Available online at www.qu.edu.iq/journalcm
JOURNAL OF AL-QADISIYAH FOR COMPUTER SCIENCE AND MATHEMATICS
ISSN:2521-3504(online) ISSN:2074-0204(print)

A Novel Approach to Automated Detection of AI-Generated Text

Halah Mohammed Abbas

Computer science and information technology, Al-Qadisiyah University, Al-Qadisiyah, Iraq. E-mail: halahmohammedabass87@gmail.com

ARTICLE INFO

Article history:
Received: 4 /1/2025
Rrevised form: 10 /1/2025
Accepted : 13 /1/2025
Available online: 30 /3/2025

Keywords:

Machine-generated text detection

Zero-shot prompt

SBERT

Graph attention network

Graph convolutional network.

Each keyword to start on a new line

ABSTRACT

Detecting machine-generated text involves identifying whether text has been created by artificial intelligence models or written by humans. This task has become increasingly significant due to the potential misuse of AI-generated text for producing fake news, reviews, or spam that can mislead people. The aim of this study is to develop a model capable of determining if a tweet's author is human or a robot. To achieve this, we utilized a zero-shot prompt with a pre-trained model and fine-tuned SBERT using various transformer models. Additionally, we employed graph attention network and graph convolutional network models to analyze the author's writing style. The findings indicate that using the graph convolutional network model to extract writing style characteristics yields the highest accuracy, reaching 93.60%. Detecting machine-generated text is vital for preventing the abuse of AI models and ensuring the reliability of content on online platforms by effectively distinguishing between human and AI-generated text.

MSC..

https://doi.org/10.29304/jqcsm.2025.17.11958

1. Introduction

The rapid advancements in artificial intelligence have revolutionized natural language processing, leading to the development of highly sophisticated language models capable of producing human-like prose [1]. These models, powered by attention-based transformers, have overcome word count limitations and can now generate texts that closely resemble human writing [2]. These developments have led to an increasing trend of more complex language models, with both positive and negative consequences. Writing helpers and chatbots have developed alongside language models. These resources now provide far more assistance than just simple grammar and punctuation [3]. They may aid with content creation, offer contextual recommendations, and even support language translation. They have consequently developed into important tools for people and organizations trying to enhance their textual

∗Corresponding author

Email addresses:

Communicated by ‘sub etitor’

communication in multiple languages. Numerous industries, including journalism, creative writing, and customer service, have adopted these ideas [4].

While these advancements have brought numerous benefits, there are also concerns about potential misuse, including spam, phishing, social media overload, and fake news [5]. Professionals and academic writers have also been misusing these language models. Furthermore, the reliability, authenticity, and ethical considerations surrounding machine-generated writing have raised important concerns [6, 7]. Astroturfing, in particular, has emerged as a serious concern, where AI-based social media can be used to distort information and manipulate public opinion [8, 9].

State-of-the-art (SOTA) language models can be manipulated to evade detection by traditional anti-spam technologies, highlighting the need for enhanced detection tools and increased public awareness to mitigate the impact of malicious activities [2].

Building on this foundation, this research seeks to develop advanced techniques using zero-shot prompts and pre-trained SBERT models to detect machine-generated text. The proposed method aims to accurately identify whether a tweet's author is a human or a robot. We'll use techniques built on pre-trained models and zero-shot prompts and be able to identify if the input text was produced by a robot or by a human using these techniques. There have been four test phases carried out. This study examines user preferences and the impact of not utilizing them since user preferences can be useful in determining the author of a tweet. In the first two phases, we use the zero-shot prompt for this reason. Using the zero-shot prompt can expedite problem-solving because of its quick execution caused by the lack of training data. Moreover, we classify tweets in the third and fourth stages using the pre-trained model. Large amounts of data are used to train these models, which typically perform better and have a higher capacity for generalizing to new data. Additionally, it is feasible to leverage the experiences and information gathered in a variety of sectors and lessen the need to start from scratch by employing pre-trained models. In this study, we used various pre-trained models to fine-tune the SBERT model. In addition, the author's writing style can be a unique fingerprint that can help in identifying the author. By using Graph Attention network (GAT) and graph convolution network models, we can extract stylistic features from the text of the tweets. Incorporating these features into the model's training can significantly improve its accuracy in author recognition.

The remainder of this article is organized as follows: In Section 2, the studies that have looked at the identification of machine-generated text are examined. The proposed methodology used in this study will be explained in the 3 Section. The implementation is covered in detail in the 4 Section. The 5 Section concludes with a summary of the subjects covered and recommendations for facilitating further research.

2. Related works

AI-generated text detectors can be divided into two distinct groups including "Prepared" and "Post-hoc" detectors" based on possession and practical applicability. Watermarking and retrieval-based detectors are the two main methods used in prepared detection techniques. The model owner must actively participate in the text production process in order to use these strategies. Conversely, post-hoc detectors use fine-tuned classifiers or zero-shot detection methods that are accessible to other parties [10].

2.1 Prepared Detectors

2.1.1 Watermarking Methods

A well-known idea in the literature, watermarking has been applied extensively to hide data's contents. But in its early stages of execution, its distinct nature presented serious difficulties [11]. In the past, methods for inserting watermarks into already-written content included synonym substitution [12], synthetic structural restructuring [13], and paraphrasing [13]. However, with recent developments in neural language models [14], rule-based methods have been superseded by improved techniques that make use of mask-infilling models [15]. These days, end-to-end models are used for both the encoding and decoding phases of watermarking [16]. Large language

model-specific watermarks have drawn a lot of interest as a reliable way to recognize text produced by machines [11, 12, 17].

Watermark and Detect are the two probabilistic polynomial-time algorithms that make up a general watermarking method [17]. In order to encode a signal into the text that is produced, the Watermark algorithm is intended to take a language model L as input and modify the model's outputs. This makes it possible to determine whether a particular text sequence was produced by a particular language model or by another model. If the text sequence was produced by L^, the Detect method can produce a 1, and if it was produced by another model, it can produce a 0. Two branches including pseudo-random watermarks and biased-sampler watermarks comprise the approaches that employ watermarking techniques.

2.1.1.1 Biased-Sampler Watermarks

Biased-sampler watermarking is a method that changes the distribution of tokens to prioritize the selection of tokens from a particular category. This is done by utilizing a predetermined "green list" of tokens to impact the sampling process at each time interval [18].

A novel watermarking framework using a paraphrase-based lexical substitution mechanism is described in the Qiang et al study [19]. The system uses a pre-trained paraphraser to generate replacement candidates for each token in a sequence, then embeds them while maintaining the original meaning.

