ROBUST CLASSIFICATION OF HUMAN WRITTEN VS LLM GENERATED ESSAYS USING ADVERSARIAL TRAINING

A PROJECT REPORT

Submitted by

YOGESHWARAN R

(2023246036)

A report for the phase-I of the project
submitted to the Faculty of

INFORMATION AND COMMUNICATION ENGINEERING

in partial fulfillment
for the award of the degree
of

MASTER OF TECHNOLOGY
in
INFORMATION TECHNOLOGY

DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY
COLLEGE OF ENGINEERING, GUINDY
ANNA UNIVERSITY
CHENNAI 600 025

JANUARY 2025

ANNA UNIVERSITY

CHENNAI - 600 025

BONAFIDE CERTIFICATE

Certified that this project report titled **ROBUST CLASSIFICATION OF HUMAN WRITTEN VS LLM GENERATED ESSAYS USING ADVERSARIAL TRAINING** is the bonafide work of **YOGESHWARAN R (2023246036)** who carried out project work under my supervision. Certified further that to the best of my knowledge and belief, the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or an award was conferred on an earlier occasion on this or any other candidate.

PLACE:
DATE:

Dr.ABIRAMI MURUGAPPAN
PROFESSOR
PROJECT GUIDE
DEPARTMENT OF IST, CEG
ANNA UNIVERSITY
CHENNAI 600025

COUNTERSIGNED

DR. S. SWAMYNATHAN
HEAD OF THE DEPARTMENT
DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY
COLLEGE OF ENGINEERING, GUINDY
ANNA UNIVERSITY
CHENNAI 600025

ABSTRACT

The rapid advancement of large language models (LLMs) has made it increasingly difficult to distinguish between human-written and LLM-generated content. This poses significant challenges in domains like academic integrity, content moderation, and the ethical use of AI in creative fields. As LLMs generate increasingly sophisticated text, developing robust methods for accurately identifying machine-generated content becomes crucial, particularly in adversarial settings like paraphrasing or word substitutions.

This project presents a novel approach to classify texts as human-written or LLM-generated by combining semantic embeddings with linguistic features. Embeddings are extracted using a fine-tuned autoencoder based on the roberta-base architecture, leveraging the [CLS] token for global semantic information. In parallel, linguistic features such as readability metrics (e.g., Flesch-Kincaid Grade Level), stylometric patterns (e.g., n-grams, part-of-speech tagging), and entity grid-based coherence analysis are computed to capture textual structure and style. These features complement the embeddings, addressing limitations of purely semantic approaches.

The hybrid feature set, integrating embeddings and linguistic insights, is fed into a feed-forward neural network for binary classification. The network includes ReLU activation, dropout regularization, and batch normalization to enhance learning and mitigate overfitting. Performance metrics such as accuracy and F1 score demonstrate the effectiveness of the proposed framework in identifying LLM-generated content, even under adversarial manipulations, highlighting its potential for real-world applications.

ABSTRACT TAMIL

ACKNOWLEDGEMENT

I would like to express my deep sense of appreciation and gratitude to my project guide **Dr. ABIRAMI MURUGAPPAN,** Professor, Department of Information Science and Technology, College of Engineering, Guindy, Anna University, Chennai for her invaluable support, supervision, guidance, useful suggestions and encouragement throughout this phase. Her moral support and continuous guidance enabled me to complete my work successfully.

I thank **Dr. S. SWAMYNATHAN,** Professor and Head of the Department of Information Science and Technology, College of Engineering, Guindy, Anna University, Chennai for the prompt and limitless help in providing the excellent computing facilities to do the project and to prepare the thesis.

We are grateful to the project committee members **Dr. S. SRIDHAR,** Professor,**Dr. G. GEETHA,** Associate Professor and **Dr. D. NARASHIMAN,** Teaching Fellow, Department of Information Science and Technology, College of Engineering, Guindy, Anna University, Chennai for their review and valuable guidance throughout the course of our project.

We also thank the faculty member and nonteaching staff members of the Department of Information Science and Technology at College of Engineering Guindy, Anna University, Chennai for their valuable support throughout the course of our project work.

YOGESHWARAN R

(2023246036)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

The rapid advancements in Natural Language Processing (NLP) and Deep Learning have fundamentally transformed the ways machines generate and interpret human language. Large Language Models (LLMs), such as Generative Pre-trained Transformer (GPT) and its derivatives, have reached unprecedented capabilities in generating coherent, contextually relevant, and human-like text. These models leverage transformer architectures to process and understand textual data with remarkable accuracy, enabling a wide range of applications across industries. However, their growing ubiquity also brings concerns about potential misuse, especially when it becomes increasingly difficult to distinguish between human-written and LLM-generated content. This challenge has significant implications for academic integrity, journalism, publishing, and other domains where authenticity is paramount.Early approaches to text classification primarily relied on traditional machine learning techniques, such as decision trees, logistic regression, and support vector machines (SVMs). These methods, while effective to an extent, often depended on surface-level textual metrics such as word frequency and n-grams, which failed to capture the deeper semantic nuances of the text. The emergence of transformer-based architectures, such as BERT and GPT, has revolutionized the ability to process text at a deeper level. By leveraging attention mechanisms, these models can extract intricate patterns and dependencies within the text, making them indispensable for modern text classification tasks.In this project, we introduce a robust classification framework for distinguishing human-written from LLM-generated essays. Our approach integrates transformer-based

embeddings from a fine-tuned autoencoder with linguistically motivated features such as readability measures, stylometric analysis, lexical diversity, and entity grids. By combining semantic and linguistic insights, this framework captures subtle differences in textual structure and style, paving the way for a reliable and accurate classification system. Such a system addresses the pressing need for solutions that ensure the integrity and authenticity of written content in an era of rapidly evolving generative AI.

1.2 PROBLEM STATEMENT

The rapid advancement of generative AI and natural language processing has led to the development of LLMs capable of producing text that is nearly indistinguishable from human-written content. Models like GPT, BERT, and their successors can generate syntactically valid, contextually coherent, and highly relevant text across a variety of domains. While these capabilities have unlocked immense potential, they also pose significant challenges. Educational institutions, for instance, struggle to identify AI-generated assignments, raising concerns about academic integrity. Similarly, industries such as publishing and journalism face difficulties in verifying the authenticity of articles, potentially questioning trust and credibility.Moreover, adversarial techniques like paraphrasing, fine-grained word substitutions, and stylistic adjustments further complicate the task of distinguishing between human-written and LLM-generated text. Existing classification methods often fall short when faced with such manipulations, underscoring the need for a robust system that not only identifies the origin of the text but also withstands adversarial attacks. The ability to accurately classify text in these challenging scenarios is critical for ensuring the ethical use of AI in creative and professional settings.The ability to accurately classify text in these challenging scenarios is critical for ensuring the ethical use of AI in creative and professional settings. Developing such a system requires leveraging advanced techniques that combine semantic understanding

with linguistic analysis, paving the way for reliable classification even under adversarial conditions.

1.3 OBJECTIVE

The objective of this project is to develop a robust and accurate classification system that can distinguish between human-written text and LLM-generated text. With the rapid advancements of generative AI models, the need for such classification systems is critical, especially in areas such as academic integrity, content moderation, and automated text generation. This project aims to achieve the following primary objectives:

- Develop a deep learning-based classification system that accurately distinguishes between human-written text and LLM-generated text.
- Extract linguistic features, including text statistics, readability metrics, stylometry, lexical diversity, and entity grids, to enhance the model's ability to discern subtle differences between human and LLM-generated text.
- Combine transformer-generated embeddings with linguistic features to improve classification accuracy and provide deeper insights into textual differences.
- Evaluate the model's performance and effectiveness in distinguishing between human-written and LLM-generated text.

The ultimate aim of this project is to develop a classification system capable of accurately identifying human and LLM-generated text, enhancing the understanding of textual differences, and providing valuable insights into content creation using generative AI.

