Mental Health Indicators AI HACKATHON

TEAM NAME: AI Avengers

TEAM LEAD: Sejal Sharma

This Report corresponds to our final submission in the AI\_Hackathon conducted by IIT\_BHU.

Problem Statement: Using the primate dataset create and train a model that processes the input text and predicts 9 mental health indicators by automatically analysing the texts.

Solution:

The above problem falls under the Natural Language Processing domain of AI. Specifically it is a Multi-Label -Classification Problem. For these types of problem many methodologies can be used like Classical ML algos SVM classifier, Decision Trees, Random Forest Classifiers, or Deep Learning Methods like LSTMs, RNNs, GRUs, Bidirectional LSTMs and more advanced methods like Transformers and LLMs. **For our submission we propose fine tuning “DistillBERT” (src:** [**https://huggingface.co/docs/transformers/model\_doc/distilbert**](https://huggingface.co/docs/transformers/model_doc/distilbert)**) which is a distilled, smaller version of BERT (Bidirectional Encoder Representations from Transformers).** It is designed to be computationally more efficient and faster while retaining much of the performance of BERT.

Transformers, a groundbreaking architecture in natural language processing (NLP), have revolutionized the way machines understand and generate human language. Introduced by Vaswani et al. in the paper "Attention is All You Need" in 2017, transformers represent a departure from traditional recurrent and convolutional neural networks, offering superior performance in capturing long-range dependencies and contextual information.

At the heart of transformers lies the self-attention mechanism, enabling the model to weigh the importance of different words in a sentence dynamically. This mechanism allows transformers to process input sequences in parallel, avoiding the sequential nature of earlier architectures and significantly reducing training times. Transformers consist of an encoder-decoder structure, but for NLP tasks, particularly in fine-tuning scenarios, we often focus on the encoder side.

What is DistilBERT?

DistilBERT, a distilled version of BERT (Bidirectional Encoder Representations from Transformers), is one such transformer-based model that has proven effective in various NLP applications. DistilBERT retains the key aspects of BERT but reduces its size, making it more computationally efficient and suitable for resource-constrained environments. This reduction is achieved through a process known as knowledge distillation, where a larger pre-trained model (like BERT) imparts its knowledge to a smaller one (DistilBERT).

The power of transformers, and consequently models like DistilBERT, lies in their ability to capture context and relationships between words in a given context. Unlike traditional models that process words sequentially, transformers consider the entire sequence simultaneously, making them adept at understanding nuances, dependencies, and contextual meanings in natural language.

Hugging Face's transformers library has played a pivotal role in democratizing access to pre-trained transformer models and facilitating their fine-tuning for specific tasks. The library provides a comprehensive collection of pre-trained models, tokenizers, and utilities for various NLP tasks, making it accessible for researchers and developers to leverage state-of-the-art models without starting from scratch.

**Proposed Method: Fine Tuning DistilBERT**

Fine-tuning with transformers involves taking a pre-trained model on a vast corpus (such as BERT or DistilBERT) and adapting it to a specific task with a smaller, task-specific dataset. This transfer learning approach allows models to learn from a general language understanding and apply that knowledge to more specialized tasks, yielding impressive results even with limited labeled data. And we have implemented this approach.

Features of DistilBERT

* The distilled model is trained to reproduce the behavior and predictions of the larger model, but with fewer parameters.
* DistilBERT has fewer parameters than BERT, resulting in a smaller memory footprint and faster inference.
* Like BERT, DistilBERT uses self-attention mechanisms that allow it to focus on different parts of the input sequence when encoding information.This enables the model to capture long-range dependencies and understand the context of each word in relation to the entire input.
* DistilBERT is often used as a base model for transfer learning in various natural language processing (NLP) tasks.
* Hugging Face's `transformers` library provides a user-friendly interface for working with pre-trained transformer models, including DistilBERT.
* The library includes tokenizers, pre-trained models, and utilities for fine-tuning, training, and inference.
* Fine-tuned DistilBERT models can be used for a variety of text-based tasks, such as text classification, sentiment analysis, named entity recognition, and more.

