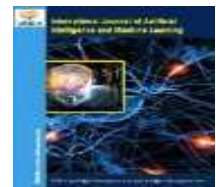




International Journal of Artificial Intelligence and Machine Learning

Publisher's Home Page: <https://www.svedbergopen.com/>



Research Paper

Open Access

HSS-GAT: A Memory-Efficient Hybrid Structural and Semantic Learning Using Graph Attention Networks and TF-IDF Features for AI-Generated Text Classification

Saleem Raja Abdul Samad¹, Pradeepa Ganesan², Sujatha Krishna³, Justin Rajasekaran⁴, Singaravelan Murugan⁵, Sathya Ramasamy⁶, Peerbasha Shebbeer Basha⁷, Bala Samuvel Joseph Gnana Sekar⁸

^{1,2,3,4} Department of Computing and Information Sciences, University of Technology and Applied Sciences-Shinas, Sultanate of Oman

⁵ Department of Computer Science and Engineering, Vel Tech Rangarajan Dr. Sagunthala R & D Institute of Science and Technology, Chennai, Tamil Nadu, India.

⁶ Department of Computer Science, PSG College of Arts & Science, Coimbatore, India.

⁷ Department of Computer Science, Jamal Mohamed College, Affiliated to Bharathidasan University, Trichy, Tamilnadu, India

⁸ Department of Mathematics, SRM Institute of Science and Technology, Ramapuram Campus, Chennai, Tamil Nadu, India Email: saleem.abdulsamad@utas.edu.om

Abstract

The rapid growth of content generated by large language model (LLM)-based applications has posed significant challenges to the society especially in domain where content originality, authorship and factual accuracy are critical. Therefore, there is an increasing need to develop robust and reliable classification methods for classifying AI-generated text. Such methods, can help this will help maintain information credibility, preventing spreading misinformation, detecting academic plagiarism, protecting digital platforms from malicious content, and ensuring transparency in online communication. Although, researchers have introduced plenty of detection methods and frameworks. there is still a need of more reliable solutions due to rapid advancement of LLM models which are capable of generating content that closely resemble human-written contents. The proposed framework presents the hybrid graph-based AI-generated text classification approach (HSS-GAT) that combines structural graph learning with TF-IDF based global semantic features to improve the classification performance. The experiment was conducted using two benchmark datasets (HC3 and DAIGT v2) which includes the generated contents from diverse LLM models. The experimental result on combined (HC3 + DAIGT v2) dataset demonstrates that the proposed framework achieved an accuracy of 98.15% and an F1-Score of 97.93% highlighting the effectiveness of the proposed framework.

Keywords: AI-generated text classification, Large Language Models (LLMs), Graph Attention Network (GAT), Bidirectional Long Short-Term Memory (BiLSTM), Hybrid Deep Learning, TF-IDF feature extraction.

1. Introduction

Advancement in digital communication has led to an enormous accumulation of data through diverse sources email, social media, news articles, online reviews, research publications and conversational systems. A significant portion of these contents are now produced either fully or partially by generative language models due to its capability of generating contents across domain with more fluent, coherent and contextually relevant contents [1][2]. Recent reports states that most of the recent reviews and comments of research papers were generated by the LLM especially chatgpt [3][4][5]. As the volume of AI-generated text contents continuously to grow, distinguishing between human-written contents and AI-generated contents has become increasingly important [6][7]. Especially in the education sector, AI-generated text classification is highly important to preserve academic integrity. Recent studies highlighting that students are increasingly relaying on various LLM-based applications to complete assignments, essays, reports, coding tasks and research documents that closely resemble human style writings, making it difficult for educators to differentiate between original and AI-generated contents. This growing dependence can seriously be affecting the critical thinking skills and problem-solving skills which will diminish the learning process [8][9]. Educators have two choices either redesign the assessment methods or develop a robust automated system for AI-generated text classification that ensures fair assessment practices without changing the existing assessment methods and maintaining the credibility of the academic evaluations. Among these, developing an effective AI-generated text classification system is the more practical and preferable approach [10]. The issues extend beyond education. LLMs can generate convincing news articles, blogs and social media contents and supporting productivity and saving valuable human time. On the other side, they also support for generating large scale dissemination of misinformation, fake news and manipulated narratives that appears credible to the readers. Such contents spoil the public trust on journalisms, influence public opinion and disturbs social harmony [11]. Recently LLMs have also used to

write scientific reports and review comments, seriously raising concern about research integrity [5]. These reports may have fabricated citations, technical inaccuracies and generalized or inaccurate interpretations of scientific findings. Moreover, increasing number of scientific publications has intensified concern about maintaining publication quality and originality [12][13][14]. The increasing capability of LLMs enable cybercriminals to create phishing emails, fraudulent messages and social engineering attacks more easily and effectively. AI-generated contents are grammatically correct, more personalized and highly convincing, increasing the chance of successful cyberattacks. Additionally, cybercriminals can also automate the generation and spreading of deceptive messages at large-scale, posing serious threats to individuals and organizations [15] [16] [17]. This highlights the need of timely detection and preventing users from falling prey to such attacks. Researchers have introduced various methods and frameworks for classifying AI-generated text contents. Some of the exiting approaches focus only on specific LLM, while others were evaluated on limited dataset which restrict the scalability and applicability of the proposed solutions. Moreover, increasing capabilities of the LLMs enable them to bypass the most of the existing detection techniques. This challenge demands computationally efficient and robust framework for classifying AI-generated text.

To address this challenge, the proposed framework presents a hybrid graph-based text classification approach (HSS-GAT), that combines structural graph learning with TF-IDF-based global semantic features to improve classification performance. Initially, the input dataset was pre-processed by converting all text into a uniform string format (lowercase) to ensure the consistency. Each document is transformed into two complementary representations that captures different characteristics of the given input text. The first representation is a graph-based where each document is converted into graph consisting of nodes and edges. Each individual words are represented as nodes and the relationship between neighbouring words are represented using an adjacency matrix. Each node contains structural information such as word length, special character ratio and normalized word position. These graph features are processed using memory efficient multi-head Graph Attention Network (GAT) [18] to learn the importance of neighbouring nodes, followed by a two-layer Bidirectional Long Short-Term Memory (BiLSTM) network [19] that captures the sequential contextual dependencies. At last, max-pooling is applied to generate a compact graph-level feature vector representing the structural information of the document.

The second representation captures the global semantic importance of words using Term Frequency - Inverse Document Frequency (TF-IDF) techniques [20] with the 1500 most informative features. This representation identifies most important words across the complete dataset. This resulting TF-IDF feature vector is then processed by dense neural network to generate a compact semantic feature representation. Finally, the structural features obtained from graph and semantic features obtained from the TF-IDF are then combined to form a fused representation. This fused representation is passed through the fully connected classification layer with a sigmoid output function to perform final classification of AI-generated text. The experiments were conducted on two benchmark datasets (HC3 and DAIGT v2). These datasets contain generated text contents from multiple LLMs. The experimental results indicates that the proposed framework HSS-GAT, achieved an accuracy of 98.15% and an F1-score of 97.93% on the combined dataset, highlighting its effectiveness in accurately classifying generated text.

Contribution:

- The proposed framework HSS-GAT, combines graph-based structural features with global semantic features using TF-IDF, enabling the system to simultaneously learn local word relationship and document-level semantics.
- In the proposed framework, compact graph representations, lightweight attention and optimized tensor processing are implemented, making the system suitable for large-scale implementation.
- The experimental results are evaluated five-fold stratified cross-validation which ensuring an unbiased performance estimation.
- The proposed framework performance on chosen dataset was evaluated using multiple metrics includes Accuracy, Precision, Recall, F1-score, ROC-AUC, Matthews Correlation Coefficient, Cohen's Kappa, confusion matrix, ROC curve, Precision-Recall curve, execution time analysis, and error analysis, provides a complete assessment.
- The proposed framework achieves better classification performance without increasing computational complexity.

The rest of the paper is organized as follows. Section 2 discusses recent related research works in the problem domain. Proposed framework is explained in the section 3 and Section 4 presents the results and discussion. Finally, section 5 concludes the paper.

