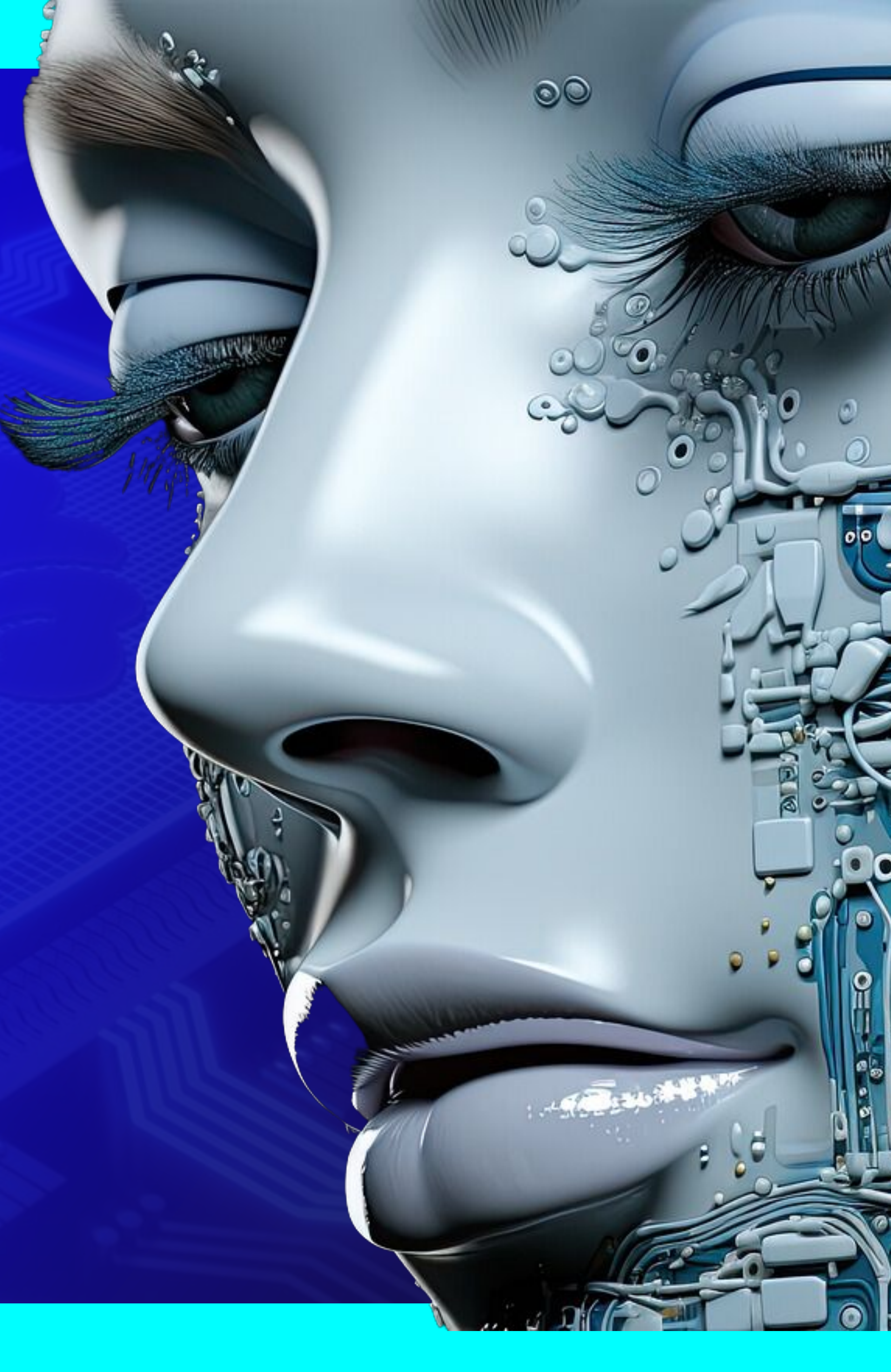
UNIVERSITÀ
DELLA CALABRIA

DIPARTIMENTO DI MATEMATICA
E INFORMATICA

LLM - DETECT AI GENERATED TEXT

CRISTIAN CIRIACO CAMPAGNA 242604
GIOVANNI IANNUZZI 214900
PIERPAOLO SESTITO 242707



Overview

Problem description

- Distinguish between human-generated and language model generated text.

