



A Hybrid Framework Combining Stylometric, Semantic, and SBERT Features for AI-Generated Text Detection

Surapong Wiriya¹, Porawat Visutsak², and Chiranun Kamolsin^{1*}

¹ Faculty of Science and Technology, Chaopraya University, Nakhon Sawan, 60240, Thailand

² Faculty of Applied Science, King Mongkut's University of Technology North Bangkok, Bangkok, 10800, Thailand

* Correspondence: chiranun.k@cpu.ac.th

Citation:

Wiriya, S.; Visutsak, P.; Kamolsing, C. A hybrid framework combining stylometric, semantic, and SBERT features for AI-generated text detection. *ASEAN J. Sci. Tech. Report.* **2026**, *29*(11), e265688. <https://doi.org/10.55164/ajstr.v29i11.265688>

Article history:

Received: May 20, 2026

Revised: June 26, 2026

Accepted: August 15, 2026

Available online: October 14, 2026

Publisher's Note:

This article is published and distributed under the terms of Thaksin University.

Abstract: The rapid advancement of Large Language Models (LLMs) has greatly improved the ability of artificial intelligence systems to generate human-like text, making AI-generated text detection more challenging. This study introduces a hybrid framework that merges stylometric and semantic features with Sentence-BERT (SBERT) embeddings to enhance classification performance and robustness. For evaluation, experiments used the AI vs Human Text Dataset. Documents under 50 words were excluded to ensure data quality. The final dataset was balanced, with 100,000 documents evenly split between AI-generated and human-written texts. Stylometric features captured writing style, while semantic features measured sentiment, readability, information density, and lexical. SBERT embeddings captured deep contextual semantics. To improve computational efficiency, Principal Component Analysis (PCA) at 95% variance preservation reduced dimensionality. Four machine learning algorithms—Logistic Regression, Support Vector Machine (SVM), XGBoost, and LightGBM—were tested with 10-fold cross-validation. The hybrid framework combined with XGBoost performed best, achieving 98.63% accuracy, a macro F1-score of 0.9863, and an ROC-AUC of 0.9987. The Wilcoxon Signed-Rank Test confirmed that the approach significantly outperformed stylometric, semantic, and SBERT-only representations (p -value = 0.0020). Findings show that combining stylometric, semantic, and contextual embedding features captures both surface-level patterns and deep semantics, providing a reliable solution for detecting AI-generated text.

Keywords: Stylometric feature; semantic feature; SBERT feature; AI-generated text detection; machine learning

1. Introduction

Artificial intelligence (AI) has become an important technology that enables users to work more efficiently and quickly across many fields, like business, agriculture, entertainment, and education. In education, generative AI is increasingly being adopted for teaching, learning, and assessment. However, these tools sometimes generate inaccurate or misleading information, which can affect users. By using AI detection tools, users can verify whether data was generated by AI, enabling thorough verification. In this context, AI detection is the process of determining whether text was generated by AI or a human. It uses natural language processing and machine learning [1]. Nevertheless, it is important to note that AI detection tools do not guarantee the accuracy of their assessments. Humans write in diverse styles, and AI sometimes mimics human

writing. Therefore, modern AI models attempt to replicate human writing patterns, making detection increasingly difficult.

AI-generated text detection has become a key topic in NLP since the emergence of Large Language Models (LLMs). These models generate text that is highly similar to human language. In a previous study by Georgiou [2], the differences between human-written and AI-generated texts were examined with linguistic feature analysis. The analysis was performed at the levels of phonology, morphology, syntax, and lexicon. Results showed that AI-generated texts had higher lexical density and used more difficult content words. Human-written texts contained more function words, pronouns, and auxiliary verbs. The researchers used the Wilcoxon rank-sum test and LASSO regression to identify linguistic features that significantly influence the classification of AI and human-written texts. This work demonstrates that stylometric and linguistic features remain crucial for recognizing AI-generated text, even as LLMs become more natural.

Building on this foundation, deep learning architectures that use contextual embeddings have also been used for AI-generated text detection. In Das et al. [3], it was demonstrated that these models can learn contextual semantic representations. They used BERT embeddings with CNN and DistilBERT with BiLSTM. When the text is highly natural, results showed that contextual embeddings can greatly enhance detection efficiency. Similarly, Kumar et al. [4] improved detection by applying several machine learning algorithms to identify AI-generated text. These included logistic regression, random forests, XGBoost, and Support Vector Machine (SVM). Instead of relying solely on accuracy, the study also evaluated vocabulary richness, spelling and grammatical errors, and readability scores. The SVM (SVC) model achieved the highest recall of 0.93, demonstrating strong performance in detecting AI-generated text. Although SVM's accuracy matched XGBoost's at 0.87, its better balance across metrics led to it being recommended as the optimal choice. Following these trends, Aityan et al. [5] presented NEULIF, a lightweight, AI-powered text detection method that addresses the computational intensity of large transformer-based or ensemble models. NEULIF breaks text into stylometric and readability features and classifies them using a compact CNN or Random Forest. Both models are smaller than transformer ensembles and can run on a standard CPU without accuracy loss. This shows that simple models, designed with language structure in mind, can match the performance of complex text-detection models. Research supports the use of semantic embeddings from Transformer models to improve the learning of text context. For example, Mitchell et al. [6] presented DetectGPT, a zero-shot method for detecting AI-generated text. DetectGPT uses the language model's probability curvature to analyze the likelihood of AI-generated text. In Bao et al. [7], Fast-DetectGPT was presented, which improves detection performance. This method maintains high accuracy and also runs faster. Reimers et al. [8] showed that Sentence-BERT (SBERT) can efficiently generate sentence embeddings that reflect contextual meaning. These embeddings are suitable for text classification and semantic similarity tasks. Chen et al. [9] studied the use of Sentence Transformers with Contrastive Learning for detecting AI-generated text. The results showed that this combination significantly improves the detection of LLM-generated text. Marchitan et al. [10] presented transformer and hybrid deep learning-based models for AI-generated text detection. They used transformer- and hybrid-based architectures. Hybrid models can enhance detection performance but may still overfit in some cases. Moreover, the rapid development of AI detection techniques has led to the emergence of commercial tools such as GPTZero [11], which leverage perplexity and burstiness metrics to identify AI-generated text. The growing challenge of detecting machine-generated content across domains, models, and languages was highlighted by the SemEval-2024 Task 8 shared task on machine-generated text detection [12]. The task demonstrated the effectiveness of advanced ensemble strategies and transformer-based detectors, particularly RoBERTa-based classifiers and other fine-tuned architectures such as Longformer. Concurrently, proactive approaches such as LLM watermarking [13] have been proposed to embed imperceptible yet statistically detectable signals into generated text during generation.

