CSE 472: Social Media Mining

Project II - Social Media Data Analysis

Prof. Huan Liu
TA: Alimohammad Beigi
abeigi@asu.edu
Released on Sept 21^{st}, 2023, Thursday
Due on November 24^{th}, 2023, 11:59PM, Friday

In this project, we will divide the project types according to the students' current programs (e.g., undergraduate and graduate programs). Read the instructions carefully before you start the project.

1 Project Type 1

All undergraduate students, including post-bachelor's, as well as MCS students should to work on this project type. In this project, students should form teams consisting of 2-members per team and notify the TA about your group members by email no later than October 6th. Students who do not email the TA with their group information will be randomly assigned to other groups.

1.1 Task: AI-Generated Scientific Paper Detection

In recent years, there has been a surge in the development of advanced large language models (LLM) that can produce fluent and highly engaging textual content. While the current capabilities of AI-generated text can be impressive, it also brings potential risks of misuse, such as academic/scientific fraud and the dissemination of AI-generated misinformation. Such challenges arise at unprecedented scales. One significant concern is how AI-generated text can exaggerate the illusion of a majority perspective, thereby amplifying marginal or deceptive viewpoints[1]. In the realm of academia, AI-generated scientific papers present a more direct challenge. Scientific papers with misinformation can harm research progress by misleading other researchers, delaying breakthroughs, and potentially harming the public. They can also erode trust in scientific publications and the peer-review process. Due to the challenges, there is increasing interest in automatic detectors for AI-generated scientific misinformation.

In this project, your objective is to develop a classifier to detect AI-generated scientific misinformation. With a dataset comprising human-written and AI-generated scientific papers (title, abstract, and introduction), your task is to correctly classify which papers are written by humans and which are generated by AI. Your classifier's performance will be evaluated using the F1 metric, emphasizing achieving the highest score possible to reduce classifier error.

We formally define the task as follows. Given a scientific paper, participants are asked to classify it into the following two classes (number indicates the label):

- **HUMAN (0)**: Text authored by humans.
- **AI (1)**: Text generated by an AI (LLM).

[1] https://www.wired.com/story/ai-generated-text-is-the-scariest-deepfake-of-all/

1.2 Resources

Following are some helpful resources that could assist you in completing the task.

Primer on Creating a Text Classifier

- Guide to understand and implement text classification in python
- Practical Text Classification With Python and Keras

AI-generated Text Detection

- How to spot AI-generated text
- How to Detect AI-Generated Text, According to Researchers

1.3 Submission Guidelines

To assess the performance of your Team, a competition on Kaggle will be conducted. Kindly follow these steps:

1. Visit the Kaggle competition.
2. Navigate to *Team* and rename your Team using the number assigned to you (e.g., **Team-11**).
3. Access *Data* to download the datasets.
4. For each level, train your model with *train.csv* and make predictions using *test.csv*.
5. Adhere to the format in *sample_submission.csv* when submitting results.
6. Note: Submissions are limited to 15 times per day to prevent avoid any potential tricks (i.e., random guessing).

For any queries regarding this task, contact **Tharindu Kumarage** at kskumara@asu.edu.

2 Project Type 2

Ph.D. and Master students with a thesis should work on this project type. You should form teams of 2-members per team and notify the TA about your group members by email no later than October 6th. Note that you may not be assigned to the task you want as we want to evenly assign the tasks. If you want to discuss the project further, please contact the mentor assigned to each task.

2.1 Task 1: Zero-shot AI-generated text detection

The objective of this task is to develop a classifier to distinguish AI-generated text from human generated one in a zero-shot manner, i.e., without any access to the training data.

For this task, contact **Raha Moraffah** at rmoraffa@asu.edu

2.2 Task 2: Neural Fake News Detection

The goal of this task is to utilize the power of Large Language Models to distinguish fake-news generated by the LLMs from the real news. This task will be conducted in three learning paradigms: (1) Zero-shot: with no access to the training data; (2) Few-shot: with access to only few samples; and (3) Supervised: which has access to the entire training dataset.

For this task, contact **Raha Moraffah** at rmoraffa@asu.edu

2.3 Task 3: Adversarial Text Purification

Deep Neural Networks (DNNs) are shown to be susceptible to adversarial attacks, which carefully craft and impose human-imperceptible perturbations to benign inputs to fool the target model into the wrong classification. There have been many works on defending DNNs against such adversarial attacks. Among these methods, adversarial purification aims to purify adversarially perturbed texts before classification. The objective of this task is to harness the power of LLMs to for the purification task.

