DETECTING FAKE ACCOUNTS THROUGH GENERATIVE ADVERSARIAL NETWORK IN ONLINE SOCIAL MEDIA

Jinus Bordbar
Department of Computer ,
Islamic Azad University,
Shiraz Branch,
Shiraz,
Iran
Jinus.Bordbar@gmail.com

Mohammadreza Mohammadrezaei *
Department of Computer ,
Islamic Azad University,
Ramhormoz Branch,
Ramhormoz,
Iran
Mohammadrezaei.m.reza@gmail.com

Saman Ardalan
Institute for Medical Informatics and Statistics,
University of Kiel,
Kiel,
Germany
Ardalan@medinfo.uni-kiel.de

Mohammad Ebrahim Shiri
Department of Mathematics and Computer Science,
Amirkabir University of Technology,
Tehran,
Iran
Shiri@aut.ac.ir

ABSTRACT

Online social media is integral to human life, facilitating messaging, information sharing, and confidential communication while preserving privacy. Platforms like Twitter, Instagram, and Facebook exemplify this phenomenon. However, users face challenges due to network anomalies, often stemming from malicious activities such as identity theft for financial gain or harm. This paper proposes a novel method using user similarity measures and the Generative Adversarial Network (GAN) algorithm to identify anomalies (fake nodes) in user accounts in a large-scale social network while handling imbalanced data issues. Despite the problem's complexity, the method achieves an AUC rate of 80% in classifying and detecting fake accounts. Notably, the study builds on previous research, highlighting advancements and insights into the evolving landscape of anomaly detection in online social networks. The findings of this study contribute to ongoing advancements in fake account detection, offering a hopeful solution for securing online spaces against fraudulent activities and anomaly detection in social networks.

***K*eywords** Online Social Networks · Generative Adversarial Networks · Machine Learning · Anomaly Detection · Imbalanced Data · Deep Learning

1 Introduction

The term social media refers to computer-based technology that facilitates the sharing of ideas, thoughts, and information through networks and virtual communities [5]. Social media relies on the internet to enable people's communications and provides users with the ability to quickly convey content. Users can engage with social media through computers, tablets, smartphones, web-based software, or applications. As of October 2021,the use of social media has become widespread, with more than 3.8 billion users utilizing this environment [39]. However, the widespread usage of social media has also led to its improper use by some profit-seeking and hostile individuals [12]. They attempt to violate the privacy of real users by creating fake accounts that do not belong to ordinary individuals. Fake accounts can be created for humorous reasons, scams, or spreading fake news and false information, causing disruptions in people's normal lives. To prevent such incidents, researchers have developed several approaches to detect fake accounts [27].

*Corresponding author: Mohammadrezaei.m.reza@gmail.com

Most methods and approaches used to identify fake accounts have been either user-level or graph-based [37, 26]. The difference between these two approaches lies in the activities that occur within a node, versus those that occur in a graph-based method, which considers the activities and connections a node has with other nodes. In both methods, a classifier is trained on data to identify fake users.[38] However, these procedures may not be effective in high-level attacks, such as when a fake user account invades privacy or fakes someone's identification using a social engineering attack. In some recent studies, a small number of fake users were incorrectly identified as normal due to the limited number of fake nodes in the dataset, leading researchers to question the validity of the learning process. To address this issue, some studies have used resampling and the Synthetic Minority Over-Sampling Technique (SMOTE) method, which has a high error rate based on their performance [24].