2. Related Works

AI-Generated text classification is one of the important research areas of current technological era. Many studies have proposed approaches based on surface level feature extraction with machine learning, advanced text representation using transformer with deep learning model and hybrid models to improve

the classification performance while maintaining computational efficiency. This section highlights the most important contributions in this problem domain. A graph-based approach for differentiating human written text and AI generated text was proposed by Andric et al [21]. Each document was transformed into co-occurrence graph, where words are represented as nodes and their co-occurrences as weighted edges and initialized node features using random embeddings, Word2Vec, fine-tuned BERT and fine-tuned RoBERTa embeddings. A Graph Attention Network (GAT) was then employed to learn structural and semantic relationships within the text for classification. The experiments were conducted on the Autextification 2023 English dataset which contains 23691 training, 10154 validation and 21832 test samples. The experimental results demonstrated that the GNN with fine-tuned BERT node features achieved an accuracy of 74.48% and macro F1-score of 74.41%. Xiaoming et al [22] proposed a coherence-enhanced framework for machine generated text detection on four datasets namely GROVER, GPT-2 WebText, and two self-constructed GPT-3.5 news datasets. This approach extracts named entities using an ELMo-based NER model and construct a coherence graph to capture both local and global document coherence through entity relationships. The graph is encoded using a relation-aware Graph Convolutional Network (GCN) with attention-based LSTM and the resulting coherence representation is combined with RoBERTa embeddings for improved classification. Experimental result indicates that CoCo achieved an accuracy of 85.30% and an F1-Score of 84.10% on the limited GPT-2 dataset. 69.93% accuracy and 61.25% F1-Score on the limited GROVER dataset and 99.72% accuracy and 99.57% F1-Score on GPT-3.5 dataset. Although the result of CoCo framework was impressive, it increases preprocessing overhead and computational complexity compared to transformed-based approaches. The performance of approach is highly dependent on accurate entity extraction which limits its applicability to short documents where coherence information is limited. Moreover, the model utilized RoBERTa embeddings which segments the longer documents, potentially resulting in the loss of contextual information.

Nidhi and Qinghua [23] proposed a graph-based framework called similarity-guided graph reasoning for AI text detection. The framework masks syntactic keywords, reconstruct them using a pretrained ALBERT-base-V2 model and constructing a similarity guided keyword graph. A two-layer Graph Convolutional Network (GCN) is then used to learn the semantic and contextual relationships between keywords for classification. This framework was evaluated on four domains such as News, Creative Writing, Student Essays and Vulnerability description using dataset includes Ghostbuster, the NVD and AI-generated text from ChatGPT. The proposed framework was compared with 4-baselines methods namely GPTZero, DetectGPT, Ghostbuster and RAIDAR using F1-Score on all four datasets. Experimental results showed that proposed framework outperformed baseline methods achieving in-domain F1 Scores of 0.79, 0.72, 0.88, and 0.96 for the News, Creative Writing, Student Essays and Vulnerability description respectively. It also maintained strong out-of-distribution F1-Score ranging from 0.67 to 0.75. However, the framework incurs higher computational overhead due to keyword extraction, graph construction and GCN based reasoning. Halah [24] proposed a graph-based framework for AI generated tweets detection based on stylistic features as graph-structured data where nodes correspond to posts and writing-style attributes while edges capture the relationship among the tweets. The evaluation includes multiple approaches including zero-shot prompting, fine-tuned sentence-BERT (SBERT), Graph Attention Networks (GAT) and Graph Convolutional Networks (GCN). Result indicates that GCN-based model achieved an accuracy of 93.60% on tweet dataset. Adilkhan et al. [25] presented a detection approach for AI generated text in academic settings. The experiment utilizes HC3 and DAIGT v2 datasets. The result indicates that the TF-IDF + Logistic Regression model achieved an accuracy of 82.87%, contextual neural model such as BiLSTM reached an accuracy of 88.86% accuracy and DistilBERT achieved an accuracy of 88.11% with ROC-AUC of 96% highlighting the importance of semantic representation over surface-level lexical features of text. This study requires high computational resources for contextual models.

3. Proposed Framework

In this proposed framework HSS-GAT introduces a novel hybrid graph-based approach for AI-generated text classification. It consists of two parallel feature extraction branches. The first branch transforms every document into a graph to capture local contextual relationships between neighbouring words while the second branch extract the global semantic information using TF-IDF. The outputs from both branches are combined to create a richer feature representation. This fused representation is then passed to a multi-layer neural network to classify the text as either AI generated or human-written text. Conventional Graph Attention Networks usually process large graph structures with high-dimensional node representations and dense attention computations, resulting in significant memory requirements during training. In contrast, the proposed framework is memory efficient and specifically designed to minimize computational and storage requirement at every stage of processing. The proposed HSS-GAT framework is illustrated in Figure 1.

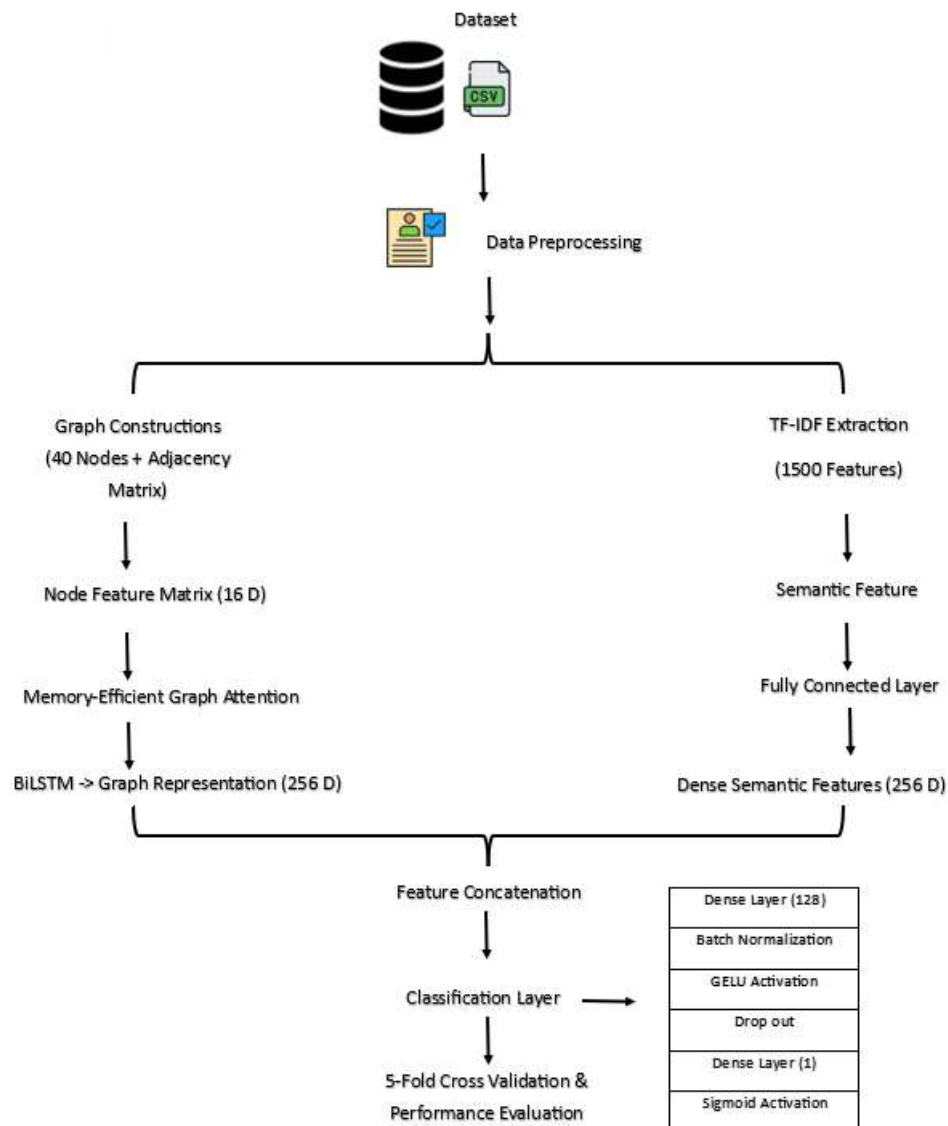


Figure 1. Architecture of Proposed HSS-GAT framework

a. Dataset

The proposed framework, utilized the two most used dataset (HC3 and DAIGT v2) for machine generated text classification. Table 1 and 2 present the HC3 dataset details and distributions. HC3 dataset is designed to compare human writings and machine generated text across diverse domains in both English and Chinese. For this experiment, considers only English answers. This dataset was collected from publicly available sources such as Hugging Face [26], GitHub [27] as originally described by Biyang et al., 2023 [28].

TABLE 1. HC3 DATASET CONTRIBUTION

	Source	Count
0	reddit_eli5	32114
1	finance	7866
2	medicine	2494
3	open_qa	2374
4	wiki_csai	1684

TABLE 2. HC3 DATASET DISTRIBUTION

	Target	Count
0	Human Written Text	23492
1	Machine Generated Text (ChatGPT)	23040

The dataset DAIGT v2 was collected from Kaggle [29] represents another large-scale English text resource specifically curated for detecting AI-generated content. It contains 44,868 text samples compiled from

multiple sources including human written essays and output generated by multiple LLMs. Every instance in the dataset includes text content and a corresponding label. The label indicates whether the text is generated (1) or human-written (0). Table 3 and 4 presents the data content and distribution.