- The Kaggle Competition, "LLM Detect AI Generated Text" provides us an opportunity to explore and address this task.

The goal

- Construct a machine learning model that excels in classifying text as either human-authored (labeled with 0) or generated by a language model (labeled with 1).

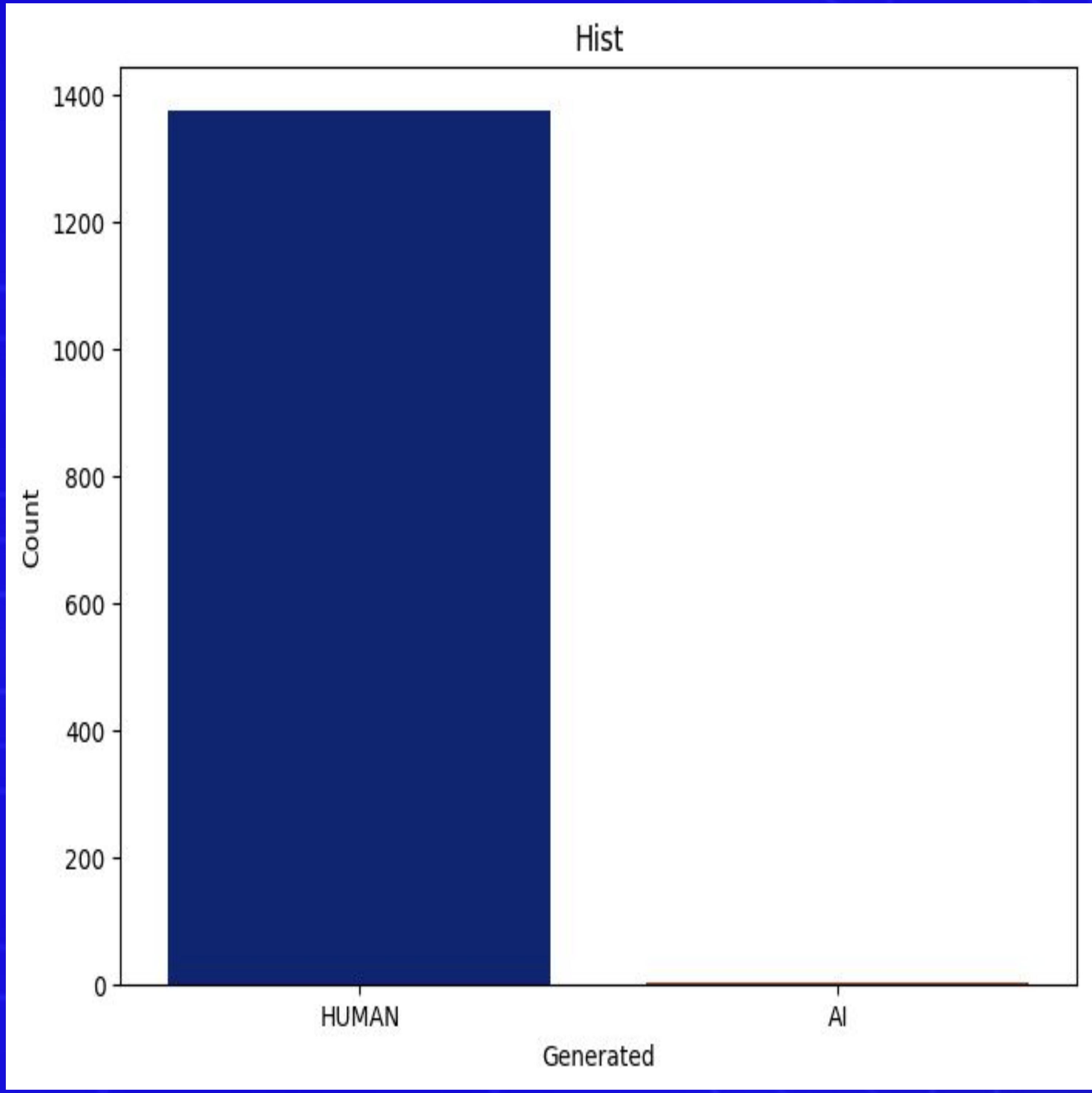
Dataset description

train_essays.csv

ID	**A unique identifier for each essay**
prompt_id	Identifies the prompt the essay was written in response to.
text	The essay text itself
generated	Whether the essay was written by a student (0) or generated by a LSSM (1).