In summary, DistilBERT is a smaller and computationally more efficient version of BERT, and its use in fine-tuning text-based models is facilitated by the `transformers` library from Hugging Face, which provides pre-trained models and tools for easy integration into NLP workflows.

**STEPS WE HAVE FOLLOWED IN OUR MODEL TRAINING AND CREATION**

**Environment: Google Colab GPU**

* 1. Downloaded the given dataset and converted it from JSON to Excel while performing preliminary annotations division
* Converted the given Json File to an Excel file

A computer screen shot of a program code

Description automatically generated

* Performed Data Analysis using Excel by converting annotations of mental health indicators into 9 separate annotations.
* Removed Outliers and saved the Excel File.

A screenshot of a computer

Description automatically generated

* Imported necessary libraries and loaded the saved excel file.
* Explored the first few rows, shape, and information of the dataset.
* Checked for duplicate entries (no duplicates found).
* Prepared the dataframe with int labels and str texts

A screenshot of a computer

Description automatically generated

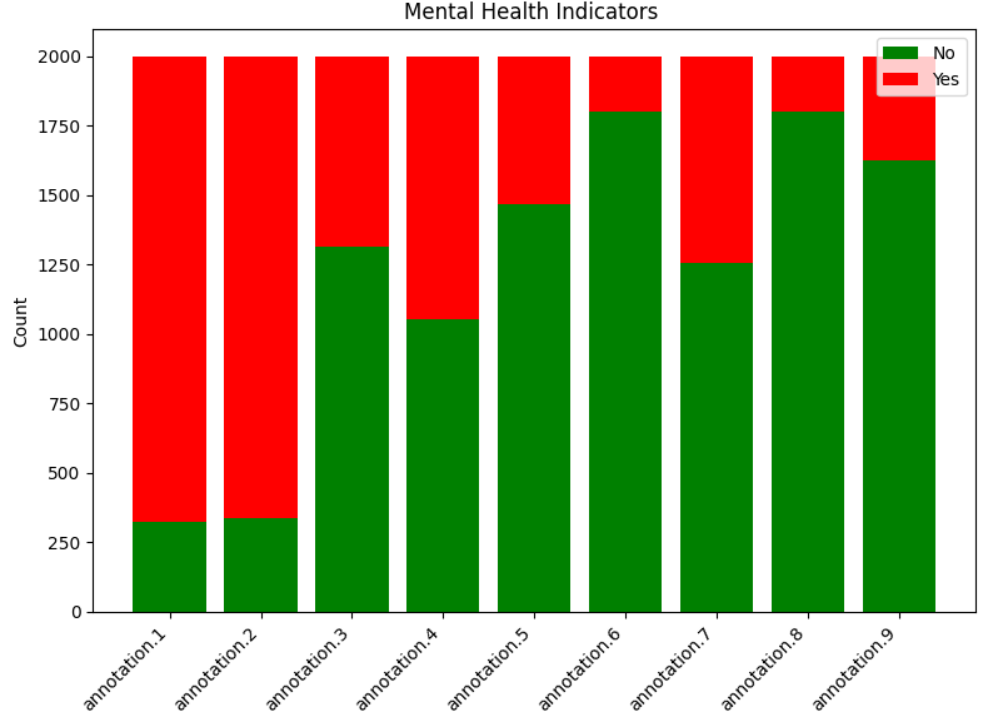
A screenshot of a computer program

Description automatically generated A screenshot of a computer code

Description automatically generated

2. Exploratory Data Analysis

* Visualized the distribution of text lengths and explored the distribution of labels in the dataset.



A graph of a number of columns

Description automatically generatedA graph of a number of blue squares

Description automatically generated

3. Data Pre-processing and Text-Cleaning

* Converted text data and labels for further processing.
* The original dataset (df) is split into training, validation, and test sets.80% of the data is used for training (train\_df), and validation (val\_df), and the remaining 10% for testing (test\_df).