Despite these advancements, existing methodologies exhibit critical limitations. Transformer-based detectors demand immense computational resources and suffer from a black-box nature, lacking interpretability. Conversely, proprietary tools like GPTZero offer limited transparency regarding their internal decision boundaries. Furthermore, studies that rely solely on raw SBERT embeddings often encounter the

curse of dimensionality, leading to overfitting. Another significant gap is that most studies treat stylometric features, semantic features, and contextual embeddings separately, which prevents models from learning both surface-level linguistic writing patterns and deep semantic representations simultaneously. Additionally, many studies lack rigorous statistical testing for significance, making it impossible to determine whether performance differences between models are statistically significant.

Novelty of the Proposed Hybrid Framework: To address these research gaps, this study proposes a novel, highly explainable Hybrid Feature Framework. Unlike existing studies that rely solely on deep embeddings or surface-level statistics, our framework innovatively fuses three distinct linguistic dimensions: stylometric (structural pacing), semantic (emotional and cohesion proxies), and context-aware SBERT embeddings. Uniquely, to mitigate dimensionality issues inherent in transformer models, we apply Principal Component Analysis (PCA) to condense the SBERT embeddings before concatenation. This hybrid integration enables traditional, lightweight ensemble models (e.g., XGBoost, LightGBM) to learn the text's structure and context, achieving state-of-the-art accuracy while maintaining high computational efficiency and full interpretability through feature-importance analysis. To ensure a robust comparison, several machine learning algorithms are evaluated under identical experimental conditions using stratified 10-fold cross-validation and an Independent Test Set, with the Wilcoxon Signed-Rank Test applied to assess statistical significance.

2. Materials and Methods

2.1 Dataset and Data Preprocessing

The experiments were conducted using the AI vs Human Text dataset (AI_Human.csv) [14]. The data preparation and experimental protocol were rigorously designed to ensure the integrity of the high performance reported in the proposed framework, explicitly mitigating risks of data memorization and data leakage.

1) Duplicate Removal Strategy: Prior to sampling, strict deduplication was enforced. All identical text strings were removed to prevent the model from memorizing repetitive instances, a common flaw that falsely inflates accuracy and ROC-AUC scores in AI-text detection literature.

2) Text Filtering: Texts containing fewer than 50 words were excluded. Short texts lack sufficient stylistic and semantic markers (such as burstiness and readability), which introduces noise into the feature extraction phase.

3) Data Splitting and Experimental Protocol: To clarify the relationship between the dataset partitions, the filtered, balanced dataset (100,000 records) was partitioned into a 70:15:15 split using stratified sampling. The training set (70%) was used exclusively for model training, feature scaling, PCA dimensionality reduction, and cross-validation. The validation set (15%) was used to monitor model stability during training and prevent overfitting (e.g., for early stopping). The independent test set (15%) was left untouched throughout the training and validation phases. It was used solely for unbiased final performance evaluation (accuracy, ROC-AUC, latency, and throughput). Keeping this set untouched ensures the reliability of the results and fully avoids data leakage.

4) 10-Fold Cross-Validation Integration: The stratified 10-fold CV was implemented strictly within the 70% training set. This decision was driven by the requirements of the Wilcoxon Signed-Rank Test to ensure robust statistical inference. Using a 10-fold CV yields 10 paired observations ($N=10$), providing sufficient statistical power to detect true significance ($p < 0.05$) when comparing the proposed hybrid framework against individual feature sets, without compromising the unseen independent test set.

5) Data Leakage Prevention (PCA and Feature Scaling): To support the credibility of the exceptionally high-performance metrics and completely avoid data leakage, rigid preprocessing pipelines were enforced. The StandardScaler (for feature normalization) and PCA ($n_components=0.95$, for SBERT dimensionality reduction) were fitted only on the 70% training set. The 15% validation set and the 15% independent test set were subsequently transformed using these pre-fitted parameters. This guarantees that no statistical distribution or variance from the unseen validation and test data influenced the model's feature space construction.

Table 1. Dataset Statistics

Statistic	Human	AI	Total
Number of documents	50,000	50,000	100,000
Average words/document	469.37	390.42	429.89
Average sentences/document	21.32	18.51	19.91
Average characters/document	2,355.01	2,127.78	2,241.40
Vocabulary size	74,018	70,300	114,827
Minimum words/document	55	55	55
Maximum words/document	1,764	1,359	1,764
Language	English	English	English

Table 1 presents the dataset, which was balanced using stratified random sampling. This method divides data into groups, or strata, and samples proportionally from each group. As a result, there is equal representation of AI-generated and human-written texts.

2.2 Feature Extraction

The present work uses feature extraction to identify key textual characteristics of writing style and semantic context. This supports the distinction between human and AI-generated text. The process comprises three feature groups: stylometric features, semantic features, and SBERT embeddings.

2.2.1 Stylometric Feature

Stylometric features are features used to analyze the writing style of text. They are popular in Authorship Attribution and AI-generated Text Detection because they can reflect the author's writing behavior and characteristics [15].

In this study, four stylometric features were extracted:

1) Average Sentence Length: Calculated as the mean number of words per sentence, this feature serves as a foundational metric for syntactic complexity. Variations in sentence length capture distinct stylistic signatures that can serve as critical indicators for authorship attribution and AI-generated text detection tasks.

2) Burstiness: Measured via the standard deviation of sentence lengths across the text to quantify structural variability. High burstiness indicates a dynamic, non-uniform sentence rhythm typical of human writing, whereas low burstiness reflects a more monotonous, uniform pattern often generated by language models.

3) Lexical Diversity: Represented by the Type-Token Ratio (TTR), which is the ratio of unique words to total words, to assess vocabulary richness. Significant deviations in lexical diversity help identify authorship by highlighting differences in vocabulary deployment between human authors and text-generation models.

4) Average Word Length: Evaluated as the mean character length per word, indicating morphological and vocabulary complexity. This feature captures specific structural tendencies in word choice, providing distinctive markers that help distinguish human-authored from machine-generated text.

These stylometric features provide complementary linguistic cues that effectively discriminate human-written from AI-generated texts. [16].

2.2.2 Semantic Feature

Semantic features analyze the meaning and context of a text, reflecting its deep linguistic characteristics.

Four semantic features were extracted:

1) Sentiment Score: The emotional analysis of the text using VADER sentiment analysis was used to calculate the overall sentiment score of the text.

2) Readability Score: Measured using the Flesch Reading Ease formula to assess reading ease, reflecting text surface-level and sentence structure.

3) Information Density: Assessed as the proportion of functional words (stopwords) relative to the total word count to indicate text density and information richness.

4) Lexical Entropy: Calculated using Shannon Entropy based on word frequency distributions to indicate text predictability, vocabulary diversity, and word complexity.

