AI generated text Classifier using DistilBERT 🤗

Park Sang Uk (2024149038)

1 Introduction

In recent years, Large Language Models (LLMs) have demonstrated remarkable capabilities in generating human-like text[1]. While this advancement offers numerous benefits, it also raises significant concerns regarding the spread of misinformation, plagiarism, and the blurring of lines between human and machine creativity. Consequently, the ability to distinguish between human-written text and AI-generated text has become a critical task in the field of Natural Language Processing (NLP)[1].

This project focuses on the task of **Human vs. AI Story Classification**. The goal is to build an AI pipeline capable of taking a text input (a story) and correctly classifying its origin as either human-written (Label 0) or AI-generated (Label 1).

To address this problem, this study implements and compares two distinct approaches:

- A **Naïve Baseline**: A simple heuristic or rule-based method implemented to establish a minimum performance threshold.
- An **AI Pipeline**: A deep learning-based approach utilizing a pre-trained `distilbert-base-uncased` model. This model is fine-tuned on a balanced dataset of human and AI stories to capture subtle semantic and syntactic differences.

2 Task Definition

- **Task description:** The primary objective of this project is to perform binary text classification. The system receives a text passage (a story) as input and classifies its origin into one of two categories: *Human-written* or *AI-generated*.
- **Motivation:** As Large Language Models (LLMs) become increasingly sophisticated, distinguishing between human and machine-generated content has become a critical challenge. Developing an effective detection pipeline is essential for maintaining academic integrity (plagiarism detection), filtering synthetic misinformation, and verifying the authenticity of creative writing.
- **Input / Output:**
  - **Input:** A raw text string representing a complete story. This text is preprocessed and tokenized (truncated to a maximum length of 256 tokens) before being fed into the model.
  - **Output:** A binary label, where `0` indicates a Human-written story and `1` indicates an AI-generated story.
- **Success criteria:** The system is considered successful if it achieves a high **Accuracy** on the held-out test set. Specifically, the AI pipeline (DistilBERT) must significantly outperform both a random guess (50% accuracy) and the Naïve Baseline method established in this project.

3 Methods

This section includes both the naïve baseline and the improved AI pipeline.

3.1 Naïve Baseline

To establish a minimum performance threshold, we implemented a simple rule-based baseline before applying deep learning models.

Your Baseline

- **Method description:** The baseline utilizes a keyword-based heuristic approach targeting common formatting artifacts found in raw LLM outputs. It converts the input text to lowercase and scans for specific substrings such as markdown syntax (e.g., "**", "##") or structural keywords (e.g., "title"). If any of these tokens are detected, the system predicts the text as *AI-generated* (Label 1); otherwise, it defaults to *Human-written* (Label 0).
- **Why naïve:** This approach is considered naïve because it lacks any semantic or syntactic understanding of the language. It ignores crucial linguistic features such as perplexity, narrative flow, or vocabulary richness. Instead, it relies solely on surface-level formatting cues that are neither exclusive to AI nor guaranteed to appear in all AI-generated texts.
- **Likely failure modes:** The model is expected to fail in two primary scenarios:
  1. **Clean AI Text (False Negatives):** Sophisticated or post-processed AI outputs that do not contain markdown headers or bolding will be misclassified as Human.
  2. **Formatted Human Text (False Positives):** Human-written stories that use markdown for emphasis will be incorrectly flagged as AI.