1.4 SOLUTION OVERVIEW

The need for robust text classification systems stems from the increasing prevalence of generative AI in content creation. As LLMs become more advanced, their ability to mimic human writing continues to blur the lines between machine-generated and human-authored content. This project addresses this critical challenge by proposing a comprehensive pipeline that integrates transformer-based embeddings with linguistically motivated features. The framework not only improves classification accuracy but also provides insights into the stylistic and structural differences between human-written and LLM-generated text.

1.5 ORGANIZATION OF THE REPORT

This report is organized into 6 chapters, describing each part of the project with detailed illustrations and system design diagrams.:

- **Chapter 2** discusses the existing system and various methods required for the proposed systems.
- **Chapter 3** discusses the various concepts used in the proposed system along with overall system architecture.
- **Chapter 4** discusses the implementation of the proposed framework, including preprocessing steps, algorithmic details, and system configuration.
- **Chapter 5** discusses the results of this thesis with related output figures and it also elaborates the intermediate results of the individual modules implementation
- **Chapter 6** discusses the conclusion and future work.

CHAPTER 2

LITERATURE SURVEY/RELATED WORK

2.1 OVERVIEW

This section reviews significant research efforts in detecting AI-generated text, focusing on methodologies that incorporate stylometric features, neural network approaches, zero-shot generalization capabilities, and adversarial robustness. These techniques aim to address the challenges posed by large language models (LLMs) and ensure reliable detection systems in real-world scenarios.

2.2 EXISTING SYSTEMS

2.2.1 DETECTION OF AI GENERATED TEXTS

Detection methods for AI-generated text can be broadly categorized into stylometric feature-based approaches and neural network-based approaches.

2.2.1.1 SYTLOMETRIC FEATURE-BASED APPROACHES

Stylometric features have been extensively explored to differentiate between human and machine-generated text. Kumarage et al.[**?**]utilized stylometric signals for detecting AI-generated content in Twitter timelines. Their model integrates features related to linguistic diversity, punctuation,

and phraseology, coupled with embeddings from pre-trained language models (PLMs). The method incorporates change point detection to identify shifts in content generation but struggles with attributing tweets to specific AI generators and with larger, more sophisticated language models. Markov et al.[**?**] proposed a holistic framework for undesired content detection, emphasizing stylometric features alongside metadata. The study showcased the potential of combining linguistic and contextual cues to detect generative content but noted challenges in cross-domain generalization, particularly for long-form text such as essays. Bajaj et al.[**?**] explored the use of adversarially generated text samples to test the robustness of emotion detection systems. Their findings revealed vulnerabilities in stylometric-based detection when subjected to adversarial perturbations, thereby emphasizing the need for more robust stylometric models.

2.2.1.2 NEURAL NETWORK-BASED APPROACHES

Neural network-based approaches have demonstrated superior performance in detecting AI-generated text. Crothers et al.[**?**] analyzed the robustness of neural and statistical features in detecting outputs from generative transformers. Their findings highlighted that while neural features often outperform statistical ones in accuracy, statistical features exhibit greater robustness against adversarial perturbations. The paper proposed combining these feature types to enhance detection reliability. Koike et al.[**?**] introduced the OUTFOX framework, leveraging in-context learning with adversarially generated examples to enhance detection performance. Their results demonstrated significant improvements in identifying essays produced by large language models (LLMs) under adversarial conditions, though the model's generalizability to unseen generators remains a concern.Huang et al.[**?**] evaluated the robustness of existing detectors against adversarial perturbations.

Their work underlined the vulnerabilities of current neural models when faced with paraphrasing and synonym substitution, suggesting the need for adversarial training to improve detector reliability.

2.2.2 ZERO-SHOT GENERALIZATION IN DETECTION

Zero-shot generalization is a critical area of research in AI-generated text detection. Pu et al.[**?**] investigated the zero-shot generalization capabilities of detectors trained on various LLMs. Their study revealed that detectors trained on medium-sized models generalize well to larger versions of the same generator but struggle with entirely unseen or non-public generators like GPT-4. This highlights the importance of designing detectors that are robust to unseen text generation techniques and capable of generalizing across diverse domains.Krishna et al.[**?**] explored retrieval-based defenses as an alternative to standard detection methods. By leveraging a database of human-written and AI-generated texts, their system retrieved similar samples to enhance detection accuracy. This retrieval-based approach showed promise in zero-shot settings but faced computational challenges, particularly for large-scale applications.

2.2.3 ADVERSARIAL SAMPLE GENERATION

The generation of adversarial samples has been extensively studied to improve the robustness of detection systems. Zhou et al.[**?**] proposed a rule-based adversarial sample generator (RAG) that introduces controlled perturbations to input text while preserving semantic and syntactic properties. Their findings demonstrated improved classification accuracy in low-data scenarios, suggesting the utility of such methods in enhancing robustness against adversarial attacks.Peng et al.[**?**] conducted an adversarial evaluation of AI-generated student essay detection systems. Their work revealed significant

performance drops when adversarial samples were introduced, emphasizing the need for robust adversarial training. The study also identified key weaknesses in current detectors, such as over-reliance on shallow linguistic cues. Koike et al.[**?**] extended their OUTFOX framework to include adversarially generated examples during training. This approach improved detector resilience to sophisticated attacks, highlighting the effectiveness of adversarial augmentation in real-world scenarios.

2.3 SUMMARY

The literature review highlights significant advancements in the detection of AI-generated text, encompassing stylometric feature-based approaches, neural network techniques, and zero-shot generalization strategies. Stylometric methods enhance detection accuracy through linguistic features but face limitations regarding complex AI models. Neural approaches have shown superior performance but require additional robustness against adversarial attacks. The exploration of zero-shot capabilities raises important questions about generalizability in diverse settings, while adversarial sample generation techniques offer promising avenues for bolstering detection system resilience.This review underlines the necessity for robust classification system that can accurately classify even when the detectors are faced to adversarial attacks.

CHAPTER 3

SYSTEM DESIGN

This chapter describes the overall architecture of the proposed system, detailing the workflow and individual modules that form the foundation of the Human-Written Text vs. LLM-Generated Text classification framework. The proposed system incorporates embeddings extracted via an autoencoder, combined with linguistically motivated features, to produce robust classifications of text authorship.The modular design ensures flexibility, allowing integration with new features or datasets while maintaining high accuracy and robustness against adversarial manipulations. Each module contributes to achieving precise classifications by addressing specific challenges, such as semantic similarity and stylistic variations.

3.1 SYSTEM ARCHITECTURE

The overall system architecture is depicted in Fig.3.1, illustrating the workflow from input preprocessing to the final classification output. The framework consists of three key modules: Embedding Generation, Linguistic Feature Extraction, and the Classification Module. Each module plays a vital role in extracting, combining, and processing different text representations for accurate Human written vs LLM generated text classification.The architecture begins by feeding text data into the Embedding Generation Module, which leverages a fine-tuned autoencoder to extract contextual embeddings. Next, the Linguistic Feature Extraction Module computes a range of stylistic and readability features from the input text, representing its surface-level attributes. These outputs are combined and passed to the Classification Module, where

a feed-forward neural network processes the concatenated feature vector to predict whether the text is human-written or machine-generated.

