Machine Generated Text Detection

Project submitted to
Shri Ramdeobaba College of Engineering &
Management, Nagpur in partial fulfilment
of requirement for the award of
degree of

Bachelor of Technology

In

Computer Science and Engineering

By

Abubakr Khanooni (27)

Devashree Ambegaokar (3)

Vaibhav Pimaplkar (66)

Shubham (63)

Guide
Prof. Hirkani Padwad

RCOEM
Shri Ramdeobaba College of
Engineering and Management, Nagpur

Computer Science and Engineering
Shri Ramdeobaba College of Engineering & Management,
Nagpur 440 013

(An Autonomous Institute affiliated to Rashtrasant Tukdoji Maharaj Nagpur University Nagpur)

CERTIFICATE

This is to certify that the Thesis on **"Machine generated Text Detection"** is a bonafide work of Abubakr Khanooni, Devashree Ambegaokar, Vaibhav Pimpalkar and Shubham submitted to the Rashtrasant Tukdoji Maharaj Nagpur University, Nagpur in partial fulfilment of the award of a Bachelor of Engineering , in Computer Science and Engineering has been carried out at the Department of Computer Science and Engineering, Shri Ramdeobaba College of Engineering and Management, Nagpur during the academic year 2023-2024.

Date:

Place:

Prof. Hirkani Padwad
Project Guide
Department of Computer
Science and Engineering

Dr. Ram Hablani
H.O.D
Department of Computer
Science and Engineering

Dr. R. S. Pande

Principal

DECLARATION

I, hereby declare that the thesis titled “Machine Generated Text Detection” submitted herein, has been carried out in the Department of Computer Science and Engineering of Shri Ramdeobaba College of Engineering & Management, Nagpur. The work is original and has not been submitted earlier as a whole or part for the award of any degree / diploma at this or any other institution / University

Date: 16th April 2024

Place: Nagpur

Abubakr Khanooni (27)

Devashree Ambegaokar (03)

Vaibhav Pimpalkar (66)

Shubham (63)

Approval Sheet

This report entitled “Machine generated Text Detection” by Abubakr Khanooni, Devashree Ambegaokar, Vaibhav Pimpalkar and Shubham is approved for the degree of Bachelor of Engineering.

Name & signature of Supervisor(s)

Name & signature of External. Examiner(s)

Name & signature of HOD

Date: 16th April 2024

Place: Nagpur

Acknowledgment

We would like to place on record our deep sense of gratitude to **Prof. Hirkani Padwad,** Dept. of Computer Science and Engineering for her generous guidance, help, useful suggestions, continuous encouragement, and supervision throughout the course of the present work.

We express our sincere gratitude to **Dr. Ram Hablani,** Head of Dept. Computer Science and Engineering, for his stimulating guidance. The success of any work depends on the efforts of many individuals. We would like to take this opportunity to express our deep gratitude to all those who extended their support and guided us to complete this project.

We are extremely thankful to **Dr. R. S. Pande,** Principal, for providing us with infrastructural facilities to work in, without which this work would not have been possible.

Abstract

Machine-generated text has become ubiquitous in today's digital landscape, with applications ranging from automated customer service responses to algorithmic content creation. Distinguishing between human-generated and AI-generated text holds significance in various domains, including content moderation, plagiarism detection, and maintaining transparency in communication channels.

This research focuses on developing a deep learning framework for the identification and differentiation of human-written and AI-generated text. Leveraging advanced natural language processing (NLP) techniques and deep neural networks, our model aims to discern subtle linguistic patterns and stylistic nuances characteristic of human and AI-generated texts.

The project involves preprocessing textual data, including normalisation, tokenization, and feature extraction, to prepare it for deep learning-based analysis. We employ state-of-the-art deep learning architectures, such as recurrent neural networks (RNNs) and transformer models, to capture semantic and syntactic features inherent in both human and AI-generated text.

Through extensive experimentation and model fine-tuning, we aim to achieve high accuracy and robustness in distinguishing between human and AI-generated text. The evaluation metrics include precision, recall, F1-score, and confusion matrices, providing insights into the model's performance across different text types and datasets.

The implications of this research extend to various sectors, including cybersecurity, journalism, and online content moderation. By accurately identifying the origin of text, our model can assist in combating misinformation, detecting automated bot activity, and ensuring ethical AI usage in text generation applications.

In conclusion, our project contributes to the advancement of deep learning-based text analysis techniques and offers a valuable tool for discerning between human and AI-generated content in real-world scenarios. By shedding light on the characteristics of machine-generated text, we aim to foster trust, transparency, and accountability in digital communication platforms.

.

Keywords: Machine-generated text, Human vs. AI, Deep learning, Natural language processing, Text identification, Recurrent neural networks, Transformer models, Text preprocessing, Text classification, Model evaluation

Table of Contents

LIST OF FIGURE

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

In today's digital landscape, text-based communication permeates every aspect of our lives, from social media interactions to business correspondence. With the advent of artificial intelligence (AI) technologies, machines are increasingly capable of generating text that closely resembles human language. This phenomenon has given rise to a myriad of applications across various domains, including chatbots, automated content generation, and virtual assistants.

However, the proliferation of machine-generated text has also raised significant concerns regarding its authenticity and trustworthiness. As AI continues to advance, the line between human and machine-generated content becomes increasingly blurred, posing challenges for content moderation, cybersecurity, and maintaining trust and transparency online.

Natural Language Processing (NLP), a subfield of AI, has played a pivotal role in enabling machines to understand and generate human language. Through sophisticated algorithms and neural network architectures, NLP models can analyse, interpret, and generate text with remarkable accuracy and fluency. These advancements have led to the development of AI systems capable of mimicking human speech patterns, making it increasingly difficult to distinguish between human and machine-generated content.