These semantic features allow the classifier to capture contextual consistency, emotional tendencies, and lexical complexity that are difficult to identify using structural features alone. [17]

2.2.3 SBERT

In addition to traditional linguistic features, the present work uses Sentence-BERT (SBERT) to extract sentence embeddings, thereby enhancing the ability to learn deep contextual semantics of texts [8]. SBERT is a model derived from BERT, designed to generate Sentence Embeddings that effectively capture semantic relationships in text, making it suitable for text similarity and classification tasks. In this study, the “*all-MiniLM-L6-v2*” model from SentenceTransformers was used to generate a 384-dimensional text embedding vector for each document. These SBERT features enable more effective representation of contextual and implied meanings than traditional methods such as Bag-of-Words or TF-IDF.

2.3 Hybrid Framework Feature

To improve the efficiency of classifying AI-generated versus human-written text, the framework presents a hybrid feature set integration approach that combines features from multiple data sources, including stylometric and semantic features, as well as SBERT embeddings. The hybrid feature set integrates linguistic structure with deep semantic information, enabling the model to learn both writing style and contextual meaning. In this step, features from each group are combined using Feature Concatenation, which joins the feature vectors from each source side by side (horizontally) before feeding them into the machine learning model. Combining stylometric and semantic features enables the model to analyze linguistic characteristics more effectively. SBERT embeddings enhance its ability to learn deep semantic context, enabling it to differentiate more accurately between AI-generated and human-written text.

The hybrid approach overcomes the limitations of relying on a single feature type. Stylometric features reflect only writing style, while SBERT focuses on learning contextual meaning. Combining these feature types expands the information available for classification [18]. After feature integration, all data will be processed using PCA to reduce dimensionality before proceeding to the next step of machine learning model building.

2.4 Dimensionality Reduction

The semantic features obtained from the Sentence-BERT (SBERT) model have a maximum dimensionality of 384. Using all features directly can result in a highly complex model, increased processing time, and a higher risk of overfitting, especially when multiple feature types are combined in a hybrid feature-set model. Therefore, this study uses PCA to reduce data dimensionality prior to classification. PCA is a widely used statistical dimensionality reduction technique in machine learning and high-dimensional data analysis. PCA transforms the data into principal components that preserve important information while reducing redundancy [19]. The main objective of using PCA in this study is to address the Curse of Dimensionality, which occurs when the number of features is too large, leading to a more complex model, reduced performance, and increased risk of overfitting. Various studies have indicated that dimensionality reduction techniques can help reduce model complexity, improve learning efficiency, reduce the impact of irrelevant or redundant features [20], and reduce computational load. (Computational Cost) Moreover, speed up model training and testing [21]. In this work, PCA was applied to SBERT feature vectors after the Embedding extraction process, with the Explained Variance Ratio set to 95% (0.95) to preserve most of the dataset's important information while reducing dimensionality. After PCA-based dimensional reduction, the SBERT vectors were reduced from 384 to 188 dimensions, enabling the model to learn the data more efficiently while preserving important semantic information.

2.5 Machine Learning Classifiers

To evaluate the performance of the four feature sets mentioned above, this experiment selected four classification algorithms spanning both linear and tree-based ensemble methods. All models had a fixed seed value (random_state=42) to ensure reproducibility of experimental results. Details and parameter settings are as follows:

2.5.1 Logistic Regression

Logistic Regression (LR) is the baseline model for this study due to its high interpretability and strong performance on high-dimensional text data. The maximum iteration parameter, `max_iter=1000`, is set to ensure the optimization converges to a minimum on a dataset of 50,000 items. This is consistent with the research of Shah, et al. [22], which showed that when Logistic Regression is combined with stylometric features, it can serve as a robust baseline for detecting AI-generated text with high accuracy.

2.5.2 SVM

SVM finds the optimal hyperplane that maximizes the margin for separating the data. This model is highly efficient at handling complex semantic features, such as SBERT vectors. The `kernel='linear'` parameter is used to reduce processing time on large datasets, and `max_iter=1000` limits the maximum number of iterations. In addition, `probability=True` is set to enable the model to estimate ROC-AUC, as shown in Kumar et al. [4]. This highlights its effectiveness in detecting AI-generated text.

2.5.3 XGBoost

Extreme Gradient Boosting (XGBoost) is an ensemble algorithm that builds a sequence of decision trees to correct errors from previous iterations (boosting). XGBoost can efficiently capture hidden nonlinear relationships in hybrid datasets. The parameters `eval_metric='logloss'` are set for evaluating the loss function in binary classification mode, and `use_label_encoder=False` is set to turn off outdated alerts. In the study by Krawczyk et al. [23], pre-trained fastText embeddings performed worse, whereas the XGBoost classifier achieved the highest minimum accuracy of 96.24%. For most methods, specificity and precision exceeded 99%, indicating a robust ability to distinguish student-created from AI-generated text.

2.5.4 LightGBM

Light Gradient Boosting Machine (LightGBM) is a gradient boosting framework that utilizes a leaf-wise tree growth strategy and histogram-based learning. Consequently, LightGBM trains faster and consumes less memory than XGBoost, making it well-suited for large datasets. Its default parameters prioritize performance, with `'verbose'` set to -1 to suppress output during training. In an applied scenario, Ahmed et al. [24] integrated TF-IDF with LightGBM to differentiate AI-generated from human-written text, achieving superior performance among feature-based methods.

2.6 Experimental Setup

To ensure the fairness and reproducibility of experimental results, this study uses a fixed seed value in all stochastic processes. The dataset was split into training, validation, and test sets. Stratified sampling with a 70:15:15 ratio was used to maintain class balance across all experiments; PCA dimensionality reduction, 10-fold cross-validation, and initial weights were used for all machine learning algorithms. This ensures fair performance comparisons across all models.

2.6.1 Hardware and Software Environment

All feature extraction and model training processes were conducted on a Windows workstation. The system used an AMD Ryzen 5 5500 processor, 16 GB RAM, and an RX 6600 GPU for deep learning computation.

2.6.2 Python Libraries and Frameworks

The development of the experimental framework relies on the following standard data science and natural language processing (NLP) libraries:

- 1) Data manipulation and mathematical computation: pandas is used for handling tabular data structures, and NumPy for array and matrix computation.
- 2) Text processing and feature extraction: nltk is used for tokenization and stopword removal, textstat for evaluating readability metrics, and textblob for text sentiment analysis. Sentence-transformers were also used to extract deep embeddings through the *all-MiniLM-L6-v2* model
- 3) Model building and evaluation: The scikit-learn library is integrated for PCA, cross-validation, logistic regression, and SVM operations, as well as calculating various metrics. Advanced tree-based models are run via the dedicated libraries XGBoost and LightGBM.

4) Statistical testing and data visualization: Wilcoxon Signed-Rank Tests are performed using the *scipy.stats* library, and data visualizations, such as ROC-AUC curves, Confusion Matrices, and Feature Importance, are generated using *matplotlib* and *seaborn*.

