IEEE 2nd International Conference on Computing, Applications and Systems (COMPAS 2025)
23-24 October 2025, Kushtia, Bangladesh

StackTrace-AI: Identifying Generative AI Text Origins using Ensemble Learning

Pabon Shaha[a], Md. Sabbir Hossen[b], Md. Ibrahim Hosen Sojib[a], Sanjida Akter[c], Md. Saiduzzaman[b], and Mostofa Kamal Nasir[a]
[a]Mawlana Bhashani Science & Technology University, Tangail-1902, Bangladesh
[b]Bangladesh University, Mohammadpur, Dhaka-1207, Bangladesh
[c]Northern University Bangladesh, Dhaka-1230, Bangladesh
Email: pabonshahacse15@gmail.com, sabbir.hossen@bu.edu.bd, ibrahimhosen0119@gmail.com, sanjida.nub@gmail.com, sojib.cse56bu@gmail.com, kamal@mbstu.ac.bd

***Abstract*—Generative AI models capable of producing human-like text, images, and other content are rapidly transforming the digital landscape. As their use becomes increasingly widespread across domains such as education, media, and communication, their outputs are shaping public discourse, influencing opinions, and even spreading misinformation. In this context, accurately identifying the source of AI-generated text is becoming increasingly crucial for ensuring transparency and accountability, protecting information integrity, mitigating deception, and fostering public trust in digital content. This study presents a source attribution framework designed to distinguish between texts generated by two leading large language models: ChatGPT and Gemini. A dataset of 2,786 AI-generated samples was constructed for this purpose. We employed transformer-based feature extraction techniques, including BERT, ALBERT, and RoBERTa, as well as a Hybrid model (AR-BERT, which combines ALBERT and RoBERTa), to capture distinctive linguistic signatures. This was followed by classification using nine traditional and ensemble machine learning algorithms. Among these, the AR-BERT-based features combined with a stacking ensemble classifier, StackTrace-AI, achieved the highest accuracy of 84.44%, surpassing all other combinations. The findings underscore the viability of transformer-driven source detection and offer valuable insights for advancing AI accountability, generative content auditing, and responsible AI governance.**

***Index Terms*—Large Language Models, Generative AI, Natural Language Processing, and Transformer Models.**

I. INTRODUCTION

The widespread adoption of generative AI (GenAI) has fundamentally transformed the landscape of digital communication, content creation, and information dissemination [1]. GenAI systems such as OpenAI's ChatGPT [2] and Google's Gemini [3] are capable of producing highly coherent and human-like text, enabling a wide range of applications from automated customer support to creative writing, education, and journalism [4]. However, this rapid advancement has also raised critical concerns regarding the authenticity, authorship, and accountability of AI-generated content [5]. As these models increasingly influence public discourse and digital interactions, the ability to accurately detect and attribute the source of AI-generated text has become a pressing need. Source detection is not only essential for maintaining transparency and content integrity but also for combating misinformation, preventing misuse, and fostering trust in AI-powered systems.

The outputs of generative AI models often exhibit subtle stylistic and structural differences due to variations in architecture, training data, and decoding strategies [6]. While these differences may be imperceptible to human readers, machine learning techniques can uncover underlying patterns that help distinguish between models. Recognizing these distinct signatures provides a pathway toward building AI forensics tools capable of identifying the origin of generated content. This task becomes especially important in legal, educational, and journalistic contexts, where content attribution can directly impact credibility and ethical compliance [7], [8]. Despite the importance of this challenge, the field of generative AI text source detection remains relatively underexplored, particularly for closely matched, state-of-the-art models such as ChatGPT and Gemini.

In this study, we propose a machine learning framework for detecting and classifying the source of AI-generated texts from two leading LLMs: ChatGPT and Gemini. We curated a balanced dataset of 2,786 text samples generated by both models across a diverse range of prompts. For feature representation, we utilized state-of-the-art transformer models, including BERT, ALBERT, RoBERTa, and the proposed hybrid AR-BERT. These features were then passed through a diverse set of traditional and ensemble-based machine learning classifiers to detect the source of GenAi texts. Our major contributions are as follows,

- We introduce a benchmark dataset comprising 2786 texts generated by ChatGPT and Gemini across various prompt categories.
- We utilize multiple transformer-based language models, such as BERT, ALBERT, RoBERTa, and hybrid AR-BERT, to extract deep contextual features from the generated text.
- Developed the StacTrace-AI model for detecting ChatGPT and Gemini-generated texts.
- Model evaluation is conducted using standard performance metrics such as accuracy, precision, recall, and F1-score.