TABLE 3. DAIGT v2 DATASET CONTRIBUTION [34]

Source	Count
persuade_corpus	25996
mistral7binstruct_v2	2421
chat_gpt_moth	2421
llama2_chat	2421
mistral7binstruct_v1	2421
kingki19_palm	1384
train_essays	1378
llama_70b_v1	1172
falcon_180b_v1	1055
darragh_claude_v6	1000
darragh_claude_v7	1000
radek_500	500
NousResearch/Llama-2-7b-chat-hf	400
mistralai/Mistral-7B-Instruct-v0.1	400
cohere-command	350
palm-text-bison1	349
radekgpt4	200

TABLE 4. DAIGT v2 DATASET DISTRIBUTION

	Target	Count
0	Human Written Text	27371
1	Machine Generated Text	17497

The experiment also employed a combined dataset (HC3 and DAIGT v2), with the data distribution summarized in Table 5.

TABLE 5. COMBINED DATASET DISTRIBUTION

	Target	Count
0	Human Written Text	50,863
1	Machine Generated Text	40,537

b. Preprocessing

The process begins with text preprocessing. Each sentence was converted to lowercase and leading and trailing spaces were removed. This step helps to clean and standardize the text which is necessary for vocabulary construction and graph generation [30].

c. Global Semantic Feature Extraction

The semantic information of each document is extracted using Term Frequency-Inverse Document Frequency (TF-IDF) technique. Initially each document is tokenized into words and TF-IDF assigns importance scores to words according to their occurrence within the individual documents and across the complete dataset. The TF-IDF representation is computed by combining two statistical measures. The first component is Term-Frequency (TF), measures the frequency of a particular term appears within a document. For a given term t in document d , the term frequency is expressed as expressed as

$$TF(t, d) = \frac{f_{t,d}}{\sum_k f_{k,d}}$$

where $f(t, d)$ denotes the number of occurrences (term frequency) of the term t in document d , and $\sum_k f(k, d)$ represents the total number of words contained in document d .

The second component is the inverse document frequency, which evaluates the frequency of a term across complete document collection. Words that appear in many documents contribute less and receives lower weights. The inverse document frequency is computed as

$$IDF(t) = \log\left(\frac{N}{DF(t)}\right)$$

Where N represents the total number of documents in the dataset, and $DF(t)$ represents the number of documents in which the term t appears. To avoid division by zero and improve numerical stability, a smoothing factor is added.

$$\text{IDF}(t) = \log \left(\frac{N + 1}{\text{DF}(t) + 1} \right) + 1$$

The final TF-IDF weight assigned to each term is obtained by multiplying its normalized term frequency with its inverse document frequency.

$$\text{TFIDF}(t, d) = \text{TF}(t, d) \times \text{IDF}(t)$$

After computing the TF-IDF scores, every document is represented as a numerical feature vector

$$G = [g_1, g_2, g_3, \dots, g_{1500}]$$

where each element in the vector g_i corresponds to the TF-IDF weight of one vocabulary term. This fixed representation of top 1500 features, simplifies batch processing and enables efficient integration with the neural network.

d. Graph Construction

This stage graph construction, transforms each text document into a fixed-size graph representation that preserves the local relationship between neighbouring words. Initially, each document is tokenized into individual words based on whitespace.

$$D = \{w_1, w_2, w_3, \dots, w_n\}$$

where w_i denotes the i^{th} word in the document and n represents the total number of words. In graph representation, each word is considered as a node and the contextual relationship between neighbouring words as graph edges. This representation enables the learning model to capture both positional and structural information during feature learning. To reduce the computational complexity and maintain a uniform graph size, only the first 120 tokens of each document are considered. If a document contains no valid tokens, a special placeholder token is inserted to ensure that every graph contains at least one node. The total number of graph nodes is restricted to a maximum of 40 nodes and is determined as

$$N = \min(n, 40)$$

where n denotes the total number of tokens in the document. The shorter documents are zero-padded to maintain uniform graph dimensions. Each graph node is represented using a structural feature vector of dimension 16. The node feature matrix is defined as

$$X \in \mathbb{R}^{40 \times 16}$$

where each row corresponds to one node and each column represents one structural feature extracted from the corresponding word. The feature vector for the i^{th} node is expressed as

$$x_i = [x_{i,1}, x_{i,2}, \dots, x_{i,16}]$$

where each feature captures as listed in table 6, a complementary lexical, structural, statistical, or positional property of the corresponding token, enabling the graph representation to encode richer document information before graph attention learning

TABLE 6. STRUCTURAL NODE FEATURES USED FOR GRAPH CONSTRUCTION

Feature Index	Feature Name	Description and Importance
$x_{i,1}$	Token Length	Represents the total number of characters in the token.
$x_{i,2}$	Punctuation / Non-Alphanumeric Ratio	Measures the proportion of punctuation or special characters relative to the token length. This feature captures writing style, formatting patterns, and unusual character usage that may contribute to document discrimination.
$x_{i,3}$	Relative Position Index	Normalized position of the token within the graph which preserves the sequential order of words.
$x_{i,4}$	Digit Ratio	Represents the proportion of numeric characters in a token.
$x_{i,5}$	Uppercase Character Ratio	Measures the proportion of uppercase letters in the token. It helps identify abbreviations, proper nouns, acronyms, and emphasized words, thereby improving structural understanding.
$x_{i,6}$	Lowercase Character Ratio	Computes the proportion of lowercase letters in the token. This feature complements uppercase information and captures the dominant writing pattern of each word.
$x_{i,7}$	Vowel Ratio	Calculates the proportion of vowels within the token. The vowel distribution provides additional lexical characteristics that help distinguish different word structures.
$x_{i,8}$	Consonant Ratio	Represents the proportion of consonants in the token. Combined with the vowel ratio, this feature characterizes word composition and improves structural representation.
$x_{i,9}$	Punctuation Count	Records the number of punctuation and special characters in a token, capturing formatting patterns.
$x_{i,10}$	Digit Count	Represents the total number of numeric characters in the token.

$x_{i,11}$	Normalized Local Token Frequency	Measures how frequently a token appears within the current document after normalization. Frequently occurring words often indicate document-specific themes and improve local contextual representation.
$x_{i,12}$	Inverse Document Frequency (IDF)	Measures a token's uniqueness across the corpus, assigning higher weights to rare, informative words
$x_{i,13}$	Global Term Frequency	Represents a token's overall frequency across the entire corpus, providing global importance information that complements the local frequency measure.
$x_{i,14}$	Stopword Indicator	A binary feature indicating whether a token is a stopwords (1) or not (0), helping the model distinguish common functional words from informative content words.
$x_{i,15}$	Normalized Character ASCII Mean Code	The normalized mean ASCII value of a token's characters, capturing character encoding patterns.
$x_{i,16}$	Relative Sentence Position Score	Represents the token's relative position in a sentence, helping the model understand local context and word relationships.

The graph connectivity is defined by constructing an adjacency matrix that captures the relationships between the graph nodes. The adjacency matrix is represented as

$$A \in \mathbb{R}^{40 \times 40}$$

where each element A_{ij} denotes the connection strength between the i^{th} and j^{th} graph nodes. In the proposed framework, each document graph to a maximum of 40 nodes, the adjacency matrix has a fixed dimension of 40×40 , enabling efficient batch processing during model training. Initially, each node is connected to itself by introducing self-loops, ensuring that its own feature information is preserved during neighbourhood aggregation as defined as

$$A_{ii} = 1, i = 1, 2, \dots, 40$$

where the diagonal elements of the adjacency matrix are assigned a value of one. To preserve the sequential flow of the original text, each node is connected to its immediate predecessor and successor using bidirectional edges as represented as

$$A_{i,i-1} = A_{i-1,i} = 1,$$

for all valid neighbouring nodes. Assigning a connection weight of one indicates a strong contextual dependency between consecutive words. These bidirectional edges preserve the natural ordering of the document and facilitate effective information propagation between adjacent tokens during graph-based feature learning. Additionally, the proposed graph incorporates second-order neighbourhood relationships by connecting nodes that are separated by one intermediate token. These connections are assigned lower weights to reflect their relatively weaker contextual influence and are expressed as

$$A_{i,i-2} = A_{i-2,i} = 0.5.$$

The reduced edge weight allows the graph to capture broader contextual dependencies without introducing excessive graph density or computational overhead. Consequently, each node aggregates information not only from directly adjacent words but also from nearby contextual words, resulting in a richer structural representation. The resulting adjacency matrix contains three categories of edge weights. Self-loop connections have a weight of 1.0, immediate neighbouring nodes are connected with a weight of 1.0, and second-order neighbouring nodes are connected with a weight of 0.5. All remaining node pairs remain unconnected with a weight of zero. This sparse graph structure significantly reduces computational complexity while preserving the essential contextual relationships required for graph attention learning. At last, each document is represented as a graph

$$G = (V, E, X, A)$$

where V denotes the set of graph nodes corresponding to document tokens, E represents the collection of graph edges describing contextual relationships, $X \in \mathbb{R}^{40 \times 16}$ denotes the node feature matrix summarized in table 6, and $A \in \mathbb{R}^{40 \times 40}$ represents the adjacency matrix defining node connectivity.

