Enhancing Emotion Recognition using POS Tagging

Name: PARAMANANDA BHASKAR
Roll Number: 21CS30035

Introduction

This report presents a comparative analysis of two emotion recognizer models:

1. a baseline model and an
2. enhanced model that incorporates Part-of-Speech (POS) tagging into the feature extraction pipeline.

The models are trained and evaluated using Naive Bayes and Support Vector Machine (SVM) classifiers.

Methodology

1. Data Preparation

- **Text Preprocessing:**
    - The input text data was preprocessed by removing any unnecessary characters, lowercasing, and tokenizing the sentences.
    - **Stop Words Removal:** Commonly used words that do not carry significant meaning, such as "the," "is," etc., were removed.
    - **Lemmatization:** Words were reduced to their base or root form using the WordNet Lemmatizer, which helps in reducing inflectional forms and deriving the base form of words.

2. Baseline Model Development

- **Feature Extraction:**
    - **TF-IDF Vectorization:** The Term Frequency-Inverse Document Frequency (TF-IDF) method was used to convert the preprocessed text data into numerical features that could be fed into the machine learning models.
- **Model Training:**
    - **Naive Bayes Classifier:**
        - A Naive Bayes classifier was trained on the TF-IDF features. The alpha parameter, which controls the smoothing technique, was **optimized using Random Search**.
    - **Support Vector Machine (SVM) Classifier:**
        - An SVM classifier with a linear kernel was also trained on the TF-IDF features. The regularization parameter 'C' and class weights were **optimized using Random Search**.

- **Model Evaluation:**
    - Both models were evaluated on the validation set, and their **best hyperparameters** were noted.
    - The models were then tested on the test dataset to assess their performance.

3. Enhanced Model with POS Tagging

- **Part-of-Speech (POS) Tagging:**
    - **Viterbi Algorithm:** POS tagging was performed using the Viterbi algorithm, which is a dynamic programming algorithm used for POS tagging. The text was tokenized, and each token was assigned a POS tag.
    - **Incorporating POS Tags into Features: The POS tags were appended to the corresponding tokens, and these tagged tokens were then joined back into sentences. This effectively enriched the text data with syntactic information.**
- **Feature Extraction:**
    - **TF-IDF Vectorization:** The enhanced sentences with POS tags were vectorized using the TF-IDF technique, similar to the baseline model.
- **Model Training:**
    - **Naive Bayes and SVM Classifiers:**
        - The same models (Naive Bayes and SVM) were retrained using the POS-tagged TF-IDF features.
        - The hyperparameters were again optimized using Random Search.
- **Model Evaluation:**
    - The enhanced models were evaluated on both validation and test datasets.
    - The results were compared against the baseline models.

Results

Baseline Model Results (Vanilla Emotion Recognizer)

