MiniProject 3: Classification of Textual Data

COMP 551, Fall 2023, McGill University
Contact TAs: Lang liu and Huiliang Zhang

October 2023

Please read this entire document before beginning the assignment.

1 Preamble

- This mini-project is due on Nov. 16 at 11:59pm (EST, Montreal Time). There is a penalty of 2^k percent penalty for k days of delay, which means your grade will be scaled to be out of $100 - 2^k$. No submission will be accepted after 6 days of delay.
- This mini-project is to be completed in groups of three. All members of a group will receive the same grade except when a group member is not responding or contributing to the project. If this is the case and there are major conflicts, please reach out to the group TA for help and flag this in the submitted report. Please note that it is not expected that all team members will contribute equally. However, every team member should make integral contributions to the project, be aware of the content of the submission and learn the full solution submitted.
- You will submit your assignment on MyCourses as a group. You must register your group on MyCourses and any group member can submit. See MyCourses for details.
- We recommend using Overleaf IEEE template

  https://www.overleaf.com/latex/templates/ieee-conference-template/grfzhhncsfqn for writing your report and Google colab for coding and running the experiments. The latter also gives access to the required computational resources. Both platforms enable remote collaborations.

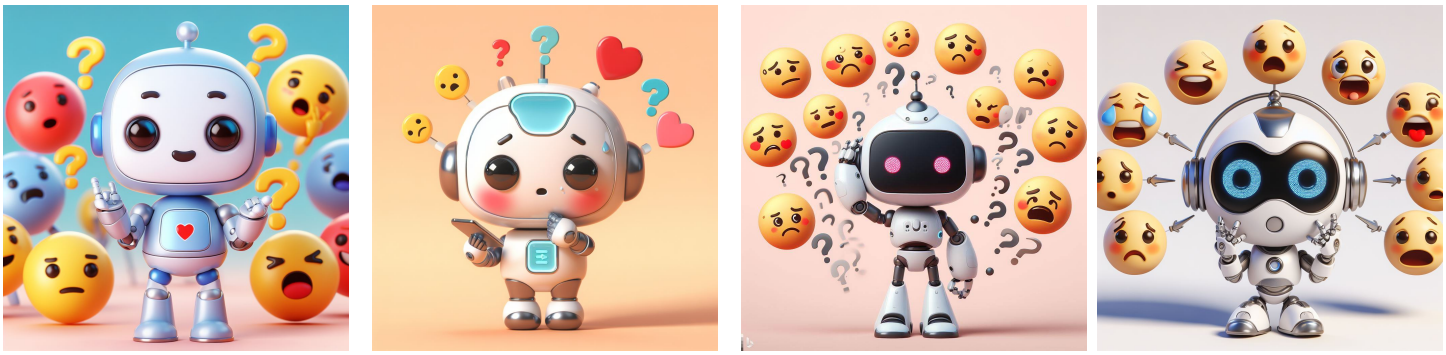

Figure 1: AI generated images for emotion detection.

- You should use Python for this and the following mini-projects. You are free to use libraries and packages.
- In particular, in most cases you should implement the models and evaluation functions yourself, which means you should not use pre-existing implementations of the algorithms or functions as found in Scikit-learn, and other packages.

2 Background

Emotion detection enables machines to detect various emotions. The technique that helps machines and computers to be capable of detecting, expressing and understanding emotions is known as emotional intelligence. In this miniproject you will implement naive Bayes from scratch and BERT-based model with pertained weights through package and finetuning, and compare these algorithms on Emotion dataset. The goal is to gain experience implementing machine learning algorithm from scratch and running the modern deep learning libraries, and getting hands-on experience comparing their performances on the real-world textual dataset.

3 Task 1: Preprocess dataset

The Emotion data can be downloaded from here: https://huggingface.co/datasets/dair-ai/emotion. You need to use only data in the "train" category for training and report the performance from the "test" category. You need to work with the text documents to build your own features.

For the naive Bayes method, you need to design the data preprocessing pipeline that turns the unstructured text data into numerical features. Specifically, you should use the bags of words representation using the scikit-learn function CountVectorizer1.

For BERT, you can use the transformers package to tokenize the input text and convert the tokens into numerical features https://pytorch.org/hub/huggingface_pytorch-transformers/. You are free to use any Python libraries you like to extract features and preprocess the data.

