Assignment - 1

NLP: CS60075
Autumn Semester 2024
IIT Kharagpur

Enhancing Emotion Recognition Using POS Tagging

Sample Code - https://colab.research.google.com/drive/1sfARw_asCMJwiFlGIUBUernwnuC8uoaN?usp=sharing

Assignment Beginning: 08:00 PM, 18th August 2024

Assignment Deadline: 11:59 PM, 3rd September 2024

OVERVIEW

The objective of this assignment is to deepen your understanding of Natural Language Tagging or more specifically Part-of-Speech (POS) tagging which involves assigning grammatical categories (such as nouns, verbs, and adjectives) to words in a sentence.
One of the pivotal NLP tasks is Emotion Recognition which involves determining the emotions expressed in a piece of text, especially in order to determine the writer's state of mind towards a particular topic, product, etc. Some typical emotions incorporate joy, sadness, anger, fear, love and surprise.

While POS tagging might seem like a primary linguistic task, its impact extends far beyond syntax. By accurately identifying the grammatical roles of words in a sentence, POS tagging contributes to understanding the context, semantics, and even emotions conveyed by the text. This is where the concept of this assignment emerges – the exploration of the integration of POS tagging and emotion recognition..

We have seen that some words can be used as multiple parts of speech in the English language. For example:

*"Kevin has dark hair and **fair** skin."*	*(ADJECTIVE)*
*"The new **fair** is boring."*	*(NOUN)*

The addition of POS tags introduces a new dimension of linguistic insight that may significantly enhance emotion recognition accuracy and depth.

You will implement a POS tagging system, apply it to a Classical Emotion Recognizer, and evaluate its impact on emotion recognition accuracy.

TASKS

1. **POS Tagger** Implementation **(from scratch).** **[40% of TOTAL]**
    a. Use the treebank corpus (from nltk) for training data. [nltk downloader]
    b. Implement the Viterbi Algorithm (dynamic programming) for POS Tagging. **[It is mandatory to implement the algorithm yourself. No in-built library should be used for this task.]**
    c. **Keep all the POS tags for this task as that will give you proper transition and emission probabilities.**
2. **Vanilla Emotion Recognizer** **[15% of TOTAL]**
    a. Use the twitter_messages corpus for data. [Use Hugging Face APIs to get the dataset. In the shared dataset's link, train/validation/test splits are already present. Strictly use only "split" Subset (as per the link shared above) for training, validation and testing. Code snapshot to download data is shared below.]

```
from datasets import load_dataset

ds = load_dataset("dair-ai/emotion", "split")
```

    b. Use Tfidf for creating sentence embeddings.
    c. Train a classical Classifier (Naive Bayes or SVM from sklearn) for emotion recognition using the above features. Each text has to be classified into one amongst the six labels (joy, sadness, anger, fear, love or surprise). Kindly look into the dataset for more details.
3. **Improved Emotion Recognizer** **[25% of TOTAL]**
    a. Use the POS Tagger in Task 1 for POS tagging the dataset.
    b. Implement a pipeline to integrate the POS tag features along with the sentence embeddings. (Use your own creativity and **mention your strategy in the report**)
    c. Train the same Classifier (as chosen in Task 2) again for emotion recognition using the new features.
4. **Report** **[20% of TOTAL]**
    a. Add your observations to a report (submit in pdf format).

b. Compare the performance of your POS-tag-enhanced model with a baseline model that doesn't use POS tags. [Don't worry about scores]
c. Make sure to include the classification reports and confusion_matrix of both models in the report for the test split.
d. Make sure to highlight any advanced modifications that you've done in your report.

Learning Outcomes

- POS Tagging
- Emotion Recognition
- Custom Pipeline with Additional Features
- Work Presentation

Submission Guidelines

- You have to use IPython Notebooks for coding. (Use Google Colab, Kaggle for running your assignments).
- Include small documentation on each cell of the notebook. Further, properly add comments to your code as and when required.
- Make sure your IPython Notebook has the outputs from each cell. The classification report and confusion matrix must be present in the submitted notebook as well. Absence of these results will lead to deduction in marks.
- You have to submit the IPython notebook and Assignment report as mentioned above. Name each of them as NLP_Assignment_1_Roll_No.ipynb and NLP_Assignment_1_Roll_No.pdf, respectively.

  [**For eg:** NLP_Assignment_1_23CS91R06.ipynb, NLP_Assignment_1_23CS91R06.pdf]

- Submit these two files in **.zip format** on **Google Form**. **Note:** Your zip file must be named NLP_Assignment_1_Roll_No.zip and if you submit more than once, only your **latest submission** will be considered.

  [**For eg:** NLP_Assignment_1_23CS91R06.zip]

- Of course, **Plagiarism in any form is strictly prohibited**.

HAPPY LEARNING