ISSN 2007-9737

Detecting AI-Generated Text Using Machine Learning and Deep Learning Approaches

Annepaka Yadagiri, L D M S Sai Teja, Partha Pakray*, Chukhu Chunka

Department of Computer Science & Engineering,
National Institute of Technology Silchar,
India

{annepaka22_rs, lekkalad_ug, partha, chukhu}@cse.nits.ac.in

Abstract. Recent advancements in natural language processing have the potential to enable artificial intelligence systems to produce text indistinguishable from human-authored content. Such developments could lead to significant ethical, legal, and societal implications. This study addresses this challenge by designing a robust AI detection model to differentiate between AI-generated and human-written text. To achieve this, we employ k-fold cross-validation to evaluate a range of established machine learning and deep learning models, including Logistic Regression, Extra Trees Classifier, CNN, RNN, and LSTM networks. Our experimental results reveal that the CNN-based model outperforms other approaches in accurately identifying AI-generated content. In addition to presenting our findings, we thoroughly review existing research in AI-generated text detection, comprehensively analyzing current methodologies and their limitations. Our testing demonstrates promising outcomes, with the proposed CNN-based approach emerging as the most effective solution. Specifically, the LSTM and RNN models achieve accuracies of 0.83, while the Detect_CNN model attains the highest accuracy of 0.85. Beyond technical performance, we also explore the broader societal implications of this research, emphasizing its potential benefits across various sectors. Furthermore, we address critical ethical considerations and environmental sustainability concerns, underscoring the need for responsible development and deployment of such technologies.

Keywords. Convolutional neural networks, machine learning classifiers, natural language processing.

1 Introduction

Artificial Intelligence (*AI*) [19] has seen remarkable progress in recent years, spanning from generative models in computer vision [29, 31] to Large Language Models (*LLMs*) in Natural Language Processing (*NLP*) [6, 39]. LLMs are now capable of generating high-quality text with broad applicability. One of the most prominent examples is the release of ChatGPT [1] by OpenAI, a versatile language model capable of handling a wide range of linguistic tasks, such as question answering [36], creative writing [4], and providing personal assistance [33]. However, this growing capability also raises the critical need to detect text generated by ChatGPT to distinguish it from human-written content, ensuring its responsible use. could be exploited to produce fake news or fraudulent reviews [21], contributing to public misinformation and manipulating web content for social engineering in ways that can have negative impacts on society [2, 37]. Some news articles rewritten by AI have led to many fundamental errors [9]. Additionally, ChatGPT may be misused for plagiarism, infringing on intellectual property rights [12]. These potential abuses of AI-generated content pose significant societal risks that necessitate proactive regulation and detection methods.

[1] https://chatgpt.com/

Natural Language Generation (*NLG*) models have demonstrated significant potential for AI-based writing assistants, particularly those built on large pre-trained NLG models. The quality of AI-generated text in terms of coherence, consistency, and grammar has shown continuous improvement, progressing from GPT-2 [28] to GPT-3 [6], and later to InstructGPT [25]. These advancements in NLG technology have enhanced tools like autocomplete and facilitated more complex and controllable writing processes [35].

Large language models [39] fine-tuned on GPT-3.5 with Reinforcement Learning from Human Feedback (*RLHF*) have attracted significant interest. ChatGPT has shown impressive capabilities across various tasks, particularly in producing text that closely mimics human writing. According to findings by [15], the Turing test highlights the difficulty that individuals unfamiliar with ChatGPT have in differentiating between text generated by the model and text composed by humans.

In the next two years, it is predicted that 99% of the content on the internet could be generated by AI. The growing prevalence of AI-generated content on social media raises concerns about the potential spread of disinformation and harmful narratives as distinguishing such content from human-written text becomes increasingly challenging [15]. As a result, identifying text generated by LLMs has become a crucial research focus. Effective detection methods can enhance information oversight and accountability, thereby clarifying the sources of information.

This paper is organized as follows: Section 2 examines the relevant literature, while Section 3 details the proposed model. Subsequently, Section 5 presents the analytical outcomes, and Section 6 provides concluding remarks on the study.

2 Related Work

The machine learning(*ML*) algorithm proposed in [23] was tested against the widely-used Generative Pretrained Transformer (*GPT*) to evaluate its ability to differentiate between AI-generated text and human-written text. Assessing the effectiveness of ML techniques in differentiating between AI-generated and human-written text, [3] collected responses from computer science students on essay and programming tasks. Using this dataset, the authors trained and evaluated various ML models, including Support Vector Machines (*SVM*), Logistic Regression (*LR*), Neural Networks (*NN*), Random Forests (*RF*), and Decision Trees (*DT*).

A novel approach for distinguishing between human-written and AI-generated texts using language models was proposed by [7]. The researchers compiled and publicly shared a dataset named OpenGPT Text, comprising rephrased outputs from ChatGPT following preprocessing steps. A preliminary evaluation of ChatGPT's learning effectiveness was conducted by [26], comparing the impact of its hints with those provided by human tutors on two algebraic subjects: elementary and intermediate algebra.