Modules used in this architecture are listed below:

- Data Loader Module
- Embedding Generation Module
- Linguistic Feature Extraction Module
- Classification Module

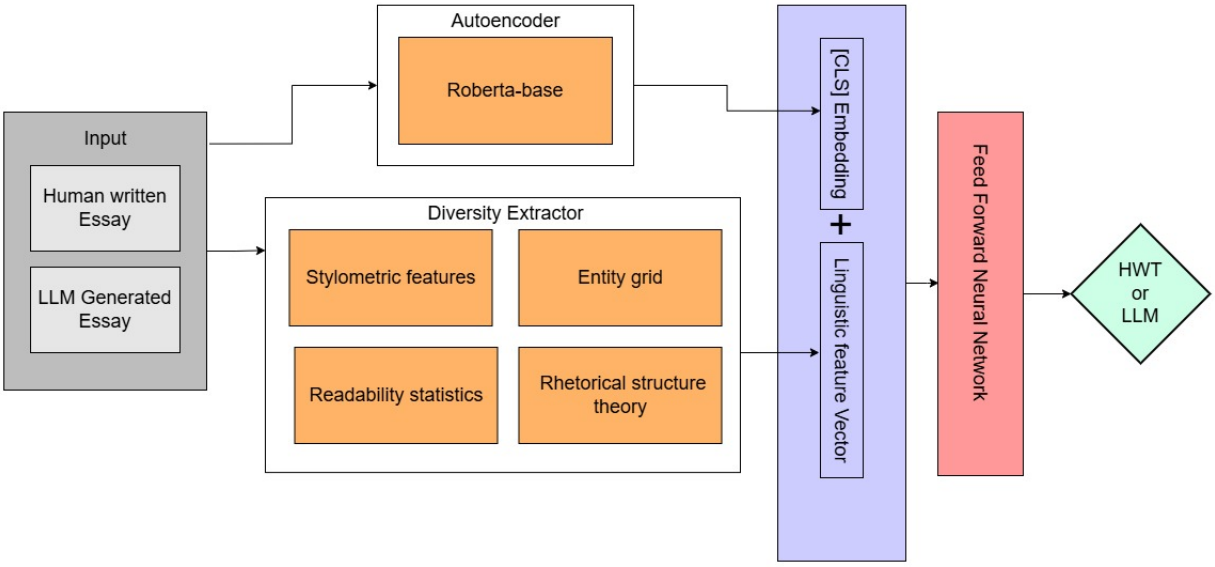

Figure 3.1: LLM Generated Essay detection architecture

3.2 DATA LOADER

The input module forms the foundation of the system, preparing raw text data for downstream processing. The text is cleaned and preprocessed to ensure compatibility with the embedding generation and feature extraction pipelines. The input consists of plain text files, which undergo tokenization and normalization to align with the requirements of the subsequent modules.

- **Text preprocessing:** The text is tokenized into subword units using a WordPiece tokenizer, which splits text into smaller components for better representation. Punctuation and special characters are removed, and sequences are padded to a fixed length to maintain uniformity across the dataset.
- **Normalization:** The text is converted to lowercase, and whitespace is standardized to ensure consistency in embeddings and feature calculations.

The preprocessed text forms the input to both the Embedding Generation Module and the Linguistic Feature Extraction Module, ensuring seamless integration between different parts of the system.

The dataset used in this project consists of two parts: a training dataset and a test dataset. The training dataset comprises 28,800 rows, including both native-speaker student-written essays and LLM-generated essays. The test dataset consists of 1,000 rows, with a similar mix of student-written and LLM-generated essays. This dataset was obtained from the paper [**?**]

This dataset serves as the foundation for training and evaluating the system, ensuring that it captures the nuanced differences between human-written and LLM-generated text, which is critical for accurate detection.

3.3 EMBEDDING GENERATION

The Embedding Generation module serves as a critical component of the proposed architecture, leveraging the fine-tuned RoBERTa-base model to generate semantic and contextual representations of input text. RoBERTa, a robust transformer-based language model, builds on the foundational BERT

architecture by introducing optimizations such as dynamic masking, removal of next-sentence prediction, and pretraining on larger datasets. These improvements enhance its ability to capture deep contextual features, making it an ideal choice for distinguishing between human-written and LLM-generated text.

The input to this module is preprocessed using a Byte Pair Encoding (BPE) tokenizer, which breaks down text into subword units, ensuring efficient representation of both common and rare words. Special tokens like [CLS] and [SEP] are added to the input sequence, where the [CLS] token serves as a global summary of the input text. Positional and segment embeddings are also incorporated to account for the sequential and contextual nature of the data. This processed input is then fed into RoBERTa's transformer layers, which employ multi-head self-attention and feed-forward networks to compute token embeddings.

For this project, the RoBERTa-base model is fine-tuned on a dataset comprising human-written and adversarially manipulated machine-generated texts. The fine-tuning process modifies the model's weights to optimize its performance in identifying subtle linguistic differences. A binary cross-entropy loss function is used, with a learning rate of 2e-5 and early stopping after three epochs to prevent overfitting. Regularization techniques such as L2 norm and dropout layers ensure stability and generalization during training. After processing the input, a fixed-length, 768-dimensional vector corresponding to the [CLS] token is extracted as the final embedding. This vector encapsulates the global semantic representation of the text, capturing both its syntactic and semantic nuances.

The embeddings generated by this module form the foundation for the subsequent components of the system. They are particularly advantageous

due to their robustness in adversarial scenarios, ability to capture deep contextual relationships, and scalability for handling essay-length texts. By integrating this fine-tuned embedding mechanism, the system ensures a reliable representation of text for downstream classification tasks.

3.4 LINGUISTIC FEATURE EXTRACTION

The Linguistic Feature Extraction module complements the embedding generation process by analyzing text through a range of linguistically motivated features. These features capture stylometric, syntactic, and semantic attributes that differentiate human-written and machine-generated content. The diversity and depth of these features ensure the system's robustness, especially in scenarios involving adversarial text manipulations like paraphrasing or word substitutions.The module extracts five key feature categories:

1. **Stylometric features :** These include metrics derived from lexical, syntactic, and structural properties of text, such as average sentence length, word frequency distribution, and punctuation usage. Part-of-speech tagging is also utilized to calculate the frequency of grammatical patterns, providing insight into stylistic tendencies.

2. **Readability statistics :** Readability metrics assess the complexity of text using established formulas like the Flesch Reading Ease, Flesch-Kincaid Grade Level, and Linsear Write metric. These metrics quantify the ease with which the text can be read and understood, capturing differences in sentence construction and vocabulary.

3. **Entity grid features :** Text coherence is evaluated using entity grids, which map the transitions of entities (subjects, objects, and others)

across sentences. The entity grid structure provides insights into the logical flow of information within the text, a key characteristic often lacking in machine-generated content.

4. **Lexical diversity :** The richness of the text's vocabulary is evaluated using metrics such as Type-Token Ratio (TTR), Maas TTR, and Hypergeometric Distribution (HDD). These measures provide an understanding of how varied the vocabulary is within the text, aiding in identifying LLM generated text, which often exhibits limited lexical diversity.

5. **Rhetorical Structure Features :** Inspired by rhetorical structure theory, this component examines the organization of discourse elements, identifying logical relationships like cause-effect and elaboration.

The extracted features are represented as a dense vector, normalized for consistent scaling. While transformer embeddings capture high-level semantic and contextual details, linguistic features offer granular insights into stylistic and structural differences. The combined use of these two modalities enhances the overall discriminative power of the system.

3.5 CLASSIFICATION MODULE

The Classification Module integrates the embeddings from the autoencoder with the linguistic feature vector to perform the final text classification. This fusion allows the system to leverage both deep contextual semantics and surface-level linguistic patterns, ensuring comprehensive analysis.

The architecture of this module is a feed-forward neural network

with two fully connected layers. The input to the network is a concatenated vector comprising the 768-dimensional embedding from the autoencoder and the normalized linguistic feature vector. The first hidden layer uses ReLU (Rectified Linear Unit) as the activation function to introduce non-linearity, ensuring the model can capture complex relationships between input features. Batch normalization is applied at each layer to stabilize training by normalizing the input distributions, and dropout with a probability of 0.5 is used to mitigate overfitting.

The model is trained using a binary cross-entropy loss function optimized with the Adam optimizer. The learning rate is set to 5e-5, with early stopping implemented to prevent overfitting during training. The network's final layer outputs a probability score, where a threshold determines whether the input is classified as human-written or machine-generated.

The modular design of the classification network ensures adaptability to new features or embeddings. Its architecture also allows scalability, enabling integration with datasets of varying size and complexity. By combining multiple input modalities, the Classification Module achieves high accuracy, robustness against adversarial manipulations, and interpretability of results.

CHAPTER 4

IMPLEMENTATION DETAILS

This chapter provides a comprehensive account of the implementation of the proposed classification framework designed to distinguish human-written text from LLM-generated text. The details include the setup of the development environment, the description of the model architecture, and the methodologies for integrating embedding generation, linguistic feature extraction, and neural network classification. Each component is discussed in detail in the following sections.

