Project Phase 2: Implementation

Introduction:

In this phase of the project, you will be combining all the data generated by each member of your group and train models to implement the system for authorship attribution (i.e. determining which of the twitter users scraped by each member of your group are responsible for posting a given tweet in the test set.

Task 1: Feature Preparation

Task 1.1: Bag of Word Features

Combine all the tweets scraped by each member, split the data using an appropriate split ratio and build a collective vocabulary and corresponding bag of word features using the functions you implemented in phase 1. The idea then was to get you to write those functions to understand how they work by demonstrating their usage within your own tweets, now you will be using that implementation on the collective group data.

Task 1.2: Embeddings

In order to familiarize you with the current state-of-the-art method of text representation, in this part you will be using embeddings to get your features. The advantage of using this over features based on pure counts is the inclusion of context and semantic information which means that your features code richer information and can therefore lead to models performing better given that they provide a higher quality input. These embeddings are obtained using transformer models, you can read more about transformers here. To understand sentence embeddings better, you can refer to this article.

For the purpose of this task, you will be using BERT Sentence Embeddings. The specific transformer model is “all-MiniLM-L6-v2”. Use this to first encode your tweets and get the representations for your dataset and then split these into train and test sets using an appropriate split ratio.

Task 2: KNNs

Now that your features are all set and ready to go, use Sickit Learn’s KNN model to get predictions for authorship attribution.
Do 5-fold cross-validation and report the validation and training losses using graphs. Use scikit-learn’s accuracy_score function to calculate the accuracy, classification_report to calculate macro-average (precision, recall, and F1), and confusion_matrix function to calculate confusion matrix on the test set.

Do this for both types of features separately.

Task 3: NNs

Use scikit-learn's Neural Network implementation to train and test the Neural Network on the provided dataset. Do 5-fold cross-validation and report the validation and training losses using graphs. Use scikit-learn's accuracy_score function to calculate the accuracy, classification_report to calculate macro-average (precision, recall, and F1), and confusion_matrix function to calculate confusion matrix on the test set.

Do this for both types of features separately.

Task 4: Ensemble Methods

Use ensemble methods for training and prediction. There are multiple methods you could use: Bagging, Boosting, Voting, etc. You are free to try these out and use the one you feel performs best on your dataset for the final submission. Use scikit-learn's accuracy_score function to calculate the accuracy, classification_report to calculate macro-average (precision, recall, and F1), and confusion_matrix function to calculate confusion matrix on the test set.

Do this for both types of features separately.

Task 4: Theoretical Understanding

Answer the following questions in a markup cell
Q1: Which model performed best and why do you think that is?
Q2: Which features gave better results for each model? Explain.
Q3: What effect would increasing the classes to 150 have?
Q4: Suggest improvements to text preparation, feature extraction, and models that can be made to perform this task better.
Q5: What - in your understanding - are the applications of authorship attribution?

Note:

For each of the models, you have to repeat the training and prediction on both types of features.

Ensure that your project notebook is well formatted and organized with proper headings, subheadings, introductions in required, and appropriate comments. Presentation will carry marks so make sure you pay attention to this along with getting results on your models

Submission Instructions:

Submit a zip file named as roll_numbers.zip containing .py and .ipynb files also named as roll_numbers.zip.