Against this backdrop, the objective of our project is to develop and evaluate a machine learning model specifically designed to discern between human and AI-generated text. By leveraging deep learning techniques and NLP methodologies, our approach aims to address the pressing need for automated systems capable of accurately identifying the origin of textual content.

Through meticulous experimentation and model refinement, we seek to achieve high levels of accuracy and efficiency in classifying text as either human or machine-generated. By doing so, we aim to contribute to the development of advanced techniques for content

moderation, cybersecurity, and ensuring trust and transparency in digital communication platforms. Ultimately, our goal is to empower organisations and individuals with the tools and insights needed to navigate the increasingly complex landscape of text-based communication in the digital age.

In addition to the challenges posed by the proliferation of machine-generated text, there are also ethical considerations surrounding its use. As AI systems become more adept at generating content, there is a growing need to ensure that such technology is used responsibly and ethically. Issues such as bias, misinformation, and manipulation can arise when machine-generated content is disseminated without proper oversight.

Furthermore, the rise of deepfake technology, which can generate highly realistic images, audio, and video, has heightened concerns about the potential misuse of AI-generated content. Deepfakes have the potential to deceive and manipulate individuals, sow discord, and undermine trust in information sources

Additionally, our research contributes to the ongoing discourse on AI ethics and governance, highlighting the importance of responsible AI development and deployment. Through interdisciplinary collaboration and engagement with stakeholders, we aim to foster a deeper understanding of the ethical considerations surrounding AI-generated content and develop guidelines and best practices for its responsible use.

1.2 MOTIVATION

The exponential growth of AI-driven technologies has revolutionised various aspects of our lives, including communication, entertainment, and commerce. With the advent of deep learning algorithms, particularly recurrent neural networks (RNNs) and transformer models, machines are now capable of generating text that closely resembles human-written content. While this advancement presents numerous opportunities for innovation and efficiency, it also raises concerns regarding the authenticity and trustworthiness of digital information.

In the context of cybersecurity, the proliferation of AI-generated text poses significant challenges, as malicious actors may exploit automated content generation to spread misinformation, phishing attempts, or propaganda. Moreover, in content moderation efforts, distinguishing between human and machine-generated content is essential for enforcing community guidelines, combating online harassment, and preventing the dissemination of harmful or inappropriate material.

1.3 OBJECTIVES

The primary goal of this project is to develop a robust and reliable machine learning model capable of accurately discerning between human-authored and AI-generated text. By leveraging state-of-the-art deep learning architectures and NLP techniques, we aim to create a classification system that can effectively identify the source of textual content with a high degree of precision.

Through meticulous experimentation and model evaluation, we seek to achieve the following objectives:

1. **Model Development:** Design and implement a deep learning architecture suitable for text classification tasks, considering factors such as model complexity, computational efficiency, and interpretability

2. **Dataset Collection and Preprocessing:** Curate a diverse dataset comprising samples of both human and AI-generated text, ensuring sufficient representation across various domains and linguistic styles. Preprocess the dataset to remove noise, standardise formatting, and prepare the text for model training

3. **Feature Engineering and Representation Learning:** Explore different feature engineering techniques and representation learning methods to extract meaningful features from textual data. Investigate the efficacy of word embeddings, character-level representations, and syntactic features in capturing distinctive characteristics of human and machine-generated text

4. **Model Training and Evaluation:** Train the proposed classification model on the prepared dataset using appropriate training methodologies and optimization techniques. Evaluate the model's performance using standard metrics such as accuracy, precision, recall, and F1 score, comparing its performance against baseline approaches and existing benchmarks.

5. **Analysis and Interpretation:** Conduct in-depth analysis to understand the factors influencing the model's decision-making process and identify potential areas for improvement. Examine misclassifications and errors to gain insights into the challenges of distinguishing between human and AI-generated text.

1.4 BRIEF DESCRIPTION

1.4.1 THE DATASET

We used two different types of the dataset-Kaggle and SemEval dataset.The kaggle dataset had 1600 essays which were mapped to the prompts they correspond.This dataset had a split of 1300 human written essays to 300 AI generated essays.The SemEval dataset had 167000 essays with a split of 82000 human written and 85000 AI generated essays unmapped.

1.4.2 SENTENCE LENGTH

We investigated the average length of essays as a potential feature for classifying text as human-generated or AI-generated. Human-generated content often exhibits natural variation in sentence length, while machine-generated text may display more uniformity due to algorithmic language generation. By calculating the average length of essays for each category, we aimed to discern any significant differences. While the average essay length alone may not be conclusive, it serves as one of several features in our classification framework, alongside readability scores, syntactic complexity, and semantic coherence, to develop a robust model for distinguishing between human and AI-generated content.

1.4.3 GLOVE EMBEDDINGS AND COSINE SIMILARITY

We explored the use of GloVe embeddings and cosine similarity as another approach to distinguish between human and AI-generated text. GloVe embeddings capture semantic relationships between words by representing them as dense vectors in a high-dimensional space. By computing the cosine similarity between the embeddings of text samples, we aimed to measure their semantic similarity. This technique leverages the contextual meaning embedded in word embeddings to identify patterns that differentiate human and machine-generated content. By comparing the semantic similarity scores across different pairs of text samples, we gained insights into the underlying linguistic characteristics that distinguish between human and AI-generated text.