For this task, contact **Raha Moraffah** at rmoraffa@asu.edu

2.4 Task 4: LLMs as Components in an ML Pipeline

Solving a task using machine learning methods requires a series of steps that often require large amounts of human effort or labor. Some examples of this are (i) data collection, (ii) data curation or something similar, (iii) data labeling, (iv) data augmentation, etc. Furthermore there might be more steps *after* the training the ML model, such as evaluation, explaining the behavior of the model, interpreting model outputs, etc. Many of these steps are also often human labor intensive. In this project, we will investigate how to effectively use Large Language Models (LLMs) to automate various aspects of this pipeline. You will be working closely with a mentor to tackle one of these components in the ML pipeline and build a system where an LLM can perform the task instead.

For this task, contact **Amrita Bhattacharjee** at abhatt43@asu.edu

2.5 Task 5: Neural Authorship Attribution

A pivotal component of AI-generated-text forensics involves identifying the source LLM used to generate a specific text, a process referred to as neural authorship attribution. The importance of neural authorship attribution cannot be overstated, as it provides a critical function within the broader forensic process. The ability to attribute a piece of AI-generated text to a particular LLM aids in unmasking the underlying characteristics of malicious actors and campaigns that misuse these LLMs.

Recently, we see a rapid development in the open-source LLMs that are fine-tuned versions of a base LLM such as Llama-2. This rapid development of new LLMs can occur in multiple ways, such as: 1) parameter-efficient tuning of a base LLM, 2) instruction-tuning of a base LLM, and 3) direct supervised tuning of a base LLM.

The goal of this task is to analyze the neural authorship attribution problem under the above-mentioned rapid-development setting. You will be mainly working on 1) data generation/collection from multiple LLMs, 2) hierarchical authorship attribution (hierarchical clustering and graph-based methods) and 3) detecting the tuning method incorporated to create the variant LLM.

For this task, contact **Tharindu Kumarage** at kskumara@asu.edu

2.6 Task 6: Idea Proposal

This task is for teams who are willing to pitch their own idea for the project. To do this, you need to **write a short proposal** that contains (1) definition of the problem (2) motivation (3) your proposed approach and the data sets needed to solve the problem. Please email your proposal to **abeigi@asu.edu** with the list of members of your team by October 6th. We will review your proposal and tell you whether we approve or disapprove the idea.

For this task, contact **Ali Beigi** at abeigi@asu.edu

3 Submission

Students are required to submit their source code, dataset, report (in PDF), and a text file named "*README.md*". *README.md* should contain instructions about how to execute the code (e.g., OS environment, command lines, directory structure, etc.). For the submission, **we only expect 1 submission per group. i.e., only one of the members will submit the project on Gradescope**.

Source code should be written in Python and it should be executable on TA's side. Thus, you should provide a file named "***requirements.txt***" that resolves package dependencies when using the command "*pip install -r requirements.txt*". Your python code should be the extension of ".py". if you implemented your codes through iPython, Jupyter, or Colab, please convert them into python file.

If the dataset size is too big, you can submit the file by compressing it to *.zip*. If the file size still exceeds the limit, please provide the link for your google drive for the dataset in *README.md*.

Your report should include at least 7 sections (1) Abstract (2) Introduction (3) Related Works (4) Model Description (5) Experiment (6) Future works (7) References. For the reference section, **cite all the papers, tutorials, packages, software and libraries** you used for your program. The experiment should be solid enough to show the distinction of your model with appropriate metrics. The report should not be less than 2 pages and should include a description of the data pre-processing, model, and validation results.

4 After Submission

You will be expected to give a short presentation (2-minutes long) about your project. Further instructions will be delivered through the canvas announcement.

5 Academic Integrity

- To prevent any potential plagiarism, we will randomly select teams and check if the result is reproducible. i.e., we check if your model automatically produce the reported result.
- Your codes in the submission will be automatically checked by the similarity detection tools. i.e., make sure to implement your codes from scratch.
- You can refer to and utilize existing tutorials, packages, software, and libraries you need for the project, but make sure to cite them in the "Reference" section in the project report.
- For all the steps, you can only *refer* to others' code and use libraries, software, and packages but it is NOT permissible to copy any existing code from others.