2.7 Evaluation Metrics

To evaluate the performance of AI-generated and human-written text classification models, this experiment employs multiple criteria to assess accuracy, class balance, and overall performance. The evaluation criteria used include:

1) Accuracy

Accuracy is the proportion of samples the model correctly classifies relative to the total number of samples. It is calculated using equation (1) [25].

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

Where: TP (True Positive) is the number of AI-generated texts correctly classified.

TN (True Negative) is the number of human-written texts correctly classified.

FP (False Positive) is the number of human-written texts incorrectly classified as AI.

FN (False Negative) is the number of AI-generated texts incorrectly classified as Human.

2) Precision

Precision measures the accuracy of the model's predictions in the group it classifies as AI text. It is calculated from equation (2).

$$\text{Precision} = \frac{TP}{TP+FP} \quad (2)$$

3) Recall

Recall, or sensitivity, measures the model's ability to detect AI-generated text correctly. It is calculated from equation (3).

$$\text{Recall} = \frac{TP}{TP+FN} \quad (3)$$

4) Macro F1-score

Macro F1-score is the average of the F1-scores in each class, which helps reflect the balance between precision and recall. It is calculated from equation (4).

$$\text{F1 - score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (4)$$

In the experiments, the macro F1-score was chosen as the main indicator because it reflects the model's performance across classes in a balanced way.

5) ROC-AUC score

The Receiver Operating Characteristic–Area Under the Curve (ROC-AUC) is a widely used metric for assessing a classifier's ability to distinguish between two classes. The ROC curve displays the True Positive Rate (TPR) versus the False Positive Rate (FPR) across various classification thresholds, and the AUC quantifies the model's overall discriminative performance on a scale from 0 to 1. An AUC value approaching 1 signifies superior discrimination between AI-generated and human-written text [26].

6) Computational Efficiency

This section explains the computational performance metrics—Latency and Throughput—used to evaluate the practical suitability of models for automated text detection systems. In addition to measuring accuracy, assessing these metrics with the independent test set enables a comprehensive evaluation.

(1) Latency

Latency refers to the average time the model takes to predict one sample of data, which reflects the model's "response speed," calculated in milliseconds per sample (ms/sample) as shown in equation (5) [27]:

$$\text{Latency} = \frac{\text{Inference Time}}{\text{Number of Samples}} \times 1000 \quad (5)$$

Where: Inference Time is the total time taken to predict all data in the test set.

Number of Samples is the number of data points in the test set.

A low latency value indicates that the model can respond quickly.

(2) Throughput

Throughput refers to the number of data samples that a model can process within one second (samples per second), which reflects the model's "workload capacity" and is calculated from equation (6):

$$\text{Throughput} = \frac{\text{Number of Samples}}{\text{Inference Time}} \quad (6)$$

A high throughput value indicates that the model can efficiently process large amounts of data, making it suitable for systems that require high-volume processing.

In this experiment, the latency and throughput of each model were measured in the same environment on an independent test set to enable a fair comparison of computational performance between models and feature representations.

2.8 Statistical Significance Testing

This section describes the approach used to determine whether differences in model results are statistically significant. Specifically, the Wilcoxon Signed-Rank Test is applied to the macro F1-score obtained from stratified 10-fold cross-validation.

Extending this approach, the Wilcoxon Signed-Rank Test is a nonparametric test for comparing paired samples that does not assume normality [28]. Because each model is evaluated on the same fold during cross-validation, the resulting macro F1-scores are considered paired. This test has become widely adopted in text classification and natural language processing research to compare model performance [29]. In this study, the proposed framework served as the reference model and was compared against other feature sets, with statistical significance set at $p < 0.05$.

3. Results and Discussion

The experiments compared four feature sets: stylometric, semantic, SBERT, and a hybrid. These were tested across four models: Logistic Regression, SVM, XGBoost, and LightGBM. Performance was evaluated using accuracy, precision, recall, macro F1-score, ROC-AUC, latency, and throughput. Statistical significance was assessed with the Wilcoxon Signed-Rank Test. The following sections present and discuss these comparative results in detail.

3.1 Performance Comparison of Feature Sets

Table 2. Performance comparison of feature sets

Feature set	Model	Accuracy	Precision	Recall	F1-score
Stylometric	Logistic Regression	0.7739	0.7739	0.7739	0.7739
	SVM	0.5683	0.6274	0.5683	0.5117
	XGBoost	0.8294	0.8302	0.8294	0.8293
	LightGBM	0.8222	0.8230	0.8222	0.8220
Semantic	Logistic Regression	0.8008	0.8008	0.8008	0.8008
	SVM	0.5214	0.6178	0.5214	0.3982
	XGBoost	0.8335	0.8335	0.8335	0.8335
	LightGBM	0.8340	0.8340	0.8340	0.8340
SBERT	Logistic Regression	0.9382	0.9382	0.9382	0.9382
	SVM	0.6899	0.6899	0.6899	0.6899
	XGBoost	0.9750	0.9750	0.9750	0.9750
	LightGBM	0.9554	0.9554	0.9554	0.9553
Hybrid proposed (Styl+Sem+SBERT)	Logistic Regression	0.9536	0.9536	0.9536	0.9536
	SVM	0.7210	0.7246	0.7210	0.7199
	XGBoost	0.9863	0.9863	0.9863	0.9863
	LightGBM	0.9712	0.9713	0.9712	0.9712

Table 2 demonstrates that the hybrid framework delivers the best results among all feature sets. The proposed framework, combined with XGBoost, achieved the highest accuracy of 0.9863, along with precision, recall, and macro F1-scores of 0.9863, which are higher than those of all other methods tested. These results confirm the model's ability to accurately and consistently classify AI-generated and human-written text. The hybrid framework, combined with LightGBM, ranked second, yielding an accuracy and macro F1-score of 0.9712, while SBERT with XGBoost achieved an accuracy and macro F1-score of 0.9750. Considering only traditional features, semantic features performed slightly better than stylometric features, with semantic features with LightGBM yielding an accuracy of 0.8340, while stylometric features with XGBoost yielded an accuracy of 0.8294. Conversely, SVM yielded the lowest results in many cases, particularly in semantics, where it achieved an accuracy of only 0.5214 and a macro F1-score of 0.3983, indicating that SVM may not be suitable for complex semantic and multidimensional data.

The findings indicate that combining multiple feature types significantly improves AI-generated text detection. The model effectively learns both linguistic structure and semantic context simultaneously. Stylometric features reflect writing styles such as sentence length, vocabulary variety, and text inconsistencies, while semantic features analyze text consistency and meaning. SBERT embeddings learn deep contextual semantics. The results with the proposed framework model, combined with XGBoost, show the highest accuracy and macro F1-score, demonstrating that ensemble learning efficiently captures relationships among

multiple feature types, especially in heterogeneous data. Moreover, the similar precision and recall values indicate balanced classification with fewer false positives and false negatives.