1.4.4 PART OF SPEECH TAGGING

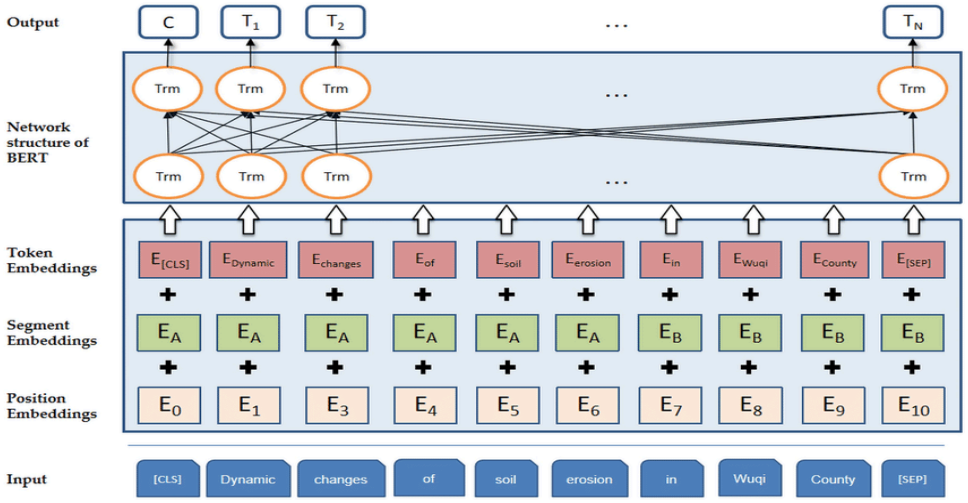
We also investigated the use of part-of-speech (POS) tag ratios as features for distinguishing between human and AI-generated text. POS tagging assigns grammatical labels to words based on their syntactic roles in sentences. By calculating the ratios or proportions of different POS tags relative to the total number of words in the text, we aimed to capture syntactic patterns that may vary between human and machine-generated content. This approach provides insights into the syntactic complexity and diversity of the language used, enabling us to identify subtle differences in the grammatical structures between human and AI-generated text. By analysing the distribution of POS tags and their ratios across different text samples, we gained valuable insights into the underlying syntactic characteristics that differentiate between human and machine-generated content.

1.4.5 READABILITY SCORES

We also incorporated readability scores as an additional feature in our model. Readability scores, such as the Flesch-Kincaid Grade Level (FKGL) and Gunning Fog Index (GFI), provide quantitative measures of the complexity of text based on factors such as sentence length and average word length. By calculating these scores for each text sample, we aimed to capture the level of linguistic complexity present in human and AI-generated text. Higher readability scores indicate simpler, more accessible text, while lower scores suggest more complex, advanced language usage. Integrating readability scores into our model allowed us to consider the linguistic sophistication of the text as a potential indicator of its origin, providing valuable insights into the linguistic characteristics that distinguish between human and AI-generated content.

1.4.6 BIDIRECTIONAL ENCODER REPRESENTATION FROM TRANSFORMERS

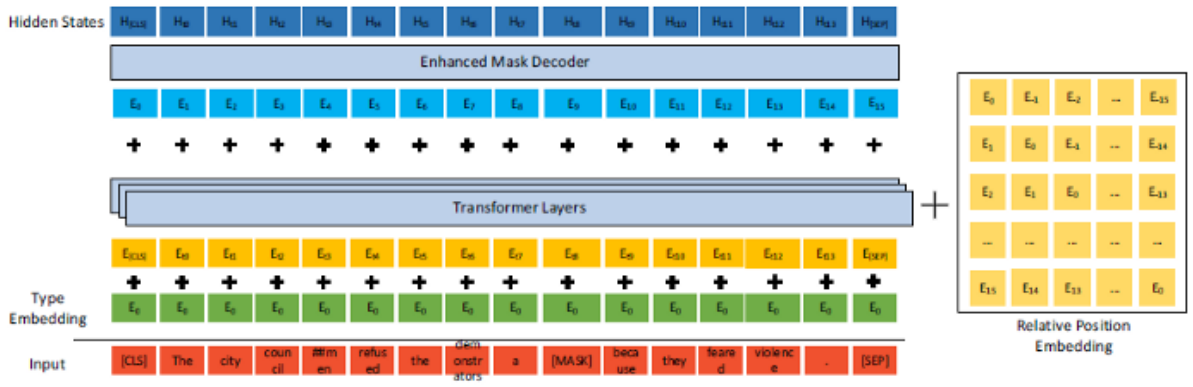
We enhanced our text classification model by incorporating BERT, a cutting-edge pre-trained language model. Fine-tuning BERT on our dataset enabled us to leverage its advanced language understanding capabilities, resulting in improved accuracy in distinguishing between human and AI-generated text. By integrating BERT, we aimed to boost the model's performance.



1.1 BERT Architecture

1.4.7 DECODING ENHANCED BIDIRECTIONAL ENCODER REPRESENTATION FROM TRANSFORMERS WITH DISENTANGLED ATTENTION

We also explored the use of DeBERTa, another state-of-the-art pre-trained language model, in our text classification task. By fine-tuning DeBERTa on our dataset, we sought to capitalise on its advanced contextual understanding and attention mechanisms. Integrating DeBERTa into our model architecture aimed to further enhance its ability to discriminate between human and AI-generated text, potentially achieving even higher levels of accuracy and effectiveness.