4 Task 2: Implement Naive Bayes and BERT models

You must implement the Naive Bayes model from scratch (i.e., you cannot use Scikit Learn or any other pre-existing implementations of these methods). In particular, your two main tasks in the part are to:

- Implement naive Bayes from scratch, using the appropriate type of likelihood for features.
- Implement a BERT-based model with pre-trained weights with a package. There are lots of existing pre-trained Bert models online, for example, you could find a pre-trained model

  https://huggingface.co/bhadresh-savani/bert-base-uncased-emotion and you could use any other one if you prefer.
- Modify or finetune the BERT-based model by using any methods that could work, and compare your results with the pre-trained one. For example, when you finetune the pre-trained model, you have the choice to finetune all the weights of the BERT model, or only the last (or last few layers). If you try both methods, you can compare in your report the pros and cons of each.

Naive Bayes Model Details

For the Naive Bayes model, you must use Python and you must implement the model from scratch (i.e., you cannot use Scikit Learn or similar libraries). Using the numpy package is encouraged. Regarding the implementation, we recommend the following approach:

- Implement naive Bayes model as a Python class. You should use the constructor for the class to initialize the model parameters as attributes, as well as to define other important properties of the model.
- Your model class should have (at least) these functions:
  - Define a **fit** function, which takes the training data (i.e., X and Y)—as well as other hyperparameters (e.g., the learning rate and/or number of gradient descent iterations)—as input. This function should train your model by modifying the model parameters.
  - Define a **predict** function, which takes a set of input features (i.e., X) as input and outputs predictions (i.e., $\hat{Y}$) for these points.
  - Define a functions **evaluate_acc** to evaluate the model accuracy. This function should take the true labels (i.e., Y), and target labels (i.e., $\hat{Y}$) as input, and it should output the accuracy score.

BERT Model Details

For the BERT model, you can use pre-trained weights by downloading the already existing pre-trained BERT model by Google or others. You can then use these pre-trained weights to do the Emotion prediction task. Then you need to finetune your model or choose to modify the structure of the model to see whether it could help you to improve the final performance. You are not required to implement this model from scratch.

5 Task 3: Run experiments

The goal of this project is to have you explore traditional machine learning and deep learning NLP techniques. You will need to conduct the multip-classification experiment on Emotion data and report the performance using accuracy. You are welcome to perform any experiments and analyses you see fit (e.g., to compare different features), but at a minimum you must complete the following experiments in the order stated below:

- In a single table, compare and report the performance of the Naive Bayes, BERT-based models and your Bert-based model, on the Emotion classification task, and highlight the winner and explain the potential reasons.
- Examine the attention matrix between the words and the class tokens for some of the correctly and incorrectly predicted documents. You will need to choose one of transformer blocks and use a specific attention head for the multi-layer multi-headed transformer architecture. You are free to do more experiments to your choosing. As a conclusion, you must answer the following question:
  - Is pretraining on an external corpus (like BERT does) good for the Emotion prediction task? What do you think pretraining does that might help with this task in particular?
  - What conclusions can you make about the performance difference between deep learning and traditional machine learning methods?

These questions are open-ended and must be answered based on your experiment results. Try to demonstrate curiosity, creativity, and an understanding of the course material in how you run your chosen experiments and how you report on them in your write-up.

6 Deliverables

You must submit three separate files to MyCourses (using the exact filenames and file types outlined below):

- code.zip: Your data processing, models and evaluation code (as some combination of .py and .ipynb or other files).
- results.ipynb: The experiments' visualization part of your models, like the attention visualization, and performance comparison. **Please keep the running results (like figures or prints results) in your .ipynb file and double-check it before you submit your file.**
- writeup.pdf: Your (max 5-page) project write-up as a pdf (details below).

Project write-up instruction

- Your team must submit a project write-up that is a maximum of five pages.
- Abstract. Summarize the project task and your most important findings.
- Introduction. You should include background information and citations to relevant work (e.g., other papers analyzing these datasets).
- System Models. Briefly introduce the core idea of the models or methods you used.
- Experiments and conclusion (at least three pages), including the Datasets, preprocessing method, experiments settings, experiments results and conclusions. You can also present the exploratory analysis you have done to understand the data, e.g. class distribution.

7 Final remarks

You are expected to display initiative, creativity, scientific rigour, critical thinking, and good communication skills. You don't need to restrict yourself to the requirements listed above - feel free to go beyond, and explore further. You can discuss methods and technical issues with members of other teams, but you cannot share any code or data with other teams.