Transformer-BERT Take Home Assessment
April 2, 2025

==========Instructions==========

1. **Submission Format:** A zip file with the name DL-Week1_<RollNo>.zip containing all the files.
2. You are not allowed to use any external libraries other than PyTorch, transformers, and required libraries for preprocessing along with scikit-learn, scipy, numpy, pandas etc.
3. Plagiarism will lead to **100% penalty** to all the involved parties. You are neither allowed to discuss among each other nor share code artifacts among each other.
4. Scoring will be automated script-based, so maintain the proper file naming. Try to make the best-effort submission.
5. **Please** make sure to follow the correct file requirements in terms of command line arguments and proper file naming conventions.

==========

This week, we will extend our previous assignment by using Deep Learning models to build a classifier that labels a social media post as either real (verified information) or fake (misinformation).

Dataset
We will use the dataset from Constraint@AAAI-2021, which focuses on detecting COVID-19-related fake news in English. It consists of 10,600 social media posts from platforms labeled as real or fake

We will fine-tune BERT-based models for text classification. Instead of using traditional deep learning models like DNNs and CNNs, we will use pre-trained transformer models. The goal is to train these models to accurately classify social media posts as real or fake using AutoModelForSequenceClassification from Hugging Face's Transformers library.

Task-1: Prepare dataset. [same as you have done]
Task-2: Preprocessing Social Media Post. [same as you have done]

Task-3: **Obtaining Representations using Bert-based Model:**
We will be considering three BERT-based models in this assignment:

- bert-base-uncased
- covid-twitter-bert
- twhin-bert-base
- socbert

Task-4: Training Classifiers with Hyperparameter Tuning

In this step, you will fine-tune transformer-based models by optimizing key hyperparameters such as learning rate, batch size, number of epochs, and others to achieve better classification performance. Hyperparameter tuning helps in finding the best configuration for training, reducing overfitting, and improving generalization. We may use libraries like Optuna for automated hyperparameter optimization or manually experiment with different values. The best-performing model based on validation performance will be selected for final evaluation.

Task-5: Evaluating Models.

Now that your machine learning models are tuned and ready for evaluation, prepare a report that includes a confusion matrix, classification accuracy, f1-score, precision, and recall (both micro and macro) on the test split that you obtained in Task 1. You can use the classification_report routine of scikit-learn for this task, but for the sake of understanding, we would recommend writing a routine of your own. Also include the final best set of hyperparameters obtained specific to each model in the report.