A method to distinguish publications generated by ChatGPT from those authored by researchers was demonstrated by [16]. Using a supervised ML methodology, the research demonstrated effective techniques for distinguishing AI-generated articles from those authored by scientists. Their algorithmic technique achieved high precision in identifying ChatGPT-generated publications. When students utilize AI tools, particularly LLMs like ChatGPT and Gemini, in formal assessments, the implications of academic integrity were investigated by [27]. The research investigated the development of these technologies and emphasized the role of LLMs in supporting students' digital writing education, encompassing composition and instructional practices. Additionally, the authors examined potential human-AI collaborations, improvements in Automated Writing Evaluation (*AWE*) systems, and the advantages for English as a Foreign Language (*EFL*) learners.

Chat2VIS, a novel system leveraging LLMs to address the complexities of language understanding, was introduced by [22]. The system provides more straightforward and accurate end-to-end solutions compared to traditional approaches. Chat2VIS illustrates the capability of LLMs to consistently produce accurate visualizations from natural language queries, even when the input is vague or contains significant inaccuracies. Research on detecting AI-generated text is still

limited, showing the need for better detection methods. As Deep Learning (*DL*) models advance, the number of parameters increases, which can lead to overfitting. Critical parameters such as the number of epochs, batch size, and learning rate play a significant role in determining the performance of CNNs. Manual tuning these hyperparameters is time-consuming and error-prone, making optimization algorithms essential. This research emphasizes enhancing the learning rate and refining the selection process for CNN models.

The key contributions of this paper are as follows:

1. We have utilized the HC3-Plus dataset, a robust benchmark dataset, for our problem statement. To ensure the reliability and generalizability of our results, we employed k-fold cross-validation during the evaluation process.
2. We implemented advanced feature extraction methods, including TF-IDF vectorization, to preprocess and transform the textual data into a suitable format for analysis.
3. We introduced a novel Detect_CNN model specifically designed for the task. Our proposed model demonstrated superior performance to existing approaches, achieving state-of-the-art results on the HC3-Plus dataset.

3 Proposed Methodology

3.1 Problem Statement

In this research, we aim to identify AI-generated text using the HC3 Plus dataset. Our dataset D consists of n text samples, each classified as either human-generated (H) or AI-generated (A). The objective is to develop a robust classification model $C : D \rightarrow \{H, A\}$.

To formalize the problem, we define:

— $D_H \subset D$: the subset of samples that are human-generated.

— $D_A \subset D$: the subset of samples that are AI-generated.

— The HC3 Plus dataset, denoted as $D_{HC3Plus}$, will serve as our training set, providing features X and corresponding labels Y.

Our primary goal is to minimize the classification error when identifying AI-generated text. The classification error can be mathematically expressed as:

$$\text{Error}(C) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}(C(d_i) \neq Y_i),$$

where: - $\mathbb{I}$ is the indicator function defined as:

$$\mathbb{I}(C(d_i) \neq Y_i) = \begin{cases} 1 & \text{if } C(d_i) \text{ is not equal to } Y_i \\ 0 & \text{otherwise.} \end{cases}$$

We explored various feature extraction techniques and classification algorithms to enhance the model's ability to differentiate between human and AI-generated texts.

3.2 Dataset Description

In this paper, the HC3-Plus dataset will be used to identify AI-generated text, including several widely used high-quality corpora.

The CNN/DailyMail dataset [32] is an English-language resource comprising a vast collection of distinct news articles authored by journalists from CNN and the Daily Mail. Each entry includes both a news article and its corresponding highlights. The Xsum dataset [24] is an English-language collection encompassing a diverse array of topics and features a BBC article and a corresponding one-sentence summary. The LCSTS dataset [18] is a substantial collection for short-text summarization in Chinese, derived from naturally annotated web content on Sina Weibo, a social media platform similar to Twitter. The news2016 corpus, introduced by CLUE benchmark [38], is sourced from the Chinese We Media (self-media) platform. Each entry includes an article along with its associated title.

Finally, HC3-SI (*HC3 Semantic Invariance*), being approximately twice the size of HC3, was merged with HC3 to form an expanded dataset referred to as HC3-Plus [34]. The detailed statistics of the HC3-Plus dataset, including the labels for human (0) and machine (1) text, along with their

train, validation, and test splits, are summarized in Table 1. Fig 1 illustrates a graphical representation of the dataset distribution.