4.1 ENVIRONMENT SETUP

The model implemented using PyTorch 2.0, which is a robust and flexible library for deep learning tasks, providing flexible options for model development and training optimization.

Software Libraries

- **PyTorch:** A robust and flexible library for building and training deep learning models, used for implementing the neural network architecture.
- **Hugging Face Transformers:** Provides pre-trained transformer models like roberta-base and tools for fine-tuning and embedding generation.
- **Other Libraries:** Standard data-handling libraries like NumPy and pandas, with matplotlib for visualization.

Development Hardware

- NVIDIA GTX 1650 GPU with 4GB VRAM
- Intel I5 11th Gen processor
- 8GB DDR4 RAM

This configuration ensured efficient execution of computationally demanding tasks such as transformer fine-tuning, neural network training, and feature computation.

4.2 MODEL ARCHITECTURE DESCRIPTION

The architecture of the framework combines pre-trained transformer embeddings, linguistic feature extraction, and a feed-forward neural network. Each module in the system is carefully designed to contribute to capturing the nuanced differences between Human written and LLM generated text. Below are the descriptions of the individual components:

4.2.1 EMBEDDING GENERATION

In this section, we describe the methodology used to preprocess the input essays and extract dense vector representations using the [CLS] embeddings from the RoBERTa-base transformer model. Furthermore, these embeddings were compressed using a custom-built autoencoder to capture latent representations for downstream tasks.

4.2.1.1 Preprocessing Essays and Generating Embeddings

The initial step involved tokenizing the input text data and passing it through the roberta-base transformer model to extract the [CLS] token embeddings. These embeddings serve as compact and semantically rich representations of the essays.

- **Tokenizer and model :** The RobertaTokenizer and RobertaModel from the Hugging Face Transformers library were used for preprocessing and embedding generation, respectively.
- **Tokenization :** Each essay was tokenized with truncation to handle a maximum sequence length of 512 tokens, ensuring compatibility with RoBERTa's architecture. Padding was applied to maintain uniform input dimensions.
- **Embedding Extraction :** The model was executed in evaluation mode (model.eval()), and the last hidden state of the transformer was used to extract the [CLS] embedding, representing the entire essay's semantic meaning.

Challenges encountered during this process were addressed effectively by truncating large input sequences to a maximum of 512 tokens, ensuring context preservation while maintaining compatibility with the RoBERTa model architecture. Additionally, efficient computation was achieved using GPU acceleration, enabling the processing of thousands of essays. The resulting embeddings were stored as a tensor of shape (n,768), where n represents the number of essays, and 768 is the embedding dimension specific to RoBERTa-base.

4.2.1.2 Autoencoder for Embedding Compression

To reduce the dimensionality of the embeddings while preserving critical semantic information, a custom autoencoder architecture was implemented. This facilitated the extraction of latent features from high-dimensional [CLS] embeddings.

The forward transformations for encoding ($f(x)$) and decoding ($g(z)$) were defined as:

$$f(x) = \text{ReLU}(W_1 x + b_1), \quad g(z) = \text{ReLU}(W_2 z + b_2) \tag{4.1}$$

where x is the input embedding, z is the latent vector, and W_1, W_2 are the weight matrices.

4.2.1.3 Loss Function

The reconstruction loss was computed using Mean Squared Error (MSE), defined as:

$$L = \frac{1}{n} \sum_{i=1}^{n} \|x_i - \hat{x}_i\|^2 \tag{4.2}$$

where x_i is the original embedding, and $\hat{x}_i$ is the reconstructed embedding.

4.2.1.4 Training Details

- **Optimizer:** Adam optimizer with a learning rate of 1×10^{-4}.
- **Batch Size:** 8 embeddings per batch.

- **Epochs:** The autoencoder was trained for 20 epochs.
- **Evaluation:** The reconstruction loss was monitored across epochs to ensure convergence.

4.2.1.5 Algorithm for Embedding Generation

The workflow for generating embeddings, including tokenization, [CLS] extraction, and autoencoder-based compression, is outlined in Algorithm 4.1.

Algorithm 4.1 Embedding Generation with Autoencoder Compression

Input: Dataset of essays E, RoBERTa tokenizer T, RoBERTa model M, Autoencoder AE
Output: Compressed embeddings Z
Initialize an empty list C to store [CLS] embeddings
Load RoBERTa tokenizer T and model M in evaluation mode
for each essay e_i in E **do**
    Tokenize e_i using T with truncation and padding
    Pass tokenized input to M and extract [CLS] embedding c_i
    Append c_i to C
end for
Convert C to tensor format
Train the Autoencoder AE with C for N epochs
Extract compressed embeddings Z from the encoder's bottleneck layer
Save Z for downstream tasks
Return Z

4.2.2 LINGUISTIC FEATURE EXTRACTION

To enhance the understanding of textual characteristics, three key categories of linguistic features are extracted: Readability Statistics, Diversity Stylometry, and Entity Grid Analysis. These features are essential for evaluating

textual coherence, complexity, and lexical diversity. Below is an in-depth explanation of each method and how the features are derived.

4.2.2.1 Readability Statistics

Readability statistics quantify the complexity and ease of understanding of a text. These metrics are derived using well-established formulas that evaluate sentence structure, word difficulty, and syllable patterns. The features extracted include:

- **Flesch-Kincaid Grade Level:** Calculates the grade level required to comprehend the text.

$$\text{FKGL} = 0.39\left(\frac{\text{Total Words}}{\text{Total Sentences}}\right) + 11.8\left(\frac{\text{Total Syllables}}{\text{Total Words}}\right) - 15.59 \quad (4.3)$$

- **Flesch Reading Ease:** Rates text on a 100-point scale, where higher scores indicate easier readability.

$$\text{FRE} = 206.835 - 1.015\left(\frac{\text{Total Words}}{\text{Total Sentences}}\right) - 84.6\left(\frac{\text{Total Syllables}}{\text{Total Words}}\right) \quad (4.4)$$

- **Gunning Fog Index:** Estimates years of formal education needed to understand the text.

$$\text{GFI} = 0.4\left(\frac{\text{Total Words}}{\text{Total Sentences}} + \text{Percentage of Complex Words}\right) \quad (4.5)$$

- **SMOG Index:** Focuses on polysyllabic words for predicting comprehension difficulty.

$$\text{SMOG} = 1.043\sqrt{\text{Number of Polysyllabic Words} \times \left(\frac{30}{\text{Total Sentences}}\right)} + 3.1291 \quad (4.6)$$

- **Coleman-Liau Index:** Uses characters instead of syllables for readability scoring.

$$\text{CLI} = 0.0588L - 0.296S - 15.8 \tag{4.7}$$

where:
L = Average number of letters per 100 words,
S = Average number of sentences per 100 words.

- **Automated Readability Index (ARI):** Focuses on character counts for estimating complexity.

$$\text{ARI} = 4.71\left(\frac{\text{Characters}}{\text{Words}}\right) + 0.5\left(\frac{\text{Words}}{\text{Sentences}}\right) - 21.43 \tag{4.8}$$

- **Dale-Chall Readability Score:** Assesses the text's difficulty based on a list of familiar words.

4.2.2.2 Diversity Stylometry

Stylometric analysis explores lexical richness, syntactic patterns, and the diversity of word usage within a text. The extracted features include:

- **Type-Token Ratio (TTR):** Measures the ratio of unique words to total words.

$$\text{TTR} = \frac{\text{Unique Words}}{\text{Total Words}} \tag{4.9}$$

- **Maas TTR:** Adjusts TTR to account for text length.

$$\text{Maas TTR} = \log(\text{Total Words}) - \log(\text{Unique Words}) \tag{4.10}$$

The Hypergeometric Distribution Diversity (HDD) employs probabilistic sampling to estimate lexical diversity, while the Moving Average TTR

(MATTR) calculates the average Type-Token Ratio over sliding windows within the text to account for variations in text length. The Measure of Textual Lexical Diversity (MTLD) evaluates how frequently new vocabulary is introduced, providing insights into the consistency of lexical variety. Additionally, linguistic processing tools enable the extraction of Part-of-Speech (POS) tags, such as nouns and verbs, along with word shape features, including capitalization and numerical forms, to capture syntactic structure and word usage patterns. Together, these features offer a comprehensive analysis of both lexical and structural aspects of text.