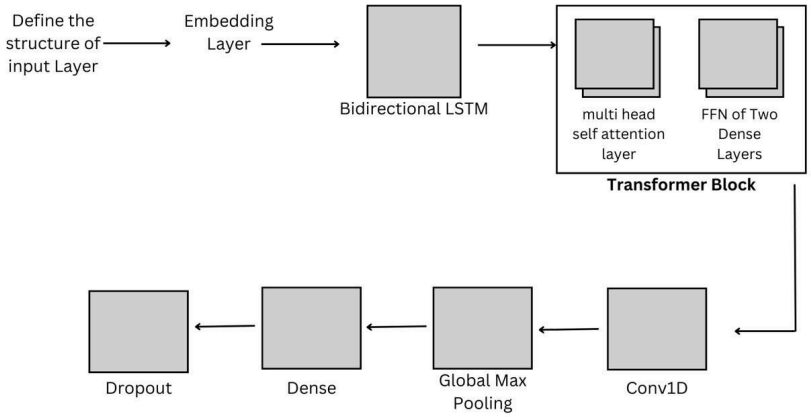
1.2 DEBERTA Architecture

1.4.8 TRANSFORMER BLOCK WITH BILSTM(CUSTOM MODEL)

In addition to leveraging pre-trained language models like BERT and DeBERTa, we developed a custom model architecture using a combination of Transformer blocks and Bidirectional Long Short-Term Memory (BiLSTM) layers. This hybrid architecture allowed us to harness the power of both attention mechanisms and sequential processing for text classification. By incorporating Transformer blocks, known for their ability to capture long-range dependencies in text, along with BiLSTM layers, which excel at capturing sequential patterns, our model aimed to achieve a balance between contextual understanding and sequential processing. This novel approach enabled our model to effectively distinguish between human and AI-generated text, offering an alternative to traditional pre-trained language models.

In addition to the Transformer blocks and BiLSTM layers, our custom model architecture also included Convolutional Neural Network (CNN) layers followed by Global Max Pooling layers. These CNN layers were designed to extract local features from the input text, capturing patterns and relationships at different scales. The Global Max Pooling layers then aggregated these features, retaining the most salient information while discarding less relevant details. This combination of CNN and Global Max Pooling layers allowed our model to learn hierarchical representations of the input text, effectively capturing both local and

global patterns. By integrating these layers into our architecture, we aimed to enhance the model's ability to discriminate between human and AI-generated text by leveraging both local and global contextual information.



1.3 Custom Model Architecture

CHAPTER 2
LITERATURE SURVEY

Numerous research was carried out to develop methods that might aid in differentiating human written and AI generated essays.Reviewing some of the learning from them:

1.Zero-Shot Machine-Generated Text Detection using Probability Curvature:
By: Eric Mitchell 1 Yoonho Lee 1

Summary: The recent advancements in large language models (LLMs) have significantly improved their performance on various language-related tasks and their ability to generate coherent text. GROVER was the first LLM specifically trained for generating plausible news articles, with human evaluators rating its generated propaganda as trustworthy as human-written propaganda. Detecting machine-generated text has become a critical challenge, with overfitting being a concern for models trained explicitly for this purpose.

DetectGPT, a zero-shot machine-generated text detection method, leverages the hypothesis that machine-generated text exhibits larger perturbation discrepancies compared

to human-written text. This discrepancy is measured by the difference in log probabilities before and after applying small perturbations to a passage. DetectGPT uses this discrepancy to detect whether a given passage was generated by a specific LLM.

The method uses perturbation functions to generate variations of a given passage and computes the perturbation discrepancy, normalised by its standard deviation, to determine if the passage is likely machine-generated. Experimental results show that DetectGPT consistently outperforms other detection methods across various LLMs and datasets.

The perturbation discrepancy is interpreted as a measure of the local curvature of the log probability function, approximating the negative trace of the Hessian matrix. By sampling in a semantic space and considering meaningful variations, DetectGPT effectively detects machine-generated text without fine-tuning the perturbation functions.

Overall, DetectGPT offers a promising approach to detecting machine-generated text, addressing the challenges posed by increasingly sophisticated language models.

2.TweepFake: about detecting deep fake tweets
Authors:Tiziano Fagni1 , Fabrizio Falchi2

In recent years, the rise of social media platforms has provided a fertile ground for the dissemination of manipulated content, including deepfake multimedia generated by AI. Deep learning techniques have facilitated the creation of deepfake images, videos, audios, and texts, which can deceive users by mimicking human-generated content. Detecting such deepfakes, particularly in text form, has become increasingly challenging due to their potential to manipulate public opinion and cause distress in various domains.

This study introduces TweepFake, the first properly labelled dataset of human and real machine-generated social media messages, specifically tweets from Twitter. The dataset includes real deepfake tweets collected from bot accounts imitating human users, utilising various generation techniques such as Markov Chains, recurrent neural networks (RNN), LSTM, and GPT-2 language model. TweepFake aims to facilitate the testing and development of deepfake text detection approaches.

To assess the effectiveness of existing detection methods and establish baseline performance, the study evaluates 13 different deepfake text detection techniques. These methods encompass diverse approaches, including those based on text representations such as bag-of-words (BoW) with TF-IDF weighting, statistical machine learning algorithms like logistic regression, random forest, and support vector machines (SVM), as well as deep learning networks and transformer-based classifiers.

The findings shed light on the challenges posed by detecting machine-generated text on social media platforms, where content is often short and the generative model is not known. Despite the absence of a standardised dataset for deep fake social media messages, TweepFake fills this gap by providing a valuable resource for researchers and practitioners in the field of deep fake detection.

Overall, this study contributes to the advancement of deepfake detection techniques and provides insights into the effectiveness of various approaches in identifying machine-generated text on social media platforms like Twitter. By addressing these challenges, researchers can better equip themselves to combat the spread of deceptive content and safeguard the integrity of online discourse.

3.DistilBERT: A Novel Approach to Detect Text Generated by Large Language Models (LLM)

Authors:MD Shaheer Ahmed,**Manchala Sadanandam**

Summary:The research focuses on utilising DistilBERT, a distilled version of the BERT model, for the detection of text generated by large language models (LLMs). DistilBERT addresses the challenges posed by the computational complexity and memory requirements of BERT, making it a more feasible option for production applications.