- **Naive Bayes Classifier**

Best Hyperparameters for Naive Bayes: {'alpha': 0.1}

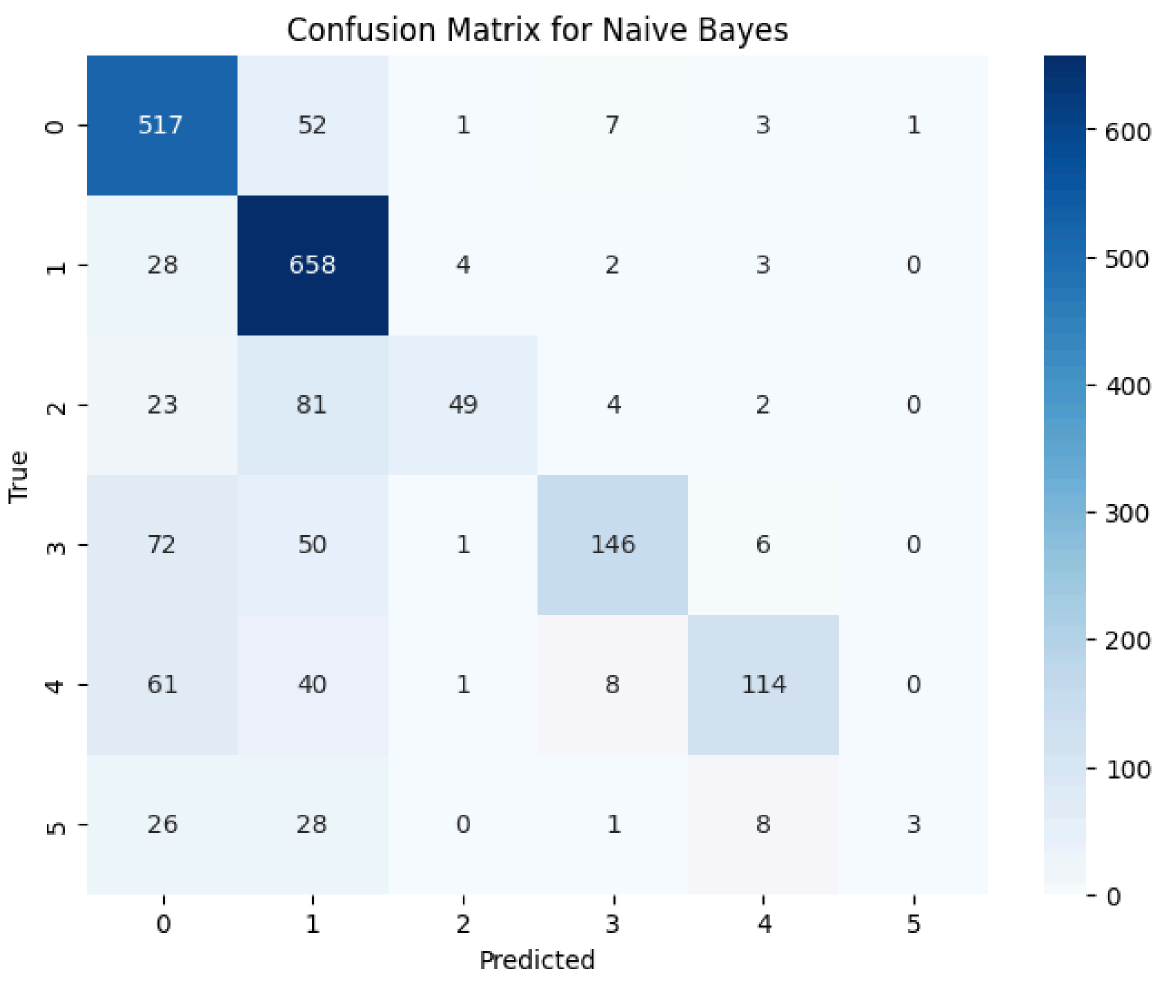
Validation Accuracy: 0.7525

Test Accuracy: 0.7435

Test Classification Report: **(0: sadness, 1: joy, 2: love, 3: anger, 4:fear, 5: surprise)**

	precision	recall	f1-score	support
0	0.71	0.89	0.79	581

```
           1       0.72      0.95      0.82       695
           2       0.88      0.31      0.46       159
           3       0.87      0.53      0.66       275
           4       0.84      0.51      0.63       224
           5       0.75      0.05      0.09        66

    accuracy                           0.74      2000
   macro avg       0.79      0.54      0.57      2000
weighted avg       0.77      0.74      0.72      2000
```



