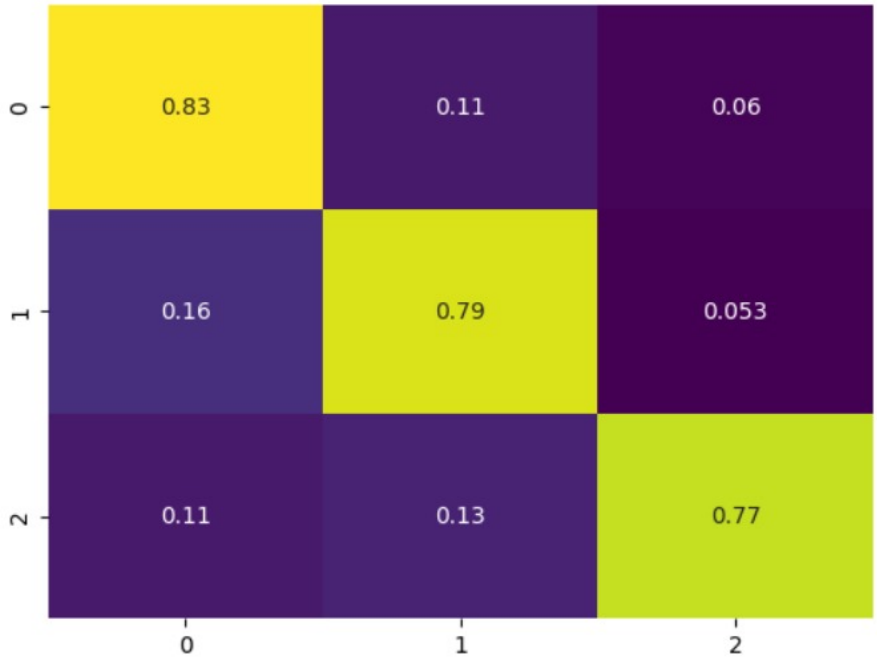


Figure 7: confusion matrix of BiLSTM

8.4 BERT results

	Precision	Recall	f1-score	support
NEG	0.81	0.85	0.83	2537
NTL	0.82	0.85	0.83	2195
POS	0.85	0.85	0.83	2186
ACCURACY			0.83	6918
macro avg	0.83	0.82	0.83	6918
weighted avg	0.83	0.83	0.83	6918

Table 3: BERT results

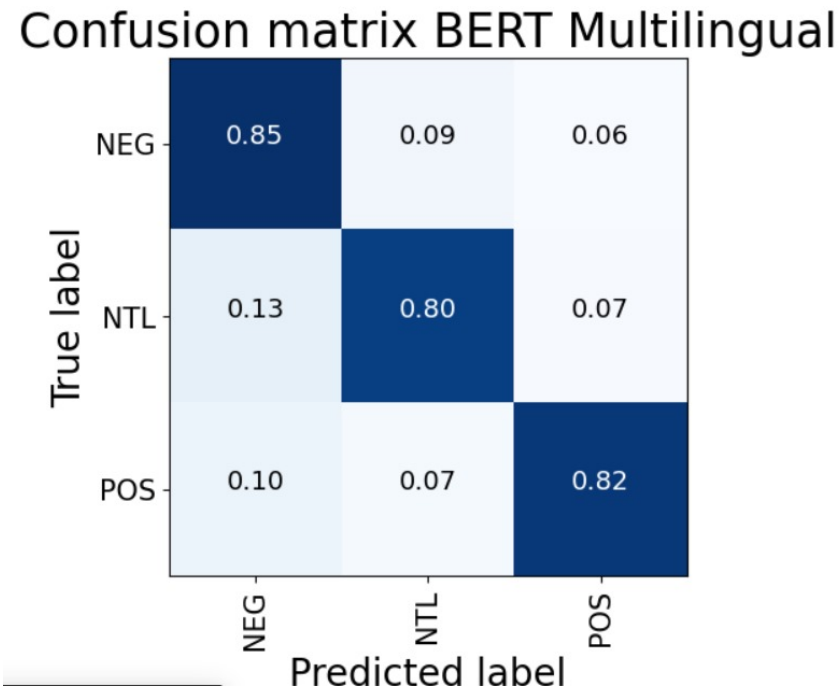


Figure 8: Confusion Matrix of BERT

8.5 XLM RoBERTa results

	Precision	Recall	f1-score	support
NEG	0.81	0.83	0.82	2537
NTL	0.83	0.85	0.83	2195
POS	0.81	0.85	0.83	2186
ACCURACY			0.82	6918
macro avg	0.82	0.82	0.82	6918
weighted avg	0.82	0.82	0.82	6918