The proposed method integrates user-level and graph-based techniques to enhance the effectiveness of detecting fake accounts on social networks in three ways: (1) the method uses similarity criteria to identify connections between accounts that are more likely to be real, therefore using those connections to help identify fake accounts[2]. Essentially, the method tries to learn about information on fake connections by repetitive training to better identify these accounts that seem to be similar to real connections. (2) The data was formatted as a table in the Excel application, with 1,000,000 data points separated by predefined methods in Excel. Feature extraction methods such as PCA (Principal Component Analysis) were employed to address the high dispersion of crucial features in a matrix, thereby reducing the number of variables. This is important because with many variables, the model can become overfit to the data, which means it may work well on the training data but not generalize well to test data or new data. (3) The method uses deep learning algorithms (e.g., GAN) to handle large amounts of data. Deep learning is a powerful technique that can help to identify complex patterns in data, which can be particularly useful when dealing with large and complex datasets like social networks[35]. By using deep learning, the method can better identify fake accounts, even when they are very similar to real ones. In this proposed method, the social network was a graph mapped into a matrix, where each node represented a user and edges expressed connections between users. Activities of accounts and connections with other nodes were learned through criteria and measures such as common friends, total friends, Jaccard measure, cosine measure, and other criteria of a node. The Principal Component Analysis (PCA) algorithm was used for feature extraction. The GAN algorithm checked connections between nodes, and the detection of fake user accounts was approved with high accuracy. Details of the dataset used are presented in the Evaluation section in table 2.

The remaining sections of this paper are arranged as follows: The related work is presented in Section 2. Concepts are discussed in Section 3. Section 4 provides a description of the methodology of the study. The evaluation and performance results on the dataset are described in Section 5. Section 6 presents the conclusions and future work.

2 Related Work

Many studies have looked into the problem of spotting fake accounts in online social networks. They have used different methods, including graph-based techniques, user-profile analysis, and machine learning. This section highlights the most relevant approaches and their limitations. It aims to position our proposed method within the current research landscape.

Yousefi et al. [42] developed a method for detecting profile cloning on social media by mapping the social network into a graph and calculating the resemblance between selected profiles and normal profiles. They also presented a user-based detection method that extracts information from profiles to search for similar profiles and classifies suspicious identities as fake if they meet certain criteria. Two detection methods were proposed and evaluated using an offline dataset of Facebook users and their attributes. A set of fake identities was assumed and added to the dataset for testing purposes. Results show that the new approach outperforms previous approaches in detecting fake profiles accurately, with a higher true positive rate and lower false positive rate.

Najari et al. [25] shows the study "Adversarial Botometer" introduces an adversarial framework for social bot detection, focusing on the interaction between bots and bot detectors. Unlike traditional models, which rely on static datasets, this paper models the detection process as a dynamic game between a bot and its detector. By using Generative Adversarial Networks (GAN), the study simulates a tug-of-war where the bot continuously generates deceptive examples to confuse the detector, while the bot detector tries to differentiate between real and fake content. The paper evaluates various bot detection scenarios, demonstrating how adversarial training can enhance detection models' robustness against evolving deceptive tactics in social media environments. This approach is particularly relevant in the context of detecting advanced social bots that mimic human behavior, a challenge addressed through adversarial training and synthetic attack generation.

Lee et al. [17] proposed a new scheme to identify malicious accounts on Twitter. The proposed scheme clustered accounts based on similar characteristics and then classified the clusters as normal or suspicious. The evaluation results showed that the proposed scheme performed well in terms of clustering and classification. The reason for this was

that the similarities between the accounts had a resemblance to each other. However, the performance of the method deteriorated when the similarities were too close to one another.

Afterward, Yang [41] presented two innovations for detecting fake accounts in social networks. Firstly, real-time data on the behavior of fake accounts was used to create a similarity-based real-time detector. The second innovation of this article is the topological description of the graph of fake accounts on a social network. They showed that a threshold-based classifier with reasonable computational efficiency can capture up to 99% of fake accounts with minimal false positives and false negatives. One of the issues in this method was the threshold, which in some cases caused difficulties in finding fake accounts.

Zhou et al. [44] paper proposes an enhanced oversampling technique for imbalanced classification tasks, using a conditional GAN (CGAN) with Wasserstein distance to mitigate issues such as gradient vanishing and mode collapse. It addresses noise and boundary-blurring problems in generated minority class samples by integrating K-means and K-nearest neighbor algorithms. Experimental results on multiple datasets demonstrate significant improvements in evaluation metrics like Recall, F1-score, G-mean, and AUC, surpassing traditional methods such as SMOTE and ADASYN. The method's ability to generate high-quality minority samples makes it effective in improving classifier performance on imbalanced datasets.