The Zhao et al [20] research paper presented GINSEW, a new technique to safeguard language generation models from model extraction attacks. The proposed method embedded an invisible sequence watermark that can be used to detect attempts at model extraction since the extracted model will carry the watermarked signal.

2.1.1.2 Pseudo-Random Watermarks

Pseudo-random watermarking schemes are designed with the objective of minimizing the discrepancy between the original distribution and the one after watermarking. This particular methodology is effective in ensuring that the watermark remains imperceptible and does not introduce any form of bias [21, 22].

In the Christ et al [22] study, the challenge of detecting AI-generated text is explored, and Pseudo-Random watermarking is utilized to embed invisible watermarks into language models. The authors presented a concept of undetectable watermarks inspired by cryptography, requiring a secret key for detection. Even when the model is probed with diverse prompts, these watermarks remain undetectable.

The Idrissi et al study [23] proposed a technique that employed a pseudo-random watermarking approach by adjusting the temperature parameter in text generation. This adjustment results in varying degrees of confidence in the model's predictions, forming a distinctive watermark for distinguishing machine-generated text.

Hou et al [24] introduced SEMSTAMP, an algorithm for sentence-level semantic watermarking utilizing locality-sensitive hashing (LSH) to partition the semantic space of sentence embeddings. This strategy aimed to counteract the vulnerability of token-level watermarking methods to paraphrase attacks by functioning within the semantic space.

2.1.2. Retrieval-based detectors

Retrieval-based detectors are commonly utilized to safeguard against paraphrasing attacks, as indicated by recent research. For this purpose, the retrieval-based detection strategy is implemented to mitigate the susceptibility of detection algorithms to paraphrasing [25, 26].

A new technique is known as CEDAR was introduced in the Nashid et al [27] study for creating prompts in few-shot learning tasks related to code. CEDAR automatically retrieved code examples that closely matched the developer's task through embedding or frequency analysis.

The research paper by Krishna et al [28] examined the problem of detecting AI-generated text and how current detection algorithms are susceptible to paraphrase attacks. The authors presented a new model called DIPPER, which can bypass several detection algorithms. In response to this vulnerability, they suggested a defense mechanism that relies on finding similar paraphrased text by using retrieval-based detectors to search a database of previously generated sequences.

2.2. Post-hoc Detectors

Post-hoc detection of AI-generated text after it has been created is a difficult task that can be done without prior preparation or collaboration with the model owner. While it is more challenging than prepared detection text, it has a wider range of applications. In situations where independent or uncooperative individuals use freely available language models, post-hoc detection is the only method for identifying AI-generated text [10].

2.2.1. Zero-shot Detection

Zero-shot text detection does not necessitate access to particular machine-generated or human-authored text samples. Instead, it is predicated on the notion that generic text sequences generated by a language model encompass discernible information that a detector can pinpoint. This identification can be accomplished utilizing a pre-existing language model that may differ from the original model, or via an entirely distinct statistical methodology. The fundamental idea is to recognize and highlight pertinent information within the text, even in the absence of prior exposure to specific instances [29, 30].

The research presented in Mitchell et al study [26], explored DetectGPT as a technique for identifying machine-generated text by analyzing the probability curvature, eliminating the necessity of training a distinct classifier. It capitalized on the observation text produced by extensive language models often resided in regions of negative curvature within the model's log probability function. DetectGPT exhibited enhanced efficacy in distinguishing fabricated news articles generated by GPT-NeoX. Through zero-shot detection, the approach scrutinized the log probabilities of the original text alongside perturbed samples to ascertain whether a passage originates from a particular language model.

The primary focus of the investigation by Bao et al [31] was the introduction of Fast-DetectGPT, a methodology designed to identify machine-generated text within Large Language Models (LLMs). It employed conditional probability curvature to pinpoint discrepancies in vocabulary selection between LLMs and content created by humans.

2.2.2. Methods based on Training and Fine-tuning of Classifiers

This method has a long-established background, with extensive research focused on refining classifiers for this purpose [32]. Detectors in this category do not require access to model parameters and can operate in completely black-box conditions. However, unlike the zero-shot setup, the detector's training requires supervised training samples in the form of text generated by both humans and machines. Despite not needing access to model parameters, detectors falling within this classification still require supervised training samples, which sets them apart from the zero-shot configuration [10].

The investigation conducted by Mohammadi et al [33] sought to assess the efficacy of machine-learning models in discriminating between human-authored and machine-generated text, aiming to pinpoint the most efficient algorithms for this particular task. In pursuit of this objective, the author delved into the differentiation between human-generated and machine-generated text in the Persian language. To realize the core concept delineated in the article, the researchers devised a Persian Machine-Generated paraphrase dataset by harnessing the capabilities of

the ChatGPT model to produce rephrased renditions of human-composed text. Various machine learning models, such as dense neural networks employing one-hot encoding, LSTM networks integrated with sentence transformers, and convolutional layers, were utilized to differentiate between human and machine-generated text.

The study presented by Xiong et al [34] introduced SemEval-2024 Task 8, which is designed to identify machine-generated texts originating from various LLMs. This task encompassed three distinct subtasks, namely Binary classification in monolingual and multilingual settings (Subtask A), Multi-class classification (Subtask B), and Mixed text detection (Subtask C). The examination primarily concentrated on Subtasks A and B within SemEval-2024 Task 8.

Other studies that have addressed this topic within the field of machine-generated text detection have been

Ref.	Year	Category	Dataset	Evaluation
[20]	2023	Biased-Sampler	The machine translation task: IWSLT14 and WMT14 The story generation task: ROCstories	improvement of 19 to 29 points in mean average precision (mAP)
[67]	2024	Pseudo-Random	OpenGen, C4 and Essays	95% match rate
[27]	2023	retrieval-based detectors	A dataset with 104,804 samples	In assertion generation: exact match accuracy: 75.79% plausible match accuracy: 77.40%
[28]	2023	retrieval-based detectors	PAR3	detect 80% to 97% of paraphrased generations
[26]	2023	Zero-shot Detection	XSum stories, SQuAD, Wikipedia contexts	DetectGPT has outperformed other criteria in detection accuracy.
[31]	2023	Zero-shot Detection	XSum, SQuAD, WritingPrompts	the acceleration in detection with a factor of 340.
[33]	2023	Training and Fine-tuning of Classifiers	COPER pn summary Digikala Comments	The best classifier is the model with CNN, LSTM, and dense layers.
[34]	2024	Training and Fine-tuning of Classifiers	CommonCrawl	The highest Dev score in Subtask A is 0.783 and in Subtask B is 0.735 for LoRA-RoBERTa and XLM-RoBERTa.