The remainder of this paper is organized as follows. Section II discusses related work in generative AI detection and model attribution. Section III describes the dataset creation, feature extraction, and classification pipeline. Section IV outlines the results and comparative analysis. Finally, Section V concludes with key insights and future directions in generative AI source detection research.

II. LITERATURE REVIEW

Recent advancements in generative AI have prompted extensive research into the detection of AI-generated texts,

979-8-3315-5525-2/25/$31.00

particularly in academic and professional domains. For instance, Yadagiri et al. [9] tackled this challenge using the llm-detect-ai-generated-text dataset from Kaggle, applying TF-IDF, Word2Vec, and word embeddings. Their traditional machine learning approach achieved remarkable performance with 98% accuracy and F1-score, even surpassing fine-tuned RoBERTa-based models. This indicates the continued relevance of linguistic feature engineering in AI text detection. Bataineh et al. [10] introduced a new dataset to differentiate between AI-generated and human texts. They employed various feature extraction methods, including BoW, TF-IDF, and BERT, along with UMAP for dimensionality reduction. Their models—spanning from Logistic Regression and Random Forest to deep learning architectures such as LSTM and CNN-LSTM—achieved over 90% accuracy. Moreover, the best-performing TF-IDF + Random Forest model was deployed using Streamlit for easy accessibility by non-technical users, and SHAP was applied to ensure model transparency and interpretability. Ibrahim et al. [11] proposed a machine learning framework using TF-IDF and traditional classifiers to differentiate between human- and chatbot-generated content. Among the tested models, Random Forest combined with TF-IDF achieved the highest performance (88% accuracy and F1-score). However, their study did not explore transformer-based embeddings or incorporate deep learning and ensemble strategies, which limits its scope in comparison to more recent approaches. Zhang et al. [12] emphasized the growing difficulty in detecting AI-generated text due to increasingly sophisticated LLMs. Their hybrid methodology, which combined TF-IDF with classifiers such as Bayesian models, SGD, CatBoost, and multiple DeBERTa-v3-large variants, showed superior performance across extensive experiments. As a result, their ensemble-based approach was validated as a robust solution against advanced synthetic content. Alhijawi [13] proposed AI-Catcher, a multimodal model that integrates MLP and CNN to detect scientific texts generated by ChatGPT. By leveraging linguistic, statistical, and sequential features from the AIGTxt dataset, the model achieved a 37.4% improvement in accuracy compared to baseline methods. This demonstrates its strong capability in identifying AI-generated academic content. Arshed et al. [14] focused on identifying AI-generated financial texts, particularly from ChatGPT and QuillBot. They applied a combination of TF-IDF and Word2Vec with an optimized window size of one, alongside a Random Forest classifier. Consequently, their model achieved a notable AUC of 0.94 for GPT-generated text and 72% overall accuracy in a multiclass setting. Their findings offer valuable insights into fine-tuned feature engineering for domain-specific AI generated text source detection. Despite promising results, several gaps remain in existing studies. Many rely on traditional ML models without fully leveraging transformer-based embeddings or deep ensemble techniques. Additionally, limited explainability in deep models and a lack of cross-domain generalization and practical deployment hinder broader applicability and real-world impact.

III. MATERIALS & METHODS

This section outlines the overall methodology adopted to build the StackTrace-AI framework. Our approach comprises several key stages: data collection, preprocessing, feature extraction, and classification. Each stage is designed to ensure accurate identification of the GenAI source based on textual responses. An architectural overview of the entire pipeline is illustrated in Fig. 1.