train_prompt.csv

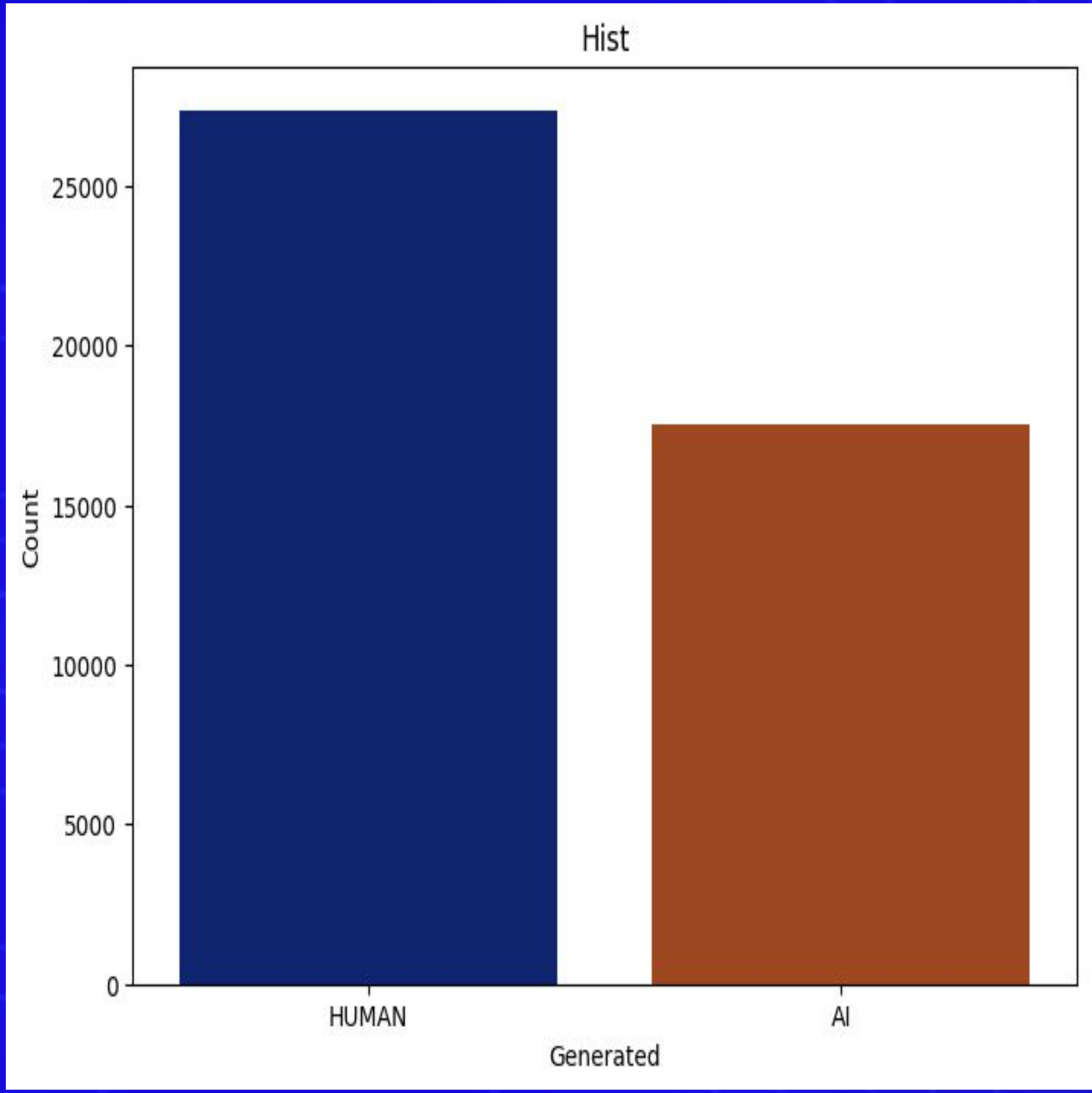
prompt_id	**A unique identifier for each prompt**
prompt_name	The title of the prompt.
instructions	The instructions given to students.
source_text	The text of the article(s) the essays were written in response to.