Table .Comparison of reviewed studies :1

3. Methodology

Figure 1 illustrates the proposed approach for determining whether the author of a tweet is human or a robot. A series of four experiments were undertaken for this purpose. Initially, data from the PAN 2019 dataset is obtained and subjected to pre-processing. Subsequently, the four experiments are conducted. The first and second experiments revolve around the zero-shot prompt concept and language models to ascertain if a tweet was authored by a human or a robot. In the third and fourth experiments, a pre-trained model is utilized for classification, focusing on extracting features from the text. During the model testing phase, the user is categorized using classification models such as logistics regression, multilayer Perceptron, or LSTM. The fourth experiment specifically considers the features of the author's writing style. Lastly, the performance of each experiment is evaluated using predefined criteria.

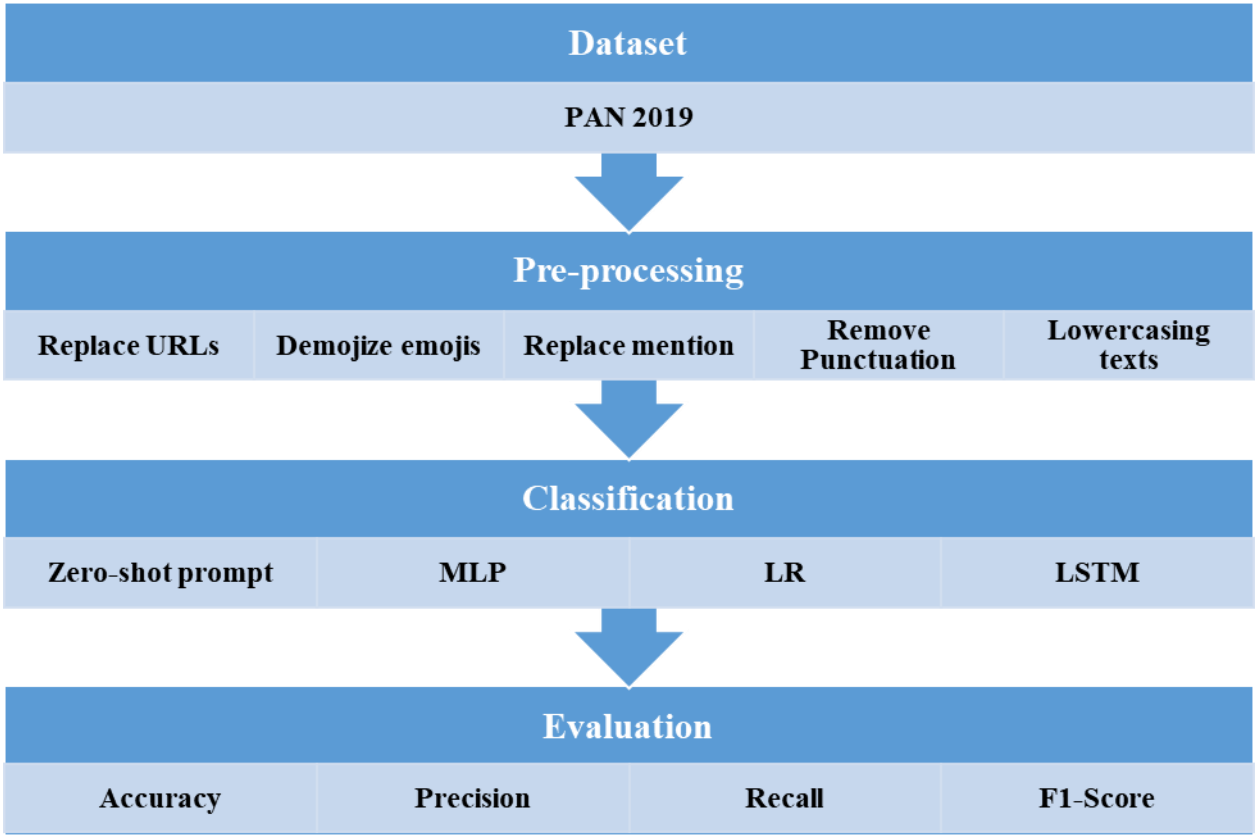

Figure 1: The proposed method.

3.1. Dataset

The PAN-AP-2019 dataset [35] comprises tweets gathered from the years 2013 to 2018. This dataset serves the purpose of distinguishing whether a tweet's author is a human or a robot. In the stage of implementation, the PAN data set underwent a division into a 60 to 40 percent ratio. A total of 6760 data points were utilized, with 4120 allocated for training and 2640 for testing purposes. The dataset is equilibrium in the corpus of bots and humans.

3.2. Pre-processing

The pre-processing involves the following steps:

3.2.1. Replacing URLs and user's mentions

In this step, URLs have been replaced with a general token like "LINK". It reduces computational complexity and overfitting, as well as addresses privacy and security concerns related to sensitive or confidential content [36]. Similarly, user mentions have been replaced with one general word. This replacement improves text data quality, reduces vocabulary size, preserves privacy, and enhances generalization. By using standardized tokens for URLs and user mentions, models can better generalize across diverse content types and prioritize textual context over specific hyperlinks [37].

3.2.2. Emoticon handling

The conversion of emojis to textual representations during the initial tweet analysis phase was executed. Emojis play a crucial role in conveying emotional information that may not be fully captured by textual content alone. They are frequently utilized in expressing sentiment, tone, and emotion in social media posts, and overlooking their significance could lead to inaccurate sentiment analysis outcomes [38].

3.2.3. Punctuations

Removing punctuation is the next preprocessing that was used in this study. Punctuation marks do not contribute to the informative content of the text and can inflate the vocabulary size, complicating text structure analysis. Eliminating punctuation aids in enhancing visual interpretation, particularly with extensive datasets. Furthermore, natural language processing algorithms struggle with accurate punctuation handling, and its removal can streamline algorithmic tasks and minimize errors [38, 39].

In the preprocessing phase, URLs were substituted with the term "LINK", while user mentions were replaced with "MENTION". Moreover, usernames were eliminated from the tweet content to safeguard user privacy. Additionally, emoticons were converted into textual representations. For instance, the emoji " " was transformed into the phrase "thumbs up" to ensure consistency. This conversion is essential as humans often employ emojis in context-specific manners, unlike bots. Consequently, translating emojis into text enables the algorithms to capture emoji-related insights as part of the author's writing style. Furthermore, extraneous punctuation and words exceeding a length of 20 characters were excluded. All text characters within the tweets also standardized to lowercase to enhance algorithmic text processing capabilities.