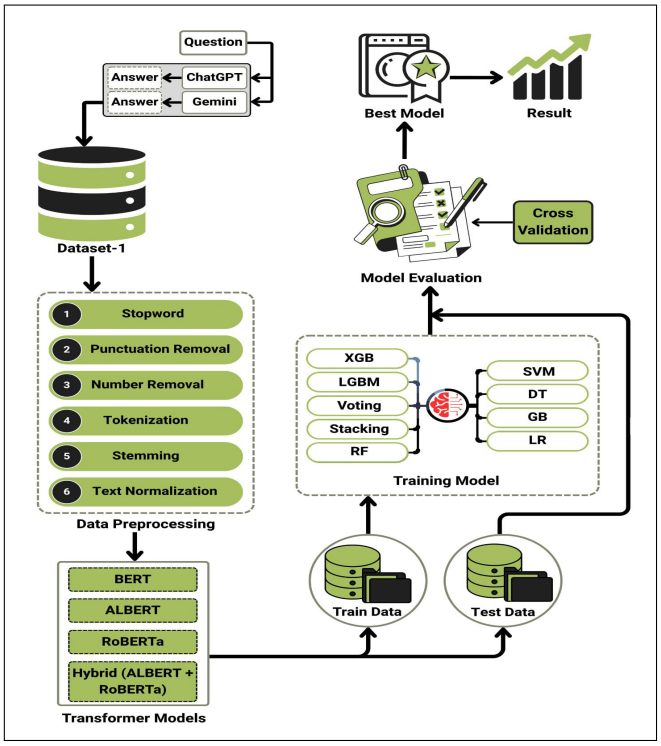


Fig. 1. Proposed Workflow for AI-Generated Text Source Detection

A. Data Collection

We collected the data used in this study through a structured data collection process. To construct the dataset, we engaged 100 university students from different academic years (1st–4th year) of the Department of Computer Science and Engineering at the Bangladesh University. The data collection process was conducted over a period of nearly one month, from May to June. We asked each of them to generate their unique questions based on their individual understanding and curiosity, primarily focused on computer science and engineering topics. While the questions varied from student to student, we instructed them to submit every question to both ChatGPT and Gemini to ensure parallel responses from the two models. The students then collected the responses from each model and labeled them accordingly to indicate which answer came from which AI system. To ensure data quality, the labeling was cross-verified by two

TABLE I
SAMPLE QUESTION ALONG WITH THE ANSWER FROM THE DATASET

Question	Answer	Source
What is an Operating System?	An Operating System (OS) is software that acts as an intermediary between computer hardware and users. It manages computer hardware resources and provides services for computer programs.	ChatGPT
What is an Operating System?	An operating system (OS) is the software that manages a computer's hardware and software resources. It provides a platform for applications to run and interact with the computer.	Gemini

independent students, and any discrepancies were resolved through discussion before finalizing the dataset. The final dataset was manually compiled into three columns: Question, Answer, and Source. This collaborative process resulted in a total of 2,786 labeled text samples, 1,368 from each model, forming a balanced dataset suitable for training and evaluating our generative AI source detection framework. Table I shows a sample question along with the corresponding answers generated by both models from the dataset.

B. Exploratory Data Analysis

To better understand the textual distribution in the dataset, a WordCloud was generated Answer field. Figure 2 shows the most frequent words, highlighting key terms such as *data*, *network*, *system*, and *model*. This visualization provides an overview of the dataset's dominant topics and ensures relevance to the task of AI-generated text classification.

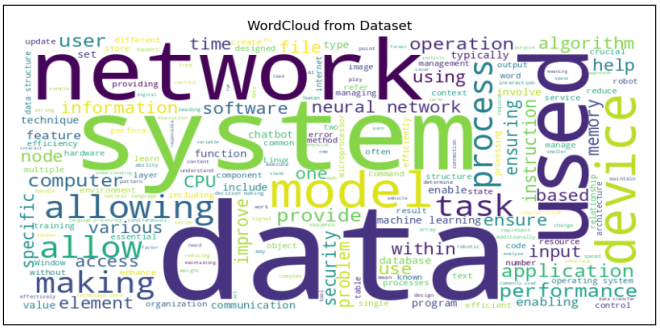


Fig. 2. WordCloud, shows the most frequent word from the dataset

C. Data Preprocessing

Before applying transformer-based models for feature extraction, the collected text data underwent several preprocessing steps to ensure cleanliness, consistency, and noise reduction. The preprocessing included converting all text to lowercase, removing special characters, HTML tags, unnecessary punctuation, extra whitespace, and irrelevant symbols. Additionally, stop words were removed to reduce redundancy and improve data quality. Since the data was generated by AI models, minimal spelling correction was needed. No stemming or lemmatization was applied, as transformer tokenizers effectively handle subword representations. These preprocessing steps helped standardize and normalize the input, preparing the text for efficient tokenization and embedding by the respective transformer models.

D. Data Partitioning

The dataset divided into two parts using an 80:20 ratio. Out of a total of 2,786 samples, 2,228 were used for training and 558 for testing. Table II shows the distribution of data samples for each partition.