Table 4: XLM RoBERTa results

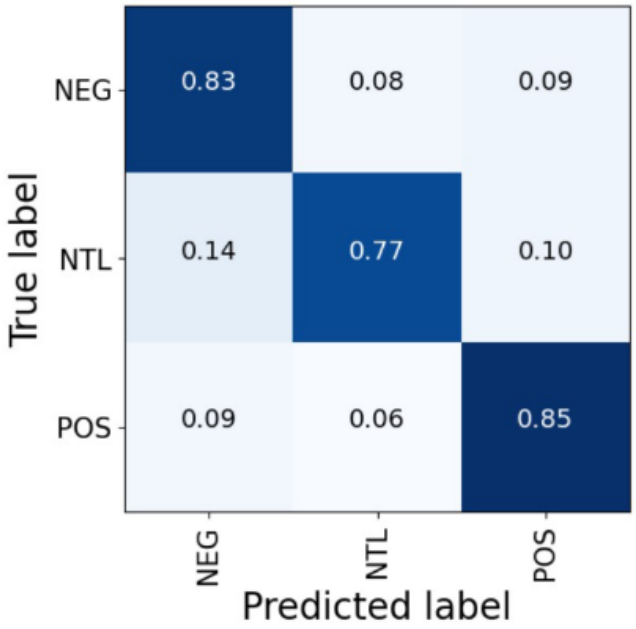


Figure 9: Confusion matrix of XLM RoBERTa

Table 5: Model Accuracy

Model	Accuracy (%)
RNN	78
BILSTM	80
BERT	83
XLM-RoBERTa	82

9 Future Scope and Conclusion

This research marks a significant milestone in the field of sentiment analysis, with a specific focus on the complex nuances of Code-Mixed Telugu-English Text (CMTET). The main aim of this study was to create and assess new methods for sentiment analysis in CMTET, a domain characterized by the fusion of Telugu and English languages commonly found in informal digital interactions. The study recognizes the difficulties presented by CMTET, such as language variations, transliterations, and informal expressions, which make traditional sentiment analysis techniques less efficient.

To tackle these challenges, the research introduces innovative unsupervised data normalization techniques designed specifically for the unique features of CMTET. By utilizing linguistic knowledge and computational methods, the normalization process aimed to standardize the data while maintaining its original meaning and sentiment. This fresh approach not only improves the dataset's quality but also establishes a basis for more precise sentiment analysis.

At the core of this study is the development of a carefully annotated dataset of CMTET, consisting of a diverse array of text samples gathered from social media platforms, forums, and messaging applications. The annotation process involved categorizing each text snippet based on its sentiment polarity, covering positive, negative, and neutral sentiments. This annotated dataset acts as a valuable asset for training and assessing sentiment analysis models tailored specifically for CMTET.

The research methodically assesses the performance of various machine learning models on the annotated dataset, including traditional classifiers like Naive Bayes, Logistic Regression, Support Vector Machines, Random Forest, and contemporary deep learning architectures such as Multi-Layer Perceptron. Through extensive experimentation and analysis, the study identifies the most effective models for sentiment analysis in CMTET, offering insights into their strengths and limitations.

References

[1] https://www.geeksforgeeks.org/bidirectional-lstm-in-nlp/

[2] https://www.geeksforgeeks.org/bidirectional-lstm-in-nlp/

[3] . https://github.com/rayalaupendar/Code-Mixed-Tel-Eng-34k

[4] https://jalammar.github.io/illustrated-bert/

cmtett

ORIGINALITY REPORT

7%	4%	8%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Marco Siino, Elisa Di Nuovo, Ilenia Tinnirello, Marco La Cascia. "Fake News Spreaders Detection: Sometimes Attention Is Not All You Need", Information, 2022 Publication	1%
2	"ECAI 2020", IOS Press, 2020 Publication	1%
3	Rajesh Kumar Das, Mirajul Islam, Md Mahmudul Hasan, Sultana Razia, Mocksidul Hassan, Sharun Akter Khushbu. "Sentiment analysis in multilingual context: Comparative analysis of machine learning and hybrid deep learning models", Heliyon, 2023 Publication	1%
4	"Innovations in Electrical and Electronic Engineering", Springer Science and Business Media LLC, 2024 Publication	1%
5	Manjubala Bisi, Rahul Maurya. "Ensemble learning and stacked convolutional neural network for Covid-19 situational information	1%

analysis using social media data", Multimedia Tools and Applications, 2024
Publication

6	dokumen.pub Internet Source	1 %
7	www.coursehero.com Internet Source	1 %
8	ijaas.iaescore.com Internet Source	1 %
9	medium.com Internet Source	1 %
10	www.nitmz.ac.in Internet Source	1 %