e. Hybrid Graph Dataset Formation

After graph construction, the node feature matrices, adjacency matrices, TF-IDF vectors, and class labels are combined into a unified dataset. The node feature matrix (X) contains the sixteen structural attributes extracted for each of the forty graph nodes, while the adjacency matrix (A) preserves the connectivity between neighbouring words using self-connections, first-order links, and second-order weighted links. The TF-IDF vector (G) represents the overall semantic characteristics of the complete document. These three representations are paired with the corresponding class label y_i to form a single training instance is represented as

$$S_i = \{X_i, A_i, G_i, y_i\}$$

These data structures are converted into tensors for efficient processing using the deep learning framework. This hybrid dataset allows simultaneous learning from graph structure and document-level semantics during model training.

f. Memory-Efficient Graph Attention Layer

The graph representation is first processed by a Memory-Efficient Graph Attention layer to extract structural information from the constructed graph. Each node initially contains a 16-dimensional feature vector that describes the lexical and structural properties of the corresponding token. These node features are first projected into a higher-dimensional feature space using a learnable linear transformation. Instead of learning from a single attention mechanism, the proposed layer employs four parallel attention heads, allowing different neighbourhood relationships to be captured simultaneously. For every pair of connected nodes i and j , an attention score is computed using separate source and destination attention vectors. The computed score is then transformed through the LeakyReLU activation function [31][32] to improve feature discrimination as expressed as

$$e_{ij} = \text{LeakyReLU}(a_s^T h_i + a_d^T h_j)$$

where e_{ij} denotes the unnormalized attention score between connected nodes i and j , h_i and h_j are the transformed node feature vectors, a_s and a_d are the learnable source and destination attention vectors. Instead of evaluating attention across every node pair, the proposed approach restricts attention computation only to the connections defined by the adjacency matrix. Therefore, the adjacency matrix is acted as a masking mechanism, assigning a very small value to all non-connected node pairs before normalization. This significantly reduces unnecessary computations and lowers memory consumption without losing important neighbourhood information. The attention coefficients are then normalized using the Softmax function as

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k \in \mathcal{N}(i)} \exp(e_{ik})}$$

where e_{ij} denotes the unnormalized attention score between node i and its neighboring node j , and $\mathcal{N}(i)$ represents the set of neighboring nodes connected to node i . These normalized coefficients determine the contribution of each neighboring node during feature aggregation. The updated node representation is computed as the weighted sum of neighbouring node features using the normalized attention coefficients. This operation enables each node to integrate contextual information from its connected neighbours while assigning higher importance to more relevant nodes. The updated representation of node i is expressed as

$$h'_i = \sum_{j \in \mathcal{N}(i)} \alpha_{ij} h_j$$

where α_{ij} denotes the normalized attention coefficient between nodes i and j , h_j represents the feature vector of the neighboring node, and $\mathcal{N}(i)$ denotes the neighbourhood of node i . The output generated by each attention head is concatenated to produce a unified node representation with a dimensionality of $4 \times 32 = 128$. The resulting 128-dimensional graph representation is subsequently processed by the Bidirectional Long Short-Term Memory (BiLSTM) network for sequential context learning.

g. Sequential Context Learning Using BiLSTM

The enriched node features produced by the graph attention layer are passed to a two-layer Bidirectional Long Short-Term Memory (BiLSTM) network to capture contextual dependencies in both forward and backward directions. Given the graph feature sequence $G = \{g_1, g_2, \dots, g_n\}$, the BiLSTM generates forward and backward hidden states, which are concatenated as

$$h_t = \begin{bmatrix} \vec{h}_t & \overleftarrow{h}_t \end{bmatrix}$$

which contains information from both temporal directions. The BiLSTM uses two stacked layers with a hidden size of 128 units in each direction and a dropout rate of 0.3 to reduce overfitting during training. After processing all node representations, a global max-pooling operation is applied over the sequence to identify the most informative contextual features across all graph nodes. The pooled representation is computed as

$$h_{\text{graph}} = \max_{t=1, \dots, n} (h_t)$$

where the maximum value is selected independently for each feature dimension. This operation generates a fixed-length graph-level representation regardless of the document length.

h. Global Feature Processing

The TF-IDF vectors are processed independently through a fully connected layer to generate a compact dense representation. Batch normalization is then applied after the dense layer to normalize the feature distribution and improve the stability and convergence of the training process. The normalized features are then passed through the Gaussian Error Linear Unit (GELU) activation function [33] which introduce

smooth nonlinearity and enables the network to learning complex semantic patterns from the transformed feature representation. A dropout layer (0.3) reduces overfitting by randomly disabling neurons during training, converts sparse TF-IDF vectors into compact semantic representation.

i. Feature Fusion

The 256-dimensional graph-based structural features and 256-dimensional TF-IDF semantic features are combined to form a 512-dimensional fused feature vector. This representation captures both structural and semantic information, enabling the classifier to accurately differentiate between AI-generated and human-written text contents

j. Classification and Model Evaluation

The fused features are passed through fully connected layers with batch normalization, GELU activation, and a 0.3 dropout rate to learn discriminative features and reduce overfitting during training. Then the last dense layer uses a sigmoid activation function to generate a probability score to perform binary classification. The model is trained using the binary cross-entropy function. The AdamW optimizer is used with a learning rate of 0.001 to update the model parameters. Training is performed for 10 epochs using a batch size of 256. During each epoch, the network weights are updated through backpropagation to minimize the prediction error and improve the classification performance. Stratified Five-Fold Cross Validation is applied by dividing the dataset into five balanced subsets. In each fold, four subsets are used for training and one subset is used for validation, and the average result across all folds is reported as the final performance. The proposed framework is evaluated using accuracy, precision, recall, F1-score, MCC, Cohen's kappa, ROC-AUC, confusion matrix, ROC curve, Precision-Recall curve and document length-based F1-score evaluation for short, medium, and long texts.

4. Result and Discussion

The experiments were carried on a Windows 11 platform with an Intel Core i7 processor, 24 GB RAM, and an NVIDIA GeForce RTX 2050 GPU with 4 GB VRAM. The development environment used VS Code with Python as the primary programming language. The study utilized two separate datasets, and their detailed data distributions are summarized in Tables 1–5. The proposed model HSS-GAT configuration is presented in table 7

TABLE 7. PROPOSED HSS-GAT CONFIGURATION

Component	Configuration
Graph Attention Network (GAT)	
Input Node Features	16
Number of Attention Heads	4
Output Dimension per Head	32
Combined GAT Output	128 Features
Attention Activation	LeakyReLU (Negative Slope = 0.2)
Graph Connectivity	Self-loop + First-order + Second-order Neighbour Connections
BiLSTM Network	
Input Size	128
Hidden Units	128
Number of Layers	2
Direction	Bidirectional
Sequence Aggregation	Max Pooling
Dropout	0.30
TF-IDF with Fully Connected Layer	
TF-IDF Vocabulary Size	1500
Dense Layer Size	256
Batch Normalization	Yes
Activation Function	GELU
Dropout	0.30
Final Fully Connected Classifier	
Input Dimension	512 (256 Graph + 256 TF-IDF Features)
Hidden Layer	128
Batch Normalization	Yes
Activation Function	GELU
Dropout	0.30
Output Layer	1 Neuron
Output Activation	Sigmoid
Classification Type	Binary Classification

Table 8 presents the experimental result obtained on the HC3 dataset. The proposed framework achieved an accuracy of 97.62% and F1-Score of 97.59% indicates its ability of correct classification and well-balanced trade-off between precision and recall, showing that the model effectively minimizes both false positive and false negative predictions. The Matthew's correlation coefficient (MCC) of 95.25% and Cohen's Kappa coefficient of 95.25 % demonstrates an excellent level of agreement between the predicted and actual class labels beyond random chance. The obtained ROC-AUC of 99.71% further confirms that the proposed framework's discriminative capability.

TABLE 8. EXPERIMENTAL RESULT ON HC3 DATASET

Accuracy %	Precision %	Recall %	F1-Score %	MCC %	Kappa %	ROC-AUC %
97.62	97.97	97.21	97.59	95.25	95.25	99.71

Figure 2 presents the complete evaluation of the proposed framework on HC3 dataset. The ROC-AUC curve (a) achieves 99.71% indicates an exceptional class separation with minimal false positives. The Precision-Recall curve (b) maintains high precision across most recall values, confirming consistently high precision under varying thresholds. The training accuracy (c) increases from approximately 94% during the first epoch to nearly 99.8% within a few epochs, accompanied by a steady decline in training loss (d) from approximately 0.145 to almost zero, reflecting efficient feature learning. The validation accuracy (e) also mirrors the similar trend, reaches approximately 97.6% and remains stable, while validation loss (f) decreases rapidly and stays low, indicates the robust generalization.