The experimental results demonstrate that the proposed hybrid framework significantly outperforms individual feature sets across all evaluated metrics. Furthermore, comparative analysis reveals distinct performance characteristics among the machine learning algorithms. Among the ensemble learners, XGBoost demonstrated marginally superior performance compared to LightGBM. This can be attributed to their underlying tree-growth mechanisms. At the same time, LightGBM employs a histogram-based, leaf-wise strategy optimized for speed; XGBoost utilizes a level-wise growth strategy with exact greedy algorithms, allowing it to capture more complex, non-linear interactions between sparse stylometric features and dense SBERT embeddings. Conversely, SVM exhibited the lowest performance, particularly when handling semantic and hybrid features. Given the use of a linear kernel for computational feasibility, the SVM failed to draw an optimal hyperplane through the highly non-linear and heterogeneous feature space, where low-variance statistical counts (stylometric) are mixed with high-dimensional dense vectors (SBERT). In contrast, tree-based models natively exploit these hierarchical dependencies, leading to their robust classification performance.

In terms of practical implications, this hybrid framework offers a scalable solution for real-world AI detection systems, such as academic integrity tools and misinformation filters. By combining PCA-reduced SBERT with lightweight stylometric features, the framework achieves state-of-the-art accuracy (98.63%) without the computational overhead of deploying massive transformer models in production. Moreover, unlike deep learning black-box models, the hybrid approach maintains high transparency; educators and system moderators can interpret feature importance—such as perplexity or burstiness metrics—to justify why a specific text was flagged as AI-generated. A confusion matrix for the proposed framework is shown in Figure 1. The XGBoost model has two categories: Human (0) and AI (1), each with 7,500 samples. There are 84 false positives, where AI (1) was predicted but the sample was Human (0). There are 121 false negatives, where Human (0) was predicted but the sample was AI (1).

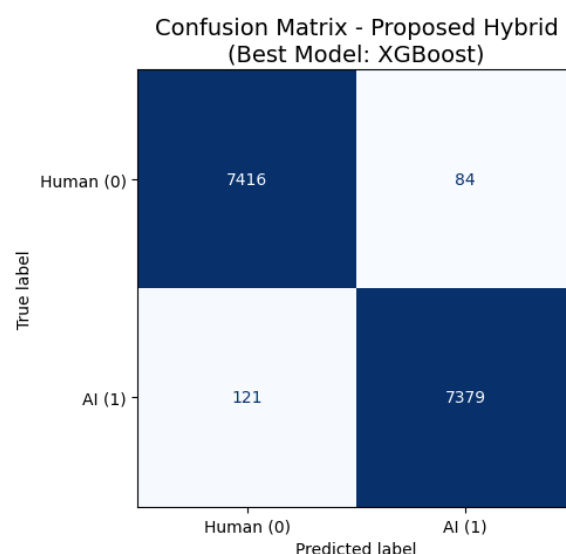


Figure 1. Confusion Matrix for the hybrid framework with XGBoost.

3.2 ROC-AUC Analysis

Based on the ROC-AUC graph in Figure 2, the proposed framework achieves the highest differentiation between AI-generated and human-written text classes among the feature sets. This framework, combined with XGBoost, achieves the highest ROC-AUC of 0.9987, followed by the proposed framework with LightGBM at 0.9966 and SBERT with XGBoost at 0.9969. Among stylometric features, XGBoost achieves an ROC-AUC of 0.9153, while semantic with XGBoost and semantic with LightGBM yield ROC-AUCs of 0.9197 and 0.9180, respectively. SVM consistently yields the lowest results, with semantic and SVM achieving only a 0.6374 ROC-AUC, which is considerably lower than that of other methods.

The high ROC-AUC results from the proposed framework indicate that the model excels at distinguishing AI-generated from human-written text. Demonstrating strong performance across all decision thresholds, the findings show that the hybrid framework allows the model to capture both writing patterns and semantic context more comprehensively, yielding highly accurate classification. Moreover, the hybrid framework's ROC-AUC value of 0.9987 with XGBoost confirms the model's superior class-probability ranking, which is crucial for robust AI-generated text detection. These findings also establish that SBERT is critical to improving ROC-AUC performance, as contextual embeddings capture text semantics more effectively than traditional methods.

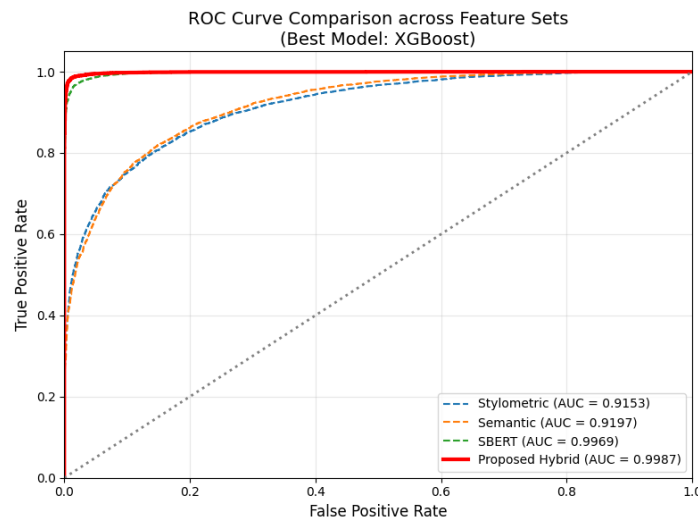


Figure 2. ROC Curve Comparison across Feature Sets.

3.3 Effectiveness of Hybrid Feature

Combining these three feature groups into a hybrid feature set enables precise detection of human and AI writing styles across multiple dimensions, addressing the limitations of each method as follows:

Stylometric features encompass structural and stylistic patterns, such as varied sentence length and lexical diversity. For instance, humans often shift between short and long sentences, while AI typically generates sentences of uniform length and demonstrates limited lexical diversity.

Semantic features detect emotion, fluency, and linguistic complexity. AI-generated text generally avoids aggressive language and favors neutral, accessible, and grammatically accurate prose, often relying on probabilistic word prediction, which can result in overly smooth transitions. Human writing, by contrast, shows pronounced emotional expression. AI-generated text often lacks linguistic complexity.

The third group, **SBERT embeddings**, extracts deep contextual representations by transforming text into vectors that capture high-level linguistic understanding, such as subject-verb-object relationships, synonyms, and holistic context.