3.3 Used methods

The methods utilized in this investigation encompass Zero-shot prompting and classification with a pre-trained model for identifying Twitter users. Additionally, the author's writing style extraction involves employing the GCN and GAT. These methodologies will be further scrutinized in the subsequent sections.

3.3.1. Zero-shot prompting

Zero-shot prompting serves as a methodology that capitalizes on the pre-training data of a LLM to generate appropriate responses to prompts without specific examples. This technique enables the LLM to utilize its existing linguistic knowledge to deliver precise and contextually suitable outputs [40]. The LLMs employed in this phase consist of Falcon-7B, LLaMA-2-7B, Mistral-7B, Vicuna-7B, Llama-2-7B.

3.3.2. Sentence-BERT

Sentence-BERT (SBERT) is a technique that facilitates the creation of fixed-size sentence embeddings by leveraging a pre-trained BERT model. It employs a siamese architecture with two parallel BERT models and pooling mechanisms to acquire sentence embeddings [41]. The SBERT model generates fixed-length (768) sentence embeddings using models like all-MiniLM [42], mpnet [43], mpnet-multilingual [44], and T5-Large [45]. The SetFit–Efficient Few-shot Learning framework was utilized for fine-tuning sentences to the SBERT model [46].

3.3.3. Graph Attention Network

The GAT is a specialized neural network architecture tailored to process graph-structured data. By incorporating attention mechanisms, it can discern the importance of nodes and edges in a graph, enabling focused attention on pertinent information for specific tasks [47].

GATs can be effectively tailored to extract writing style features from Twitter text [48]. In this method, each tweet is represented as a node in a graph, with edges denoting various relationships such as user interactions, shared hashtags, or chronological order. The tweet content serves as the node features, while edges encode contextual information like replies, retweets, or shared hashtags [49]. GATs learn attention weights for each edge, enabling the model to prioritize the most relevant neighboring tweets during the message-passing process. This attention mechanism is essential for capturing the context and dependencies between tweets, which is vital for identifying writing style features [50]. GATs aggregate information from neighboring tweets and compute node representations by considering both the intrinsic features of each node and the attention-weighted features of its neighbors [51]. From these learned representations, distinctive writing style features emerge, highlighting patterns such as sentence length, punctuation usage, and specific vocabulary. These features are then extracted from the tweet content, offering a detailed understanding of the author's writing style.

3.3.4. Graph Convolutional Network

The GCN is an architecture designed for analyzing graph-based data, particularly for semi-supervised learning. GCNs, based on modified convolutional neural networks, can handle intricate graph structures. They learn through hidden layer representations that encompass local graph structures and node features, making them suitable for tasks like node classification and link prediction [52]. GCNs have applications in machine-generated text detection, especially in text classification, where they exploit the interconnectedness of words or documents to infer document labels by transforming text into a word graph and applying graph convolution operations [53].

In GCNs, the content of each tweet is used as node features [54]. Unlike GATs, GCNs utilize a fixed aggregation method where each node combines features from its neighboring nodes based on their adjacency [55]. This approach captures the structural context and dependencies between tweets. By stacking multiple GCN layers, the model can gather information from a broader context, improving its ability to capture writing style features [56]. GCNs compute node representations by averaging the features of neighboring nodes, weighted by the graph structure, which aids in extracting unique writing patterns from the tweet content [57].

3.4. Implementation

We conducted four experiments as previously mentioned. Experiments 1 and 2 entailed the utilization of the zero-shot prompt. Within these experiments, the model was provided with a prompt corresponding to Figure 2 to discern whether the input data originated from a human or a robot.

```
Classify the user tweets as 'human' or 'bot.'
### User Tweets:
{tweets}
### Answer:
```

Figure 2: prompt given to language models in the first and second experiment.

The first experiment incorporated the ALL-IN-ONE technique, involving the summation of 20% of a user's test data utilizing Gensim. This process resulted in favorable user preferences, particularly in the depiction of users through tweets. Subsequently, we fed this data simultaneously to language models to ascertain whether the user is categorized as a human or a robot.

In the second experiment, we employed the All-in-All technique, which encompassed considering 20% of all users' tweets. We designed this experiment to overlook user preferences and investigate the variations in outcomes. We gave each user's tweets one by one to the language model, and following the model's labeling of the user, majority voting was employed to determine the user's classification as a robot or human.

In experiments 3 and 4, we used a pre-trained model to categorize tweets. The tweet text was transformed into vectors to make it understandable for the model. In these tests, sentences with a fixed length needed to be embedded. Therefore, sentence transformer models such as all-MiniLM, MPNet, MPNet-multilingual, and T5 were utilized. The SBERT model was fine-tuned using the training data through the SetFit-Efficient Few-shot Learning framework. This framework converts the meaning of sentences into sentence embeddings. Consequently, we employed the SetFit-Efficient Few-shot Learning framework to fine-tune SBERT, with sentences being fine-tuned to the model in pairs. In addition, we applied logistic regression and Multi-Layer Perceptron (MLP) on top of the sentence-transformers for the experiments. A batch size of 16, a language model fine-tuning epoch number of 5, and a classifier epoch number of 15 were employed across experiments 3 and 4. In the third experiment, the SBERT output was classified on the test data using MLP and logistic regression models.

Our goal for the fourth experiment is to identify and analyze the distinctive aspects of the author's writing style. LMs have demonstrated potential in quickly learning to identify bots. However, using LMs to analyze an author's writing style, which involves understanding subtle and unique writing styles, can be challenging. Author stylistic information includes factors such as tone, vocabulary, and linguistic patterns that make an author's work distinctive. While LMs are effective at capturing general language patterns and context, accurately capturing an individual's writing style may require more data and fine-tuning. This study suggests that an author's word usage is a distinct characteristic that conveys meaning when connected with other words. These connections can also link bot-generated tweets in a graph, even if the user is human. Therefore, these connections provide valuable information for representing both bots and humans. Based on this, we represent an author's tweets as graph embeddings to understand their linguistic behavior and establish graph embeddings as a way to represent dependency information through nodes (words, authors) and edges (connections between words).

In the fourth experiment, the characteristics of the author's writing style were derived using GAT and GCN. Both GATs and GCNs can be adapted to extract writing style features from Twitter text. In this approach, each tweet is represented as a node in a graph, with edges denoting various relationships between tweets, such as user interactions, shared hashtags, or temporal order [58]. These features were then classified using a LSTM network, alongside features extracted by the SBERT model.