A black text on a white background

Description automatically generated

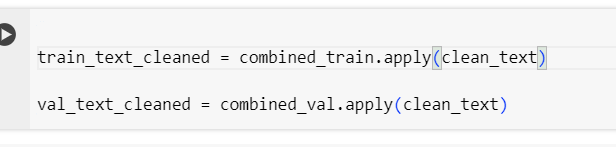
* Columns 'post\_text' and 'post\_title' are excluded from the label columns (not\_chosen\_columns). And the remaining columns are selected as label columns (label\_columns). Labels are converted to lists (train\_labels, val\_labels, test\_labels).
* Text data is combined from 'post\_title' and 'post\_text' columns into new columns (combined\_train, combined\_val, combined\_test). These combined text columns are converted to lists (train\_text, val\_text, test\_text).

A screenshot of a computer code

Description automatically generated A screen shot of a computer program

Description automatically generated

* Performed data cleaning. Converted all text to lowercase for consistent representation.
* Eliminated URLs from the text to remove any web links.
* Handled emojis by converting emojis to text representation and also removed punctuation marks from the text.
* Eliminated common English stopwords, such as "the" and "is," to focus on meaningful content.
* Reduced multiple consecutive whitespaces to a single space and removes leading/trailing whitespaces.



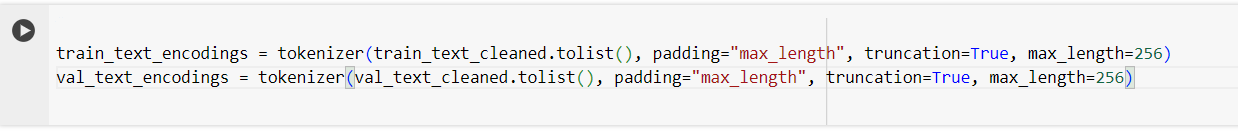
These cleaning steps aim to enhance the quality of the text data by eliminating noise, irrelevant details, and ensuring a consistent format. The cleaned text is then more suitable for training and evaluating machine learning models in natural language processing tasks.

4. Tokenization and Model Building:

* Used DistilBERT tokenizer to convert cleaned text data into numerical tokens for a multi-label classification task. The DistilBERT model is then initialized for sequence classification with a 9 number of output labels corresponding to 9 mental health indicators.
* This prepares the data and the model for training and evaluation.
* For tokenization sequences are padded to the maximum length specified (256 in this case). Padding is crucial for creating uniform input sequences, as models typically expect inputs of consistent length. It helps handle variable-length sequences by adding padding tokens to shorter texts.
* If a text exceeds the maximum length (256), truncation is applied to reduce it. This is essential to ensure that all sequences are within the model's input size limits. Truncation prevents potential issues with excessively long texts that may not fit into the model.

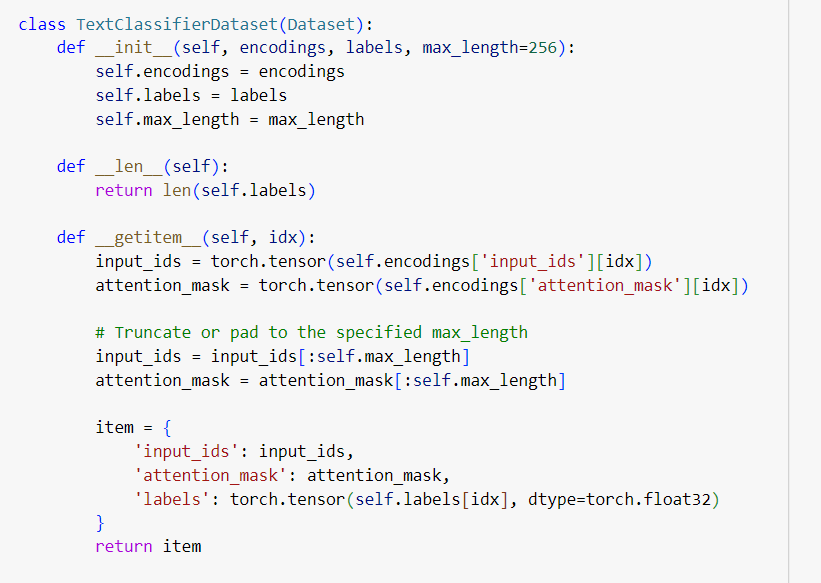
A computer screen shot of a computer code