TABLE II
DATASET PARTITIONING INTO TRAINING AND TESTING SETS

Source	*Training Data*	*Testing Data*	*Total Sample*
ChatGPT	1114	279	1393
Gemini	1114	279	1393
Total	**2228**	**558**	**2786**

E. Feature Vectorization

To convert raw text into high-quality numerical representations suitable for classification tasks, we employed four cutting-edge transformer-based language models for feature extraction: BERT, ALBERT, RoBERTa, and the proposed hybrid AR-BERT. Each input text was first preprocessed and then tokenized using the respective tokenizer from the HuggingFace Transformers library. This step involved inserting special tokens such as [CLS] (used to represent the entire sentence) and [SEP] (to separate segments), applying padding or truncation to standardize input length, and generating attention masks to differentiate between real tokens and padding. The tokenized sequences were then fed into the pre-trained transformer models. From the final hidden layer, we extracted the contextual embedding corresponding to the [CLS] token, which served as a condensed fixed-length vector representation of the entire input sequence. These vectors, typically 768-dimensional, captured the semantic, syntactic, and contextual features of the text, making them highly effective for downstream machine learning classifiers. This approach enabled the model to leverage the deep language understanding embedded in each transformer architecture, enhancing the overall performance and interpretability of the classification framework.

1) BERT

Bidirectional Encoder Representations from Transformers (BERT), developed by Google [15], is one of the most influential pre-trained language models. It utilizes a deep bidirectional transformer encoder trained on large corpora, such as BookCorpus and Wikipedia, employing masked language modeling and next-sentence prediction [16]. BERT captures context from both directions, making it effective for tasks that require understanding of word relationships in a sentence.

2) ALBERT

ALBERT is an optimized and more efficient version of BERT introduced by Google Research [17]. It reduces model size significantly by employing parameter sharing across layers and factorized embedding parameterization. Despite having fewer parameters, ALBERT achieves performance comparable to or better than BERT on several NLP benchmarks. In our study, we used the Albert-base-v2 model, which maintains a good balance between computational efficiency and representational power.

3) RoBERTa

RoBERTa is an enhancement of BERT, developed by Facebook AI [18], which removes the next sentence prediction objective and trains with dynamic masking on a much larger dataset, including CommonCrawl News, WebText, and more. This leads to more robust performance across various NLP tasks. We used the roberta-base variant, which, like BERT-base, contains 12 layers and 110 million parameters, but benefits from a more optimized training strategy and larger training data.

4) AR-BERT: A Hybrid Fusion Feature Vectorization Model

To enhance model performance, we developed AR-BERT. This hybrid feature fusion architecture leverages the contex-

tual strengths of both ALBERT and RoBERTa by extracting and concatenating their respective [CLS] token embeddings. For a given input text, AR-BERT tokenizes and encodes the sequence using both models individually. It then retrieves the final hidden state of the [CLS] token from each model, forming a fused representation by concatenating the embeddings. This dual-model strategy leverages ALBERT's parameter efficiency and RoBERTa's robust contextual encoding, resulting in a richer and more discriminative text representation for downstream machine learning tasks.

F. Model Selection

For the classification task, we selected a diverse set of machine learning algorithms based on their effectiveness in handling high-dimensional feature spaces and their proven success in text classification tasks. The Support Vector Machine (SVM) was chosen for its ability to find optimal decision boundaries in complex feature spaces, particularly when working with high-dimensional embeddings, such as those from transformers. Logistic Regression (LR), a strong linear baseline, was included due to its simplicity, interpretability, and reliable performance in binary classification tasks. Decision Tree (DT) was selected for its ease of interpretation and ability to capture non-linear relationships. Random Forest (RF), an ensemble of decision trees built via bootstrap aggregation (bagging), was employed to improve generalization and reduce overfitting by combining predictions from multiple randomized trees. Gradient Boosting (GB) and Extreme Gradient Boosting (XGB) were included as powerful ensemble learners known for their robustness and ability to handle feature interactions through sequential learning. LightGBM, a fast and efficient gradient boosting framework, was incorporated for its scalability and accuracy on large datasets. To further enhance classification performance, we employed ensemble learning strategies, specifically Voting and Stacking, to combine the strengths of individual classifiers. The Voting ensemble aggregates predictions based on majority rule (hard voting) or average predicted probabilities (soft voting), improving stability and reducing the variance of individual models. In contrast, the proposed StackTrace-AI employs a Stacking ensemble that combines multiple base classifiers, including Gradient Boosting, Support Vector Machine, XGBoost, and Random Forest, whose outputs are fed into a Logistic Regression meta-classifier. This design captures complex interactions among diverse learners, enhancing generalization and predictive accuracy. As our framework is dedicated to detecting the source of generative AI text, we named the Stacking model StackTrace-AI to reflect its goal of tracing AI-generated content.