4. Results

The outcomes of the test set are displayed in Table 2 for the first experiment. Table 5-4 illustrates the complexity of bot detection in social media, with LLama-2-7B emerging as the top-performing model boasting an accuracy of 53.33%, and Falcon-7B achieving an F1-score of 48.83%.

Table 1: The results of first experiment.

Model	Accuracy	Precision	Recall	F1-Score
Mistral-7B + all-in-one	48.93	47.53	48.93	40.46
LLama-2-7B + all-in-one	**53.33**	**66.82**	**53.33**	**41.63**

Falcon-7B + all-in-one	48.82	48.77	48.82	48.83
Vicuna-7B + all-in-one	50.26	75.06	50.26	33.91

The outcomes of the second experiment are presented in Table 3. This particular study aimed to assess the impact of disregarding user preferences on progress. The results from Falcon-7B and LLaMA-2-7B experiments, the most effective models from the comprehensive experiments, reveal that neglecting user preferences or semantics could result in substantial information loss.

These discoveries offer insights into the examination of user stylistic patterns in social media posts. Therefore, the research indicates that utilizing LLMs for few-shot learning may not be optimal due to limitations in input capacity and the vast variability in tweet components. This often leads to surpassing the input length constraints of LLMs. Furthermore, the obstacles linked to extended input length and tweet variability necessitate a more extensive dataset to enable LLMs to display characteristics akin to few-shot learners. Nevertheless, the current input cap impedes our ability to effectively meet this necessity.

It is crucial to emphasize that LLMs are categorized as few-shot learners; however, the dilemma arises from the absence of a distinct structure in tweets typically seen in few-shot scenarios. Moreover, the complexity of the task is compounded by the sporadic presence of bots generating human-like tweets, thereby intensifying the challenge faced by LLMs in managing such diverse behaviors.

Table 2: The results of second experiment.

Model	Accuracy	Precision	Recall	F1-Score
Falcon-7B + all-in-all	**45.83**	**45.30**	**45.83**	**44.27**
Llama-2-7B + all-in-all	50.00	25.00	50.00	33.33

The outcomes of the third experiment, as depicted in Table 4, entail varied setups of models and refinement methods utilizing SBERT with diverse pre-trained language models and additional elements like LR or MLP. The objective is to execute classification assignments with different quantities of labeled data (100 or 400 instances) for training. The assessment metrics are outlined for each trial, illustrating the models' efficacy.

The remarkable accuracy of 92.46% attained with T5-Large, notably in conjunction with the MLP classifier, is significant. Moreover, the revelation that these outcomes were achieved using a small portion of the training set—below 3% of the data—implies that LMs serve as effective few-shot learners for both bot and human identification tasks. The success of these experiments signifies that LMs can adjust adeptly to the nuances of distinguishing between bot and human conduct with minimal adjustments. This efficiency is especially beneficial as it suggests that considerable performance enhancements can be achieved with a relatively limited amount of labeled data in the fine-tuning phase. Overall, these discoveries underscore the promise of LMs for efficient few-shot learning in the realm of bot and human identification.

Table 3: The results of third experiment.

Model	Accuracy	Precision	Recall	F1-Score
SBERT (all-MiniLM) + FewShot finetuning (100)	88.16	88.60	88.10	88.06
SBERT (all-MiniLM) + FewShot finetuning (100) + LR	88.18	88.59	88.18	88.15
SBERT (all-MiniLM) + FewShot finetuning (100) + MLP	88.21	88.68	88.21	88.18
SBERT (mpnet) + FewShot finetuning (100)	89.84	89.87	89.84	89.84
SBERT (mpnet) + FewShot finetuning (100) + LR	90.90	90.95	90.90	90.90
SBERT (mpnet) + FewShot finetuning (100) + MLP	91.89	91.93	91.89	91.89
SBERT (mpnet-multilingual) + FewShot finetuning (100)	86.40	86.88	86.40	86.35
SBERT (mpnet-multilingual) + FewShot finetuning (100) + LR	86.89	87.27	86.89	86.86
SBERT (mpnet-multilingual) + FewShot finetuning (100) + MLP	86.74	87.03	86.74	86.71
SBERT (T5-Large) + FewShot finetuning (100)	89.31	90.01	89.31	89.27
SBERT (T5-Large) + FewShot finetuning (100) + LR	89.35	89.93	89.35	89.31
SBERT (T5-Large) + FewShot finetuning (100) + MLP	**92.46**	**92.52**	**92.46**	**92.45**
SBERT (mpnet) + FewShot finetuning (400)	91.36	91.39	91.36	91.36
SBERT (mpnet) + FewShot finetuning (400) + LR	91.15	91.15	91.15	91.15
SBERT (mpnet) + FewShot finetuning (400) + MLP	91.66	91.17	91.66	91.66

In the fourth experiment, both the GAT and GCN models were experimented with. According to these experiments and the data showcased in Table 5, the models underwent training and assessment with similar hyperparameters

and identical datasets. In scenarios involving classification of brief texts, RNNs, particularly LSTM, exhibit commendable performance. Nevertheless, complications arise when handling extensive texts. The representation of such texts poses challenges and extracting sequential information appropriately becomes a hurdle for RNNs. Word sequence holds significance in tasks like sentiment analysis, hence RNN models are apt for such undertakings. Nevertheless, for this particular task, where word co-occurrences along with semantics play a crucial role, GCNs emerge as a fitting choice. Each writer may have distinct word preferences, and by capturing these inclinations, we may uncover favored word selections for distinct categories. GCNs enable the incorporation of word co-occurrences and inter-word relationships through edge weights, alongside word semantics represented as nodes. Consequently, GCNs disregard word order and prioritize word usage with semantics, rendering them well-suited for the task at hand. The employment of GCNs atop few-shot learners led to a heightened accuracy of 93.60% in bot identification tasks, as evidenced.

Table 4:The results of fourth experiment.

Model	Accuracy	Precision	Recall	F1-Score
SBERT (T5-Large) + FewShot finetuning (100) + prep + GCN	**93.60**	**93.67**	**93.60**	**93.60**
SBERT (T5-Large) + FewShot finetuning (100) + prep + GAT	92.72	92.80	92.72	92.72

According to the obtained results, the best model considering the user's preferences is the LLama-2-7B model with an accuracy of 53.33%. Also, when the features of the author's writing style were extracted with the GCN model, the highest accuracy of 93.60% was obtained.