Table 1. HC3-Plus dataset statistics

Label	Train	Valid	Test
Human (0)	83,582	9,349	8,529
Machine (1)	64,820	7,148	6,311
Total	**148,402**	**16,492**	**14,840**

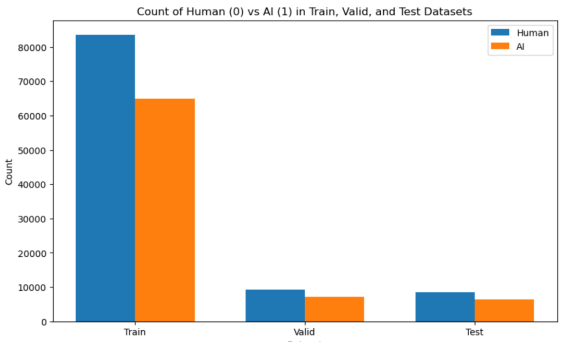

Fig. 1. Graphical representation of the dataset

4 Proposed Model Approach

This paper introduces an automated model for detecting human-generated and AI-generated text called the Detect_CNN model. The goal of the proposed Detect_CNN method is to examine the model's decision-making process and uncover any identifiable patterns. The model involves four stages: feature extraction using word embedding, application of traditional methods with k-fold cross-validation for optimal accuracy, and generation of classification reports including accuracy, precision, recall, F1-scores, MCC, and CNN-based hyperparameter tuning. The overall flow of the Detect_CNN approach is depicted in Fig 2.

4.1 Feature Extraction

In this study, Term Frequency-Inverse Document Frequency (*TF-IDF*) [1], a well-established method for word vectorization, is utilized to emphasize the most significant words in a document. TF-IDF is applied for generating word embeddings, whereas the count vectorizer technique is employed for feature extraction. Feature extraction in data mining involves simplifying the data and making it more manageable, especially when working with large datasets. One of the main challenges in analyzing complex text arises from numerous variables.

Typically, processing large or complex texts requires significant memory and computational resources. This frequently results in applying classification methods that excel on training data but struggle to generalize effectively to unseen instances. The authors highlighted that feature extraction, particularly in applications with many features, serves a similar function to dimensionality reduction. Applying feature extraction methods to input data before it enters the classifier can achieve more refined results and improved classification performance.

4.2 Term Frequency (*TF*)

The process of calculating *TF-IDF* consists of two steps. The first step is to compute the term frequency (*TF*), which measures how frequently a term (either a word or a phrase) appears within a document. Let the term be represented by t, the document by d, the entire set of documents (*corpus*) by D, and the total number of documents by N. The TF is calculated by dividing the count of a term's occurrences in a document by the total number of terms present. Equation 1 provides the mathematical representation of this TF calculation.

$$TF(t,d) = \frac{n_{t,d}}{\sum_k n_{k,d}}, \tag{1}$$

where $n_{t,d}$ is the number of occurrences of term t in document d, and $\sum_k n_{k,d}$ represents the total number of terms in document d.

4.3 Inverse Document Frequency (*IDF*)

It quantifies the importance of a term by evaluating how frequently it appears across a collection of documents (*corpus*). As shown in Equation 2, IDF is computed by taking the logarithm of the ratio of

Table 2. Hyperparameters utilized across all experiments

Parameter	Value
Activation Functions	Sigmoid, Softmax
Optimizer	AdamW, Adam
Loss Function	binary_crossentropy
Learning Rate	1e-5, 2e-5
Batch Size	16, 32
Number of Epochs	05
Dropout	0.2
ModelCheckpoint	Yes
EarlyStopping	Yes
Patience	02

the total number of documents in the corpus to the number of documents containing the specific term is calculated:

$$IDF(t, D) = \log\left(\frac{N}{df(t)}\right). \quad (2)$$

4.4 TF-IDF

Finally, the TF-IDF score is determined by multiplying the TF and IDF values derived from the previous steps, as represented in Equation 3.

The TF-IDF score is computed as:

$$tf\text{-}idf = tf_{t,d} \times \log(idf). \quad (3)$$

Additionally, this study employed a word embedding technique for feature extraction.

4.5 Model Selection

In this section, we have selected both ML and DL models for experimental purposes.

Logistic Regression: It is a statistical technique primarily employed for binary classification tasks. It models the relationship between the input variables and the target class to generate predictions [11].

Support Vector Machines: It is used for classification and regression tasks. The main objective of SVM is to identify a hyperplane in an N-dimensional space that effectively separates the data points [10].

Decision Tree: It creates a tree-like structure based on patterns identified in the data. The model traverses this tree during inference to determine the most suitable class [5].

Random Forest: It is a supervised ML algorithm that aggregates multiple decision trees using the technique of attribute bagging.

AdaBoost: AdaBoost, or Adaptive Boosting, is an ensemble boosting algorithm in ML. It is particularly effective when dealing with noisy data [13].

Bagging Classifier: The Bagging Classifier is an ensemble meta-estimator that fits base classifiers on randomly sampled subsets of the original dataset. It then aggregates the predictions from each classifier through averaging or voting to produce an outcome.

Multi-layer Perceptron: It is based on artificial neurons, interconnected units or nodes that mimic the neurons in a biological brain. Each connection between neurons, similar to synapses in the brain, enables the transfer of signals to neighboring neurons. After receiving input signals, an artificial neuron processes them and sends output to connected neurons. The output of each neuron is determined by a non-linear function applied to the sum of its inputs [30].