TRAIN_ESSAYS.CSV

CLASS IMBALANCE

- 1375 labeled as 0
- 3 labeled as 1



DAIGT-V2

DATASET PROVIDED BY THE KAGGLE COMMUNITY

- 27371 labeled as 0
- 17497 labeled as 1

More equitable distribution but not at all!

Extended dataset

1

Cosine similarity

- Calculated for pairs of textual records attributed to human authors.
- Entries with similarity >= 0.9 were pruned to reduce redundancy.

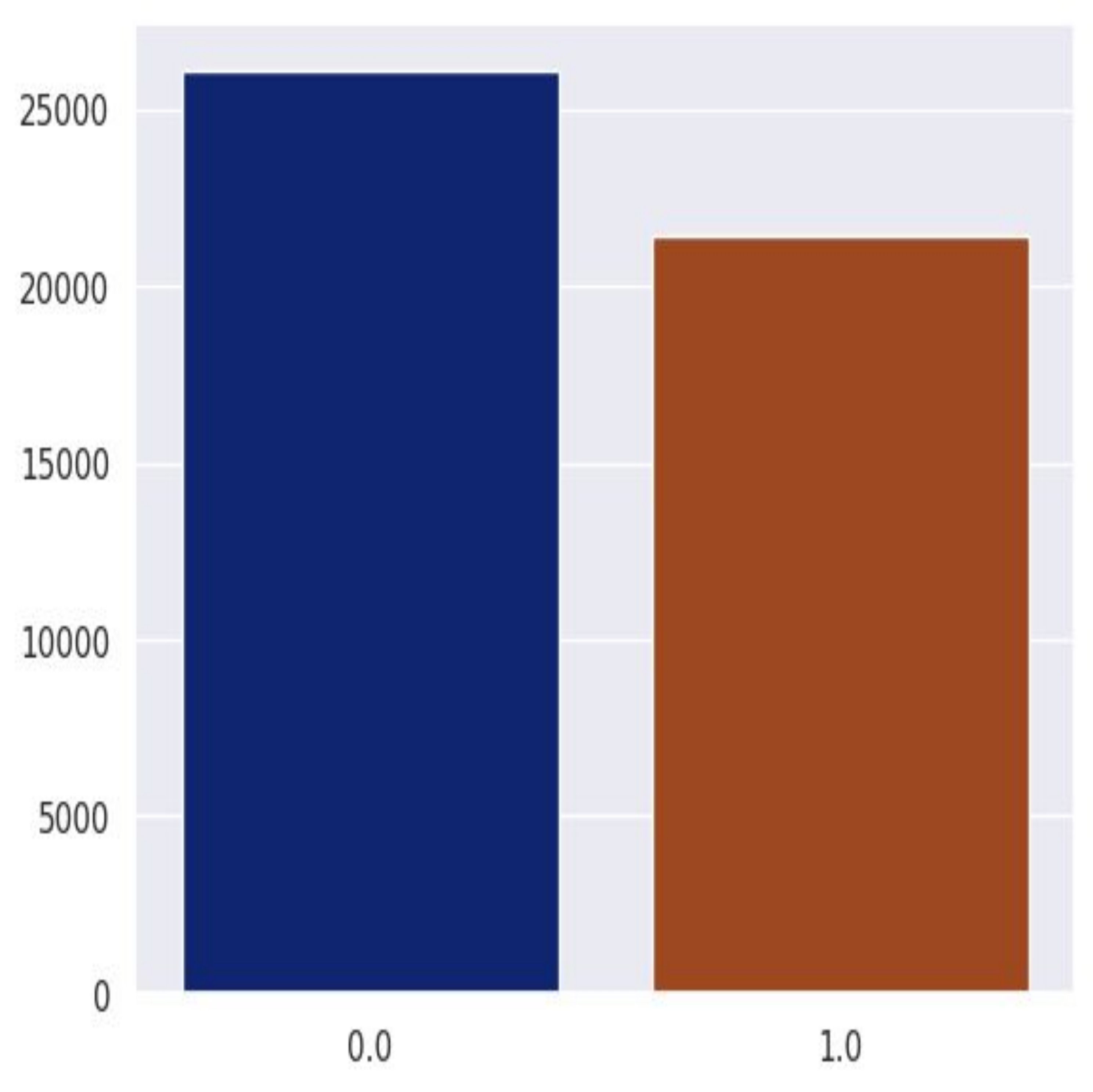
2

Record generation

- Introducing new record to decrease the unbalance.
- Cohere and Mistral-7b-v0.1 as generative models for text-generation.

MISTRAL AI_

cohere



DAIGT-V3

OUR NEW EXTENDED DATASET

- 26105 labeled as 0
- 21397 labeled as 1

Preprocessing

Irrelevant columns Removal

(prompt_name, source, RDizzl3_seven, id)

Stop word removal

- Removed commonly occurring words that contribute little semantic value to the text.