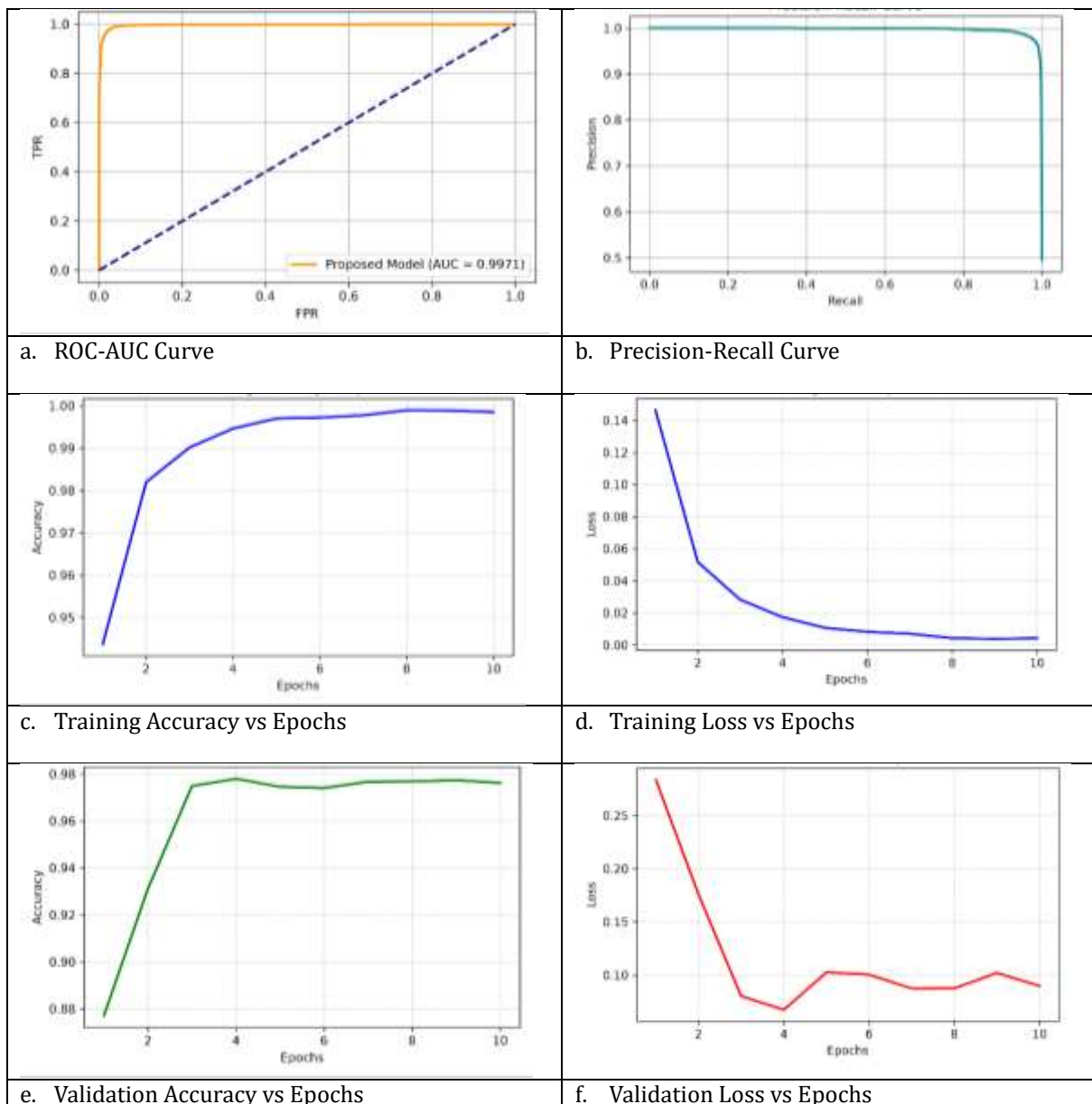


FIGURE 2. COMPREHENSIVE PERFORMANCE EVALUATION ON HC3 DATASET

Table 9 shows that the proposed HSS-GAT framework performs differently across text lengths, achieving F1-score of 70.77% for short texts (<50 words), 97.61% for medium-length texts (50-200 words) and 98.05% for long text (>200 words), indicates the significance of contextual information in classification performance. Despite of the limited textual diversity, fewer sequential dependencies and sparse graph structures in short documents, the proposed framework. Even though, short documents contain limited lexical diversity, fewer sequential dependencies, and sparse graph structures, the proposed framework sufficiently discriminative representations through its graph attention mechanism and BiLSTM architecture.

TABLE 9. STRATIFIED EVALUATION RESULT ON HC3 DATASET

Category	Sample Count	Stratified F1-Score %
Short (<50 words)	440	70.77
Medium (50-200 words)	4631	97.61
Long (>200 words)	4567	98.05

The Wilcoxon significance distribution shown in Figure 3 indicates that the model achieves highly consistent accuracy scores, ranging from approximately 0.97525 to 0.97675 with the median centered within this narrow range, demonstrating the proposed framework is stable and reliable performance across different evaluations.

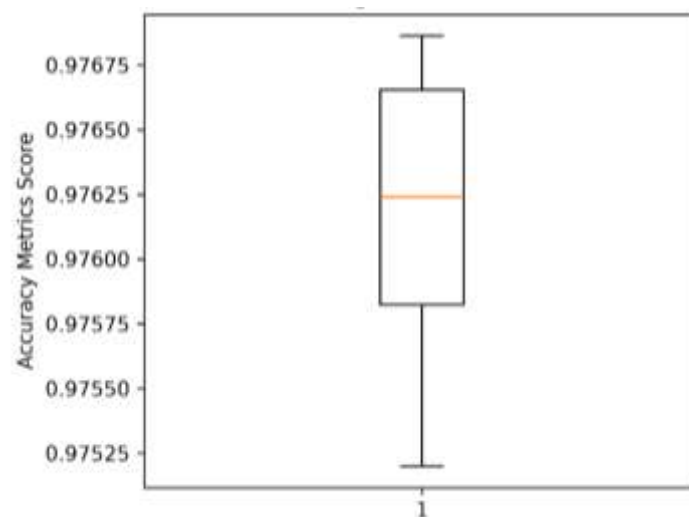


FIGURE 3. WILCOXON SIGNIFICANCE DISTRIBUTION OF ACCURACY METRICS ON HC3 DATASET

Table 10 summarize the evaluation result on the DAIGT v2 dataset, which contains AI-generated text from by multiple LLMs. The experimental results shows that the proposed framework achieved an accuracy of 99.40%, and F1-Score of 99.23%, which reflects proposed framework's ability to accurately classify the text and balanced classification. It also attains an MCC of 98.74%, Cohen's Kappa of 98.73% and ROC-AUC of 99.96%, confirms strong correlation between predicted and actual class labels, genuine agreement across both classes and near-perfect class separation.

TABLE 10. EXPERIMENTAL RESULT ON DAIGT v2 DATASET

Accuracy %	Precision %	Recall %	F1-Score %	MCC %	Kappa %	ROC-AUC %
99.40	99.63	98.83	99.23	98.74	98.73	99.96

Figure 4(a) shows ROC-AUC curve (AUC=0.9996) indicating a near-ideal decision boundary with a true positive rate approaching 1 at minimal false positive rates. This trend is reaffirmed by the precision-recall curve (b) which maintain near perfect precision across almost the entire recall range. During training as shown in (c) and (d), the proposed framework exhibits rapid and stable convergence. Training accuracy steadily increases from 0.980 to 0.999 around epoch 5 while training loss decreases smoothly from 0.075 to nearly 0 indicating effective optimization. Validation accuracy (e) consistently remains in impressive range of 0.990-0.994. Although, both the validation accuracy and validation loss (f) curve exhibit fluctuations across epochs. the proposed framework performs consistently maintain high validation

accuracy and its near perfect ROC-AUC indicates that these fluctuations have only minimal impact on the framework's generalization performance.