Liu et al. [19] This paper addresses the challenge of anomaly detection in imbalanced data streams with concept drift. It proposes an ensemble learning method that integrates GAN-based oversampling, stacking ensemble classifiers, and a consistency check module. By incorporating double encoders into the GAN and leveraging Wasserstein distance, the method improves the quality of generated samples. Experimental results show that the method outperforms traditional approaches in terms of detection accuracy and robustness against noise, especially in dynamic environments with concept drift. The method demonstrates significant advantages in handling both imbalanced data and concept drift.

Mohammadrezaei et al. [24] used three algorithms: support vector machine (SVM), logistic regression, and Gaussian SVM to identify fake user accounts. By comparing these three models, they concluded that the Gaussian SVM model was superior to the other two with an accuracy of 97.6%. The dataset had 10 fake nodes, and SMOTE was used to balance the minority class, but it had a large error rate. This model was also weak when dealing with big data.

Yuan et al. [43] had an imbalanced dataset in which one group of classes had fewer samples. In order to ensure unbiased classification, it is necessary to have classes in almost the same proportion. Therefore, they proposed a deep adversarial insider threat detection (DAITD) framework using Generative Adversarial Networks (GAN) to simulate true anomalous behavior diffusion and address the issue of imbalanced data.

Sahoo et al. [29] presented an automatic method for detecting fake news in the Chrome environment, which can detect fake news on Facebook. In particular, they employed deep learning to analyze the account's behavior by utilizing a variety of features, including some news content features. They compiled a raw dataset from multiple user posts to identify fake news on Facebook. These data were both relevant and irrelevant, based on the chosen features. They used both machine learning algorithms (KNN, SVM, Logistic Regression) and deep learning (LSTM) for news detection. Table 2 shows most of the advantages and disadvantages of the mentioned papers.

Putra Wanda et al. [40] also proposed a model to train extensive features with dynamic deep-learning architecture and classify malicious vertices using node-link information. They presented a function known as WalkPool pooling to improve network performance and construct dynamic deep learning. Throughout the training process, the convolutional layer aimed to compute feature extraction, specifically link information features. WalkPool pooling was used in this experiment to achieve maximum accuracy with only a slight loss in the network architecture training session. By performing nonlinear computation and reducing the parameters, they constructed the function to achieve high performance like an accuracy of 98.04%. However, this method only works for samples that are not in the form of a graph.

Shafqat et al. [32] The paper presents a methodology to address data imbalance in recommendation systems using hybrid GAN models. The approach involves preprocessing the data, performing statistical and pattern analysis, and passing it to a hybrid GAN model consisting of conditional GAN, WGAN-GP, and PacGAN. The conditional GAN architecture helps to explicitly condition the minority class, while the loss function of WGAN-GP and auxiliary classifier (AC) loss generates samples that belong to the minority class. The sampled input to the discriminator, as in PacGAN, eliminates model collapse and improves performance. The evaluation phase assesses the synthetic data's quality and the recommendation system's performance using the synthetic data. The results demonstrate that the proposed methodology can effectively address data imbalance and enhance the recommendation system's performance.

Elsewhere, Anuraganand Sharma et al. [33] Propose a novel knowledge transfer-based two-phase oversampling strategy that combines the strengths of SMOTE and GAN. addresses issues of class imbalance by combining the SMOTE and the Generative Adversarial Network (GAN). In a variety of benchmark datasets that were tested, the experimental

results demonstrate that the sample quality of minority classes has been improved. On F1-score measurements, its performance outperforms the next-best algorithm by up to around 94%.

In summary, while earlier studies have made important efforts in detecting fake accounts using classifiers, clustering, and sampling methods, many still struggle with problems like imbalanced datasets, limited feature extraction, and generalization across platforms. These challenges inspire our approach, which combines graph-based similarity measures with deep generative learning (GAN) to enhance detection performance on large-scale social network data.