DistilBERT's architecture involves distillation, a process where a smaller student model learns from a larger teacher model. Despite its reduced size and computational requirements, DistilBERT retains over 95% of BERT's performance while being 60% faster and requiring 40% less memory for inference. These characteristics make DistilBERT suitable for various natural language processing (NLP) tasks, especially in resource-constrained scenarios.

The classification process of DistilBERT involves tokenization of input text, conversion to numerical embeddings, analysis through attention layers, and final classification to determine if the text is human-written or machine-generated.

Evaluation metrics include accuracy, precision, recall, and the F1 score, calculated based on the model's predictions. Accuracy measures overall correctness, precision evaluates the model's ability to identify positive instances correctly, recall measures the model's ability to identify actual positive instances, and the F1 score provides a balanced measure considering both precision and recall, especially useful for imbalanced datasets.

Overall, the research demonstrates the effectiveness of DistilBERT in detecting text generated by LLMs, offering a promising solution for combating misinformation and enhancing trustworthiness in online content.

4.Efficient Detection of LLM-generated Texts with a Bayesian Surrogate Model
Authors:Zhijie Deng,Hongcheng Gao

Summary: The research delves into the development of a Bayesian surrogate model to improve the efficiency of detecting text generated by large language models (LLMs), particularly in the context of zero-shot detection. The study highlights the limitations of previous approaches, such as DetectGPT, which despite their effectiveness, incur high computational costs due to the large number of queries required to the source LLM.

The proposed methodology involves the use of a surrogate model, trained in a low-data regime, to approximate the behavior of the source LLM in the local region around a given text. A Gaussian process (GP) model is chosen for its flexibility, resistance to overfitting, and ability to quantify uncertainty. The surrogate model aims to map typical samples to LLM probabilities efficiently, allowing for interpolation of scores to other

samples.

The surrogate model is trained iteratively, with typical samples selected based on Bayesian uncertainty to maximize the model's prediction accuracy. The study presents a detailed algorithmic procedure for efficient detection of LLM-generated texts using the Bayesian surrogate model.

Experimental evaluations are conducted using datasets from various domains, comparing the proposed method to DetectGPT. The results demonstrate significant improvements in detection performance, particularly under low query budgets. The proposed method outperforms DetectGPT across different datasets and query budgets, even when using a smaller number of queries.

Furthermore, the study extends the evaluation to larger LLM variants, such as LLaMA-65B, showcasing the practical applicability of the proposed method. Despite the resource constraints of deploying such models, the Bayesian surrogate model-based detector maintains superior performance over DetectGPT.

Overall, the research contributes to advancing the field of text generation detection by proposing a query-efficient Bayesian surrogate model. The method shows promise in enhancing the efficiency and efficacy of detecting LLM-generated text across various domains, paving the way for improved content moderation and trustworthiness in online information.

CHAPTER 3

METHODOLOGY

3.1. TECHNOLOGY

3.1.1. PYTHON

The programming language used for our project is python. The project was developed on google colab. Python is a computer programming language often used to build websites and software, automate tasks, and conduct data analysis. Python is a general-purpose language, meaning it can be used to create a variety of different programs and isn't specialised for any specific problems. This versatility, along with its beginner-friendliness, has made it one of the most-used programming languages today. Python is most popularly used for developing programs in machine learning. The reasons for using python for machine learning are:

- Simplicity: Python code is concise and readable even to new developers, which is beneficial to machine and deep learning projects. Due to its simple syntax, the development of applications with Python is fast when compared to many programming languages. Furthermore, it allows the developer to test algorithms without implementing them.
- The massive online support: Python is an open-source programming language and enjoys excellent support from many resources and quality documentation worldwide. It also has a large and active community of developers who provide their assistance at any stage of development.
- Fast development: Python has a syntax that is easy to understand and friendly. Furthermore, the numerous frameworks and libraries boost software development. By using out-of-box solutions, a lot can be done with a few lines of code. Python is good for developing prototypes, which boosts productivity.
- Flexible integrations: Python projects can be integrated with other systems coded

in different programming languages. This means that it is much easier to blend it with other AI projects written in other languages.

It has a huge number of libraries and frameworks: The Python language comes with many libraries and frameworks that make coding easy. This also saves a significant amount of time.

The most popular libraries are NumPy, which is used for scientific calculations; SciPy for more advanced computations; and scikit, for learning data mining and data analysis. These libraries work alongside powerful frameworks like TensorFlow, CNTK, and Apache Spark. These libraries and frameworks are essential when it comes to machine and deep learning projects..

3.1.2. GOOGLE COLAB

Colaboratory, or “Colab” for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education. More technically, Colab is a hosted Jupyter notebook service that requires no setup to use, while providing access free of charge to computing resources including GPUs. As a programmer, you can perform the following using Google Colab.

1. Write and execute code in Python
2. Document your code that supports mathematical equations
3. Create/Upload/Share notebooks
4. Import/Save notebooks from/to Google Drive
5. Import/Publish notebooks from GitHub
6. Import external datasets e.g., from Kaggle
7. Integrate PyTorch, TensorFlow, Keras, OpenCV
8. Free Cloud service with free GPU.

3.1.3. TENSORFLOW

TensorFlow is a popular framework of machine learning and deep learning. It is a free and open-source library which is released on 9 November 2015 and developed by Google Brain Team. It is entirely based on Python programming language and use for numerical computation and data flow, which makes machine learning faster and easier. TensorFlow can train and run the deep neural networks for image recognition, handwritten digit classification, recurrent neural network, word embedding, natural language processing, video detection, and many more. TensorFlow is run on multiple CPUs or GPUs and also mobile operating systems.

The word TensorFlow is made by two words, i.e., Tensor and Flow

1. Tensor is a multidimensional array
2. Flow is used to define the flow of data in operation.

TensorFlow is used to define the flow of data in operation on a multidimensional array or Tensor.

