# Tweets Sentiment Analysis with Transformers

## Introduction:

After some research into the state of the art in this field, I discovered that **BERT** (Bidirectional Encoder Representations from Transformers) is the leading model. So, I focused on developing architectures based on BERT to both enhance my understanding of **transformers** and **attempt to improve the model's performance**. I chose to use the **PyTorch framework** to further develop my skills in this advanced area of deep learning.

## Hardware Setup:

I conducted my experiments on Kaggle, using their NVIDIA Tesla P100 GPU for efficient model training and testing.

## Data Preprocessing:

I first removed unnecessary elements such as tags, hashtags, URLs, punctuation, multiple spaces, and numbers to clean the text. However, I chose not to remove stop words, as several studies suggest that doing so can alter the context of sentiments; for instance, "not happy" has a different meaning than "happy." Then , I divided the dataset into training, testing, and validation sets with proportions of 80%, 12%, and 8%, respectively.

## Text or Selected-text ?:

Here we have two columns: "text" and "selected text," where the second is a subset of words from the "text" column, (the method of selection was not mentionned). To understand this relationship, I used BERT's tokenizer and model to embed both columns of each sample, then calculated **the similarity matrix between them (matrice de correlation)**.
The average similarity score was **0.72**, which is significant. So I opted to work with **the full text to better preserve the context of the sentences**.

## Modeling:

**I started my modeling with a basic BERT model. Then I tried to improve the accuracy, with adding LSTM and GRU layers, creating the BERTLSTM and BERTGRU models. These combinations were intended to better retain the textual context and enhance memory within the model, aiming for improved results :**

1. **BERT:**utilizd a pre-trained BERT model(bert-base-uncased) for feature extraction, followed by a custom classifier(simple nn)
2. **BERTLSTM**: combined the pre-trained BERT model (bert-base-uncased) for initial text embedding with an LSTM layer .The LSTM is configured with options for multiple layers, bidirectionality, and dropout for regularization (pour éviter l'overfitting)
3. **BERTGRU:** combined the pre-trained BERT model with a GRU (Gated Recurrent Unit). The GRU is designed with multiple layers, bidirectional capabilities, and dropout to effectively handle long-range dependencies in text.

## Loss Function :

I initially used **cross-entropy** due to its effectiveness in multi-classification tasks. However, given the **slight imbalance** in the dataset, where 'neutral' sentiments were more prevalent, I experimented with **focal loss**, which is often recommended in this case. Unfortunately, focal loss did not give good results. I also tried using **weighted cross-entropy to fix class imbalance**, but this approach also led to **unsatisfactory results.** So, I decided to continue with the **standard cross-entropy loss**, with the **Adam optimizer**

## Models comparison :

| | Batch size | Epochs | Learning rate | Params (Millions) | GFLOPs (1e9) | Train Accuracy | Test Accuracy | Inference time (seconds) |
|---|---|---|---|---|---|---|---|---|
| **BERT** | 32 | 40 | 2e-5 | 109.52 | 82.16150 | 0.99 | **0.78** | 0.012 |
| **BERT + LSTM** | 32 | 40 | 2e-5 | 113.16 | 82.16032 | 0.75 | **0.69** | 0.013 |
| **BERT + GRu** | 32 | 40 | 2e-5 | 112.24 | 82.16032 | 0.72 | **0.71** | 0.013 |

## conclusion :

The standalone BERT model achieved the highest accuracy of 0.78, compared to 0.69 for BERT-LSTM and 0.71 for BERT-GRU. This suggests that BERT alone is more effective for our dataset than when combined with recurrent neural network layers. A possible explanation for this outcome could be the length of the training sentences. If the sentences are relatively short, the additional complexity and memory capabilities of LSTM and GRU layers do not significantly contribute to performance improvements.
Therefore, the simpler BERT model without recurrent layers appears to be the most efficient approach for this specific sentiment analysis task.

## Similar work and results

I have worked on a similar sentiment analysis task on IMBD movies' reviews dataset. I worked with Tensorflow and tested different combinations of recurrent NNs and pre-trained Embedders.

**Github repository:**
https://github.com/HaykelBargouguy/Sentiment-Analysis-Pretrained-Embedders-and-Recurrent-NNs-combinations/tree/main