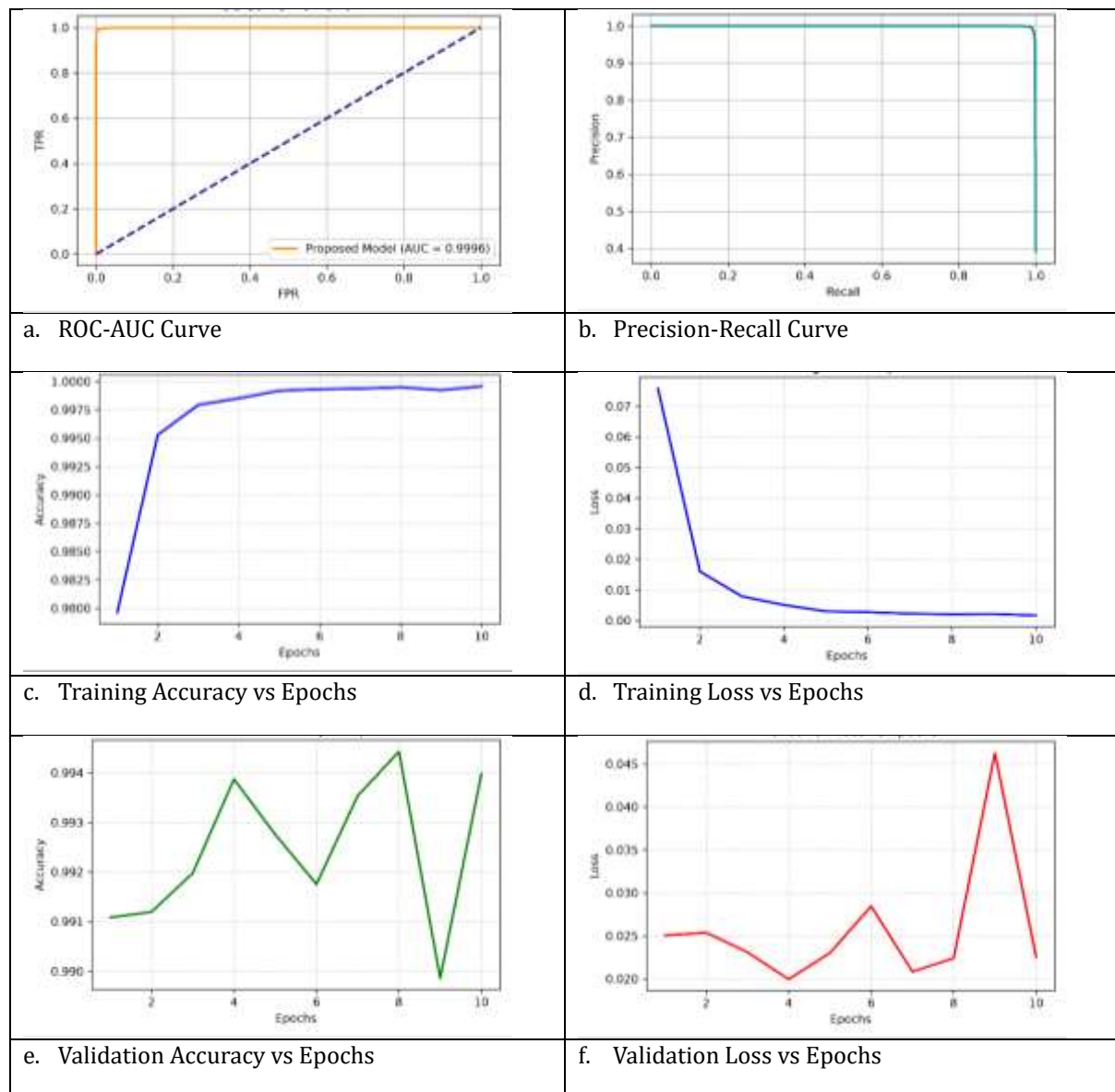


FIGURE 4. COMPREHENSIVE PERFORMANCE EVALUATION ON DAIGT v2 DATASET

Table 11 presents the stratified F1-scores of the proposed framework across different text-length categories on the DAIGT V2 dataset. Since the dataset contains no texts shorter than 50 words, so the evaluation primarily reflects performance on medium and long text sequences. The proposed framework performs exceptionally well on these longer contexts achieving a stratified F1-score of 99.28% on the long texts and 98.54% on medium length text sequences. These results demonstrate that the framework maintains consistently high classification performance across different text length categories.

TABLE 11. STRATIFIED EVALUATION RESULT ON DAIGT v2 DATASET

Category	Sample Count	Stratified F1-Score %
Short (<50 words)	0	0
Medium (50-200 words)	749	98.54
Long (>200 words)	8225	99.28

The box-plot on figure 4, illustrates the remarkable stability in the accuracy across cross-validation folds. The scores are clustered within a narrow range of approximately 0.9914-0.9940 with a median close to 0.9935 which indicates the consistently strong predictive performance of the proposed framework under different data partitions

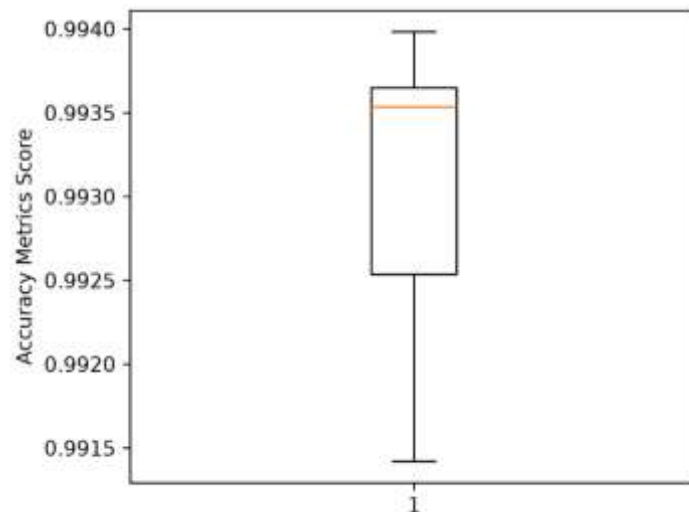


FIGURE 4. WILCOXON SIGNIFICANCE DISTRIBUTION OF ACCURACY METRICS ON DAIGT v2 DATASET

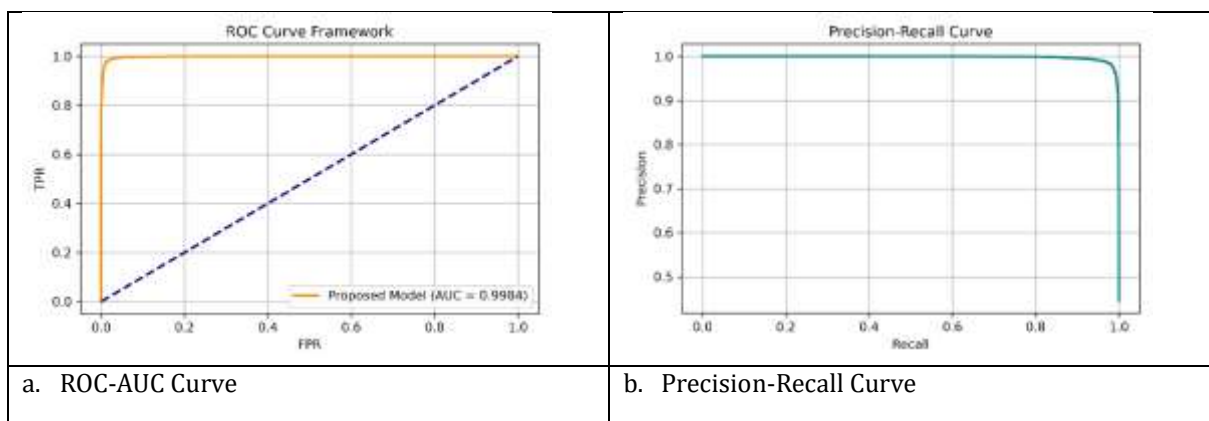
Accuracy %	Precision %	Recall %	F1-Score %	MCC %	Kappa %	ROC-AUC %
98.12	98.64	97.12	97.87	96.21	96.20	99.84

At last, framework was evaluated on combined dataset comprising both the HC3 and DAIGT v2. As shown in table 12, it maintained consistently strong performance across all evaluation when compared to that achieved on the separate dataset evaluation. The proposed framework achieved an accuracy of 98.12% and the F1-score of 97.87%, demonstrating accurate classification and strong balance between precision and recall. It also achieved an MCC of 96.21% and a Cohen's Kappa of 96.20% indicating high agreement between predicted and actual labels and reliable performance even with class imbalance. Moreover, the ROC-AUC of 99.84% indicates its near-perfect ability to differentiate AI-generated and human written text across different thresholds.