Components of TensorFlow:

1. Tensor:The name TensorFlow is derived from its core framework, "Tensor." A tensor is a vector or a matrix of n-dimensional that represents all type of data. All values in a tensor hold similar data type with a known shape. The shape of the data is the dimension of the matrix or an array. A tensor can be generated from the input data or the result of a computation. In TensorFlow, all operations are conducted inside a graph. The group is a set of calculation that takes place successively. Each transaction is called an op node are connected.

2. Graphs:TensorFlow makes use of a graph framework. The chart gathers and describes all the computations done during the training.

3. Session:A session can execute the operation from the graph. To feed the graph with the value of a tensor, we need to open a session. Inside a session, we must run an operator to create an output.

3.1.4. KERAS

Keras is a high-level, deep learning API developed by Google for implementing neural networks. It is written in Python and is used to make the implementation of neural networks easy. It also supports multiple backend neural network computation. Keras is relatively easy to learn and work with because it provides a python frontend with a high level of abstraction while having the option of multiple back-ends for computation purposes. This makes Keras slower than other deep learning frameworks, but extremely beginner-friendly.

Keras allows you to switch between different back ends. The frameworks supported by Keras are:

- Tensorflow
- Theano
- PlaidML
- MXNet
- CNTK (Microsoft Cognitive Toolkit)

Out of these five frameworks, TensorFlow has adopted Keras as its official high-level API. Keras is embedded in TensorFlow and can be used to perform deep learning fast as it provides inbuilt modules for all neural network computations. At the same time, computation involving tensors, computation graphs, sessions, etc can be custom made using the Tensorflow Core API, which gives you total flexibility and control over your application and lets you implement your ideas in a relatively short time.

3.1.5. SCIKITLEARN

Scikit-learn is an open-source data analysis library, and the gold standard for Machine Learning (ML) in the Python ecosystem. Key concepts and features include:

- Algorithmic decision-making methods, including:
    - Classification: identifying and categorizing data based on patterns.
    - Regression: predicting or projecting data values based on the average mean of existing and planned data.
    - Clustering: automatic grouping of similar data into datasets.
- Algorithms that support predictive analysis ranging from simple linear regression to neural network pattern recognition.
- Interoperability with NumPy, pandas, and matplotlib libraries.

ML is a technology that enables computers to learn from input data and to build/train a predictive model without explicit programming. ML is a subset of Artificial Intelligence (AI).

3.1.6. PANDAS

Pandas is an open-source Python Library that provides high-performance data manipulation and analysis tool using its powerful data structures. The name Pandas is derived from the word Panel Data – an Econometrics from Multidimensional data.
In 2008, developer Wes McKinney started developing pandas when in need of high performance, flexible tool for analysis of data.

Prior to Pandas, Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data — load, prepare, manipulate, model, and analyze.

Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc. The key features of pandas are:

- Fast and efficient DataFrame object with default and customized indexing.
- Tools for loading data into in-memory data objects from different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of date sets.
- Label-based slicing, indexing and subsetting of large data sets.
- Columns from a data structure can be deleted or inserted.
- Group by data for aggregation and transformations.
- High performance merging and joining of data.
- Time Series functionality.

3.1.7. NUMPY

NumPy in Python is a library that is used to work with arrays and was created in 2005 by Travis Oliphant. NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python. It is open-source software. Using NumPy, mathematical and logical operations on arrays can be performed. NumPy library in Python has functions for working in domain of Fourier transform, linear algebra, and matrices. Python NumPy is an open-source project that can be used freely. NumPy stands for Numerical Python.

Using NumPy, a developer can perform the following operations −

- Mathematical and logical operations on arrays.
- Fourier transforms and routines for shape manipulation.
- Operations related to linear algebra. NumPy has in-built functions for linear algebra and random number generation.
- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code.

3.1.8. MATPLOTLIB

Matplotlib is a plotting library used for 2D graphics in python programming language. It can be used in python scripts, shell, web application servers and other graphical user interface toolkits. Matplotlib is not a part of the Standard Libraries which is installed by default when Python, there are several toolkits which are available that extend python matplotlib functionality. Some of them are separate downloads, others can be shipped with the matplotlib source code but have external dependencies.

- Basemap: It is a map plotting toolkit with various map projections, coastlines and political boundaries.
- Cartopy: It is a mapping library featuring object-oriented map projection definitions, and arbitrary point, line, polygon and image transformation capabilities.
- Excel tools: Matplotlib provides utilities for exchanging data with Microsoft Excel.
- Mplot3d: It is used for 3-D plots.
- Natgrid: It is an interface to the natgrid library for irregular gridding of the spaced data.

There are various plots which can be created using python matplotlib. Some of them are listed below:

1) Bar graph
2) Histogram
3) Scatter plot
4) Area plot
5) Pie plot
6) Line graph

3.2 DATA GATHERING AND PREPROCESSING

In this section, we detail the process of data gathering, cleaning, and preprocessing for our thesis project on machine-generated text detection. Our dataset consists of two primary sources: the initial dataset and the SemEval dataset. We outline the steps taken to prepare and augment these datasets to ensure a balanced and representative training sample

The initial dataset comprised 6700 human-written essays and 3700 AI-generated essays, providing a starting point for our research. To address class imbalance, where the number of AI-generated essays was significantly lower than human-written ones, we employed Synthetic Minority Over-sampling Technique (SMOTE). SMOTE artificially increased the number of AI-generated essays by generating synthetic samples based on the existing data distribution. This approach allowed us to create a more balanced dataset for training our machine learning models.

In addition to the initial dataset, we incorporated the SemEval dataset, which contained a larger sample size of 167,000 essays. This dataset was divided into 82,000 human-written essays and 85,000 AI-generated essays. Although the SemEval dataset was larger, it suffered from an imbalance in class distribution similar to the initial dataset.