Long Short-Term Memory: Long Short-Term Memory (*LSTM*) is a recurrent neural network that retains information over long sequences, making it suitable for tasks involving temporal dependencies [17].

Extremely Randomized Trees: Extremely Randomized Trees (*Extra Trees*) share similarities with Random Forests, as they construct multiple decision trees using the entire dataset during the training process. However, unlike Random Forest, which splits data based on the best splitting criteria, Extra Trees performs random splits at each decision node [14].

Convolutional Neural Networks (*CNN*): CNNs represent a category of DL models specifically developed for analyzing structured data, including images and sequential inputs. These networks utilize convolutional layers to capture spatial feature hierarchies, allowing them to recognize patterns while maintaining translation invariance. These models have been widely adopted for

various tasks, including image classification, natural language processing, and AI-generated text detection, due to their robust feature extraction capabilities [20].

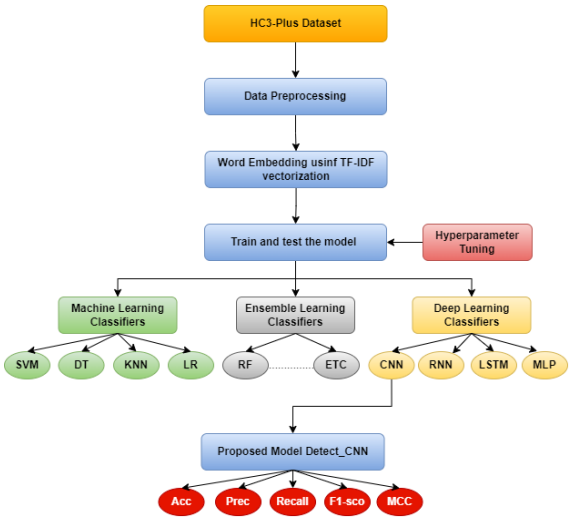

Fig. 2. Proposed model architecture

5 Results and Discussion

5.1 Experimental Setup

The experiment was carried out using a Jupyter Notebook on a machine featuring an *Intel® Xeon® W-2155 CPU @ 3.30GHz with 20 cores and an NVIDIA Quadro P2000 GPU* for processing LLM tasks. The system was further equipped with 64 GB of RAM. Python was used as the programming language, along with libraries such as `Numpy`, `Pandas`, `SKlearn`, and `TensorFlow`.

Evaluation metrics assess the model's performance, offering different perspectives. This study used Performance evaluation metrics such as accuracy, precision, recall, F1 score, and Matthews Correlation Coefficient (*MCC*), commonly used to assess the effectiveness of ML models.

These metrics provide insights into different aspects of classification performance, including overall correctness, positive prediction reliability, sensitivity to actual positives, harmonic balance between precision and recall, and a comprehensive assessment of model performance considering all confusion matrix components.

Accuracy quantifies the proportion of correctly classified instances, as defined in Equation 4. It is computed by dividing the total number of correctly predicted samples by the overall number of samples:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}. \tag{4}$$

Precision represents the proportion of correctly predicted positive instances among all instances classified as positive, whereas **Recall** indicates the fraction of actual positive cases correctly identified. The mathematical expressions for precision and recall are provided in Equations 5 and 6, respectively:

$$\text{Precision} = \frac{TP}{TP + FP}, \tag{5}$$

$$\text{Recall} = \frac{TP}{TP + FN}. \tag{6}$$

As presented in Equation 7, the **F1 score** is the harmonic mean of precision and recall, offering a more balanced evaluation compared to precision and recall alone:

$$\text{F1-score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{7}$$

Lastly, the MCC is a more stable metric that considers all four components of the confusion matrix. This metric is regarded as more comprehensive and reliable than the others mentioned above [8]. Equation 8 provides the formula for computing MCC:

$$\text{MCC} = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}. \tag{8}$$

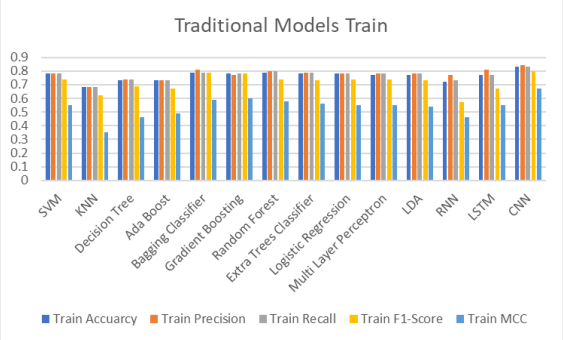

Fig. 3. Traditional models train evaluation metrics

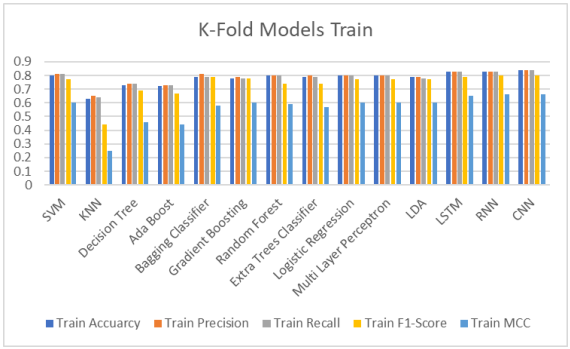

Fig. 5. K-fold models train evaluation metrics

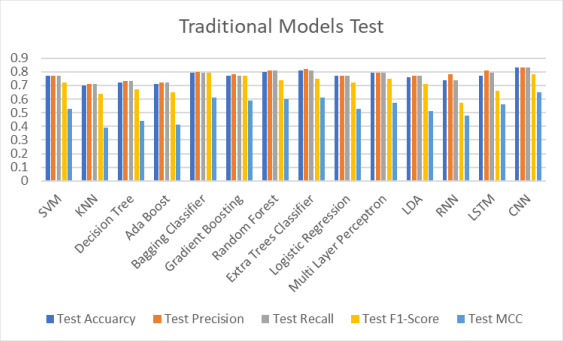

Fig. 4. Traditional models test evaluation metrics

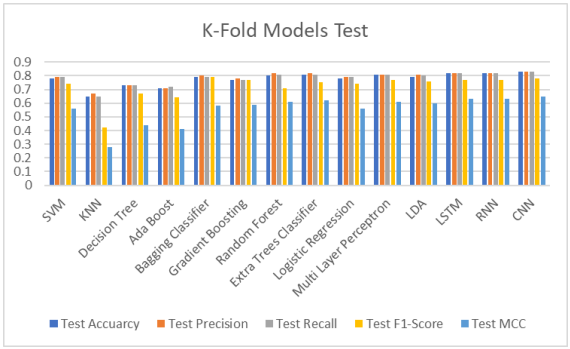

Fig. 6. K-fold models test evaluation metrics

5.2 Experimental Results

We evaluated 14 models from traditional ML, Ensemble learning, and DL categories, as detailed evaluated results in Tables 3, 4, 5, and 6. Among all the models, the (*CNN*) consistently delivered the best performance in both the training and testing phases, whether using non-k-fold or k-fold cross-validation methods.

In the training phase, using various hyperparameter tuning from Table 2 to all the models, the CNN achieved a high accuracy of 83% with non-k-fold validation and 85% with k-fold cross-validation using (*5-fold*), which can be seen in Figures 3 and 5.

Similarly, in the testing phase, the CNN model demonstrated superior performance with an accuracy of 83% in non-k-fold validation confusion matrices as shown in Fig. 7 and 84% in k-fold cross-validation confusion matrices as shown in Fig. 8, which can be seen in Figures 4 and 6. The CNN model outperformed traditional and ensemble models and other DL models, such as Recurrent Neural Networks (*RNN*) and LSTM.

Furthermore, the CNN model exhibited the highest scores across all evaluation metrics, including precision, recall, F1-score, and MCC. After the CNN, the best-performing models were Random Forest, Extra Trees Classifier, and Multi-Layer Perceptron (*MLP*), surpassing the RNN and LSTM models in overall performance.

6 Conclusion

Based on the findings of this study, we conclude that the proposed Detect_CNN model, utilizing CNN, effectively distinguishes between AI-generated and human-written text. Our approach leverages k-fold cross-validation to evaluate various ML, ensemble, and DL models. Utilizing the HC3 Plus dataset effectively addresses challenges

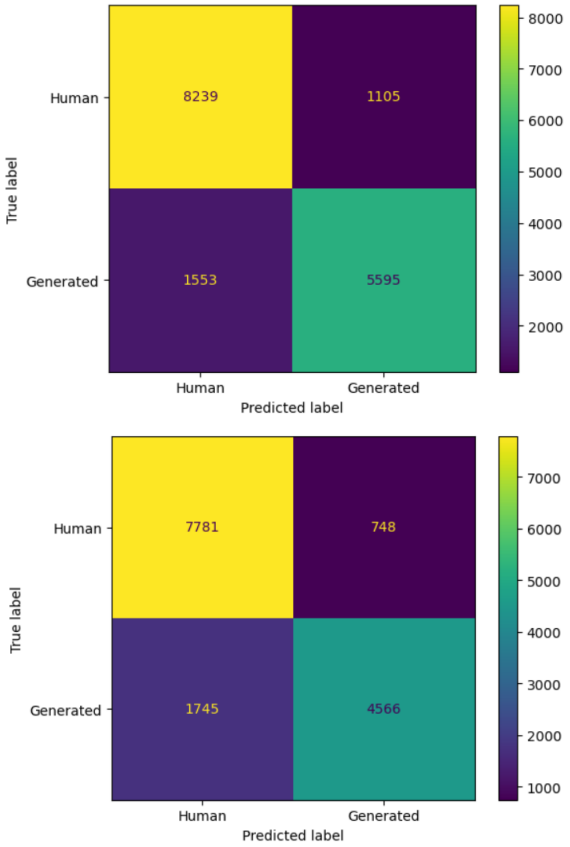

Fig. 7. Detect_CNN train and test confusion matrices (Non-k-fold)