Lowercase Conversion

- All sentences were converted to lowercase.
- To standardize text and reduce dataset's dimensionality.

Other attempts

- Strip punctuation and lemmatization.
- During the training phase they were removed as they achieved poor results.

Attempted architecture

BERT-model

Why yes:

- Renowned for its contextual language understanding.
- Promising candidate for our task.

Why not:

- 512-token limit

LongFormer model

Why yes:

- It exceed the BERT model token limit, offering a max range of 4096 token.

Why not:

- Computational resource saturation

Final architecture

Recurrent Neural Network variant

Faced with the challenges posed by pre-trained models, we turned our

attention to building a custom solution.

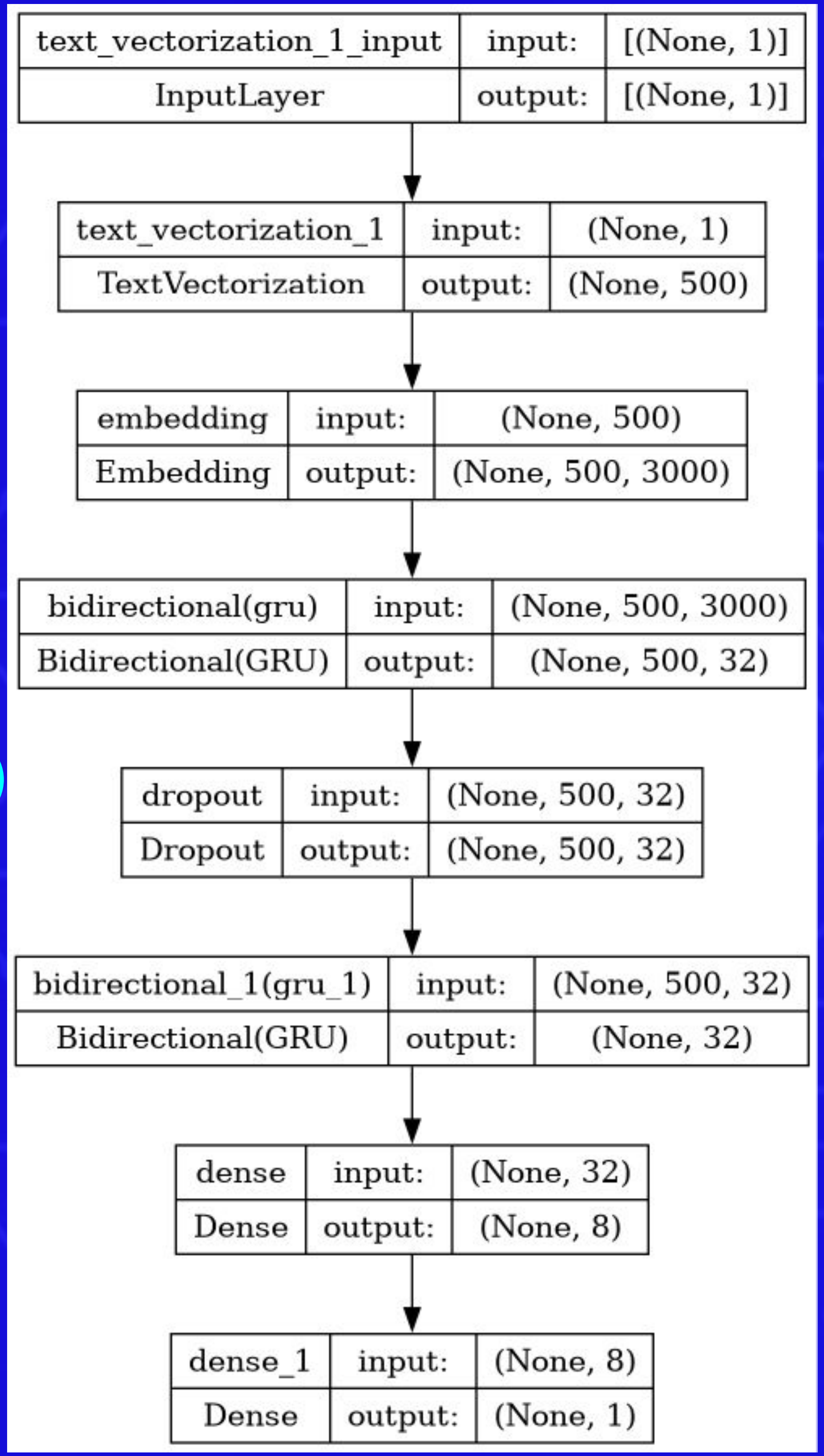
Why yes:

Well-suited for sequential data and exhibit the ability to capture dependencies over time. This approach offers more flexibility in handling datasets with varying lengths of text sequences

Final Model - Code

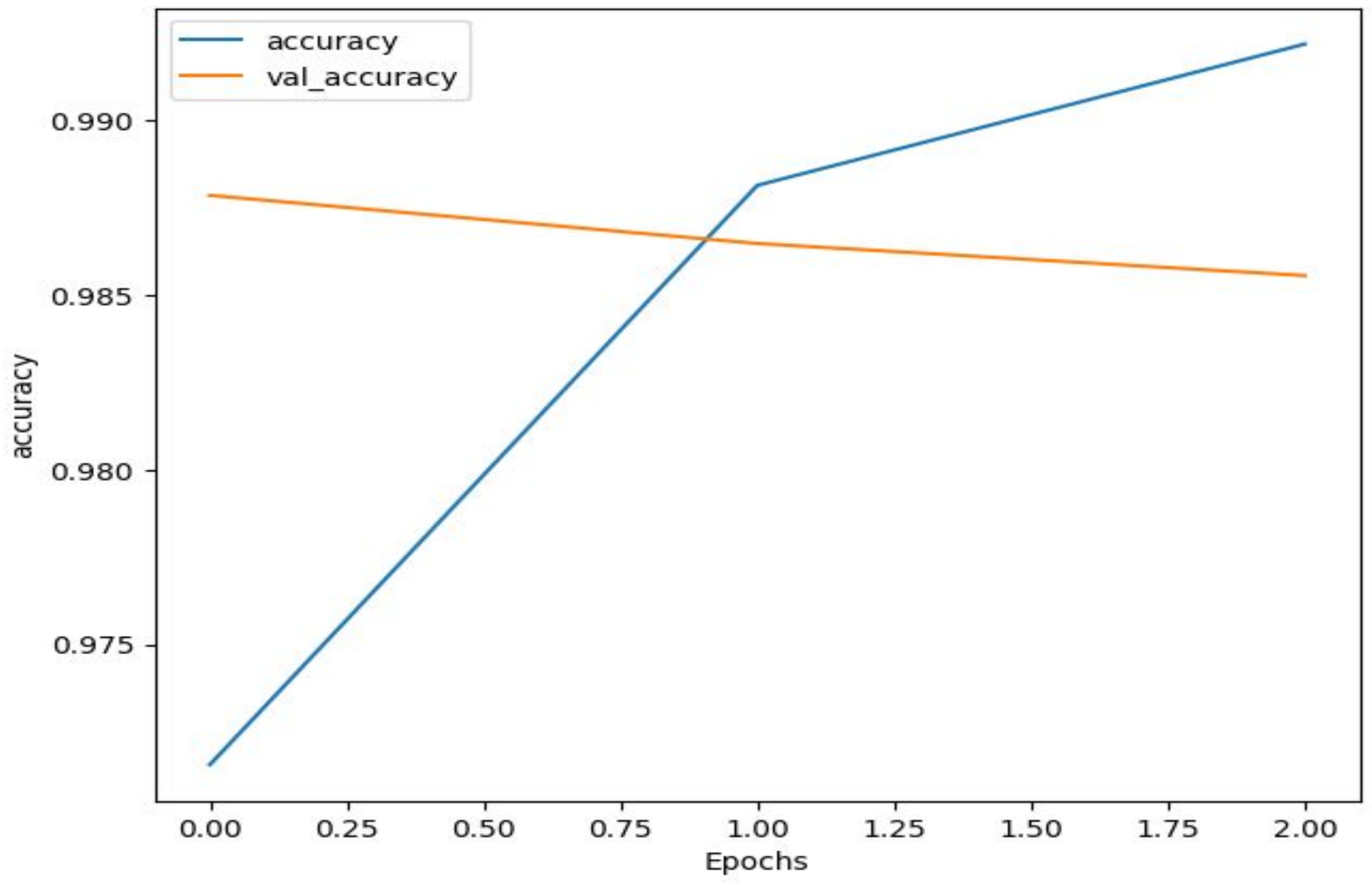
```
import tensorflow as tf

model = tf.keras.Sequential([
        tf.keras.layers.TextVectorization(
            output_sequence_length=500,
            standardize=None,
            max_tokens=8000),
        tf.keras.layers.Embedding(
            input_dim=len(encoder.get_vocabulary()),
            output_dim=3000,
            mask_zero=True),
        tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(
            units=16,
            return_sequences=True)),
        tf.keras.layers.Dropout(rate=0.5),
        tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(
            units=16)),
        tf.keras.layers.Dense(
            units=8,
            activation = 'relu',
            kernel_regularizer = regularizers.l1(0.1)),
        tf.keras.layers.Dense(1, activation='sigmoid')
    ])
```