G. Model Evaluation

To evaluate the performance of the proposed classification models, we employed several widely recognized metrics, including the confusion matrix (CM), accuracy, precision, recall, F1-score and the Receiver Operating Characteristic (ROC) curve. The confusion matrix provides a visual summary of a model's predictive capability by displaying the number of correctly and incorrectly classified instances for each class. Table III outlines the core evaluation metrics and their corresponding formulas. In binary classification tasks, True Positives (TP) and True Negatives (TN) refer to cases where the model made correct predictions, while False Positives (FP) and False Negatives (FN) represent incorrect predictions.

TABLE III
PERFORMANCE EVALUATION METRICS

Metric	Formula	Description
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$	Overall correctness of predictions; measures the proportion of all correctly classified instances among total cases.
Precision	$\frac{TP}{TP+FP}$	Proportion of predicted positives that are actually true; indicates prediction exactness.
Recall	$\frac{TP}{TP+FN}$	Ability of the model to correctly identify actual positive cases.
F1-Score	$2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision}+\text{Recall}}$	Harmonic mean of precision and recall; balances false positives and false negatives.

H. Algorithm for GenAI Source Detection

Algorithm 1 presents the step-by-step workflow of the proposed generative AI source detection framework. The process begins with initializing key components, including the dataset, preprocessing steps, transformer-based feature extractors (BERT, ALBERT, RoBERTa, Hybrid AR-BERT), and a set of machine learning classifiers. Text data is preprocessed through standard cleaning operations such as stop word removal, punctuation handling, and tokenization. Embeddings are generated using transformer models, then reduced in dimension for computational efficiency. The system applies a range of classifiers, including SVM, LR, RF, DT, GB, XGB, LGBM, Voting, and Stacking, trained on an 80:20 train-test split. The performance of each classifier is evaluated using accuracy, precision, recall, and F1-score to determine the most effective configuration for identifying the source (ChatGPT or Gemini) of AI-generated text.

Algorithm 1: Framework for Detecting Generative AI Text Sources

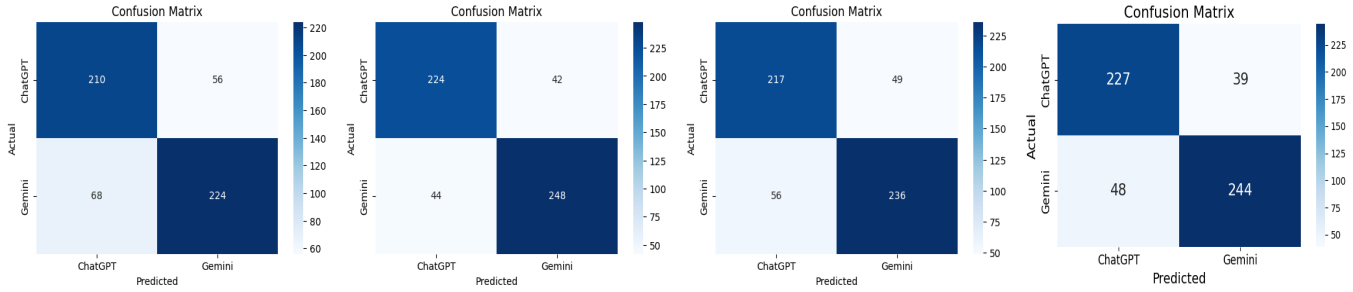
Initialize: $Dataset$, $PreProc$, $Embedder$, $Reducer$, $ClassifierSet$; // Set up modules
$Dataset \leftarrow$ Input Data, $PreProc \leftarrow$ Preprocessing Steps ;
$Embedder \leftarrow$ Embedding Techniques, $Reducer \leftarrow$ Dimensionality Reducers ;
$ClassifierSet \leftarrow$ Classification Models ;
$PreProc =$ [Token Clean-up, Lowercasing, Stemming, Normalizing] ;
$Embedder =$ [BERT, ALBERT, RoBERTa, AR-BERT] ;
$ClassifierSet =$ [LR, SVM, RF, DT, GB, XGB, LXGB, Voting, Stacking] ;
$X \leftarrow Dataset$[Text], $Y \leftarrow Dataset$[Class] ;
foreach $emb \in Embedder$ **do**
  $Features \leftarrow emb(X)$; // Extract contextual embeddings
  foreach $red \in Reducer$ **do**
    $ReducedFeatures \leftarrow red(Features)$; // Dimensionality compression
    foreach $clf \in ClassifierSet$ **do**
      $X_{train}, X_{test}, Y_{train}, Y_{test} \leftarrow$ Split$(ReducedFeatures, Y, 0.8)$;
      clf.fit(X_{train}, Y_{train}) ; // Model training
      $Y_{pred} \leftarrow clf$.predict(X_{test}) ; // Generate predictions
      Display $\rightarrow$ [Accuracy, Precision, Recall, F1] ;
Finalize: Clear all initialized components ; // Cleanup process