TABLE 12. EXPERIMENTAL RESULT ON COMBINED (HC3 +DAIGT v2) DATASET

Accuracy %	Precision %	Recall %	F1-Score %	MCC %	Kappa %	ROC-AUC %
98.12	98.64	97.12	97.87	96.21	96.20	99.84

Figure 5 shows that the ROC curve (a) achieved an AUC of 0.9984, reflecting the proposed framework's discriminative capabilities, while the precision-recall curve (b) maintains precision above 0.98 across almost entire recall range. These two metrics confirms that the proposed framework reliably distinguishes AI-generated and Human-written text with excellent accuracy. The framework training accuracy (c) increases steadily from approximately 95.7% to 99.8% over 10 epochs while training loss declines smoothly from 0.115 to nearly 0. These trends reflect the quality of the optimization process and no evidence of convergence difficulties. Validation accuracy (e) improves gradually from 97.6% to approximately 98.35% in 9th epochs demonstrating that the framework generalizes well. In contrast, the validation loss (f) fluctuates between 0.059 and 0.0755 throughout the training. However, these moderate fluctuations, are unlikely affect the framework performance as the validation accuracy remains consistently high and classification metrics indicate stable generalization.



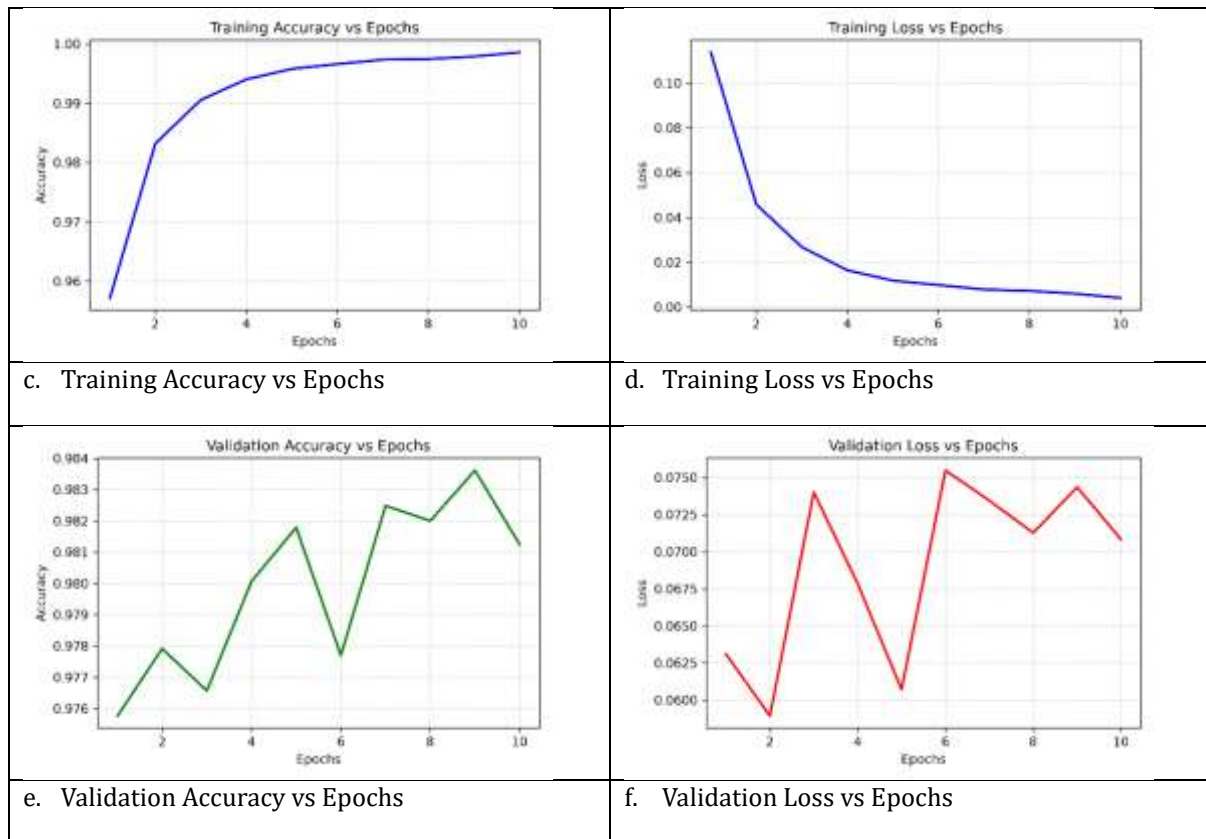


FIGURE 5. COMPREHENSIVE PERFORMANCE EVALUATION ON COMBINED (HC3+ DAIGT v2) DATASET

The stratified analysis in table 13 shows that the proposed framework's predictive accuracy even for the short text (<50 words). It achieved a stratified F1-score of 98.57% for long text (>200 words) and 97.00% for medium length text (50-200 words) indicating the consistently strong performance regardless of text length. The box plot in figure 6 demonstrates the framework's robustness across different fold. Accuracy remains tightly clustered between approximately 0.9813 and 0.9842 with median close to 0.9830 indicates very low variability across class validation folds, indicates framework's consistent behavior and generalization ability.

TABLE 13. STRATIFIED EVALUATION RESULT ON COMBINED (HC3+ DAIGT v2) DATASET

Category	Sample Count	Stratified F1-Score %
Short (<50 words)	451	79.01
Medium (50-200 words)	5259	97
Long (>200 words)	12902	98.57

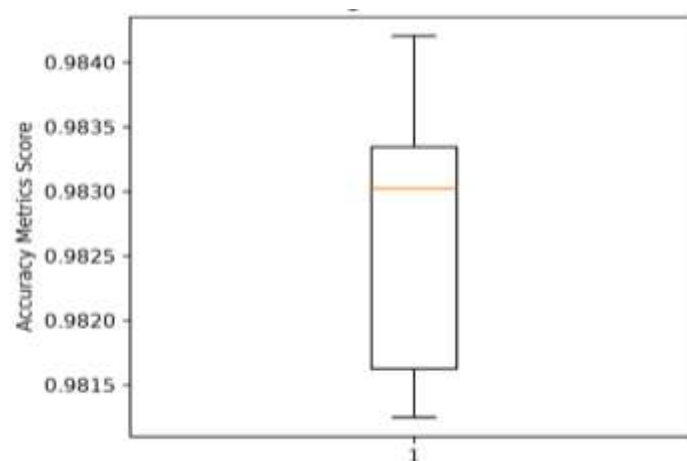


FIGURE 6. WILCOXON SIGNIFICANCE DISTRIBUTION OF ACCURACY METRICS ON COMBINED (HC3+ DAIGT v2) DATASET

5. Conclusion

The rapid advancement of large language models has made distinguishing AI-generated text from human-written content increasingly challenging due to its advancement in generation capabilities. Accurately identifying the true origin of text has therefore become essential for ensuring content authenticity, maintaining academic integrity, combating misinformation and supporting trustworthy communication. The proposed framework HSS-GAT is a memory efficient, hybrid graph-based classification framework for AI generated text classification that combines graph based structural learning and global semantic feature representation using IF-IDF. The graph representation is processed through a memory-efficient multi-head graph attention network to capture local contextual relationships whereas a BiLSTM learns sequential dependencies from the graph features. In parallel, the semantic features vector is refined using a fully connected network and both representations are fused to generate a comprehensive feature embedding for final classification. The experimental results of the proposed framework on both individual dataset and on combined (HC3+ DAIGT v2) dataset reflects the model ability and consistency of differentiating AI-generated text across different folds and different length of text. The future work may focus on refining node features and incorporating contextual word embeddings to capture better semantic relationships along with graph-based structural representation.

Declarations

Use of AI: During the preparation of this manuscript, the authors used ChatGPT to improve the language, grammar, and readability of the manuscript. The authors reviewed and edited the generated content as necessary and take full responsibility for the content of this publication.

Conflicts of interest:

The authors declare no conflict of interest.

Data availability:

Publicly Available Dataset

Funding Information (mandatory)

This research work is supported by the Research Internal Funding Program (RIFP) 2026, of the University of Technology and Applied Sciences-Shinas, Sultanate of Oman.