3.2 AI Pipeline

Your Pipeline

- **Models used:** We utilized the `distilbert-base-uncased` model from the Hugging Face Transformers library[2].
- **Pipeline stages:** The pipeline consists of the following three distinct stages:
  1. **Preprocessing & Tokenization:** Raw text data is first cleaned by handling missing values (NaNs) and converting all inputs to strings. We utilize the `DistilBertTokenizer` to tokenize the text. To optimize memory usage, inputs are truncated and padded to a fixed maximum sequence length of 256 tokens. The dataset is split into Train (60%), Validation (20%), and Test (20%) sets.
  2. **Representation Learning:** The tokenized inputs (Input IDs and Attention Masks) are passed through the DistilBERT encoder layers. The model extracts contextualized embeddings, utilizing the hidden state of the special `[CLS]` token to represent the semantic content of the entire story.
  3. **Classification & Inference:** The aggregated sequence representation is fed into a dense classification layer (dropout + linear projection) to compute logits for the two classes (Human vs. AI). The model is fine-tuned end-to-end using the AdamW optimizer with a learning rate of 2e-5.
- **Design choices and justification:**
  - **Model Selection (DistilBERT):** We chose DistilBERT over the standard BERT-base or larger LLMs to satisfy the project's constraint of running on a single consumer GPU

(NVIDIA RTX 3050 Laptop GPU). DistilBERT offers an optimal balance between computational efficiency and classification accuracy[2].

  - **Sequence Length (256):** While BERT supports up to 512 tokens, we limited the maximum length to 256. This reduction significantly decreases the memory footprint and training time without losing critical context, as the introductory portion of a story often contains sufficient stylistic markers to distinguish AI generation.
  - **Fine-tuning Strategy:** Instead of using the model solely as a fixed feature extractor, we performed fine-tuning for 3 epochs. This allows the model to adapt its pre-trained weights specifically to the unique syntactic patterns of the "Human vs. AI" dataset, yielding higher accuracy than a simple classifier on frozen embeddings.

4 Experiments

4.1 Datasets

Your Dataset Description

- **Source:** Hugging Face Datasets (`gsingh1-py/train`)[3]. The data was originally provided in a CSV format containing prompts, human stories, and multiple AI-generated variations.
- **Total examples:** A total of **14,642** examples were used after constructing a balanced dataset.
  - The original data contained significantly more AI-generated text than human text.
  - To prevent class imbalance, we down-sampled the AI-generated stories to match the exact count of human stories (1:1 ratio), resulting in a perfectly balanced dataset.
- **Splits:** The dataset was randomly shuffled and split into three subsets using a fixed random seed (42) for reproducibility:
  - **Training Set (60%):** 8,784 examples (Used for model optimization).
  - **Validation Set (20%):** 2,929 examples (Used for monitoring loss and accuracy during training).
  - **Test Set (20%):** 2,929 examples (Held-out set used solely for final evaluation).
- **Preprocessing steps:**
  1. **Data Cleaning:** Missing values (NaNs) in the text column were replaced with empty strings, and all inputs were explicitly cast to string type to prevent tokenization errors.
  2. **Tokenization:** We used the `DistilBertTokenizer` with the `do_lower_case=True` option to normalize case.
  3. **Truncation & Padding:** To ensure computational efficiency on a single GPU, all sequences were truncated or padded to a fixed maximum length of **256 tokens**. This length was chosen to capture sufficient context while maintaining a low memory footprint.

4.2 Metrics

To quantitatively evaluate the performance of our models, we employed the following metrics:

- **Accuracy:** Since our dataset is perfectly balanced (50% Human, 50% AI), Accuracy is the most intuitive and fair metric to measure performance. It is defined as the ratio of correctly predicted observations to the total observations:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

  where TP is True Positive, TN is True Negative, FP is False Positive, and FN is False Negative.

- **Cross-Entropy Loss:** For the AI pipeline (DistilBERT), we also monitored the Cross-Entropy Loss during training and validation. This metric measures the difference between the predicted probability distribution and the true distribution, allowing us to assess how confident the model is in its predictions and ensuring the model is converging properly.

4.3 Results

We evaluated both the Naïve Baseline and the AI Pipeline on the held-out test set (2,929 examples). The quantitative results are summarized in Table 1.

Table 1: Performance comparison between the Naïve Baseline and the Fine-tuned DistilBERT Pipeline on the test set.

