Social Computing Assignment 2
Aspect based Sentiment Classification from Reviews on Social Media Sites

Deadline: November 14, end of day [Hard deadline]

For any query about the assignment, you can contact TA Sourjyadip Ray (sourjyadipray@gmail.com).

Dataset:

https://drive.google.com/file/d/1Fsv7Ru8h4WErgXzDf5enJKLoyHpmQduS/view?usp=sharing

Assignment Objective:

Build machine learning models for aspect based sentiment classification for product/service reviews posted on a social media site.

Aspect Based Sentiment Classification:

Input: Review text + Aspect
Output: Sentiment label of review w.r.t the aspect.

Example:
Sentence: "The battery life is great but the screen is too small."
Aspect: "battery", **Sentiment:** Positive
Aspect: "screen", **Sentiment:** Negative

Dataset Description

The dataset contains 2 files:
train.csv - This file contains the training data in csv format.
test.csv - This file contains the testing data in csv format.

Description of csv file:

Header	**Description**
id	Unique id for each row.
Sentence	Review sentence on which the sentiment classification has to be done.
Aspect Term	Aspect term on the review.
Polarity	The possible values of the polarity field are: “positive”, “negative”,

	“conflict” (both positive and negative), “neutral”.
Category	Category of review - it is either “restaurant” or “laptop”.

Tasks:

Part A: Analysis of Aspect Based Sentiment Classification Dataset

Objectives:
1) Find the 5 most frequently occurring aspects in the training set for each category (“restaurant” and “laptop”) and visualize the data (category and frequency) using a bar graph (use matplotlib library). You should have 2 bar graphs (one for each category).
2) Find the distribution of the number of aspects per sentence (the same sentence can contain multiple aspects) in the entire training set and visualize the data using a bar graph. You can check for sentences having 1, 2, 3 and more than 3 aspects. You should have a single bar graph.

Part B: Classifiers for Aspect Based Sentiment Classification

Models to be used:
1) Support Vector Machine (SVM) based sentiment classifier **(Part B(1))**
2) BERT based classifier (bert base uncased) **(Part B(2))**

Evaluation Metrics to be used for both classifier models:
1) Precision
2) Recall
3) F1 score
4) Accuracy

How to incorporate both Sentence and Aspect in the input?

Method: Concatenate the aspect and the sentence into a single input. Treat the aspect as just another word in the input.

Example:
Sentence: "The battery life is great."
Aspect: "battery"
Concatenated Input: "battery [SEP] The battery life is great."
The [SEP] token indicates the boundary between the aspect and the sentence

This approach can be used for both parts B(1) and B(2).

Deliverables

- A single python notebook (.ipynb file) containing both parts of the assignment (including output of the cells).
- A single Project Report pdf file containing the deliverables in Part A, and evaluation metric values for both models in Part B

General Guidelines:

- The python notebook files should contain the code and output for both the parts. Marks will be awarded on the basis of the outputs.
- You may use a Google Colab or Kaggle Notebook to write and run your code. If you are using a server to run your code, you may use a Jupyter Notebook.
- **The code should be well commented to explain what you have done.**
- You can use the 'classification_report' function from scikit-learn to report the evaluation metrics for part B.
https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html
- You may use either pytorch or tensorflow frameworks to complete the assignment.
- You have to import the BERT model using the transformers library. You have to use SVM from the sklearn library.
- You can use the bert base uncased model from huggingface
https://huggingface.co/google-bert/bert-base-uncased
- You have to use TF-IDF vectors for representing the words in Part B(1) as shown in https://medium.com/@bedigunjit/simple-guide-to-text-classification-nlp-using-svm-and-naive-bayes-with-python-421db3a72d34.
- Both parts B(1) and B(2) should have the following code format in the ipython notebook:
1) Preprocessing (Data parsing and formatting)
2) Model Creation
3) Model Training (using train set)
4) Model Evaluation on the Test set
- You will have to perform hyperparameter optimization on both the models to achieve the best possible results. Marks will be allotted on the basis of relative evaluation metrics shown in the notebook file.
- The report should have the following contents:
1) Student Details
2) Dataset Analysis - It should contain screenshots of the bar plots from Part A.
3) Results and Analysis - It should contain all evaluation metrics from part B(1) and B(2) in a single table along with a confusion matrix for both parts.

PLEASE FOLLOW THE GENERAL GUIDELINES STRICTLY. FAILURE TO DO SO WILL RESULT IN DEDUCTION OF MARKS.

Submission Guidelines and Naming Conventions:

- The python notebook should be named in the following format: <roll_number>.ipynb (Example: 22CS71P04.ipynb).
- The Project Report should be named in the following format: <roll_number>.pdf (Example: 22CS71P04.pdf).
- Both the python notebook and project report should be zipped together, and submitted in the form of a single .zip file named in the format: <roll_number>.zip (Example: 22CS71P04.zip).

PLEASE FOLLOW THE NAMING CONVENTIONS STRICTLY. FAILURE TO DO SO WILL RESULT IN DEDUCTION OF MARKS.

Plagiarism: **We will be employing strict plagiarism checking. If your code matches with another student's code, all those students will be awarded zero marks for the assignment.** Therefore, please ensure there is no sharing of code. **You are not allowed to use ChatGPT or any other LLM for assistance in writing the code.** We will ensure strict checking is done regarding this. Please make use of the resources given.

Code Error: If the code doesn't run for a particular experiment, partial marks may be awarded based on structure and logic of code. If required, you may be contacted by the TAs to explain your code.

Additional Resources:

1. Support Vector Machines https://scikit-learn.org/stable/modules/svm.html
2. Using scikit learn for a classification task https://medium.com/@bedigunjit/simple-guide-to-text-classification-nlp-using-svm-and-naive-bayes-with-python-421db3a72d34
3. Transformers https://arxiv.org/pdf/1706.03762.pdf
4. BERT https://arxiv.org/pdf/1810.04805.pdf
5. Fine Tuning BERT for classification using transformers library https://luv-bansal.medium.com/fine-tuning-bert-for-text-classification-in-pytorch-503d97342db2
6. Jay Alammar Transformer blog http://jalammar.github.io/illustrated-transformer/
7. Jay Alammar BERT blog http://jalammar.github.io/illustrated-bert/
8. Jay Alammar Finetuning BERT http://jalammar.github.io/a-visual-guide-to-using-bert-for-the-first-time/