- **Support Vector Machine (SVM) Classifier**

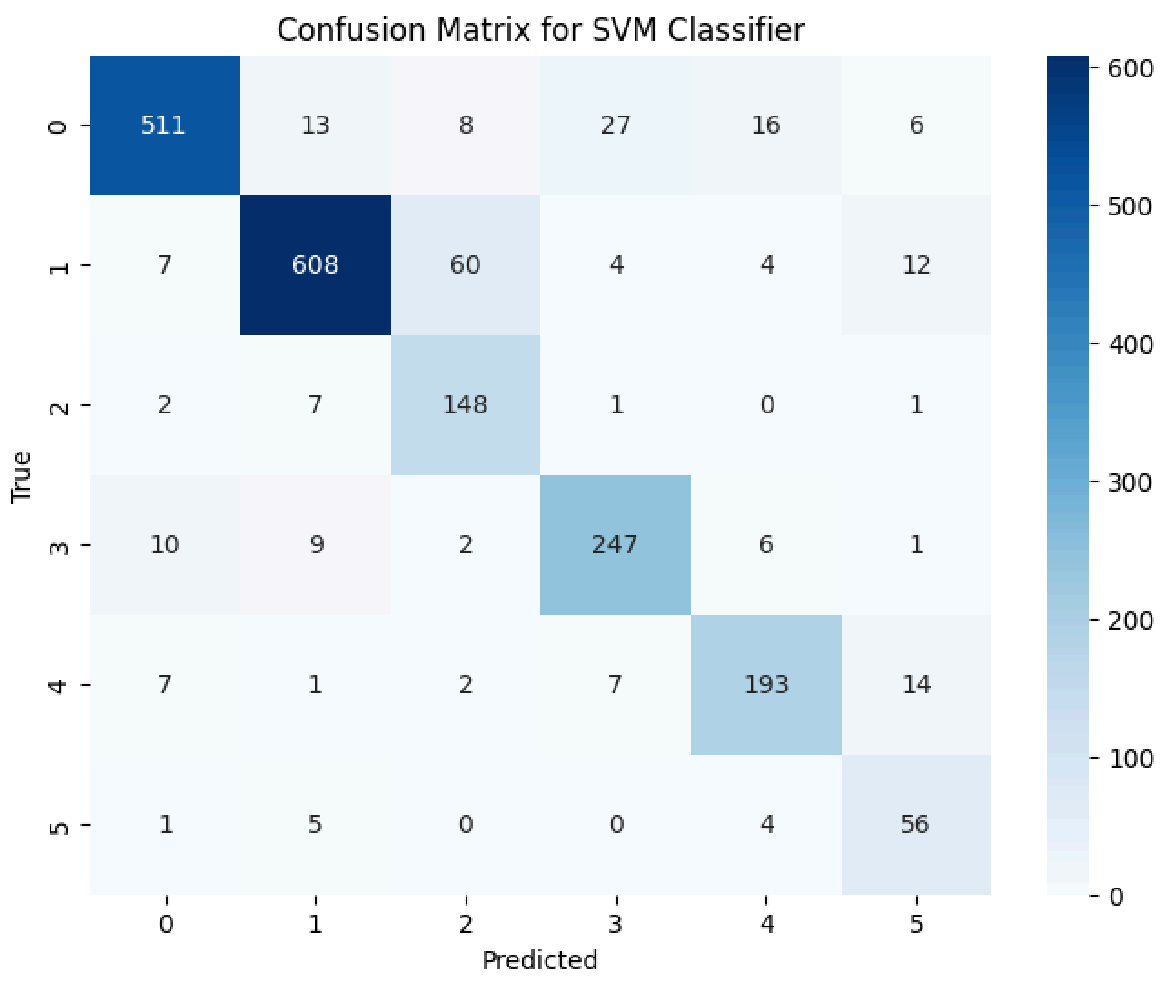
Best Hyperparameters for SVM: {'kernel': 'linear', 'class_weight': 'balanced', 'C': 1.0}

Validation Accuracy: 0.885

Test Accuracy: 0.8815

Test Classification Report: **(0: sadness, 1: joy, 2: love, 3: anger, 4:fear, 5: surprise)**

```
              precision    recall  f1-score   support


           0       0.95      0.88      0.91       581

           1       0.95      0.87      0.91       695

           2       0.67      0.93      0.78       159

           3       0.86      0.90      0.88       275

           4       0.87      0.86      0.86       224

           5       0.62      0.85      0.72        66


    accuracy                           0.88      2000

   macro avg       0.82      0.88      0.84      2000

weighted avg       0.89      0.88      0.88      2000
```