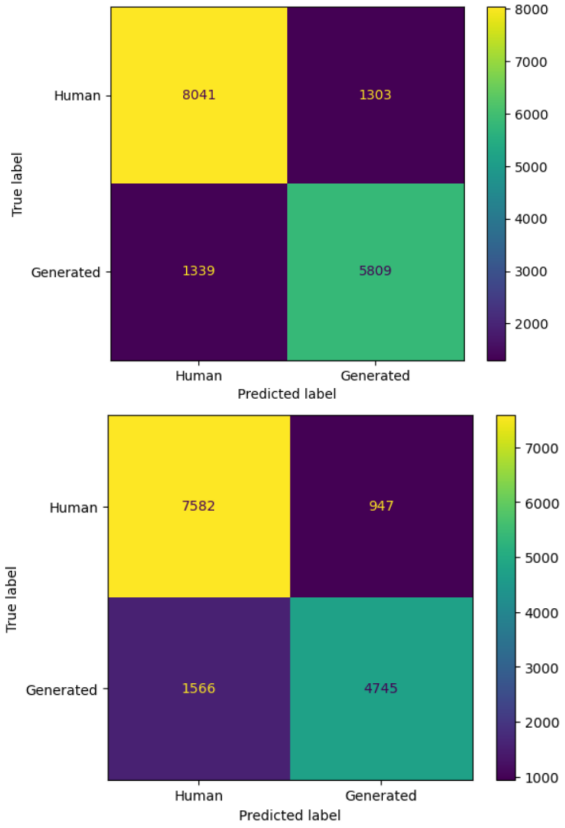

Fig. 8. Detect_CNN train and test confusion matrices (k-fold)

in accurately identifying AI-generated content in the context of AI-generated content. Among the models tested, Detect_CNN outperformed others, achieving the highest accuracy of 0.85, followed by LSTM and RNN, which achieved accuracy of 0.83.

This research also comprehensively analyzes the current state of AI-generated text detection, highlighting the importance of accurate identification methods for addressing ethical, legal, and social implications.

The successful application of CNN demonstrates its potential for practical deployment in real-world AI detection systems, offering significant benefits for industries such as education, publishing, and cybersecurity.

6.1 Limitations and Future Work

Although this study offers essential insights into detecting AI-generated text, several limitations should be considered. The models, including machine learning, ensemble learning, and neural networks like CNN, RNN, and LSTM, were evaluated using a specific dataset, which may not capture the full diversity of AI-generated content across different domains and contexts. This limits the generalizability of the findings, highlighting the need for future work with more diverse datasets.

Although the CNN model achieved the highest accuracy, its performance may vary with newer text generation systems like advanced GPT models. Additionally, deep learning models, particularly CNN, require substantial computational

Table 3. Performance metrics for various traditional models on training data

Models	Acc	Prec	Recall	F1-S	MCC
SVM	0.78	0.78	0.78	0.74	0.55
KNN	0.68	0.68	0.68	0.62	0.35
DT	0.73	0.74	0.74	0.69	0.46
AB	0.73	0.73	0.73	0.67	0.49
BC	0.79	0.81	0.79	0.79	0.59
GB	0.78	0.77	0.78	0.78	0.6
RF	0.79	0.8	0.8	0.74	0.58
ETC	0.78	0.79	0.79	0.73	0.56
LR	0.78	0.78	0.78	0.74	0.55
MLP	0.77	0.78	0.78	0.74	0.55
LDA	0.77	0.78	0.78	0.73	0.54
RNN	0.72	0.77	0.73	0.57	0.46
LSTM	0.77	0.81	0.77	0.67	0.55
Detect_CNN	**0.83**	**0.84**	**0.83**	**0.8**	**0.67**

Table 4. Performance metrics for various traditional models on test data

Models	Acc	Prec	Recall	F1-S	MCC
SVM	0.77	0.77	0.77	0.72	0.53
KNN	0.7	0.71	0.71	0.64	0.39
DT	0.72	0.73	0.73	0.67	0.44
AB	0.71	0.72	0.72	0.65	0.41
BC	0.79	0.8	0.79	0.79	0.61
GB	0.77	0.78	0.77	0.77	0.59
RF	0.8	0.81	0.81	0.74	0.6
ETC	0.81	0.82	0.81	0.75	0.61
LR	0.77	0.77	0.77	0.72	0.53
MLP	0.79	0.79	0.79	0.75	0.57
LDA	0.76	0.77	0.77	0.71	0.51
RNN	0.74	0.78	0.74	0.57	0.48
LSTM	0.77	0.81	0.79	0.66	0.56
Detect_CNN	**0.83**	**0.83**	**0.83**	**0.78**	**0.65**

resources, which could limit their deployment in resource-constrained environments. The manual hyperparameter tuning process is also time-consuming and can introduce variability.