Reference

- [1]. Weixin Liang, Yaohui Zhang, Mihai Codreanu, Jiayu Wang, Hancheng Cao, James Zou, The widespread adoption of large language model-assisted writing across society, *Patterns*, vol.6, No.12, 2025. <https://doi.org/10.1016/j.patter.2025.101366>.
- [2]. Anders Giovanni Møller, Daniel M. Romero, David Jurgens, Luca Maria Aiello. The impact of generative AI on social media: an experimental study. *Sci Rep* **16**, 9376, 2026. <https://doi.org/10.1038/s41598-026-40110-8>
- [3]. Sandeep Kumar, Samarth Garg, Sagnik Sengupta, Tirthankar Ghosal, Asif Ekbal. MixRevDetect: Towards Detecting AI-Generated Content in Hybrid Peer Reviews.. 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, vol.2, pp. 944–953, 2025. <https://aclanthology.org/2025.naacl-short.79/>
- [4]. Prabha Kannan, How Much Research Is Being Written by Large Language Models? Stanford Institute for Human-Centered Artificial Intelligence (HAI), Stanford University, May 13, 2024. Online Available: <https://hai.stanford.edu/news/how-much-research-being-written-large-language-models>. Last Accessed: Jul. 24, 2026.
- [5]. Miryam Naddaf, Major AI conference flooded with peer reviews written fully by AI, *Nature*, vol. 648, no. 8093, p. 256, Nov. 2025. <https://www.nature.com/articles/d41586-025-03506-6>
- [6]. Flávia A Rodrigues, Niclas F Sturm, Flávio L Pinheiro. A linguistic comparison between human- and AI-generated content. *iScience*. Vol. 29, Issue. 3114976, 2026. doi: 10.1016/j.isci.2026.114976.
- [7]. Junchao Wu, Shu Yang, Runzhe Zhan, Yulin Yuan, Lidia Sam Chao, Derek Fai Wong. A Survey on LLM-Generated Text Detection: Necessity, Methods, and Future Directions. *Computational Linguistics*, 51(1):275–338, 2025. <https://aclanthology.org/2025.cl-1.8/>
- [8]. Mireia Vendrell, Samantha-Kaye Johnston, Scaffolding critical thinking with generative AI: Design principles for integrating large language models in higher education, *Computers and Education: Artificial Intelligence*, vol. 10, 2026. <https://doi.org/10.1016/j.caeai.2026.100572>.
- [9]. Jiayin Zhi, Harsh Kumar, and Mina Lee. Investigating the Effects of LLM Use on Critical Thinking Under Time Constraints: Access Timing and Time Availability. In *Proceedings of the 2026 CHI*

- Conference on Human Factors in Computing Systems (CHI '26). Association for Computing Machinery, no.60, pp.1–21, 2026. <https://doi.org/10.1145/3772318.3791796>
- [10]. Mohammed Qorich, Rajae El Ouazzani, Detection of artificial intelligence-generated essays for academic assessment integrity using large language models, *Expert Systems with Applications*, vol.291, 2025. <https://doi.org/10.1016/j.eswa.2025.128405>.
- [11]. Ivan Vykopal, Matúš Pikuliak, Ivan Srba, Robert Moro, Dominik Macko, Maria Bielikova. Disinformation Capabilities of Large Language Models, *Proc. 62nd Annual Meeting of the Association for Computational Linguistics*, pp.14830–14847, 2024, <https://aclanthology.org/2024.acl-long.793.pdf>
- [12]. He Liu, Fabricated citations in the age of AI: A wake-up call for editors, reviewers, and authors, *Journal of Dental Sciences*, vol. 21, no. 1, pp. 679–680, 2026, <https://doi.org/10.1016/j.jds.2025.10.024>.
- [13]. Bassel Almarie, Paulo E. P. Teixeira, Kevin Pacheco-Barrios, Carlos Augusto Rossetti, Felipe Fregni, Editorial – The Use of Large Language Models in Science: Opportunities and Challenges, *Principles and Practice of Clinical Research*, vol. 9, no. 1, pp. 1–4, Jul. 2023. <https://journal.ppcr.org/index.php/ppcrjournal/article/view/259>
- [14]. Lisa Messeri, M. J. Crockett, The uncritical adoption of AI in science is alarming — we urgently need guard rails,” *Nature*, vol. 653, no. 8115, pp. 675–676, May 19, 2026. <https://www.nature.com/articles/d41586-026-01557-x>
- [15]. Dinushan Sivaneswaran, Chaminda T.E.R. Hewage, H.M.K.K.M.B. Herath, Rajkumar Singh Rathore, Vishal Krishna Singh, Weiwei Jiang, A systematic literature review of large language models in phishing attack generation and detection, *Array*, vol 30, 2026. <https://doi.org/10.1016/j.array.2026.100775>.
- [16]. Khalifa Afane, Wenqi Wei, Ying Mao, Junaid Farooq, Juntao Chen, Next-Generation Phishing: How LLM Agents Empower Cyber Attackers, 2024 IEEE International Conference on Big Data (BigData), pp. 2558–2567, 2024. doi: 10.1109/BigData62323.2024.10825018.
- [17]. Rommel Gutierrez, William Villegas-Ch, Jaime Govea, Cross-model evaluation of phishing detectors against LLM-generated emails. *Front. Big Data* 9:1883452. 2026. doi: 10.3389/fdata.2026.1883452
- [18]. Aman Roy, Shravika Mittal , Tanmoy Chakraborty, MG2Vec+: A multi-headed graph attention network for multigraph embedding. *Knowl Inf Syst* 65, 111–132, 2023. <https://doi.org/10.1007/s10115-022-01706-4>
- [19]. Mike Schuster, Kuldeep K. Paliwal, Bidirectional recurrent neural networks, in *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997, doi: 10.1109/78.650093.
- [20]. Thomas Roelleke , Jun Wang. TF-IDF uncovered: a study of theories and probabilities. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '08)*. Association for Computing Machinery, 435–442, 200. <https://doi.org/10.1145/1390334.1390409>.
- [21]. Andric Valdez-Valenzuela, Helena Gómez-Adorno, Manuel Montes-y-Gómez, Text Graph Neural Networks for Detecting AI-Generated Content, *Workshop on GenAI Content Detection (GenAIDetect)*, COLING, pp. 134–139, 2025. <https://aclanthology.org/2025.genaidetect-1.10.pdf>
- [22]. Xiaoming Liu, Zhaohan Zhang, Yichen Wang, Hang Pu, Yu Lan , Chao Shen, CoCo: Coherence-Enhanced Machine-Generated Text Detection Under Data Limitation With Contrastive Learning, *arXiv preprint arXiv:2212.10341*, Dec. 2022. doi: 10.48550/arXiv.2212.10341.
- [23]. Nidhi Gupta, Qinghua Li, Seeing Through the Mask: AI-Generated Text Detection with Similarity-Guided Graph Reasoning, *Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pp.1350–1360. 2025. <https://aclanthology.org/2025.findings-ijcnlp.84.pdf>
- [24]. Halah Mohammed Abbas. A Novel Approach to Automated Detection of AI-Generated Text. *Journal of Al-Qadisiyah for Computer Science and Mathematics*, 17(1), pp.1–17. 2025. <https://doi.org/10.29304/jqcs.2025.17.11958>
- [25]. Adilkhan Alikhanov, Aidar Amangeldi, Diar Demeubay, Dilnaz Akhmetzhan, Nurbek Moldakhmetov, Omar Polat, Galymzhan Zharas. AI Generated Text Detection, *arXiv*, Jan. 7, 2026. <https://arxiv.org/abs/2601.03812>
- [26]. Hello-SimpleAI, HC3: Human ChatGPT Comparison Corpus Dataset, *Hugging Face Datasets*, 2023. Available: <https://huggingface.co/datasets/Hello-SimpleAI/HC3>. Last Accessed: Feb. 8, 2026.
- [27]. Hello-SimpleAI, ChatGPT-Comparison-Detection, *GitHub repository*, 2023. Available: <https://github.com/Hello-SimpleAI/chatgpt-comparison-detection>. Last Accessed: Feb. 8, 2026.
- [28]. Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, Yupeng Wu. How Close is ChatGPT to Human Experts? Comparison Corpus, Evaluation, and Detection, *arXiv preprint arXiv:2301.07597*, 2023. <https://arxiv.org/abs/2301.07597>

- [29]. DAIGT V2 Train Dataset [Data set], Kaggle, 2023.
<https://www.kaggle.com/datasets/thedrcat/daigt-v2-train-dataset>. Last Accessed: Feb. 8, 2026.
- [30]. Abhinav Kathuria, Anu Gupta , R. K. Singla. A Review of Tools and Techniques for Preprocessing of Textual Data. In: Singh, V., Asari, V., Kumar, S., Patel, R. (eds) Computational Methods and Data Engineering. Advances in Intelligent Systems and Computing, Springer, vol 1227. 2021.
https://doi.org/10.1007/978-981-15-6876-3_31.
- [31]. Tomasz Szandała. Review and Comparison of Commonly Used Activation Functions for Deep Neural Networks. Bio-inspired Neurocomputing. Studies in Computational Intelligence, vol 903. 2021. Springer, Singapore. https://doi.org/10.1007/978-981-15-5495-7_11
- [32]. Stamatis Mastromichalakis, ALReLU: A Different Approach on Leaky ReLU Activation Function to Improve Neural Networks Performance," arXiv preprint arXiv:2012.07564, Dec. 2020.
<https://arxiv.org/pdf/2012.07564>.
- [33]. Dan Hendrycks, Kevin Gimpel, Gaussian Error Linear Units (GELUs), arXiv preprint arXiv:1606.08415, Jun. 2016. doi: 10.48550/arXiv.1606.08415.