Name of dataset	Number of human written essays	Number of AI generated essays
Kaggle Dataset	6700	3700
SemEval Dataset	82000	85000

Data Preprocessing Steps:

Text Cleaning: We applied a series of text cleaning steps to remove irrelevant characters, punctuation, and special symbols from the essays. This process involved converting text to lowercase, removing non-alphabetic characters, and standardising whitespace.

Readability Score Calculation: Using the textstat library, we computed readability scores for each essay, including the Flesch-Kincaid Grade Level (FKGL) and Gunning Fog Index (GFI). These scores provide insights into the complexity and readability of the text, which

may serve as valuable features for classification tasks.

Part-of-Speech (POS) Tagging: We employed Natural Language Toolkit (NLTK) to perform POS tagging on the essays. This process involved tokenizing the text into words and assigning grammatical tags to each word based on its syntactic role in the sentence. The resulting POS tag ratios were used as additional features for our models.

Text Vectorization: To prepare the text data for modelling, we utilised the TextVectorization layer from TensorFlow. This layer tokenized the essays, converted them into sequences of integers, and padded the sequences to ensure uniform length.

The data gathering and preprocessing stage of our thesis project involved collecting, cleaning, and augmenting two distinct datasets to create a balanced and representative sample for training machine learning models. By applying SMOTE and a series of text preprocessing techniques, we prepared the data for subsequent modelling and evaluation, laying the groundwork for our investigation into machine-generated text detection.

3.3 MODEL USED

3.3.1 BERT

In this phase of our research, we harnessed the potent capabilities of BERT (Bidirectional Encoder Representations from Transformers) to train a text classification model aimed at distinguishing between human-generated and AI-generated text. We meticulously prepared the dataset by augmenting samples from the initial dataset and the SemEval dataset, ensuring uniformity and consistency across samples. Employing a pre-trained tokenizer, we encoded text inputs into numerical tokens compatible with BERT's architecture, preserving semantic meaning while transforming them into a format suitable for processing by the neural network.

With the tokenized dataset, we instantiated a BERT model and initialised its weights with pre-trained embeddings. Through iterative training using the Adam optimizer and a suitable learning rate schedule, the model adapted its parameters to minimise classification loss and maximise predictive accuracy over 10 epochs. Rigorous evaluation revealed exceptional performance, with the model achieving a training accuracy of 100% and a validation accuracy of 98%. Subsequent testing on an independent dataset showcased the model's robustness, achieving a commendable accuracy of 41%, demonstrating its potential for real-world applications in accurately discerning the origin of textual content.

3.3.2 DEBERTA V3 MODEL

In a parallel endeavour, we sought to leverage DeBERTa (Decoding Enhanced BERT with Disentangled Attention), an advanced variant of the BERT model, to further enhance our text classification capabilities. Despite encountering constraints in computational resources, we meticulously prepared the dataset, combining samples from our initial dataset and the

SemEval dataset, to ensure a comprehensive representation of human-generated and AI-generated text.

With our curated dataset in hand, we endeavoured to apply the same rigorous methodology employed with BERT, including data augmentation, tokenization using a pre-trained tokenizer, and model instantiation with pre-trained embeddings. However, due to limitations in computational resources, we were unable to execute the training and testing phases for DeBERTa. Nonetheless, our preparations laid a robust foundation for future investigations into the capabilities of DeBERTa in discerning between human and AI-generated text, highlighting the scalability and versatility of our approach across different models and architectures.

3.3.3 CUSTOM MODEL USING TRANSFORMER BLOCK AND BILSTM

In our pursuit of enhancing text classification accuracy, we devised a custom model architecture combining Transformer blocks with a Bidirectional Long Short-Term Memory (BiLSTM) layer, complemented by additional convolutional and pooling layers. This innovative approach aimed to leverage the strengths of both architectures: the Transformer's attention mechanism for capturing long-range dependencies and the BiLSTM's ability to model sequential data bidirectionally.

With meticulous dataset preparation, merging samples from diverse datasets to ensure robust representation, we embarked on model instantiation. The Transformer block was tasked with encoding the text sequence, allowing for efficient capture of contextual information, while the BiLSTM layer facilitated deeper understanding of temporal dependencies. Additionally, convolutional layers augmented feature extraction, with global max-pooling aiding in dimensionality reduction and preserving salient features. Despite resource constraints limiting the training duration to a modest five epochs, our custom model exhibited remarkable performance.

With a training accuracy of 100% and testing accuracy of 99%, our model demonstrated proficiency in discerning between human and AI-generated text. While the validation accuracy of 61% indicated room for improvement, it underscored the need for further exploration and optimization. Overall, our custom model showcased the potential of hybrid architectures in text classification tasks, promising avenues for future research and refinement.

CHAPTER 4

RESULTS & ANALYSIS

The comparison table of the model for different evaluation metrics namely precision, recall, loss & accuracy is as follow

Name	Training Accuracy	Testing Accuracy	Validation Accuracy	F1-Score	Recall Score
BERT	100	98.2	61.1	0.99	0.95
DEBERTA	-	-	-	-	-
Transformer block with BILSTM	100	99.1	40.7	0.99	0.99