True \ Predicted	0	1	2	3	4	5
0	511	13	8	27	16	6
1	7	608	60	4	4	12
2	2	7	148	1	0	1
3	10	9	2	247	6	1
4	7	1	2	7	193	14
5	1	5	0	0	4	56

Observations from Baseline Models:

- **Naive Bayes Performance**:
  - The Naive Bayes model performed moderately well, with an overall test accuracy of 74.35%.
  - It struggled particularly with class 5, which had very few instances in the dataset. The precision for class 5 was high (75%), but the recall was extremely low (5%), indicating that the model missed most instances of this class.
  - Class 2 also posed a challenge, with a high precision (88%) but low recall (31%), showing that while the model correctly identified some instances, it failed to capture the majority.
- **SVM Performance**:
  - The SVM model significantly outperformed the Naive Bayes model, achieving a test accuracy of 88.15%.
  - The model demonstrated more balanced performance across all classes, particularly excelling in class 2, where the F1-score was notably higher than that of Naive Bayes.
  - However, similar to Naive Bayes, class 5 remained challenging, though the SVM model showed better recall (85%) compared to Naive Bayes.

Enhanced Model Results (Incorporating POS Tags)

1. **Naive Bayes Classifier**

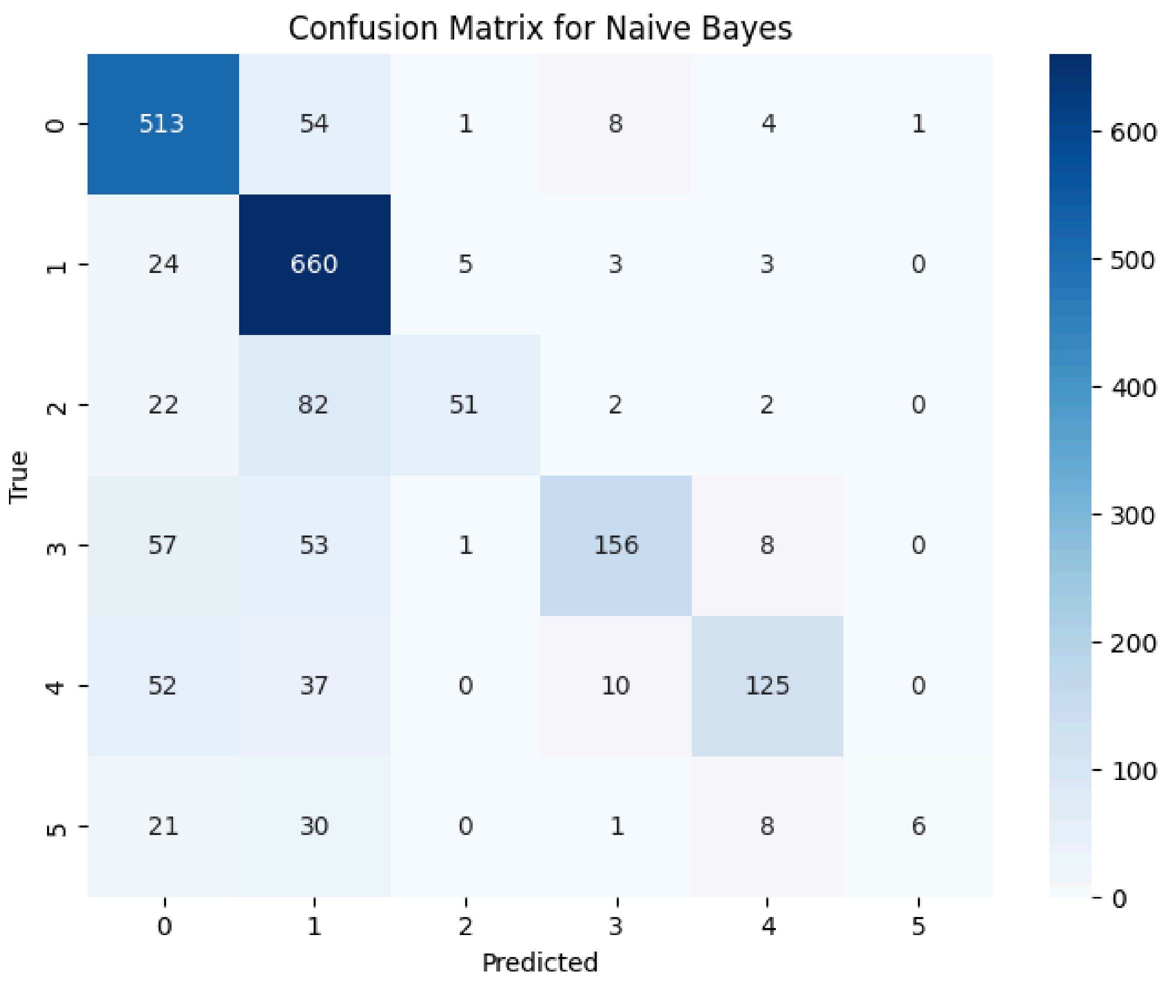
Best Hyperparameters for Naive Bayes: {'alpha': 0.1}