4.2.2.3 ALGORITHM FOR LINGUISTIC FEATURE EXTRACTION

The process for extracting linguistic features, including preprocessing, computation of readability and diversity metrics, and syntactic analysis, is detailed in Algorithm 4.2. Furthermore, the computations and methodologies described in Equation 4.1 to Equation 4.11 are summarized within Algorithm 4.2, ensuring a cohesive representation of the feature extraction pipeline.

4.2.2.4 ENTITY GRID

Entity grid analysis evaluates text coherence by tracking the transitions of entities, such as nouns, pronouns, or proper nouns, across sentences. Each entity is assigned a grammatical role within a sentence: subject (s), object (o), other (x) for non-subject/object roles, or absent (-) if the entity does not appear. Coherence is analyzed by examining transitions between these roles across consecutive sentences, such as $s \rightarrow o$ or $x \rightarrow -$. The frequency of each transition type is normalized to calculate a weighted transition probability,

Algorithm 4.2 Computation of Readability and Diversity Metrics

Require: Input text T
Ensure: Computed metrics M
Step 1: Preprocessing
Tokenize the input text T into sentences and words.
Count total words, sentences, syllables, and complex words.
Step 2: Readability Metrics Computation
Compute Flesch-Kincaid Grade Level (FKGL) using word, sentence, and syllable counts.
Compute Flesch Reading Ease (FRE) based on text length and syllables.
Calculate Gunning Fog Index (GFI) using word and sentence counts.
Compute SMOG Index using polysyllabic word and sentence counts.
Derive Coleman-Liau Index (CLI) based on character and sentence counts.
Compute Automated Readability Index (ARI) using character and word counts.
Step 3: Diversity Metrics Computation
Calculate Type-Token Ratio (TTR) as the ratio of unique words to total words.
Adjust TTR to compute Maas TTR for text length normalization.
Apply Hypergeometric Distribution Diversity (HDD) for probabilistic sampling.
Compute Moving Average TTR (MATTR) using sliding windows over the text.
Derive Measure of Textual Lexical Diversity (MTLD) to evaluate vocabulary variety.
Step 4: Syntactic and Structural Analysis
Extract Part-of-Speech (POS) tags from the text.
Extract word shape features, including capitalization and numerical forms.
Step 5: Output
Combine all computed metrics into M.
return M

which quantifies the relative importance of each transition. This is done using the following formula:

$$\text{Weight} = \frac{\text{Transition Count}}{\text{Total Transitions}} \tag{4.11}$$

Dependency parsing is used to identify entities and their roles, making this method a robust approach to measuring text coherence.

4.2.3 NEURAL NETWORK TRAINING

The classification of human-written and LLM-generated texts was achieved using a Feed-Forward Neural Network (FFNN) model. This model integrates linguistic features and autoencoder embeddings to generate predictions. This section elaborates on the design of the neural network, the data preprocessing pipeline, the training procedure, and the evaluation metrics used to measure its performance.

4.2.3.1 MODEL ARCHITECTURE

The neural network model, implemented as the CombinedModel class, is a deep feed-forward architecture tailored for binary classification. The input layer of the model accepts concatenated vectors that include autoencoder embeddings and linguistically derived features, which are then processed through a series of hidden layers.

The first hidden layer comprises 512 neurons, followed by a batch normalization step to stabilize and accelerate training by normalizing the activations. ReLU (Rectified Linear Unit) is employed as the activation function

to introduce non-linearity, enabling the network to learn complex patterns in the data. To prevent overfitting, a dropout regularization layer with a probability of 0.5 is applied. A second hidden layer, containing 64 neurons, processes the intermediate features and also uses batch normalization, ReLU activation, and dropout. The final output layer consists of two neurons corresponding to the binary classification labels, and a softmax activation is applied during evaluation to compute class probabilities.

This architecture balances complexity and generalizability, ensuring the model effectively captures patterns within the data without overfitting.

4.2.3.2 TRAINING PROCEDURE

The FFNN model was trained using the Cross-Entropy Loss function, which measures the discrepancy between the predicted probabilities and the true labels. The Adam optimizer was employed to minimize the loss function. This optimizer was configured with a learning rate of 1×10^{-5} and a weight decay of 0.01 to mitigate overfitting by penalizing large weights. Additionally, gradient clipping with a maximum norm of 1.0 was implemented to prevent exploding gradients, ensuring stability during backpropagation.

A learning rate scheduler, `ReduceLROnPlateau`, dynamically adjusted the learning rate based on validation accuracy. If the validation accuracy failed to improve for three consecutive epochs, the learning rate was reduced by a factor of 0.1. This mechanism allowed the model to refine its performance during the later stages of training.

Training was conducted over a maximum of 25 epochs, with early stopping enabled to halt training if the validation accuracy did not improve for

five consecutive epochs. This strategy prevented overfitting by ensuring the model stopped training once its performance plateaued on the validation set.

4.2.3.3 TESTING PROCEDURE

The testing phase aimed to evaluate the classification performance of the proposed framework on an independent test dataset. The dataset underwent the same preprocessing steps as the training phase, including tokenization, syllable counting, and linguistic feature extraction. Compressed embeddings generated by the fine-tuned autoencoder were combined with the extracted features and passed to the trained feed-forward neural network for classification.

To assess the model's performance, four key metrics were calculated: accuracy, precision, recall, and F1-score. These metrics provided a comprehensive evaluation of the framework's ability to differentiate between human-written and machine-generated texts. The results demonstrated the effectiveness of integrating autoencoder embeddings with linguistic features.

4.3 SUMMARY

This chapter has provided a comprehensive explanation of the implementation process for the proposed framework, which integrates fine-tuned transformer-based autoencoder embeddings with diverse linguistic features to create a robust classification model. The feature extraction methods effectively capture linguistic, structural, and semantic characteristics, while the feed-forward neural network leverages these features to perform accurate classification.By employing modern deep learning techniques, such as dropout, batch normalization, and ReLU activation, combined with linguistically motivated analyses, the system adeptly identifies subtle differences between

human-written and machine-generated texts. The modular design of the framework ensures scalability, enabling seamless integration of additional features or algorithms.

CHAPTER 5

RESULTS AND DISCUSSIONS

In this chapter, the results of the model training and evaluation will be presented and discussed. The main objective of this project was to develop a feed-forward neural network (FFNN) for distinguishing between human-written and LLM-generated text, leveraging lexical diversity and stylistic features. The results from the validation process, as well as various evaluation metrics, will be analyzed to assess the performance of the model.

5.1 TRAINING AND VALIDATION LOSS

Figures and accompanying charts were created to visualize the training and validation loss over the course of the training process. As seen in the training curve 5.1, the model demonstrated a significant reduction in the training loss from the start of the training to approximately epoch 6, where the training loss plateaued. This suggests that the model efficiently learned to minimize the loss at the early stages.However, the validation loss exhibited fluctuations, peaking at certain points (e.g., epochs 8, 14) before steadily decreasing towards the later epochs. The validation loss achieved a minimum at epoch 12, after which it again started to increase, indicating potential overfitting, where the model started to perform well on the training data but less so on unseen data. This overfitting is a common issue in machine learning and was addressed by the implementation of early stopping and the use of dropout layers in the FFNN model architecture.