Description automatically generated



5. Custom Dataset Creation

* Developed a custom dataset class TextClassifierDataset, for handling tokenized input, attention masks, and labels.
* The dataset class that inherits from PyTorch's Dataset class. This class is tailored for the specific requirements of our text classification task.
* This method extracts input\_ids and attention\_mask from the tokenized encodings for the specified sample.Then Truncates or pads input\_ids and attention\_mask to the specified max\_length.
* Finally, Constructs a dictionary (item) containing 'input\_ids', 'attention\_mask', and 'labels' for the sample.
* This ensures that the input data is in the correct format and structure for effective model training.



6. Evaluation Metrics and Training:

* Implemented evaluation metrics such as ROC-AUC, hamming loss,accuracy,precision,recall and F1 score for multi-label classification. Utilized a threshold (0.7 after hypertuning) to convert probability predictions to binary predictions.Computed a dictionary of multi-label classification metrics.
* Implementing the training process for a multi-label text classification model (DistillBert) using the Hugging Face Transformers library. Set up training arguments and initiated the Trainer from the transformers library.
* Specified training arguments such as batch size, output directory, number of epochs, and saving configurations.Used the defined model, training and validation datasets, and evaluation metrics Iterated over the specified number of epochs (num\_train\_epochs) and Updated the model parameters based on the training dataset.
* Periodically evaluated the model on the validation dataset using the defined evaluation metrics (compute\_metrics function). The trained model and associated results are saved in the specified output directory (output\_dir).
* The training process is configured with specific arguments and evaluation metrics, and the model is trained on the provided training dataset while periodically evaluating on the validation dataset. Trained the model for 6 epochs, saving the model and associated metrics.

7. Model Saving

The trained model and associated results are saved for further analysis or deployment.A close-up of a computer screen

Description automatically generated

8. Predictions:

Demonstrated how to use the trained model for making predictions on new text data.

Obtained multi-label genre predictions for a sample text.

A screenshot of a computer code

Description automatically generated

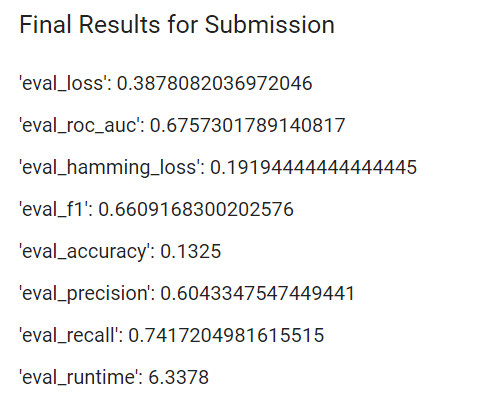
9. Testing Script

Finally we have created a testing script that takes in test.json file processes it loads our saved model and passes it for prediction and finally writes the mental health annotations and predicted labels in the result.txt file.

A screenshot of a computer program

Description automatically generated

10. Results on Test Dataset



Training Results: {'eval\_loss': 0.3003917932510376, 'eval\_roc\_auc': 0.7306568408947467, 'eval\_hamming\_loss': 0.14812161731932505, 'eval\_f1': 0.7402545892002979, 'eval\_accuracy': 0.2148997134670487, 'eval\_precision': 0.7174723878610284, 'eval\_recall': 0.8153550455631311, 'eval\_runtime': 23.3295, 'eval\_samples\_per\_second': 59.839, 'eval\_steps\_per\_second': 7.501, 'epoch': 5.0

This report outlines the process of loading data, exploring and preprocessing it, building a multi-label classification model using DistilBERT, and evaluating the model's performance. It concludes with saving the trained model and providing an example of making predictions using the saved model.

Top of Form