Combining these three methods creates a complementary relationship that fills each method's blind spots. SBERT understands deep context well but lacks sentence structure; stylometrics captures structure but lacks semantics; Semantic focuses on emotion but misses writing patterns. Concatenating them with *np.hstack* allows a model like LightGBM to capture both style and context. With a feature space spanning all linguistic dimensions, tree-based models such as LightGBM and XGBoost efficiently learn decision boundaries, achieving optimal accuracy. This leads to strong F1, ROC-AUC, and stability scores. Experimental results clearly demonstrate that the proposed framework achieves the highest performance across all key metrics, with XGBoost achieving an accuracy and macro F1-score of 0.9863 and an ROC-AUC of 0.9987. This confirms that effective AI-generated text detection requires multidimensional features, as newer LLMs, such as GPT-based models, closely mimic human linguistic structure and semantic context.

3.4 Comparative Analysis of Machine Learning Models

The experimental results show that the machine learning algorithms exhibit markedly different performance in detecting AI-generated text, especially when used with different feature sets. Ensemble learning

models, namely XGBoost and LightGBM, outperform Logistic Regression and SVM across all feature sets, particularly when used with the hybrid framework. XGBoost achieved the best accuracy, macro F1-score, and ROC-AUC across all experiments, while LightGBM performed similarly to XGBoost. This indicates that Gradient Boosting models efficiently learn complex relationships among multiple feature types, especially in a hybrid feature set encompassing all linguistic dimensions. Conversely, Logistic Regression performed significantly worse on accuracy and macro F1-score than XGBoost and LightGBM, especially when using SBERT embeddings and a hybrid feature set for multidimensional, highly complex data. Additionally, the experiments revealed that SVM yielded the lowest results across all experimental sets. Specifically, when working with semantic features, the accuracy was only 0.5214, and the macro F1-score was 0.3982. SVMs' limitations in handling large, complex datasets likely stem from the increased processing demands of Kernel Functions as data volume and feature dimensions increase. As a result, SVM is less efficient at pattern learning than ensemble models. Overall, these findings indicate that XGBoost and LightGBM, when used with a hybrid feature set, are best suited for AI-generated text detection, balancing accuracy, generalization, and efficiency. Logistic Regression remains viable for speed-focused tasks. In contrast, SVM is not recommended for embedded or highly complex data in this study.

3.5 Statistical Significance Testing

The results of the Wilcoxon Signed-Rank Test on the macro F1-score from the 10-fold cross-validation evaluation are shown in Table 3. The hybrid framework was set as the reference model for comparison with other feature sets, namely stylometric, semantic, and SBERT. The experimental results showed that the proposed framework achieved the highest average macro F1-score of 0.9838 ± 0.0010 , surpassing all other feature sets. SBERT gave a macro F1-score of 0.9727 ± 0.0017 , while stylometric and semantic gave macro F1-scores of 0.8376 ± 0.0042 and 0.8350 ± 0.0040 , respectively. The Wilcoxon Signed-Rank Test yielded a p-value of 0.0020 when comparing the hybrid framework with other feature sets, which is below the 0.05 significance level. This result confirms the superior performance of the proposed framework. The improvement is attributed to feature integration rather than random variation during cross-validation. While SBERT achieved a high macro F1-score of 0.9727, the hybrid framework still performed significantly better. This suggests that contextual embeddings alone do not capture all linguistic features necessary for AI-generated text detection. SBERT excels at learning context and meaning (contextual semantics), supporting semantic analysis and coherence. However, it may overlook detailed stylistic aspects. By contrast, stylometric features capture writing patterns—including sentence length, vocabulary variety, and structural inconsistency—that continue to distinguish human-written from language model-generated text. Additionally, the hybrid framework's low standard deviation of ± 0.0010 indicates stable, generalized performance (as shown in Figure 3)—a crucial trait for practical text detection systems.

Table 3. Wilcoxon Signed-Rank Test results on macro F1-score using 10-fold cross-validation

Feature Representation	Macro F1-score (Mean $\pm$ Std)	p-value	Result
Proposed Hybrid	0.9838 ± 0.0010	-	Reference
Stylometric	0.8376 ± 0.0042	.0020	Significant
Semantic	0.8350 ± 0.0040	.0020	Significant
SBERT	0.9727 ± 0.0017	.0020	Significant

3.6 Computational Performance Analysis

Experimental results on time performance (Table 4) show that Logistic Regression achieves the lowest latency and the highest throughput across all feature sets. When combined with stylometric features, it achieves the lowest latency at 0.000042 milliseconds per sample and the highest throughput at 2.39×10^7 samples per second, followed by its combination with the semantic feature set. Among high-accuracy models, the hybrid framework paired with XGBoost achieves a latency of 0.000775 milliseconds per sample and a throughput of 1.29×10^6 samples per second, making it computationally efficient and practical for real-world applications. In contrast, across all three feature groupings—individual features, SBERT with SVM, and hybrid framework with SVM—the SVM exhibits the highest latency and the lowest throughput. In particular, the

SBERT with SVM and the hybrid framework with SVM combinations record throughputs below 10^4 samples per second. Although Logistic Regression is the fastest, its accuracy and macro F1-score are substantially lower than those of XGBoost and LightGBM. Both XGBoost and LightGBM require marginally more processing time than Logistic Regression, but their latency remains well below 1 millisecond per sample, making them suitable for automated text detection systems. Likewise, the hybrid framework, paired with XGBoost, optimally balances accuracy and speed, achieving the highest accuracy while delivering high throughput. In contrast, SVM underperforms on both metrics, as kernel-based learning is computationally intensive when processing large volumes of high-dimensional data, particularly with SBERT embeddings and a hybrid feature set that combines diverse feature groups.

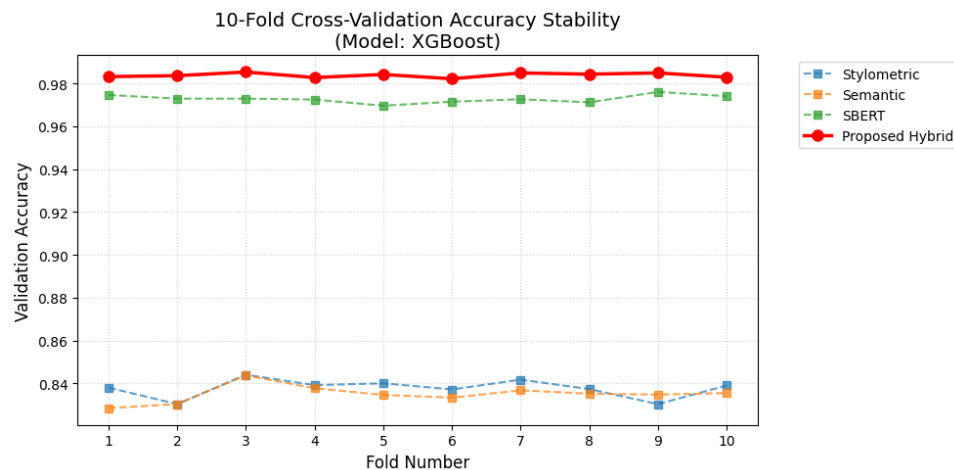


Figure 3. 10-fold cross-validation accuracy stability. The line chart illustrates the fold-wise validation accuracy of the best-performing model across the 10-fold cross-validation, demonstrating the consistency and robustness of the proposed hybrid feature set compared to individual feature sets.