5.2 VALIDATION ACCURACY

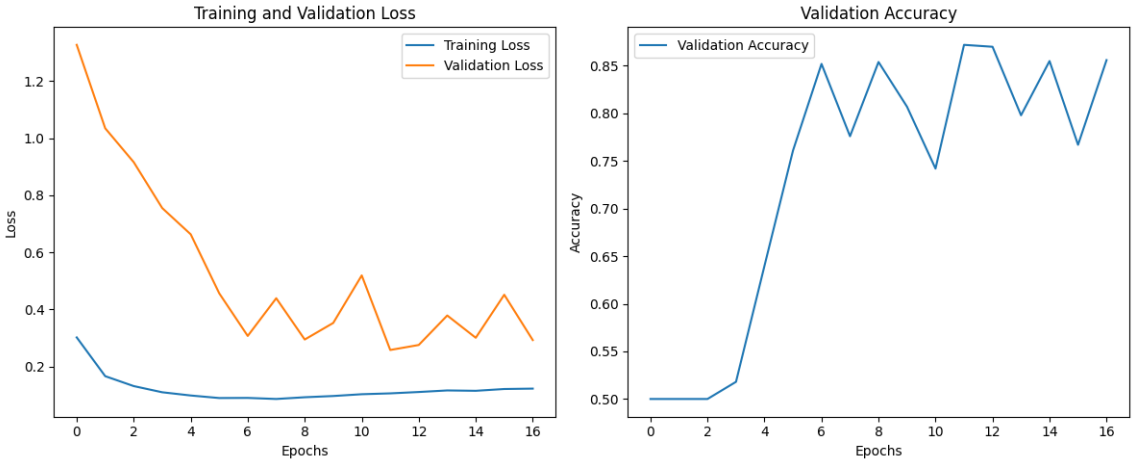

Figure 5.1: Model Performance

The validation accuracy followed a similar trend, showing a sharp increase in the first few epochs, reaching as high as 87.2 by epoch This result indicates that the model effectively learned the underlying patterns in the data, with improvements in accuracy being sustained even after the learning rate was reduced at epoch 16. The validation accuracy reached its peak value at epoch 12 (87.2), and the model maintained this performance until the early stopping criterion was met at epoch 17.The sharp jump in accuracy observed around epoch 6–7 can be attributed to the model's ability to generalize well, as the regularization techniques (dropout, batch normalization) and optimization strategies (learning rate decay) allowed it to escape local minima and converge towards better solutions.Early stopping was employed to prevent overfitting by halting the training process when the validation accuracy failed to improve for five consecutive epochs. This strategy ensured that the model did not overtrain, thus preserving its ability to generalize. Additionally, the learning rate scheduler reduced the learning rate by a factor of 10 after epoch 16, which helped the model converge more smoothly toward the optimal solution.The final model was saved at epoch 17, which was the best-performing epoch according to the validation accuracy.

5.3 RESULTS OF LLM GENERATED ESSAY DETECTION SYSTEM

The results of the LLM-generated essay detection system are presented in this section, focusing on the performance of each module within the framework. The model's ability to distinguish between human-written text (HWT) and LLM-generated text is assessed, with specific attention to the effectiveness of the Embedding Generation Module and the Linguistic Feature Extraction Module. Each module's contribution to the overall system's performance is detailed, emphasizing how the integration of RoBERTa embeddings and handcrafted linguistic features improves classification accuracy and robustness. The results also highlight the system's ability to generalize across different text types within the project's dataset, providing a comprehensive evaluation of its effectiveness in detecting LLM-generated text.

5.3.1 INPUT SAMPLE REPRESENTATION

The raw input to the framework consists of essays and their corresponding labels, as shown in the table below **??**. For illustration, a single sample essay is represented in a cell format, with the content truncated for brevity.

Table 5.1: Evaluation Metrics for the Model

Metric	**Value (%)**
Accuracy	87.2
F1 Score	86.99
Precision	89.81
Recall	87.20

5.3.2 LINGUISTIC FEATURE EXTRACTION MODULE

The feature extraction process transforms raw essay data into structured representations suitable for classification. This process is crucial

for capturing linguistic, syntactic, and stylistic features, enabling the model to distinguish between human-written and machine-generated text. The workflow for feature extraction, incorporating readability metrics, lexical diversity measures, and stylometric properties, is outlined below.

The framework employs automated tools and algorithms to quantify the coherence and complexity of the text. Key extracted features include readability scores, entity transition patterns, and diversity metrics. These features ensure the downstream classification model's performance and interpretability.

The readability metrics, as shown in Figure 5.2, quantify how easily a given text can be read and understood. Stylometric features, including entity transitions and lexical richness, are summarized in Figure 5.3. and Figure 5.4

Readability scores are calculated using well-established formulas, such as Flesch-Kincaid, Gunning Fog, and SMOG indices. Stylometric features include transition patterns between entities and lexical diversity metrics like Type-Token Ratio (TTR) and Maas TTR. These outputs, outlined in Figure 5.2 and Figure 5.3, serve as a comprehensive representation of the input data, ensuring the robustness of the classification model.

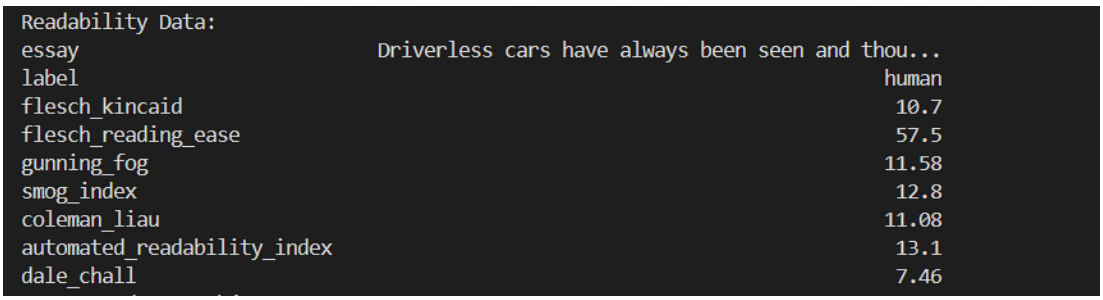

Figure 5.2: Readability Metrics for the training dataset.

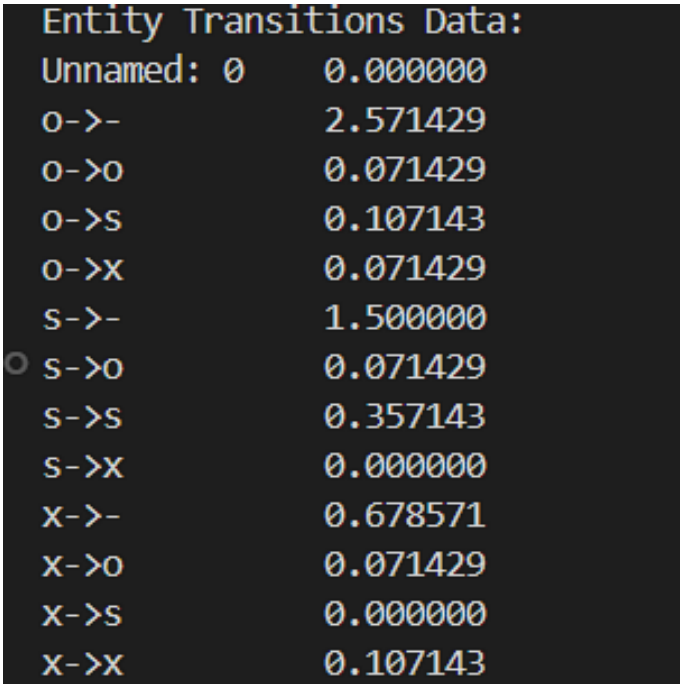

Figure 5.3: Entity Transition Patterns captured during feature extraction.