Method	**Accuracy**
Naïve Baseline	92.49%
AI Pipeline (DistilBERT)	**100.00%**

As shown in the table, the AI Pipeline achieved perfect accuracy (100.00%), completely correcting the errors made by the baseline. Notably, the baseline achieved a surprisingly high accuracy of 92.49%, indicating that heuristic rules based on formatting artifacts are quite effective for this dataset. However, the AI Pipeline's superior performance demonstrates that deep learning models can capture even the subtle nuances and edge cases that a strong heuristic approach misses.

Qualitative Analysis

To better understand where the heuristic baseline failed, we analyzed specific cases where the AI model outperformed the baseline.

- **Example 1 (Clean AI Text):**

  Text: "400000 A Tale of Three Housing Markets New York Mississippi Maryland three states one price point..."
  Analysis: The ground truth is **AI (1)**, but the baseline predicted **Human (0)**. The text appears to be a generated article title or summary but lacks the specific markdown keywords (e.g., "**", "title") required by the baseline's rules. The AI model correctly identified the synthetic nature of the text.

- **Example 2 (API Error Message):**

  Text: "Error: Error communicating with OpenAI: HTTPSConnectionPool(host='api.openai.com', port=443)..."
  Analysis: The ground truth is **AI (1)**, but the baseline predicted **Human (0)**. This is a specific artifact where the dataset contains an API error message instead of a story. Since it lacks markdown formatting, the baseline failed. The AI pipeline successfully learned that such error messages are associated with the AI class.

- **Example 3 (Formatted Human Text):**

  Text: "new video loaded:Errol Morris: 'Demon in the Freezer' transcript... MAIN TITLE CARD Demon in the Freezer..."
  Analysis: The ground truth is **Human (0)**, but the baseline predicted **AI (1)**. This error occurred because the baseline explicitly searches for the keyword "title" (intended to catch AI-generated markdown headers). The text contained the phrase "MAIN TITLE CARD", which triggered this simple keyword rule, leading to a false positive. The AI pipeline, however, correctly understood the context of a video transcript.

These examples highlight the limitation of the Naïve Baseline: it relies on superficial formatting cues. It fails when AI text is “clean” (Example 1) or contains system errors (Example 2), and it generates false positives on structured human text (Example 3). The AI Pipeline demonstrates a robust understanding of these diverse patterns.

5 Reflection and Limitations

Your Reflection

The transition from a Naïve Baseline to a fine-tuned AI pipeline yielded results that exceeded expectations, with DistilBERT achieving perfect accuracy (100.00%). This success demonstrates the power of transfer learning, as the model easily identified semantic patterns that my rule-based heuristic—which failed on structured human texts like interviews—could not capture. However, the process was not without challenges; I encountered significant issues with data quality (e.g., NaN values crashing the tokenizer) and memory constraints, which forced me to carefully truncate sequences to 256 tokens and optimize the batch size.

Regarding the evaluation, while accuracy was an appropriate metric for our perfectly balanced dataset, the perfect score raises a concern that the dataset might contain distinct artifacts (or “watermarks”) that make the task trivial for a transformer. If I had more time, I would test the model on “out-of-distribution” samples generated by newer models like GPT-4 to verify if the performance holds. Additionally, I would employ interpretability techniques (such as attention visualization) to ensure the model is learning genuine linguistic differences rather than overfitting to specific dataset quirks.

References

[1] Rajarshi Roy. A comprehensive dataset for human vs. ai generated text detection. *arXiv preprint arXiv:2510.22874*, 2025. URL https://arxiv.org/html/2510.22874v1.

[2] Amogh Lele. News classification using distilbert. https://www.kaggle.com/code/atechnohazard/news-classification-using-huggingface-distilbert, 2020. Kaggle Notebook.

[3] Gurpeet Singh. Human vs. ai generated text dataset. https://huggingface.co/datasets/gsingh1-py/train, 2024. Accessed: 2025-12-12.