IV. RESULT ANALYSIS & DISCUSSION

This section provides a comparative analysis of the performance of various machine learning models using different

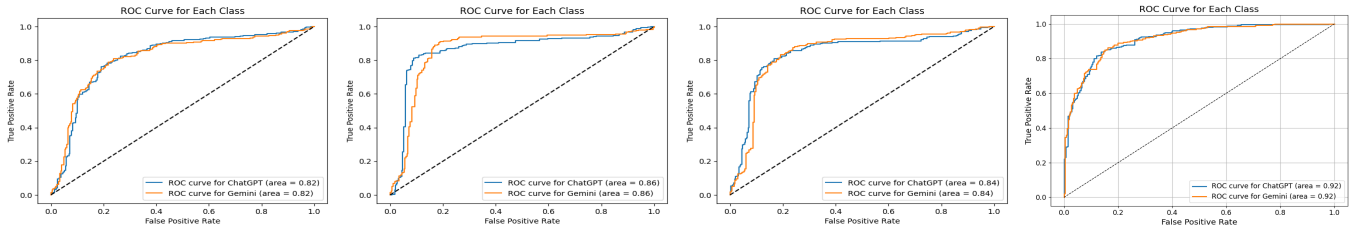
TABLE IV
PERFORMANCE COMPARISON OF VARIOUS CLASSIFIERS USING DIFFERENT TRANSFORMER-BASED FEATURE VECTORIZATION TECHNIQUES. METRICS INCLUDE ACCURACY (A), PRECISION (P), RECALL (R), AND F1-SCORE (F1).

	BERT				ALBERT				RoBERTa				hybrid(ALBERT+RoBERTa)			
Classifier	A (%)	P (%)	R (%)	F1 (%)	A (%)	P (%)	R (%)	F1 (%)	A (%)	P (%)	R (%)	F1 (%)	A (%)	P (%)	R (%)	F1 (%)
LR	75.48	75.55	75.48	75.49	82.08	82.16	82.08	82.08	77.49	77.60	77.49	77.47	82.29	82.38	82.29	82.30
SVM	74.37	74.44	74.37	74.38	80.54	80.60	80.54	80.54	79.35	79.45	79.35	79.35	80.68	80.72	80.68	80.69
RF	62.51	62.80	62.51	62.44	75.56	75.65	75.56	75.56	70.39	70.52	70.39	70.39	75.70	75.81	75.70	75.70
DT	55.73	55.85	55.73	55.74	64.62	64.78	64.62	64.63	59.21	59.29	59.21	59.21	62.97	6297	63.09	62.97
GB	67.60	67.77	67.60	67.58	76.45	76.53	76.45	76.46	74.95	75.02	74.95	74.94	78.24	78.29	78.24	78.24
XGB	67.28	67.38	67.28	67.26	77.38	77.49	77.38	77.39	74.34	74.42	74.34	74.34	79.21	79.29	79.21	79.21
LXGB	67.49	67.69	67.49	67.48	78.75	78.81	78.75	78.75	75.59	75.61	75.59	75.59	79.50	79.53	79.50	79.50
Voting	77.00	77.00	77.00	77.00	83.00	83.00	83.00	83.00	81.54	81.59	81.54	81.55	84.16	84.19	84.16	84.16
Stacking	**78.00**	**78.00**	**78.00**	**78.00**	**81.90**	**81.98**	**81.90**	**81.90**	**81.81**	**81.79**	**81.72**	**81.72**	**84.44**	**84.46**	**84.44**	**84.44**



(a) (b) (c) (d)

Fig. 3. The confusion matrices present a comparative analysis of classification performance across three individual feature extraction methods and the proposed Hybrid AR-BERT model: (a) BERT with Stacked Model, (b) ALBERT with Stacked Model, (c) RoBERTa with Stacked Model, and (d) the proposed Hybrid AR-BERT with Stacked Model. These matrices highlight the model predictions across the two classes, reflecting the accuracy and misclassification rates for each approach.