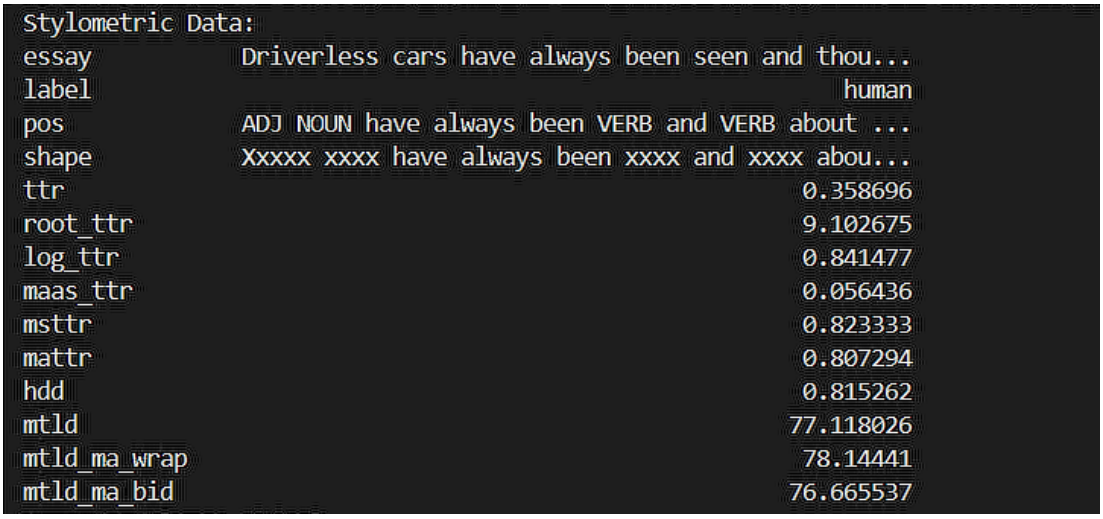

Figure 5.4: Stylometric features for the training dataset.

5.3.3 EMBEDDING GENERATION MODULE

The Embedding Generation Module demonstrated strong performance in generating effective embeddings for distinguishing human-written text (HWT) from LLM-generated text. By leveraging the RoBERTa-base model to generate contextual embeddings and applying mean pooling, the module successfully condensed token-level information into fixed-length vectors. These embeddings captured rich semantic information, providing a solid foundation for text classification tasks. An example of a generated embedding is shown in Figure 5.5, which illustrates the combined representation of both RoBERTa embeddings and handcrafted features.

Incorporating handcrafted features, such as Type-Token Ratio (TTR), Moving-Average Type-Token Ratio (MATTR), and Measure of Textual Lexical Diversity (MTLD), alongside RoBERTa embeddings enhanced the overall representation. These features added valuable statistical and stylistic insights, enabling the model to better capture lexical diversity and sentence structure, which are key in differentiating between human and LLM-generated writing. The hybrid embedding approach significantly improved classification accuracy

```
First training vector: [ 1.07000000e+01  5.75000000e+01  1.15800000e+01  1.28000000e+01
  1.10800000e+01  1.31000000e+01  7.46000000e+00  3.58695652e-01
  9.10267519e+00  8.41476689e-01  5.64363660e-02  8.23333333e-01
  8.07294118e-01  8.15262499e-01  7.71180260e+01  7.81444099e+01
  7.66655372e+01  0.00000000e+00  2.57142850e+00  7.14285750e-02
  1.07142860e-01  7.14285750e-02  1.50000000e+00  7.14285750e-02
  3.57142870e-01  0.00000000e+00  6.78571400e-01  7.14285750e-02
  0.00000000e+00  1.07142860e-01 -6.93021640e-02 -3.02274287e-01
  2.40390047e-01  7.18156815e-01 -3.35885376e-01 -2.78332859e-01
  3.30515862e-01  1.00614107e+00 -4.57032144e-01 -1.15727782e+00
  5.40218413e-01  2.57574826e-01 -8.04753721e-01  8.84188652e-01
  6.78465515e-02  1.92449361e-01  1.73478853e-02  1.03146601e+00
  6.35646358e-02  4.28001493e-01 -1.61001205e-01 -2.16741309e-01
```

Figure 5.5: Generated embedding vector resulting from the concatenation of the input essay and linguistic features.

compared to models relying solely on RoBERTa embeddings or handcrafted

features alone. The combination of semantic depth and statistical information made the embeddings more robust and adaptable to various text domains, ensuring better generalization and higher performance across different datasets.

Overall, the results highlight the effectiveness of combining deep learning-based contextual embeddings with handcrafted lexical and stylistic features. This approach not only increased classification accuracy but also provided a scalable solution, capable of handling diverse writing styles and text structures, making it highly suitable for distinguishing HWT from LLM-generated text. The generated embedding, as shown in Figure 5.5, reflects the hybrid nature of the representation and its potential to improve classification performance.

5.4 PERFORMANCE ANALYSIS

To further analyze the model's performance, a confusion matrix Fig.5.6 was generated to provide a detailed breakdown of true positives, true negatives, false positives, and false negatives. The confusion matrix reveals that the model correctly classified 500 samples as negative (human-written text), representing the true negatives (TN), and accurately identified 372 samples as positive (LLM-generated text), representing the true positives (TP). However, 128 positive samples were misclassified as negative, indicating false negatives (FN) and suggesting some difficulty in identifying certain machine-generated texts. Notably, there were no cases where human-written texts were incorrectly classified as machine-generated, resulting in zero false positives (FP). The high values for true positives and true negatives highlight the model's overall effectiveness in distinguishing between the two classes. However, the false negatives suggest areas for improvement in detecting more nuanced machine-generated text. The absence of false positives demonstrates that the model is conservative in misclassifying human-written content, which

aligns with the high precision observed in the evaluation metrics.

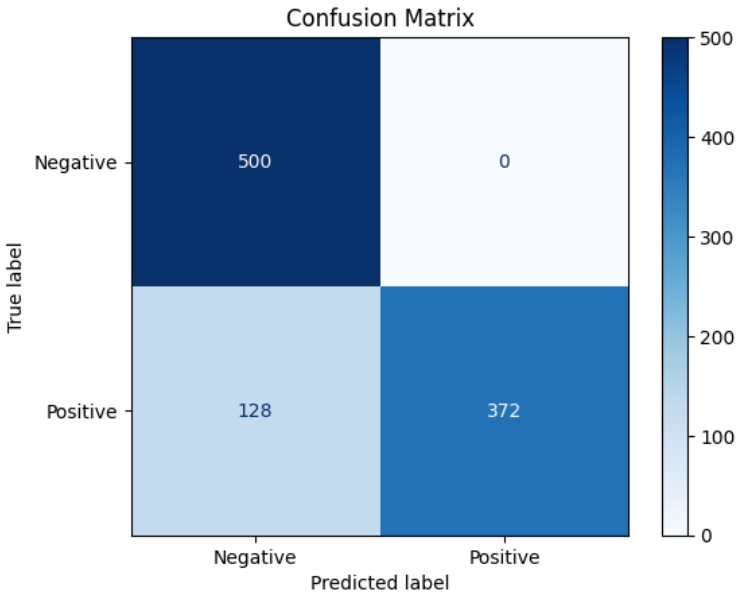

Figure 5.6: Confusion Matrix

The high values for true positives and true negatives highlight the model's overall effectiveness in distinguishing between the two classes. However, the false negatives suggest areas for improvement in detecting more nuanced machine-generated text. The absence of false positives demonstrates that the model is conservative in misclassifying human-written content, which aligns with the high precision observed in the evaluation metrics.

The evaluation of the model was based on several metrics, including accuracy, F1 score, precision, and recall. The results were as follows:

Table 5.2: Evaluation Metrics for the Model

Metric	**Value (%)**
Accuracy	87.2
F1 Score	86.99
Precision	89.81
Recall	87.20

These metrics indicate that the model achieved strong classification

performance across both classes, with particularly high precision. Precision reflects the proportion of positive predictions that were correct, suggesting that the model was effective in identifying machine-generated text when it classified a sample as such. The recall value of 87.20% indicates that the model also performed well at identifying true positives, ensuring that most of the machine-generated texts were classified correctly. The F1 score, a harmonic mean of precision and recall, further supports the conclusion that the model's performance was balanced, with a value of 86.99% suggesting that both false positives and false negatives were minimized.