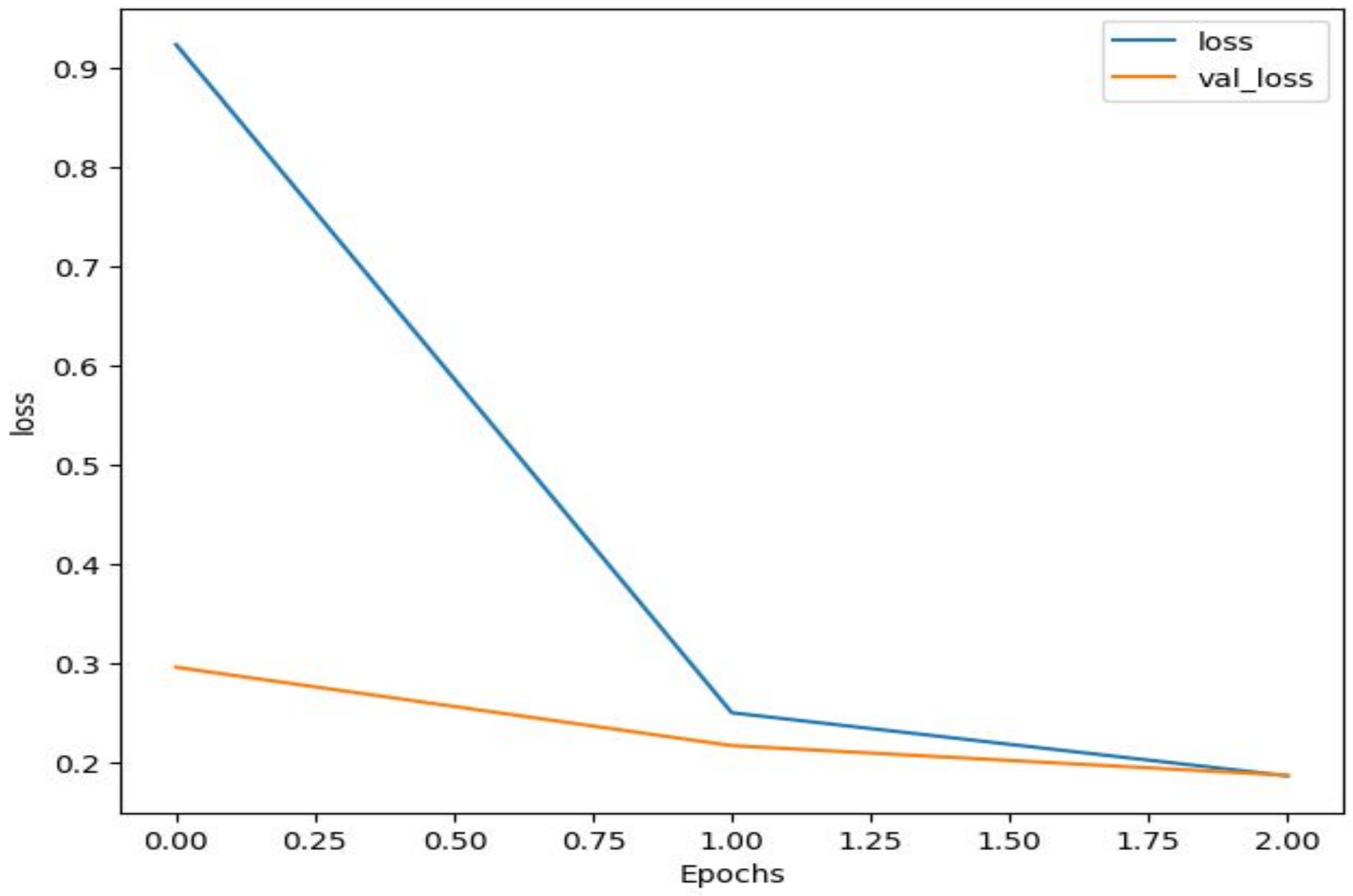
Final Model - Visual representation and description