Ethical considerations such as privacy, fairness, and the potential misuse of AI detection systems are critical and should be addressed in future research.

For future work, we plan to explore hybrid deep learning models (*e.g., CNN-RNN, CNN-LSTM*), optimize hyperparameters, and incorporate transformer-based models like BERT, RoBERTa, and DistilBERT to improve model robustness and adaptability.

Table 5. Performance metrics for various K-Fold models on training data.

K-Fold Models	Acc	Prec	Recall	F1-S	MCC
SVM	0.8	0.81	0.81	0.77	0.6
KNN	0.63	0.65	0.64	0.44	0.25
DT	0.73	0.74	0.74	0.69	0.46
AB	0.72	0.73	0.73	0.67	0.44
BC	0.79	0.81	0.79	0.79	0.58
GB	0.78	0.79	0.78	0.78	0.6
RF	0.8	0.8	0.8	0.74	0.59
ETC	0.79	0.8	0.79	0.74	0.57
LR	0.8	0.8	0.8	0.77	0.6
MLP	0.8	0.8	0.8	0.77	0.6
LDA	0.79	0.79	0.78	0.77	0.6
LSTM	0.83	0.83	0.83	0.79	0.65
RNN	0.83	0.83	0.83	0.8	0.66
Detect_CNN	**0.85**	**0.85**	**0.85**	**0.80**	**0.66**

Table 6. Performance metrics for various K-Fold models on test data

K-Fold Models	Acc	Prec	Recall	F1-S	MCC
SVM	0.78	0.79	0.79	0.74	0.56
KNN	0.65	0.67	0.65	0.42	0.28
DT	0.73	0.73	0.73	0.67	0.44
AB	0.71	0.71	0.72	0.64	0.41
BC	0.79	0.8	0.79	0.79	0.58
GB	0.77	0.78	0.77	0.77	0.59
RF	0.8	0.82	0.81	0.71	0.61
ETC	0.81	0.82	0.81	0.75	0.62
LR	0.78	0.79	0.79	0.74	0.56
MLP	0.81	0.81	0.81	0.77	0.61
LDA	0.79	0.81	0.8	0.76	0.6
LSTM	0.82	0.82	0.82	0.77	0.63
RNN	0.82	0.82	0.82	0.77	0.63
Detect_CNN	**0.84**	**0.84**	**0.84**	**0.80**	**0.67**

References

1. **Abubakar, H. D., Umar, M., Bakale, M. A. (2022).** Sentiment classification: Review of text vectorization methods: Bag of words, tf-idf, word2vec and doc2vec. SLU Journal of Science and Technology, Vol. 4, No. 1, pp. 27–33.

2. **Adelani, D. I., Mai, H., Fang, F., Nguyen, H. H., Yamagishi, J., Echizen, I. (2020).** Generating sentiment-preserving fake online reviews using neural language models and their human-and machine-based detection. Advanced information networking and applications: Proceedings of the 34th international conference on advanced infor-

mation networking and applications (AINA-2020), Springer, pp. 1341–1354.