Table 4. Performance comparison of Latency Time and Throughput

Feature set	Model	Latency Time (ms/)	Throughput (sample/s)
Stylometric	Logistic Regression	0.000042	2.388155e+07
	SVM	0.024691	4.050063e+04
	XGBoost	0.000471	2.121551e+06
	LightGBM	0.001417	7.055868e+05
Semantic	Logistic Regression	0.000044	2.265861e+07
	SVM	0.026358	3.793872e+04
	XGBoost	0.000644	1.552458e+06
	LightGBM	0.001517	6.593609e+05
SBERT	Logistic Regression	0.000740	1.351364e+06
	SVM	0.153339	6.521515e+03
	XGBoost	0.000774	1.292557e+06
	LightGBM	0.002086	4.795013e+05
Proposed Hybrid (Styl+Sem+SBERT)	Logistic Regression	0.000411	2.432617e+06
	SVM	0.138167	7.237594e+03
	XGBoost	0.000775	1.289624e+06
	LightGBM	0.002074	4.820640e+05

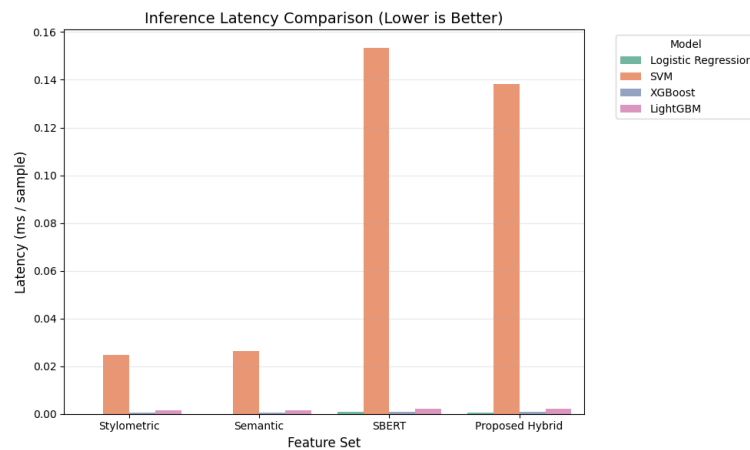


Figure 4. Latency (ms/sample) comparison across feature sets.

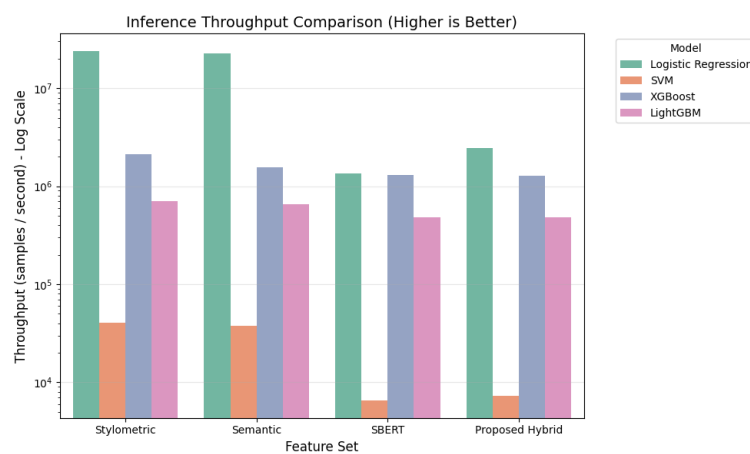


Figure 5. Throughput (sample/second) comparison across feature sets.

4. Conclusions

The present study proposes a hybrid framework for AI-generated text detection that combines stylometric and semantic features with SBERT embeddings. Experiments used the English-language AI vs Human Text Dataset, excluding documents with fewer than 50 words. A balanced random sample of 100,000 documents, including 50,000 AI-generated and 50,000 human-written texts, was selected to eliminate class imbalance and enhance the reliability of the evaluation. Experimental results demonstrated that the proposed hybrid framework consistently outperformed models that relied solely on stylometric, semantic, or SBERT features. Among the evaluated classifiers, XGBoost achieved the best performance, with an accuracy of 0.9863, a macro F1-Score of 0.9863, and a ROC-AUC of 0.9987, demonstrating the effectiveness of combining complementary linguistic representations for AI-generated text detection. Furthermore, the Wilcoxon Signed-Rank Test confirmed that the proposed framework significantly outperformed the comparison methods ($p = 0.0020$), indicating that the observed performance improvement was statistically significant rather than due to random variation. In addition, ensemble learning models, particularly XGBoost and LightGBM, maintained low inference latency and adequate throughput, making them well-suited for practical deployment in automated AI-generated text detection systems. Despite its strong performance, the proposed framework has several limitations. The experiments were conducted using only a single English-language dataset, and its applicability to multilingual and morphologically rich languages remains unverified. Moreover, the rapid advancement of large language models may gradually reduce the effectiveness of traditional stylometric and semantic features as newer models become increasingly capable of mimicking human writing styles. Future research should therefore evaluate the proposed framework on multilingual benchmark datasets, incorporate texts generated by state-of-the-art large language models such as GPT-4o, Claude, and Gemini, and investigate

the integration of deep learning architectures or large language models with hybrid linguistic features. Additionally, adversarial training strategies should be explored to improve further the robustness and generalization capability of AI-generated text detection systems.

Author Contributions: S.W., P.V. and C.K. conceived of the presented idea. S.W. experimented. S.W., P.V. and C.K. verified the analytical methods. S.W. supervised the project. S.W. wrote the manuscript. All authors discussed the results and contributed to the final manuscript.

Funding: This research was funded by Chaopray University.

Conflicts of Interest: “The authors declare no conflict of interest.”