The study of language models with stylistic information leads us to interesting findings about authors' fingerprints while choosing words. As a result of this, we obtained a rank of 5 out of 56 participants in the PAN 2019 shared task. And defeated all the baselines and some benchmark models on this task by only applying stylistic and semantic information. As a result of this, we defeated most of the works that already stood out in the task namely [59-66]. A particular work [63] utilized multi-layer CNN layers for feature extraction and multi-dense layers as a classifier. It was the only deep learning model that stood out with an accuracy of 91.82% and appeared among 10% of the best performers.

5. Discussion

The study of LMs incorporating stylistic information has led to intriguing insights into authors' unique word choices. As a result, our approach ranked 5th out of 56 participants in the PAN 2019 shared task, outperforming all baselines and several benchmark models by solely applying stylistic and semantic information. This achievement highlights the effectiveness of our method compared to many prominent works in the task. Jimenez et al. [59] aimed to identify the authors of tweets from the PAN 2019 dataset, achieving an accuracy of 91% on the English test dataset for robot/human classification. Mahmood et al. [60] utilized the TF-IDF method combined with a support vector machine (SVM) classifier, also reaching 91% accuracy. Vogel et al. [61] employed unigrams, bigrams, and n-gram characters as features, using an SVM classifier to categorize tweets, and achieved an overall accuracy of 92%. In the study by Polignano et al. [63], GloVe and FastText methods were used for word embedding, with a 2D CNN classifier determining whether the tweet's author was human or a robot. Puertas et al. [64] applied machine learning methods to categorize tweet authors, using the N-gram method for feature extraction and achieving 91.2% accuracy. Table 6 compares these studies in detail.

Table 5: Comparison of other studies with the proposed method.

Best Accuracy	Classifier	Feature extraction methods	Ref.
91 %	SVM	• document frequency • Frequency co-occurring entropy • Information gain	[59]
91 %	SVM	• TF-IDF	[60]
92 %	SVM	• Unigrams • Bigrams	[61]
91.82 %	2D CNN	• FastText • GloVe	[63]
91.2 %	• Naive Bayes • Gaussian Naive Bayes • Complement Naive Bayes • Logistic Regression • Random Forest	N-gram (2,3,4)	[64]
93.60	• **LSTM**	**SBERT(T5-Large) + GCN**	**Ours**

6. Conclusion and future works

Machine-generated text has the potential to improve human interaction with written content in various ways. It can be used for grammar correction, machine translation, writing assistance for general text and programming languages, and even poetry creation. However, challenges arise when dealing with problematic content such as misinformation and spam. Therefore, it's important to be able to distinguish between human-written and AI-generated text. Simple and easily understandable content is less likely to be perceived as artificial. Key concerns include authorship attribution, authenticity, and deviation from reality. To address these challenges, a straightforward classifier based on effective features is needed to differentiate between human and machine-generated text. The research aims to provide a method to determine whether the author of a tweet is a human or a robot using the zero-shot prompt method and pre-trained language models. The study shows that Language Models with stylistic information are effective in coping with social media tasks, especially when used in combination with GNNs. Improving machine-generated text detection can enhance content credibility on social media, helping to identify and flag misleading or false information and ultimately creating a more trustworthy and reliable social media environment. This can lead to a more informed and educated online community, promoting critical thinking and responsible sharing of information. Enhancing machine-generated text detection can significantly boost content credibility on social media. It can aid in identifying and flagging misleading or false information, thereby curbing the spread of misinformation. This improvement can foster a more trustworthy and reliable social media environment, where users can have greater confidence in the content they encounter. Furthermore, it can contribute to a more informed and educated online community, encouraging critical thinking and responsible information sharing.

6.1. Future work

In the following, several important questions and research directions related to machine-generated text recognition that have not yet been resolved are discussed.

- Regulating large language models in order to prevent abuse is a global concern. These models can be misused by spreading false information or impersonating people. A possible solution is to create a regulatory body to mark all public language models and implement a detection mechanism for protection.
- Most of the studies conducted in the field of machine-generated text detection demonstrate the performance of their models by showing improved accuracy in detecting machine-generated text from human writing. However, the dataset used may not accurately represent the diversity of human writing. For example, the fluency of a text written by language experts or native speakers is very different from a text written by non-native writers. It is important to consider these nuances when evaluating text detectors.
- Biases in text detectors can have far-reaching consequences, particularly in educational settings where they may wrongly accuse marginalized groups of plagiarism. It is crucial to address and minimize bias in text detectors, whether through perplexity scores or classifiers, to ensure fairness.
- It is evident that repeatedly applying paraphrasing attacks can significantly reduce the effectiveness of state-of-the-art text detectors. Moreover, using an iterative randomized paraphraser may eventually eliminate all traces of the original text, including watermarks, but it inevitably alters the original text's meaning. Hence, there is a balance between the intensity of the paraphrase attack and how much the text's meaning alters. Conversely, generative attacks have a bijective nature and can potentially bypass the detector without causing a significant change in text quality and meaning. Given these insights, the challenge of designing a more robust detector or watermarking scheme against potential attacks remains an open issue.
- In the present study, Twitter data was used to experiment. Data from other social networks can also be used in future research.
- In this study, Twitter data was used for experimentation. Future research can explore using data from other social networks as well.

References

[1] R. Imamguluyev, "The Rise of GPT-3: Implications for Natural Language Processing and Beyond," *International Journal of Research Publication and Reviews*, vol. 4, pp. 4893-4903, 03/03 2023, doi: 10.55248/gengpi.2023.4.33987.

[2] L. Fröhling and A. Zubiaga, "Feature-based detection of automated language models: tackling GPT-2, GPT-3 and Grover," *PeerJ Computer Science*, vol. 7, p. e443, 2021.

[3] E. Sadikoğlu, M. Gök, M. Mijwil, and I. Kosesoy, "The Evolution and Impact of Large Language Model Chatbots in Social Media: A Comprehensive Review of Past, Present, and Future Applications," vol. 6, pp. 67-76, 12/21 2023.

[4] D. Valiaiev, "Detection of Machine-Generated Text: Literature Survey," *arXiv preprint arXiv:2402.01642*, 2024.

[5] L. Dugan, D. Ippolito, A. Kirubarajan, and C. Callison-Burch, "RoFT: A tool for evaluating human detection of machine-generated text," *arXiv preprint arXiv:2010.03070*, 2020.

[6] A. Das and R. M. Verma, "Can machines tell stories? a comparative study of deep neural language models and metrics," *IEEE Access*, vol. 8, pp. 181258-181292, 2020.

[7] I. Dergaa, K. Chamari, P. Zmijewski, and H. Ben Saad, "From Human Writing to Artificial Intelligence Generated Text: Examining the Prospects and potential threats of ChatGPT in Academic Writing," *Biology of Sport*, vol. 40, pp. 615-622, 03/07 2023, doi: 10.5114/biolsport.2023.125623.