3. **Alamleh, H., AlQahtani, A. A. S., ElSaid, A. (2023).** Distinguishing human-written and ChatGPT-generated text using machine learning. 2023 Systems and Information Engineering Design Symposium (SIEDS), IEEE, pp. 154–158.
4. **Bishop, L. (2023).** A computer wrote this paper: What chatgpt means for education, research, and writing.
5. **Breiman, L. (2017).** Classification and regression trees. Routledge.
6. **Brown, T. B. (2020).** Language models are few-shot learners. arXiv preprint arXiv:2005.14165.
7. **Chen, Y., Kang, H., Zhai, V., Li, L., Singh, R., Raj, B. (2023).** Gpt-sentinel: Distinguishing human and ChatGPT generated content. arXiv preprint arXiv:2305.07969.
8. **Chicco, D., Jurman, G. (2020).** The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. BMC genomics, Vol. 21, pp. 1–13.
9. **Christian, J. (2023).** CNET secretly used AI on articles that didnâ€™t disclose that fact, staff say. Futurusm, January.
10. **Cortes, C. (1995).** Support-vector networks. Machine Learning.
11. **Cox, D. R. (1958).** The regression analysis of binary sequences. Journal of the Royal Statistical Society Series B: Statistical Methodology, Vol. 20, No. 2, pp. 215–232.
12. **Falati, S. (2023).** How ChatGPT challenges current intellectual property laws.
13. **Freund, Y., Schapire, R. E. (1997).** A decision-theoretic generalization of on-line learning and an application to boosting. Journal of computer and system sciences, Vol. 55, No. 1, pp. 119–139.
14. **Geurts, P., Ernst, D., Wehenkel, L. (2006).** Extremely randomized trees. Machine learning, Vol. 63, pp. 3–42.
15. **Guo, B., Zhang, X., Wang, Z., Jiang, M., Nie, J., Ding, Y., Yue, J., Wu, Y. (2023).** How close is ChatGPT to human experts? comparison corpus, evaluation, and detection. arXiv preprint arXiv:2301.07597.
16. **Hamed, A. A. (2023).** Improving detection of ChatGPT-generated fake science using real publication text: Introducing xfakebibs a supervised-learning network algorithm.
17. **Hochreiter, S. (1997).** Long short-term memory. Neural Computation MIT-Press.
18. **Hu, B., Chen, Q., Zhu, F. (2015).** Lcsts: A large scale Chinese short text summarization dataset. arXiv preprint arXiv:1506.05865.
19. **Hunt, E. B. (2014).** Artificial intelligence. Academic Press.
20. **Kattenborn, T., Leitloff, J., Schiefer, F., Hinz, S. (2021).** Review on convolutional neural networks (cnn) in vegetation remote sensing. ISPRS journal of photogrammetry and remote sensing, Vol. 173, pp. 24–49.
21. **Li, X., Zhang, Y., Malthouse, E. C. (2023).** A preliminary study of ChatGPT on news recommendation: Personalization, provider fairness, fake news. arXiv preprint arXiv:2306.10702.
22. **Maddigan, P., Susnjak, T. (2023).** Chat2VIS: Generating data visualizations via natural language using ChatGPT, codex and GPT-3 large language models. Ieee Access, Vol. 11, pp. 45181–45193.
23. **Merine, R., Purkayastha, S. (2022).** Risks and benefits of AI-generated text summarization for expert level content in graduate health informatics. 2022 IEEE 10th International Conference on Healthcare Informatics (ICHI), IEEE, pp. 567–574.
24. **Narayan, S., Cohen, S. B., Lapata, M. (2018).** Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. arXiv preprint arXiv:1808.08745.
25. **Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., and others (2022).** Training language models to follow instructions with human feedback. Advances in neural information processing systems, Vol. 35, pp. 27730–27744.
26. **Pardos, Z. A., Bhandari, S. (2023).** Learning gain differences between ChatGPT and human tutor generated algebra hints. arXiv preprint arXiv:2302.06871.
27. **Perkins, M. (2023).** Academic integrity considerations of AI large language models in the post-pandemic era: ChatGPT and beyond. Journal

of University Teaching and Learning Practice, Vol. 20, No. 2.

28. **Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., and others (2019).** Language models are unsupervised multitask learners. OpenAI blog, Vol. 1, No. 8, pp. 9.
29. **Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B. (2022).** High-resolution image synthesis with latent diffusion models. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 10684–10695.
30. **Rumelhart, D. E., Hinton, G. E., Williams, R. J. (1986).** Learning representations by back-propagating errors. Nature, Vol. 323, No. 6088, pp. 533–536.
31. **Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E. L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., and others (2022).** Photorealistic text-to-image diffusion models with deep language understanding. Advances in neural information processing systems, Vol. 35, pp. 36479–36494.
32. **See, A., Liu, P. J., Manning, C. D. (2017).** Get to the point: Summarization with pointer-generator networks. arXiv preprint arXiv:1704.04368.
33. **Shahriar, S., Hayawi, K. (2023).** Let's have a chat! a conversation with ChatGPT: Technology, applications, and limitations. arXiv preprint arXiv:2302.13817.
34. **Su, Z., Wu, X., Zhou, W., Ma, G., Hu, S. (2023).** Hc3 plus: A semantic-invariant human ChatGPT comparison corpus. arXiv preprint arXiv:2309.02731.
35. **Sun, S., Zhao, W., Manjunatha, V., Jain, R., Morariu, V., Dernoncourt, F., Srinivasan, B. V., Iyyer, M. (2021).** Iga: An intent-guided authoring assistant. arXiv preprint arXiv:2104.07000.
36. **Tan, Y., Min, D., Li, Y., Li, W., Hu, N., Chen, Y., Qi, G. (2023).** Can chatgpt replace traditional kbqa models? an in-depth analysis of the question answering performance of the gpt llm family. International Semantic Web Conference, Springer, pp. 348–367.
37. **Weiss, M. (2019).** Deepfake bot submissions to federal public comment websites cannot be distinguished from human submissions. Technology Science, Vol. 2019121801.
38. **Xu, L., Hu, H., Zhang, X., Li, L., Cao, C., Li, Y., Xu, Y., Sun, K., Yu, D., Yu, C., and others (2020).** CLUE: A Chinese language understanding evaluation benchmark. arXiv preprint arXiv:2004.05986.
39. **Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M., Li, X., Lin, X. V., and others (2022).** Opt: Open pre-trained transformer language models. arXiv preprint arXiv:2205.01068.

Article received on 27/12/2024; accepted on 16/10/2025.
**Corresponding author is Partha Pakray.*