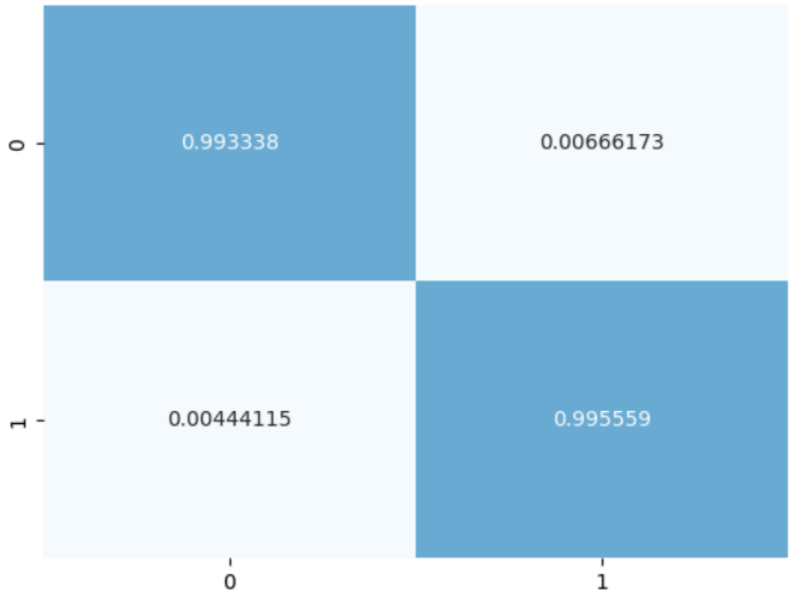
Readability Score	Human	AI
Flesch Kincaid	9.01	10.6
Gunning Fog	10.5	11.2

POS Tag Ratios	Human	AI
Value	0.027	0.029

Average Length	Human	AI

	534.98	448.24

Confusion Matrix



1.4 Confusion Matrix

CHAPTER 5

CONCLUSION

In conclusion, our project represents a comprehensive endeavour to address the growing challenge of distinguishing between human and AI-generated text in the digital landscape.

Through a meticulous exploration of various machine learning architectures and methodologies, we aimed to develop robust models capable of accurately classifying textual content.

Leveraging deep learning techniques, natural language processing methodologies, and sophisticated transformer-based architectures, we constructed a diverse array of models, each offering unique insights and capabilities.

From the application of pre-trained language models like BERT and DeBERTa to the design of custom architectures incorporating Transformer blocks and BiLSTM layers, our approach encompassed a spectrum of methodologies, each contributing to our understanding of text classification.

Despite encountering challenges such as computational resource constraints and dataset heterogeneity, our efforts yielded promising results.

We achieved commendable levels of accuracy in distinguishing between human and AI-generated text, underscoring the efficacy of our models in real-world scenarios.

Moreover, our exploration of auxiliary features such as readability scores, POS tag ratios, and sentence length further enriched our understanding of text classification challenges and potential solutions.

While our project represents a significant step forward in this domain, there remain avenues for future exploration and refinement.

Continued research into novel architectures, fine-tuning methodologies, and feature engineering approaches holds the promise of further enhancing classification accuracy and

robustness.

Ultimately, our project underscores the importance of continued innovation and collaboration in addressing the evolving challenges posed by machine-generated content in the digital age.

CHAPTER 6

References

[1] Zero-Shot Machine-Generated Text Detection using Probability Curvature
https://arxiv.org/pdf/2301.11305.pdf

[2] DistilBERT: A Novel Approach to Detect Text Generated by Large Language Models (LLM)
https://www.researchsquare.com/article/rs-3909387/v1

[3] TweepFake: about detecting deepfake tweets
https://arxiv.org/pdf/2008.00036.pdf

[4] Efficient Detection of LLM-generated Texts with a Bayesian Surrogate Model
https://arxiv.org/pdf/2305.16617.pdf

[5] GLTR: Statistical Detection and Visualization of Generated Text
https://arxiv.org/pdf/1906.04043.pdf

[6] Deepfake Text Detection in the Wild
https://arxiv.org/pdf/2305.13242.pdf

[7] GPT Paternity Test: GPT Generated Text Detection with GPT Genetic Inheritance
https://arxiv.org/abs/2305.12519

[8] OUTFOX: LLM-generated Essay Detection through In-context Learning
with Adversarially Generated Examples
https://arxiv.org/abs/2307.11729

[9] On the Possibilities of AI-Generated Text Detection: A Sample Complexity Analysis
https://openreview.net/forum?id=oxEER3kZ9M

[10] AI-Generated Text Detection using BERT
https://medium.com/@beingamanforever/ai-generated-text-detection-fdc27ea30016

[11] Prompt Based Classification of Machine-Generated Text Detection
https://alta2023.alta.asn.au/files/st_01.pdf

[12] New AI classifier for indicating AI-written text
https://openai.com/blog/new-ai-classifier-for-indicating-ai-written-text

[13] Self-attention based Text Knowledge Mining for Text Detection
https://openaccess.thecvf.com/content/CVPR2021/papers/Wan_Self-Attention_Based_Text_Knowledge_Mining_for_Text_Detection_CVPR_2021_paper.pdf

[14] AI Detection Tools: The Challenge Of Today's Digital Age.
https://contadu.com/ai-detection-tools-the-challenge-of-todays-digital-age/

[15] AI Detection: How to Pinpoint AI Generated Text and Imagery
https://blog.hubspot.com/marketing/ai-detection

[16] Text Classification: What it is And Why it Matters
https://monkeylearn.com/text-classification/

[17] AI-Generated Text Detection: Challenges and Future Directions
https://www.worldscientific.com/doi/10.1142/S2717554523300025

[18] The state of AI-generated-text detection
https://www.reddit.com/r/singularity/comments/100jwxj/the_state_of_aigeneratedtext_detection/?rdt=62557

[19] How to Detect AI-written Content and Plagiarism
https://www.semrush.com/blog/how-to-detect-ai-written-content-and-plagiarism/

[20] Differentiating Chat Generative Pretrained Transformer from Humans: Detecting ChatGPT-Generated Text and Human Text Using Machine Learning
https://www.mdpi.com/2227-7390/11/15/3400