- Text Vectorization Layer:
  - Tokenizes and vectorizes the input text.
  - Maps text into token sequences (appropriately applying truncation/padding techniques)
- Embedding Layer
  - Converts generated tokens into dense vectors.
    - Basically each token is represented by a long-vector.
- 1st Bidirectional LSTM
  - Returns the complete sequences instead of just the final output for each sequence.
    - It will provide an output of each time step in the input sequence
- Dropout Layer
- 2nd Bidirectional LSTM.
  - Does not return the complete sequences, but the final outputs.
- Dense Layer with ReLu Activation Function with L1 Regularization.
  - Adds a term to the loss function - It has property of making many weights of the model exactly to zero.
  - Prevents overfitting and improve model generalization
- Output Layer

Evaluation





Test Loss: 0.18071378767490387
Test Accuracy: 0.990139901638031

	precision	recall	f1-score	support
0	0.99	1.00	0.99	2589
1	0.99	0.98	0.99	2162
accuracy			0.99	4751
macro avg	0.99	0.99	0.99	4751
weighted avg	0.99	0.99	0.99	4751

****
0.9905283098295096

Final considerations

Problem: In the light of the results obtained, some considerations emerged on how the various technologies and resources obtained during the work could be used, in order to obtain a more larger, consistent, balanced and general dataset.

Possible solutions:

1. Use the found models for the text generation to generate a number of different prompts (~1000) and then use the same models to generate a number of records (~100) labeled with 1 (AI). After several considerations, we realized that we could use some generative models (pre-trained on datasets containing only texts written by humans) to generate the remaining records (~100), for each prompt, labeled with 0 (Human).

2. Generate more AI records with other generative models to balance at all (maybe with others models that respect our computational resources limitation).

Our results:

Competition Notebook	Run	Public Score	Best Score
LLM - Detect AI Generated Text	529.2s - GPU T4 ×2	0.855	0.855 V2

#	Team	Members	Score	Entries	Last	Join
2927	G15		0.855	10	11h	

Thanks for your attention!

- [GItHub Repository]