[8] A. Rauchfleisch, M. Sele, and C. Caspar, "Digital astroturfing in politics: Definition, typology, and countermeasures," *Studies in Communication Sciences*, vol. 18, pp. 69-85, 11/14 2018, doi: 10.24434/j.scoms.2018.01.005.

[9] J. Peng, R. K. K. Choo, and H. Ashman, "Astroturfing Detection in Social Media: Using Binary n-Gram Analysis for Authorship Attribution," in *2016 IEEE Trustcom/BigDataSE/ISPA*, 23-26 Aug. 2016 2016, pp. 121-128, doi: 10.1109/TrustCom.2016.0054.

[10] S. S. Ghosal, S. Chakraborty, J. Geiping, F. Huang, D. Manocha, and A. Bedi, "A Survey on the Possibilities & Impossibilities of AI-generated Text Detection," *Transactions on Machine Learning Research*, 2023.

[11] S. Katzenbeisser and F. Petitcolas, *Information hiding*. Artech house, 2016.

[12] U. Topkara, M. Topkara, and M. J. Atallah, "The hiding virtues of ambiguity: quantifiably resilient watermarking of natural language text through synonym substitutions," in *Proceedings of the 8th workshop on Multimedia and security*, 2006, pp. 164-174.

[13] M. J. Atallah *et al.*, "Natural language watermarking: Design, analysis, and a proof-of-concept implementation," in *Information Hiding: 4th International Workshop, IH 2001 Pittsburgh, PA, USA, April 25–27, 2001 Proceedings 4*, 2001: Springer, pp. 185-200.

[14] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[15] H. Ueoka, Y. Murawaki, and S. Kurohashi, "Frustratingly easy edit-based linguistic steganography with a masked language model," *arXiv preprint arXiv:2104.09833*, 2021.

[16] Z. M. Ziegler, Y. Deng, and A. M. Rush, "Neural linguistic steganography," *arXiv preprint arXiv:1909.01496*, 2019.

[17] X. Zhao, P. Ananth, L. Li, and Y.-X. Wang, "Provable robust watermarking for ai-generated text," *arXiv preprint arXiv:2306.17439*, 2023.

[18] S. S. Ghosal, S. Chakraborty, J. Geiping, F. Huang, D. Manocha, and A. S. Bedi, "Towards possibilities & impossibilities of ai-generated text detection: A survey," *arXiv preprint arXiv:2310.15264*, 2023.

[19] J. Qiang, S. Zhu, Y. Li, Y. Zhu, Y. Yuan, and X. Wu, "Natural language watermarking via paraphraser-based lexical substitution," *Artificial Intelligence*, vol. 317, p. 103859, 2023/04/01/ 2023, doi: https://doi.org/10.1016/j.artint.2023.103859.

[20] X. Zhao, Y.-X. Wang, and L. Li, "Protecting language generation models via invisible watermarking," *arXiv preprint arXiv:2302.03162*, 2023.

[21] R. Kuditipudi, J. Thickstun, T. Hashimoto, and P. Liang, "Robust distortion-free watermarks for language models," *arXiv preprint arXiv:2307.15593*, 2023.

[22] M. Christ, S. Gunn, and O. Zamir, "Undetectable Watermarks for Language Models," *arXiv preprint arXiv:2306.09194*, 2023.

[23] B. Y. Idrissi, M. Millunzi, A. Sorrenti, L. Baraldi, and D. Dementieva, "Temperature Matters: Enhancing Watermark Robustness Against Paraphrasing Attacks," 2023.

[24] A. B. Hou *et al.*, "Semstamp: A semantic watermark with paraphrastic robustness for text generation," *arXiv preprint arXiv:2310.03991*, 2023.

[25] J. Kirchenbauer, J. Geiping, Y. Wen, J. Katz, I. Miers, and T. Goldstein, "A watermark for large language models," *arXiv preprint arXiv:2301.10226*, 2023.

[26] E. Mitchell, Y. Lee, A. Khazatsky, C. D. Manning, and C. Finn, "Detectgpt: Zero-shot machine-generated text detection using probability curvature," *arXiv preprint arXiv:2301.11305*, 2023.

[27] N. Nashid, M. Sintaha, and A. Mesbah, "Retrieval-based prompt selection for code-related few-shot learning," in *Proceedings of the 45th International Conference on Software Engineering (ICSE'23)*, 2023.

[28] K. Krishna, Y. Song, M. Karpinska, J. Wieting, and M. Iyyer, "Paraphrasing evades detectors of AI-generated text, but retrieval is an effective defense. arXiv," *arXiv preprint arXiv:2303.13408*, 2023.

[29] M. H. I. Abdalla, S. Malberg, D. Dementieva, E. Mosca, and G. Groh, "A Benchmark Dataset to Distinguish Human-Written and Machine-Generated Scientific Papers," *Information*, vol. 14, no. 10, p. 522, 2023. [Online]. Available: https://www.mdpi.com/2078-2489/14/10/522.

[30] D. Ippolito, D. Duckworth, C. Callison-Burch, and D. Eck, *Automatic Detection of Generated Text is Easiest when Humans are Fooled*. 2020, pp. 1808-1822.

[31] G. Bao, Y. Zhao, Z. Teng, L. Yang, and Y. Zhang, "Fast-detectgpt: Efficient zero-shot detection of machine-generated text via conditional probability curvature," *arXiv preprint arXiv:2310.05130*, 2023.

[32] R. Zellers *et al.*, "Defending against neural fake news," *Advances in neural information processing systems*, vol. 32, 2019.

[33] K. Mohammadi, "Human vs machine generated text detection in Persian," ed, 2023.

[34] F. Xiong, T. Markchom, Z. Zheng, S. Jung, V. Ojha, and H. Liang, "Fine-tuning Large Language Models for Multigenerator, Multidomain, and Multilingual Machine-Generated Text Detection," *arXiv preprint arXiv:2401.12326*, 2024.

[35] F. Rangel and P. Rosso, "Overview of the 7th author profiling task at PAN 2019: bots and gender profiling in twitter," *Working notes papers of the CLEF 2019 evaluation labs*, vol. 2380, pp. 1-7, 2019.

[36] E. Psomakelis, K. Tserpes, D. Anagnostopoulos, and T. Varvarigou, "Comparing methods for twitter sentiment analysis," *arXiv preprint arXiv:1505.02973*, 2015.