(a) (b) (c) (d)

Fig. 4. The ROC Curves illustrate the performance comparison of three individual feature extraction techniques with the proposed Hybrid AR-BERT model: (a) BERT with Stacked Model, (b) ALBERT with Stacked Model, (c) RoBERTa with Stacked Model, and (d) the proposed Hybrid AR-BERT with Stacked Model. Each curve displays the classification performance across two classes (0 and 1) based on the corresponding ROC values.

feature vectorization techniques for AI source identification. Section 5-A presents the overall results, while Section 5-B discusses the findings and outlines the limitations of the study.

A. Result Analysis

Table IV compares the performance of four transformer-based feature vectorization techniques, BERT, ALBERT, RoBERTa, and hybrid AR-BERT combined with nine machine learning classifiers: LR, SVM, RF, DT, GB, XGB, LXGB, Voting, and Stacking. Notably, the stacked model using the hybrid AR-BERT vectorizer outperforms all other combinations, achieving the highest accuracy of 84.44%, along with 84.46% recall, precision, and F1-score value. Overall, stacked models across all vectorizers demonstrate outstanding performance compared to individual classifiers. The stack model, when combined with three different vectorizers, BERT, ALBERT, and RoBERTa, achieves accuracy scores of 78%, 81.90%, and 81.81%, respectively. Corresponding precision values are 78%, 81.98%, and 81.79%; recall scores are 78%, 81.90%, and 81.72%; and F1-scores are 78%, 81.90%, and 81.71%. These results indicate that the stack model outperforms all other classifiers across all evaluation metrics. In contrast, the Decision Tree (DT) classifier exhibits the weakest performance, as evident from the comparison table.

B. Confusion Matrix and ROC Curve

Figure 3 presents the confusion matrices for three transformer-based and one hybrid transformer-based feature vectorization techniques, all evaluated using a stacked ensemble model on the GenAI-generated text dataset. Notably, Figure 3(d) illustrates the performance of the hybrid AR-BERT with the stacked model, achieving 227 and 244 correct classifications, while recording 39 and 48 misclassifications, respectively.

Figure 4 illustrates the ROC curves comparing the performance of three individual feature extraction architectures and the proposed hybrid AR-BERT with the stacked model. As shown in Figure 4(d), the hybrid AR-BERT achieves the highest ROC scores—0.92 for ChatGPT and 0.91 for Gemini. Notably, it consistently outperforms all individual models, demonstrating its effectiveness in capturing richer contextual representations.

C. Discussion & Limitations

After analyzing the experimental outcomes, it is evident that the hybrid AR-BERT embedding combined with a stacking ensemble classifier delivers the most effective performance in distinguishing between text generated by ChatGPT and Gemini. This combination consistently outperformed all other configurations in terms of accuracy, precision, recall, and F1-score. The success of AR-BERT underscores the effectiveness of combining multiple transformer embeddings (ALBERT, and RoBERTa) to capture a more comprehensive set of semantic and contextual features. Meanwhile, the stacking ensemble, StackTrace-AI, was able to leverage the strengths of individual base classifiers, resulting in improved generalization and classification robustness. Despite these promising results, the research has several drawbacks. First, the dataset was limited to 2786 text samples generated from computer science and engineering-related queries. As a result, the model's generalizability to other domains or less-structured prompts remains uncertain. Second, only two generative AI models (ChatGPT and Gemini) were evaluated, which restricts the applicability of the framework to a broader range of large language models. Additionally, while transformer embeddings were highly effective, they are computationally expensive, and the dimensionality reduction process, although helpful, may still lead to a loss of fine-grained contextual information. Lastly, the study assumes all generated responses are free of prompt formatting bias or system-level noise, which may not always hold in real-world scenarios.