5.4.1 ANALYSIS OF KEY LINGUISTIC FEATURES

To gain deeper insights into the distinguishing characteristics of AI-generated and human-written texts, several linguistic features were analyzed. These features include lexical diversity metrics such as the Moving Average Type-Token Ratio (MATTR) and Maas Index, as well as readability metrics like the Flesch-Kincaid Grade Level, Gunning Fog Index, SMOG Index, and Flesch Reading Ease. The results of this analysis are summarized in Table5.3. The comparative analysis presented in Table VI highlights significant differences between AI-generated and human-written texts across various linguistic dimensions. The MATTR value for AI-generated texts (0.9548) is higher than that of human-written texts (0.9203), indicating that AI systems tend to employ a broader range of vocabulary within a text, whereas the slightly higher Maas Index for human-written texts suggests that human authors demonstrate more nuanced word usage despite a lower overall lexical diversity. In terms of readability and complexity, AI-generated texts are more challenging to read, as evidenced by their lower Flesch Reading Ease score (35.42) compared to human-generated texts (56.18). This trend is further supported by the Flesch-Kincaid Grade Level (37.07 vs. 52.96), Gunning Fog Index (41.32 vs. 56.99), and SMOG Index (10.35 vs. 13.25), which consistently reveal

simpler syntactic structures in human-authored texts. These findings suggest that human writers prioritize fluency and readability, while AI-generated content often reflects more structured and formal language patterns. These distinct linguistic characteristics formed the basis for feature extraction in the proposed classification framework, with the higher complexity and lexical diversity of AI-generated texts serving as key distinguishing features effectively captured by the feed-forward neural network (FFNN) for classification.

Table 5.3: Some Feature Values Characteristic to AI and Human-Generated Texts

Feature	AI-Generated	Human-Generated
Flesch-Kincaid Grade Level	37.07	52.96
Flesch Reading Ease	35.42	56.18
Gunning Fog Index	41.32	56.99
SMOG Index	10.35	13.25
MATTR	0.9548	0.9203
Maas	0.0180	0.0196

5.5 MODEL OVERFITTING AND GENERALIZATION

One of the critical challenges in training deep learning models is avoiding overfitting, particularly when working with relatively small datasets. While the training loss continuously decreased, the validation loss showed some fluctuations, especially beyond epoch 12. This highlights the challenge of generalization, where a model performs well on training data but struggles with unseen data. The strategies employed in this project, such as dropout, batch normalization, and early stopping, helped mitigate overfitting. The model was able to achieve high validation accuracy and evaluation metrics, which suggests that the regularization techniques were successful in promoting generalization.

5.6 DISCUSSION

The results demonstrate that the FFNN model is effective at

distinguishing between human-written and LLM-generated text based on lexical and stylistic features. The high validation accuracy, precision, and recall values indicate that the model was able to learn these features well and generalize them to new, unseen data. However, the fluctuations in validation loss and the slight overfitting observed suggest that further improvements can be made. Potential strategies include increasing the size of the training dataset through data augmentation techniques, which could help improve generalization. Additionally, implementing more advanced regularization techniques, such as L2 regularization or early stopping with a more aggressive threshold, could further reduce overfitting. Finally, fine-tuning hyperparameters such as the learning rate, batch size could yield even better results.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 CONCLUSION

The proposed framework tackles the challenging task of differentiating between Human-Written Text and LLM-Generated Text, particularly in scenarios where adversarial manipulations are involved. By combining fine-tuned autoencoder embeddings, which extract meaningful semantic patterns, with linguistically informed features—such as readability metrics, stylometry, lexical diversity, and entity grids—the model effectively integrates both semantic and stylistic information to achieve accurate classification. This combination of features plays a pivotal role in enhancing the model's understanding of subtle variations in text.

The feed-forward neural network used as the core classification engine demonstrates the ability to integrate these diverse features seamlessly. The model achieved strong classification performance, recording an accuracy of 0.872, along with a precision of 0.8981 and a recall of 0.8720. These results emphasize the framework's reliability in detecting LLM-generated text while minimizing misclassifications. More importantly, the hybrid approach—merging deep semantic insights with linguistic features to be highly effective in addressing this complex problem.

One of the key strengths of the model lies in its ability to adapt to texts with varying stylistic and linguistic patterns. The fine-grained feature extraction process allows it to handle both structured and loosely coherent texts, ensuring consistent performance. However, the experiments also revealed

certain limitations. Specifically, the model struggled against adversarial attacks involving paraphrasing. These findings highlight the need for improving the model's generalization capabilities and robustness to evasive strategies.

This study makes important contributions by integrating autoencoder embeddings with linguistic features and leveraging entity grids for enhanced text coherence analysis. While the results are promising, the limitations identified—particularly in handling adversarial attacks—present opportunities for further improvement. Addressing these challenges, alongside optimizing computational efficiency, will be crucial for deploying the model effectively in real-world applications.

6.2 FUTURE WORK

While the current study demonstrates strong performance in distinguishing human-written text from LLM-generated content, several avenues can further enhance the model's robustness and applicability. First, advanced embedding techniques such as transformer-based models (e.g., GPT, T5) and domain-adaptive pre-training will be explored. These methods can provide richer semantic representations by fine-tuning embeddings specifically for human-written and machine-generated text tasks. Additionally, multi-task learning approaches may enable the model to generalize better by simultaneously learning related tasks, such as paraphrase detection or text summarization.

Second, integrating contextual and semantic knowledge sources can address limitations in handling adversarially modified text. Future work will incorporate knowledge graphs to capture structured semantic relationships and pre-trained language models with reasoning capabilities to detect subtle changes in text coherence. This integration will enhance the model's ability

to identify adversarial attacks, such as paraphrasing, which often challenge existing classifiers.

Finally, optimizing the neural network architecture and expanding dataset diversity will further improve the model's performance and generalization. Architectures such as attention mechanisms, transformers, and Graph Neural Networks (GNNs) will be investigated to capture long-range dependencies and entity-level coherence. In parallel, expanding datasets to include multilingual texts, diverse adversarial scenarios (e.g., GPT-4, Claude-generated content), and synthetic data augmentation will ensure the model's adaptability across languages and domains. These improvements will contribute to the development of resilient and scalable text classification systems capable of detecting AI-generated content in increasingly complex and adversarial environments.

REFERENCES

[1] Tharindu Kumarage, Joshua Garland, Amrita Bhattacharjee, Kirill Trapeznikov, Scott Ruston, and Huan Liu. Stylometric detection of ai-generated text in twitter timelines. *arXiv preprint arXiv:2303.03697*, 2023.

[2] Todor Markov, Chong Zhang, Sandhini Agarwal, Florentine Eloundou Nekoul, Theodore Lee, Steven Adler, Angela Jiang, and Lilian Weng. A holistic approach to undesired content detection in the real world. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 15009–15018, 2023.

[3] Ashish Bajaj and Dinesh Kumar Vishwakarma. Evading text based emotion detection mechanism via adversarial attacks. *Neurocomputing*, 558:126787, 2023.

[4] Evan Crothers, Nathalie Japkowicz, Herna Viktor, and Paula Branco. Adversarial robustness of neural-statistical features in detection of generative transformers. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2022.

[5] Ryuto Koike, Masahiro Kaneko, and Naoaki Okazaki. Outfox: Llm-generated essay detection through in-context learning with adversarially generated examples. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 21258–21266, 2024.

[6] Guanhua Huang, Yuchen Zhang, Zhe Li, Yongjian You, Mingze Wang, and Zhouwang Yang. Are ai-generated text detectors robust to adversarial perturbations? *arXiv preprint arXiv:2406.01179*, 2024.

[7] Xiao Pu, Jingyu Zhang, Xiaochuang Han, Yulia Tsvetkov, and Tianxing He. On the zero-shot generalization of machine-generated text detectors. *arXiv preprint arXiv:2310.05165*, 2023.

[8] Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Wieting, and Mohit Iyyer. Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense. *Advances in Neural Information Processing Systems*, 36, 2024.

[9] Nai Zhou, Nianmin Yao, Jian Zhao, and Yanan Zhang. Rule-based adversarial sample generation for text classification. *Neural Computing and Applications*, 34(13):10575–10586, 2022.

[10] Xinlin Peng, Ying Zhou, Ben He, Le Sun, and Yingfei Sun. Hidding the ghostwriters: An adversarial evaluation of ai-generated student essay detection. *arXiv preprint arXiv:2402.00412*, 2024.