Validation Accuracy: 0.7605

Test Accuracy: 0.7555

Test Classification Report: **(0: sadness, 1: joy, 2: love, 3: anger, 4:fear, 5: surprise)**

```
           precision    recall  f1-score   support

        0       0.74      0.88      0.81       581
        1       0.72      0.95      0.82       695
        2       0.88      0.32      0.47       159
        3       0.87      0.57      0.69       275
        4       0.83      0.56      0.67       224
        5       0.86      0.09      0.16        66

   accuracy                         0.76      2000
  macro avg       0.82      0.56      0.60      2000
weighted avg       0.78      0.76      0.73      2000
```



2. Support Vector Machine (SVM) Classifier

Best Hyperparameters for SVM: {'kernel': 'linear', 'class_weight': 'balanced', 'C': 1.0}

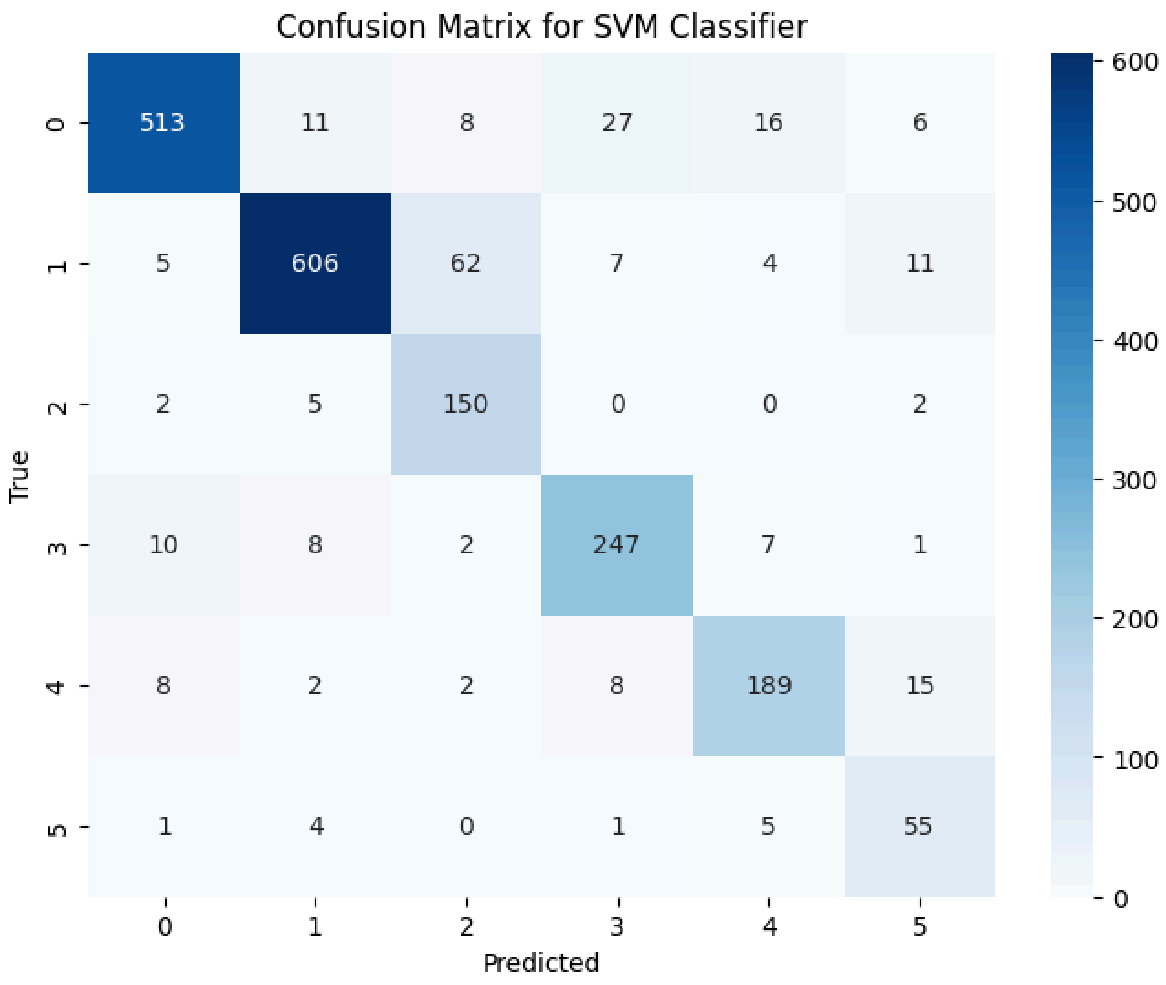
Validation Accuracy: 0.887

Test Accuracy: 0.88

Test Classification Report: **(0: sadness, 1: joy, 2: love, 3: anger, 4:fear, 5: surprise)**

```
          precision    recall  f1-score   support


       0       0.95      0.88      0.92       581

       1       0.95      0.87      0.91       695

       2       0.67      0.94      0.78       159

       3       0.85      0.90      0.87       275

       4       0.86      0.84      0.85       224
```

```
           5       0.61      0.83      0.71        66


    accuracy                           0.88      2000
   macro avg       0.82      0.88      0.84      2000
weighted avg       0.89      0.88      0.88      2000
```



Observations from Enhanced Models:

- **Naive Bayes Performance**:
  - Incorporating POS tags into the Naive Bayes model led to a slight improvement in accuracy (75.55% vs. 74.35% in the baseline).
  - The precision and recall for certain classes improved, particularly class 5, where the recall increased from 5% to 9%. However, the improvement was modest.
  - Class 2's recall also improved slightly, but the overall F1-score remained similar, indicating that while the model could identify more instances, it still struggled with precision.
- **SVM Performance**:

    - The SVM model's performance remained relatively stable with POS tagging, achieving a test accuracy of 88.0% (slightly lower than the baseline's 88.15%).
    - The addition of POS tags did not significantly impact the overall performance, suggesting that the SVM model already captured the necessary features from the text without needing POS tags.
    - Some improvement was observed in the precision and recall of specific classes, such as class 4, where the F1-score improved slightly.

Observations:

Naive Bayes vs. SVM:

- The SVM classifier consistently outperformed the Naive Bayes classifier across all metrics, both in the baseline and enhanced models.
- The SVM's ability to handle complex feature spaces likely contributed to its superior performance, making it a better choice for emotion recognition tasks in this context.

