Sentiment Analysis of Roman Urdu Using LSTM, BERT, and GPT-2

Aquib Ansari
DSSE
Natural Language Processing

***Abstract*—In this project, I tackled the unique challenge of sentiment analysis for Roman Urdu, a code-mixed language blending Urdu and English with significant spelling variations and informal vocabulary. The objective was to classify sentiments into four categories: Negative, Positive, Neutral, and Abusive. Due to the underrepresentation of the Abusive class, I implemented synonym-based data augmentation to balance the dataset. Subsequently, I fine-tuned three models—LSTM, BERT, and GPT-2—and compared their performance using metrics like accuracy, precision, recall, and F1-score. GPT-2 demonstrated superior results, achieving 70.24% test accuracy, underscoring its ability to understand complex linguistic patterns. This work contributes to addressing linguistic challenges in analyzing low-resource, code-mixed languages, providing insights for future research and applications.**

***Index Terms*—Sentiment Analysis, Roman Urdu, LSTM, BERT, GPT-2, Data Augmentation, Natural Language Processing.**

I. Introduction

Sentiment analysis has become a pivotal application of natural language processing (NLP), offering insights into public opinions, user behavior, and automated content moderation. However, analyzing Roman Urdu presents distinct challenges due to its informal nature and lack of standardization. The language combines Urdu and English, often written phonetically, resulting in diverse spellings and inconsistent syntax.

Roman Urdu's complexity makes traditional sentiment analysis models ineffective, as they are designed for structured and standardized languages like English. This project focuses on addressing these challenges, emphasizing the underrepresented Abusive class, which is vital for online content moderation. My objective was to create a robust sentiment classification pipeline, leveraging state-of-the-art models and novel augmentation techniques to overcome linguistic barriers and dataset limitations.

II. Related Work

Sentiment analysis for high-resource languages like English has benefited from advancements in deep learning models, particularly transformers such as BERT and GPT-2. These models, pre-trained on extensive corpora, excel in understanding context and semantic relationships. However, low-resource languages like Roman Urdu remain underexplored due to a lack of annotated datasets and linguistic complexities.

Existing studies on Roman Urdu sentiment analysis often rely on traditional machine learning models, such as Support Vector Machines (SVMs) and Naive Bayes, which fail to capture the intricacies of code-mixed languages. While multilingual transformers like mBERT and XLM-R show promise, they require fine-tuning and augmentation strategies to adapt to Roman Urdu's unique characteristics. This project bridges the gap by combining LSTM, BERT, and GPT-2 models with tailored preprocessing and augmentation techniques.

III. Methodology

A. Dataset Preparation

Two datasets were combined: one for sentiment analysis and another for hate speech detection. Labels were unified into four categories:

- Negative (0)
- Positive (1)
- Neutral (2)
- Abusive (3)

The Abusive class was significantly underrepresented, containing only 3,358 samples compared to over 20,000 in other classes. To address this imbalance, I employed synonym replacement-based augmentation, expanding the Abusive class to 13,374 samples.

Data exploration revealed overlapping characteristics between the Negative and Abusive classes, complicating classification. I refined the augmentation process to minimize such overlaps, ensuring clearer distinctions between these categories. This step was crucial in enhancing the model's ability to generalize across complex sentiments.

B. Preprocessing

Preprocessing involved:

- **Tokenization and Padding:** For LSTM, I used Keras' Tokenizer to convert text into sequences and padded them to a fixed length of 128 tokens. For BERT and GPT-2, I employed their pre-trained tokenizers (WordPiece and BPE, respectively), adding attention masks to manage padding effectively.
- **Label Encoding:** Labels were converted into one-hot vectors for LSTM and integer mappings for transformer models.

These preprocessing steps ensured uniformity across datasets, reducing noise and improving model performance.

C. Model Architectures

LSTM:

- Embedding Layer: Transformed tokens into dense vectors of size 128.
- LSTM Layers: Sequentially captured long-term dependencies.
- Dropout Layers: Mitigated overfitting by randomly deactivating neurons during training.
- Output Layer: Predicted one of four classes using a softmax activation function.

BERT:

- Fine-tuned the pre-trained `bert-base-uncased` model, adding a classification head.
- Optimized using AdamW with a learning rate scheduler for stability.

GPT-2:

- Fine-tuned `gpt2` with modifications for classification tasks.
- Defined a padding token (`<|endoftext|>`) and adjusted the architecture to support four-class outputs.

D. Training Details

Each model was trained on the augmented dataset:

- **Batch Sizes:** 32 for LSTM, 8 for BERT and GPT-2 due to GPU constraints.
- **Epochs:** 5 for LSTM, 3 for BERT and GPT-2.
- **Loss Function:** Categorical Cross-Entropy across all models.

Regular monitoring of metrics like training loss, validation loss, and accuracy guided hyperparameter adjustments, ensuring optimal performance.

IV. Experiments and Results

A. Training Results

Training performance revealed:

- **LSTM:** Achieved 80% training accuracy, with validation accuracy plateauing at 68%.
- **BERT:** Demonstrated balanced training (76% accuracy) and validation (73%) performance.
- **GPT-2:** Achieved the most consistent results, with 74% training accuracy and 70.24% test accuracy.

B. Testing Results

The test dataset evaluation showed:

TABLE I
Model Testing Performance

Model	Accuracy	Precision	Recall	F1-Score
LSTM	57.43%	0.54	0.57	0.54
BERT	65.05%	0.62	0.62	0.62
GPT-2	**70.24%**	**0.64**	**0.65**	**0.65**

V. Discussion

A. Challenges and Resolutions

- **Class Imbalance:** Addressed through synonym-based augmentation, increasing Abusive class samples by 300%.
- **Overlapping Labels:** Refined augmentations and tokenization strategies reduced misclassifications between Negative and Abusive sentiments.
- **Computational Constraints:** Leveraged GPUs for training while optimizing batch sizes and epochs to balance efficiency and performance.

VI. Conclusion and Future Work

This project demonstrated the effectiveness of modern NLP models in handling Roman Urdu sentiment analysis. GPT-2 emerged as the top-performing model, highlighting its ability to understand complex linguistic patterns in code-mixed text. Future work will explore:

- Multilingual models like XLM-R for enhanced performance.
- Expanding datasets to include diverse linguistic variations.
- Advanced augmentation techniques like back-translation and adversarial training.

References

[1] J. Devlin et al., "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in NAACL-HLT 2019.
[2] Hugging Face Transformers Documentation.
[3] TensorFlow and PyTorch Documentation.