V. CONCLUSION & FUTURE WORK

This study presents a comprehensive framework for detecting the source of AI-generated text, specifically distinguishing between responses generated by ChatGPT and Gemini. By leveraging the representational strength of four transformer-based models (BERT, ALBERT, RoBERTa, and a proposed hybrid AR-BERT) and combining them with nine traditional and ensemble-based machine learning classifiers, the research demonstrates that the stacking ensemble applied to the hybrid AR-BERT embeddings achieves the highest classification performance across all metrics. Our approach highlights the potential of combining multiple transformer embeddings to construct richer feature representations, which, when paired with robust ensemble methods, significantly enhance source classification accuracy. The framework also incorporates dimensionality reduction and visualization techniques to improve interpretability and scalability. This work contributes to the growing field of generative AI forensics, emphasizing the importance of identifying the origin of machine-generated content for the sake of transparency, content authentication, and ethical AI governance. In the future, we aim to expand the dataset across more domains, including different languages, and include additional generative models such as Claude, Copilot Mistral, or open-source LLMs. Investigating prompt engineering strategies, adversarial text generation, and fine-tuning transformer models on source detection tasks could also further improve accuracy and robustness.

REFERENCES

[1] S. Feuerriegel, J. Hartmann, C. Janiesch, and P. Zschech, "Generative ai," *Business & Information Systems Engineering*, vol. 66, no. 1, pp. 111–126, 2024.

[2] P. P. Ray, "Chatgpt: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope," *Internet of Things and Cyber-Physical Systems*, vol. 3, pp. 121–154, 2023.

[3] G. Team, R. Anil, S. Borgeaud, and J.-B. A. et al., "Gemini: A family of highly capable multimodal models," 2025.

[4] A. Jo, "The promise and peril of generative ai," *Nature*, vol. 614, no. 1, pp. 214–216, 2023.

[5] D. Dalalah and O. M. Dalalah, "The false positives and false negatives of generative ai detection tools in education and academic research: The case of chatgpt," *The International Journal of Management Education*, vol. 21, no. 2, p. 100822, 2023.

[6] M. T. Baldassarre, D. Caivano, B. Fernandez Nieto, D. Gigante, and A. Ragone, "The social impact of generative ai: An analysis on chatgpt," in *Proceedings of the 2023 ACM Conference on Information Technology for Social Good*, pp. 363–373, 2023.

[7] D. I. Uchendu, K. Lucas, A. Ororbia, E. Hovy, and S. Muresan, "Authorship attribution for neural text generation," in *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 3969–3983, 2021.

[8] H. Sharma, R. Kumar, *et al.*, "Ai-generated text detection: Recent advances and challenges," *arXiv preprint arXiv:2302.10368*, 2023.

[9] A. Yadagiri and P. Pakray, "Deep learning strategies for identifying machine-generated text," in *2025 19th International Conference on Ubiquitous Information Management and Communication (IMCOM)*, pp. 1–8, IEEE, 2025.

[10] A. Al Bataineh, R. Sickler, K. Kurcz, and K. Pedersen, "Ai-generated vs. human text: Introducing a new dataset for benchmarking and analysis," *IEEE Transactions on Artificial Intelligence*, 2025.

[11] M. S. Ibrahim, J. A. Eleiwy, H. M. Muhi-Aldeen, Y. Al-Yasiri, and A. A. Nafea, "Human to chatbot text classification using multi-source ai chatbots and machine learning models," *Journal of Intelligent Systems & Internet of Things*, vol. 16, no. 1, 2025.

[12] Y. Zhang, Q. Leng, M. Zhu, R. Ding, Y. Wu, J. Song, and Y. Gong, "Enhancing text authenticity: A novel hybrid approach for ai-generated text detection," in *2024 IEEE 4th International Conference on Electronic Technology, Communication and Information (ICETCI)*, pp. 433–438, IEEE, 2024.

[13] B. Alhijawi, R. Jarrar, A. AbuAlRub, and A. Bader, "Deep learning detection method for large language models-generated scientific content," *Neural Computing and Applications*, vol. 37, no. 1, pp. 91–104, 2025.

[14] M. A. Arshed, S. C. Gherghina, C. Dewi, A. Iqbal, and S. Mumtaz, "Unveiling ai-generated financial text: A computational approach using natural language processing and generative artificial intelligence," *Computation*, vol. 12, no. 5, p. 101, 2024.

[15] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pp. 4171–4186, 2019.

[16] M. S. Hossen, M. Saiduzzaman, and P. Shaha, "Social media sentiments analysis on the july revolution in bangladesh: A hybrid transformer based machine learning approach," in *Proceedings of the 17th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, IEEE, 2025.

[17] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," in *International Conference on Learning Representations*, 2020.

[18] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.