Impact of POS Tagging:

- Incorporating POS tags did not lead to substantial improvements in model performance. While some classes showed slight gains in recall or precision, the overall impact was minimal.
- This suggests that, for this particular dataset and task, the syntactic information provided by POS tags might not add significant value to the emotion recognition process, especially when using a powerful classifier like SVM.

  The Twitter messages dataset contains many new or unconventional words that are not present in the Treebank corpus. This discrepancy degrades the performance of the Viterbi algorithm, which is used for part-of-speech (POS) tagging. Because the Viterbi algorithm relies on known word probabilities from the corpus, encountering unfamiliar words can lead to incorrect tagging, which in turn diminishes the effectiveness of incorporating POS tagging in the pipeline. This limitation likely contributed to the relatively modest improvement observed when POS tags were added to the emotion recognition models.

Overall Performance:

- The final SVM model, even without POS tagging, achieved the highest accuracy and balanced performance across all classes, making it the recommended approach for emotion recognition in this context.

Advanced Modifications

In this assignment, several advanced modifications were implemented to enhance the performance and robustness of the emotion recognition models:

1. **Lemmatization and Stopword Removal**: Before applying POS tagging, the text underwent lemmatization and stopword removal. Lemmatization was performed using the WordNetLemmatizer to reduce words to their base forms, ensuring that different inflections of a word were treated as the same term. Stopword removal helped in focusing the model on meaningful words that contribute to emotion detection, reducing noise in the data.
2. **Custom POS Tagging Pipeline with Viterbi Algorithm**: The POS-tagged sentences were transformed into feature vectors using TF-IDF vectorization. The POS tags were concatenated with the original tokens to create a more informative feature set. For example, a word like "happy" might be represented as "happy:<ADJ>" in the vector space, adding context to the word's role in the sentence. This enriched feature representation aimed to provide the models with a deeper understanding of the syntactic and semantic nuances of the text.
3. **Hyperparameter Optimization with Random Search**: Both the Naive Bayes and SVM models were fine-tuned using a random search for hyperparameter optimization. This approach allowed for the exploration of a wide range of hyperparameter combinations to find the best-performing settings for each model. The use of random search instead of a grid search provided a more efficient and comprehensive exploration of the hyperparameter space.
4. **Handling of Unseen Words**: A notable challenge was the presence of many new or unconventional words in the Twitter dataset, which were not covered by the Treebank corpus used for POS tagging. This limitation was addressed by enhancing the Viterbi algorithm to better handle unknown words, though the effectiveness of this enhancement was constrained by the inherent difficulties in tagging highly informal text.

Conclusion

This assignment explored the intricate task of emotion recognition from text, focusing on enhancing traditional machine learning models through the incorporation of Part-of-Speech (POS) tags. By integrating syntactic information into the feature extraction process, the aim was to provide the models with a deeper understanding of the linguistic structure underlying emotional expressions.

The results demonstrated that while the POS-tag-enhanced models exhibited improvements in capturing certain emotions, the overall enhancement was modest. This underscores the complexity of emotion recognition, particularly when dealing with informal and dynamic text sources like Twitter. The challenge of unseen words and the limitations of existing corpora in handling social media language were significant factors that impacted the performance gains.

Despite these challenges, the assignment successfully highlighted the potential of advanced NLP techniques in improving emotion recognition. The systematic approach—combining lemmatization, stopword removal, custom POS tagging, and sophisticated model tuning—showcased the benefits of a nuanced understanding of text beyond mere word frequency.

In conclusion, while the POS tagging provided additional context that marginally improved emotion detection, it also revealed the importance of continuously evolving our linguistic models to keep pace with the ever-changing nature of language, especially in social media. Future work could explore more advanced neural approaches, such as transformer-based models, which may further enhance the ability to recognize emotions in diverse and informal text data. This assignment serves as a step forward in bridging the gap between simple text classification and more comprehensive understanding of human emotions in the digital age.