References

- [1] Gotoman, J. E. J.; Luna, H. L. T.; Sangria, J. C. S.; Santiago, C. S., Jr.; Barbuco, D. D. Accuracy and Reliability of AI-Generated Text Detection Tools: A Literature Review. *Am. J. IR* **2025**, *4*(1), 1–9. <https://doi.org/10.54536/ajirb.v4i1.3795>
- [2] Georgiou, G. P. Differentiating between Human-Written and AI-Generated Texts Using Automatically Extracted Linguistic Features. *Information* **2025**, *16*(11), 979. <https://doi.org/10.3390/info16110979>
- [3] Das, R.; Dey, R.; Mondal, S.; Das, N.; Sadhukhan, B. Enhancing AI-Generated Text Identification with BERT-CNN and DistilBERT-BiLSTM Models. In *Proceedings of the 2025 International Conference on Computing, Intelligence, and Application (CIACON)*; IEEE: Piscataway, NJ, 2025; pp 1–6.
- [4] Kumar, S.; Tiwari, S.; Prasad, R.; Rana, A.; Arti, M. K. Comparative Analysis of Human and AI Generated Text. In *Proceedings of the 2024 11th International Conference on Signal Processing and Integrated Networks (SPIN)*; IEEE: Piscataway, NJ, 2024; pp 168–173.
- [5] Aityan, S. K.; Claster, W.; Emani, K. S.; Rais, S.; Tran, T. A Lightweight Approach to Detection of AI-Generated Texts Using Stylometric Features. *arXiv* **2025**, arXiv:2511.21744. <https://doi.org/10.48550/arXiv.2511.21744>
- [6] Mitchell, E.; Lee, Y.; Khazatsky, A.; Manning, C. D.; Finn, C. DetectGPT: Zero-Shot Machine-Generated Text Detection Using Probability Curvature. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*; 2023. <https://arxiv.org/abs/2301.11305>
- [7] Bao, G.; Zhao, Y.; Teng, Z.; Yang, L.; Zhang, Y. Fast-DetectGPT: Efficient Zero-Shot Detection of Machine-Generated Text via Conditional Probability Curvature. In *Proceedings of the International Conference on Learning Representations (ICLR)*; 2024; pp 1–23.
- [8] Reimers, N.; Gurevych, I. Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*; 2019; pp 3982–3992. <https://doi.org/10.18653/v1/D19-1410>
- [9] Chen, H.; Büssing, J.; Rügamer, D.; Nie, E. Team MGTD4ADL at SemEval-2024 Task 8: Leveraging (Sentence) Transformer Models with Contrastive Learning for Identifying Machine-Generated Text. In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*; 2024; pp 1711–1718. <https://doi.org/10.18653/v1/2024.semeval-1.245>
- [10] Marchitan, T. G.; Creanga, C.; Dinu, L. P. Transformer and Hybrid Deep Learning Based Models for Machine-Generated Text Detection. *arXiv* **2024**, arXiv:2405.17964. <https://doi.org/10.48550/arXiv.2405.17964>
- [11] Tian, E. GPTZero: A Tool to Detect AI-Generated Text; GPTZero, 2023. <https://gptzero.me/> (accessed June 23, 2026).
- [12] Wang, Y.; Mansurov, J.; Ivanov, P.; Su, J.; Shelmanov, A.; Tsvigun, A.; et al. SemEval-2024 Task 8: Multidomain, Multimodel and Multilingual Machine-Generated Text Detection. In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*; 2024; pp 2057–2079.
- [13] Kirchenbauer, J.; Geiping, J.; Wen, Y.; Katz, J.; Miers, I.; Goldstein, T. A Watermark for Large Language Models. In *Proceedings of the 40th International Conference on Machine Learning (ICML 2023)*; PMLR, 2023; pp 17061–17084.
- [14] Gerami, S. *AI vs. Human Text*; Kaggle, 2026. <https://www.kaggle.com/datasets/shanegerami/ai-vs->

- [15] Juola, P. Authorship Attribution. *Found. Trends Inf. Retr.* **2008**, 1(3), 233–334.
- [16] Stamatatos, E. A Survey of Modern Authorship Attribution Methods. *J. Am. Soc. Inf. Sci. Technol.* **2009**, 60(3), 538–556.
- [17] Schütze, H.; Manning, C. D.; Raghavan, P. *Introduction to Information Retrieval*; Cambridge University Press: Cambridge, U.K., 2008; pp 234–265.
- [18] Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, 2016.
- [19] Jolliffe, I. T.; Cadima, J. Principal Component Analysis: A Review and Recent Developments. *Philos. Trans. R. Soc. A* **2016**, 374(2065), 20150202. <https://doi.org/10.1098/rsta.2015.0202>
- [20] Anowar, F.; Sadaoui, S.; Selim, B. Conceptual and Empirical Comparison of Dimensionality Reduction Algorithms (PCA, KPCA, LDA, MDS, SVD, LLE, Isomap, LE, ICA, t-SNE). *Comput. Sci. Rev.* **2021**, 40, 100378. <https://doi.org/10.1016/j.cosrev.2021.100378>
- [21] Rastogi, A. K.; Taterh, S.; Kumar, B. S. Dimensionality Reduction Algorithms in Machine Learning: A Theoretical and Experimental Comparison. *Eng. Proc.* **2023**, 59(1), 82. <https://doi.org/10.3390/engproc2023059082>
- [22] Shah, A.; Ranka, P.; Dedhia, U.; Prasad, S.; Muni, S.; Bhowmick, K. Detecting and Unmasking AI-Generated Texts through Explainable Artificial Intelligence Using Stylistic Features. *Int. J. Adv. Comput. Sci. Appl.* **2023**, 14(10), 1043–1053. <https://doi.org/10.14569/IJACSA.2023.01410110>
- [23] Krawczyk, N.; Probierz, B.; Kozak, J. Towards AI-Generated Essay Classification Using Numerical Text Representation. *Appl. Sci.* **2024**, 14(21), 9795. <https://doi.org/10.3390/app14219795>
- [24] Ahmed, S.; Hammond, T. *DependencyAI: Detecting AI Generated Text through Dependency Parsing*. *arXiv* **2026**, arXiv:2602.15514. <https://doi.org/10.48550/arXiv.2602.15514>
- [25] Sujon, K. M.; Hassan, R.; Choi, K.; Al Samad, M. A. Accuracy, Precision, Recall, F1-Score, or MCC? Empirical Evidence from Advanced Statistics, ML, and XAI for Evaluating Business Predictive Models. *J. Big Data* **2025**, 12, 268. <https://doi.org/10.1186/s40537-025-01313-4>
- [26] Draelos, R. *Measuring Performance: AUC (AUROC)*; Glass Box Medicine, 2019.
- [27] Praveen, B. C.; Sivasankaran, K. Fast and Accurate Sentiment Analysis Using a Hybrid LLM Model with OpenVINO Optimization. *IEEE Access* **2026**, 14, 16176–16187. <https://doi.org/10.1109/ACCESS.2026.3658411>
- [28] Wilcoxon, F. Individual Comparisons by Ranking Methods. In *Breakthroughs in Statistics: Methodology and Distribution*; Springer: New York, NY, 1992; pp 196–202.
- [29] Demšar, J. Statistical Comparisons of Classifiers over Multiple Data Sets. *J. Mach. Learn. Res.* **2006**, 7, 1–30.