[37] D. Effrosynidis, S. Symeonidis, and A. Arampatzis, *A Comparison of Pre-processing Techniques for Twitter Sentiment Analysis*. 2017.

[38] M. Siino, I. Tinnirello, and M. La Cascia, "Is text preprocessing still worth the time? A comparative survey on the influence of popular preprocessing methods on Transformers and traditional classifiers," *Information Systems*, vol. 121, p. 102342, 2024/03/01/ 2024, doi: https://doi.org/10.1016/j.is.2023.102342.

[39] S. Vasudevan, "Enhancing the Sentiment Classification Accuracy of Twitter Data using Machine Learning Algorithms," 2021, pp. 189-199.

[40] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, "Large language models are zero-shot reasoners," *Advances in neural information processing systems*, vol. 35, pp. 22199-22213, 2022.

[41] H. Cheng, S. Liu, W. Sun, and Q. Sun, "A Neural Topic Modeling Study Integrating SBERT and Data Augmentation," *Applied Sciences*, vol. 13, no. 7, p. 4595, 2023. [Online]. Available: https://www.mdpi.com/2076-3417/13/7/4595.

[42] H. Face, "all-MiniLM ". [Online]. Available: https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2.

[43] H. Face, "mpnet " 2022. [Online]. Available: https://huggingface.co/sentence-transformers/all-mpnet-base-v2.

[44] N. a. G. Reimers, Iryna, "paraphrase-multilingual-mpnet-base-v2," 2019. [Online]. Available: https://huggingface.co/sentence-transformers/paraphrase-multilingual-mpnet-base-v2.

[45] "sentence-transformers/gtr-t5-large," 2021. [Online]. Available: https://huggingface.co/sentence-transformers/gtr-t5-large.

[46] L. Tunstall *et al.*, "Efficient few-shot learning without prompts," *arXiv preprint arXiv:2209.11055*, 2022.

[47] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[48] Y. Yang, Z. Wu, Y. Yang, S. Lian, F. Guo, and Z. Wang, "A survey of information extraction based on deep learning," *Applied Sciences*, vol. 12, no. 19, p. 9691, 2022.

[49] B. Jiang, L. Wang, J. Tang, and B. Luo, "Context-Aware Graph Attention Networks," *arXiv preprint arXiv:1910.01736*, 2019.

[50] J. Chen and H. Chen, "Edge-featured graph attention network," *arXiv preprint arXiv:2101.07671*, 2021.

[51] E. Alothali, M. Salih, K. Hayawi, and H. Alashwal, "Bot-mgat: A transfer learning model based on a multi-view graph attention network to detect social bots," *Applied Sciences*, vol. 12, no. 16, p. 8117, 2022.

[52] S. Zhang, H. Tong, J. Xu, and R. Maciejewski, "Graph convolutional networks: a comprehensive review," *Computational Social Networks*, vol. 6, no. 1, pp. 1-23, 2019.

[53] S. Zhang, C. Zhou, Y. Li, X. Zhang, L. Ye, and Y. Wei, "Irregular Scene Text Detection Based on a Graph Convolutional Network," *Sensors*, vol. 23, no. 3, p. 1070, 2023. [Online]. Available: https://www.mdpi.com/1424-8220/23/3/1070.

[54] N. R. Aljohani, A. Fayoumi, and S.-U. Hassan, "Bot prediction on social networks of Twitter in altmetrics using deep graph convolutional networks," *Soft Computing*, vol. 24, no. 15, pp. 11109-11120, 2020.

[55] L. Zhang, H. Song, N. Aletras, and H. Lu, "Node-Feature Convolution for Graph Convolutional Networks," *Pattern Recognition*, vol. 128, p. 108661, 2022/08/01/ 2022, doi: https://doi.org/10.1016/j.patcog.2022.108661.

[56] X. Zhu, L. Zhu, J. Guo, S. Liang, and S. Dietze, "GL-GCN: Global and local dependency guided graph convolutional networks for aspect-based sentiment classification," *Expert Systems with Applications*, vol. 186, p. 115712, 2021.

[57] L. Zhang and H. Lu, "A feature-importance-aware and robust aggregator for GCN," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020, pp. 1813-1822.

[58] H. B. Giglou, M. Rahgouy, A. Rahmati, T. Rahgooy, and C. D. Seals, "Profiling Irony and Stereotype Spreaders with Encoding Dependency Information using Graph Convolutional Network," in *CLEF*, 2022, pp. 1613-0073.

[59] V. Jimenez-Villar, J. Sánchez-Junquera, M. Montes-y-Gómez, L. Villaseñor-Pineda, and S. P. Ponzetto, "Bots and gender profiling using masking techniques: Notebook for PAN at CLEF 2019," in *CEUR Workshop Proceedings*, 2019, vol. 2380: RWTH Aachen, pp. 1-8.

[60] A. Mahmood and P. Srinivasan, "Twitter Bots and Gender Detection using Tf-idf," in *CLEF (Working Notes)*, 2019.

[61] I. Vogel and P. Jiang, "Bot and Gender Identification in Twitter using Word and Character N-Grams," in *CLEF (Working Notes)*, 2019.

[62] R. Goubin, D. Lefeuvre, A. Alhamzeh, J. Mitrovic, E. Egyed-Zsigmond, and L. G. Fossi, "Bots and Gender Profiling using a Multi-layer Architecture," in *CLEF (Working Notes)*, 2019.

[63] M. Polignano, M. G. de Pinto, P. Lops, and G. Semeraro, "Identification Of Bot Accounts In Twitter Using 2D CNNs On User-generated Contents," in *Clef (working notes)*, 2019.

[64] E. Puertas, L. G. Moreno-Sandoval, F. M. Plaza-del Arco, J. A. Alvarado-Valencia, A. Pomares-Quimbaya, and L. Alfonso, "Bots and gender profiling on twitter using sociolinguistic features," *CLEF (Working Notes)*, pp. 1-8, 2019.

[65] D. Kosmajac and V. Keselj, "Twitter User Profiling: Bot and Gender Identification: Notebook for PAN at CLEF 2019," in *Experimental IR Meets Multilinguality, Multimodality, and Interaction: 11th International Conference of the CLEF Association, CLEF 2020, Thessaloniki, Greece, September 22–25, 2020, Proceedings 11*, 2020: Springer, pp. 141-153.

[66] T. Fagni and M. Tesconi, "Profiling Twitter Users Using Autogenerated Features Invariant to Data

[67] W. Qu *et al.*, "Provably Robust Multi-bit Watermarking for AI-generated Text via Error Correction Code," *arXiv preprint arXiv:2401.16820*, 2024.