DS605-Deep Learning for Low Resource NLP

Sentitel: Telugu Emotion Detection using BERT

-Gorrepati Hasitha(12140710)

-Kummari Vasanthi(12140950)

Problem Statement

This project focuses on building an Emotion Detection System for Telugu Text using a fine-tuned multilingual BERT model. It predicts the intensity (in percentages) of four core emotions: Happy, Sad, Anger, and Fear from a given paragraph or sentence written in Telugu.

Dataset

The dataset for this project was prepared manually due to the scarcity of labeled Telugu emotion datasets. The preparation involved the following steps.

1) **Data Collection:**Telugu paragraphs were collected from multiple online resources, including:Telugu blogs,Short stories, Folk tales, News articles, Quotes and motivational content.

Care was taken to ensure the dataset covers a diverse range of emotional expressions in Telugu.

2) **Including CoT:**The dataset used consisted of Telugu paragraphs along with CoT (Chain-of-Thought) reasoning texts, with corresponding emotion intensity scores for each of the four emotions. The scores originally ranged from 0 to 10 and were normalized by dividing each score by 10 to bring them into the range [0,1].

Preprocessing

- Missing values in the Paragraph and CoT fields were replaced with empty strings.
- The text was tokenized using the BERT Multilingual Cased Tokenizer.
- Each input text was prepared by concatenating the Paragraph and CoT, separated by a special token (`[SEP]`).
- Tokenized inputs were padded and truncated to a maximum length of 256 tokens.

Model Architecture

- The model is built on top of the BERT Multilingual Cased model.

- After extracting the pooled output (`[CLS]` token representation) from BERT, it is passed through:
    - A dropout layer to reduce overfitting.
    - A shared dense layer (256 units with ReLU activation).
    - Four separate fully connected layers, each predicting one emotion score (Happy, Sad, Anger, Fear).
- The model predicts four emotion scores simultaneously for each input.

Loss Function and Optimizer

- **Loss Function**: Binary Cross Entropy with Logits Loss (`BCEWithLogitsLoss`).
- **Optimizer**: AdamW optimizer with a learning rate of `2e-5` and weight decay of `0.01`.

Training

- The model was trained for 100 epochs using a batch size of 8 on train_data
- During training, after each epoch, Mean Squared Error (MSE), Mean Absolute Error (MAE), and R-squared (R^2) metrics were calculated to monitor performance.

Evaluation

After training, the model was able to predict emotion intensities fairly accurately. The evaluation metrics showed a good fit, indicating that the model successfully learned patterns related to different emotions in Telugu text.

Prediction Method

A prediction function was developed to:

- Take a new paragraph and an optional CoT as input.
- Tokenize the text and pass it through the trained model.
- Apply a softmax function to the model's outputs to get probabilities.
- Normalize and rescale the probabilities so that the emotion intensities sum to 10.
- Handle rounding adjustments carefully to ensure the final intensities meet the required constraint.

Improvements in Phase-3

Compared to the earlier phase of the project, several important improvements were made:

1. **Intensity Prediction**:
   Instead of predicting emotion probabilities, the model was modified to predict direct intensity values for each emotion, making the output more meaningful for the task.
2. **Separate Layers for Each Emotion**:
   Instead of using a single output layer, four different fully connected layers were used —

one for each emotion (Happy, Sad, Anger, Fear). This allowed the model to better specialize in predicting each emotion's intensity.

3. **Incorporating Chain-of-Thought (CoT)**:
   To improve context understanding, a CoT reasoning text was included along with each paragraph. This provided additional context, helping the model make more accurate predictions.
4. **Softmax in Final Prediction**:
   During the final prediction step, a softmax function was applied to the raw outputs to ensure that the predicted intensities were properly normalized and could be scaled appropriately.

Results:

```
Test Set Evaluation:
Test MSE: 4.1612
Test MAE: 1.7736
Test R2 Score: -534.1198
```

```
# Example Prediction
new_paragraph = "ఇలా నాలో నేను మాట్లాడుకుంటుంటే... ఏయ్...నువ్వు ఇప్పట్లో అడిగేలా లేవ్ కాని నేనో విషయం అడగనా ?
predicted_emotions = predict_emotions(model, tokenizer, new_paragraph)

print("\nPredicted Emotion Percentages:")
for emotion, percentage in predicted_emotions.items():
    print(f"{emotion}: {percentage}")
```

```
Predicted Emotion Percentages:
Happy: 1
Sad: 6
Anger: 2
Fear: 1
```