3 Concepts

Principal Component Analysis (PCA) and Generative Adversarial Network (GAN) are the core algorithms utilized in this proposed method. In this section, the operations of these two algorithms will be discussed in more detail.

3.1 Principal Component Analysis (PCA)

The PCA method is popular due to its simplicity in extracting information from complex datasets [11] and yields the best results when applied to linear algebraic applications. It breaks down a multi-dimensional variable into a set of unrelated components, each of which is a linear combination of the original variables. The method is mainly used to analyze the principal components, reduce the number of variables, and find a communication structure among the variables. However, one of the significant issues in the PCA method is selecting the number of essential core components. In this paper, the first twenty columns with the highest variance are used to achieve better classification. At this stage, PCA is used to pass each altered similarity criterion once through the PCA stage [28][22]. The selection of the number of core components is one of the most significant issues in the PCA method. An unofficial method selects the total number of high variations based on the cumulative percentage, with the highest precision typically considered to be between 80 and 90 percent. To calculate the number of principal components or the number of dimensions, one formal group method is Kaiser's rule [15], which uses eigenvalues greater than one.

3.2 Generative Adversarial Network (GAN)

By framing the problem as supervised learning, GANs are an effective method for training. GAN is a deep generative model with two sub-models consisting of the discriminator model, which in this case tries to classify examples as either real or fake, and the generator model, which is trained to generate new structures of nodes [13]. In a zero-sum game adversarial, the two models are trained together until the discriminator model is tricked about half of the time, indicating that the generator model is producing plausible examples. GANs are an exciting and rapidly changing field. In this paper, generator's ability was used to generate the node and also utilized the discriminator's ability to learn the node's features. In other words, we did not utilize the generator during the inference phase to generate synthetic data; instead, we employed it during the training phase to assist the discriminator in enhancing its fake detection capabilities.

The steps involved in this method are demonstrated in Figure 1. We used a publicly available social network dataset. A subset of 1,000,000 nodes was selected to support the findings of this study: [8], this tabular dataset contains 5,384,162 users with 16,011,445 links among them. In this paper, 1,000,000 nodes were selected from the data, with almost 10,000 of them being fake. By chunking method nodes then mapped into a graph. By adjacency matrix the graph was converted into a matrix to analyze the relations between each node [14]. In order to find relationships between all nodes and achieve better separation of fake classes, similarity measures were calculated. These similarity measures are explained in detail in the following section. Ten similarity measures were applied to each matrix. Then, by calculating a node's own values and extracting diagonal values—representing the node's centrality, clustering, etc.—we captured its structural properties. Additionally, row-wise statistics (mean, max, variance) were computed to analyze how a node's neighbors behave. Furthermore, column-wise influence (sum of interactions) was calculated to measure how much influence a node has within the matrix. This process reduces the dimensionality of 10 matrices of size 1,000,000 × 10 to 10 matrices of size 1,000,000 × 5. Due to having large features, all matrices of similarities were passed through PCA. By the help of Kaiser's rule, dimension of the matrix reduce to 20, it means twenty most important features were adopted. Thus, a matrix of 1,000,000 * 20 dimensions remained. Afterwards, around 10,000 fake nodes separated and divided into two parts, train and test, in the ratio of 0.7 and 0.3, respectively, and this ratio was also maintained over multiple test steps. In the work by [24], SMOTE was used to address class imbalance. However, SMOTE generates synthetic samples based on linear interpolation between K-nearest neighbors in the minority class. In our experiments, this method produced samples that did not accurately represent the structural patterns of real fake nodes in graph-based data, leading to decreased classification performance. GAN solved this problem by synthesizing fake nodes with the generator model. The fake nodes were learned by the discriminator and could be identified from other nodes.

4 Proposed Method

4.1 Mapping Social Network's Data to Graph

To analyze the dataset, the data was first converted into a graph using similarity measures [14]. At this stage, each user was represented as a node, and each relation between users was represented as an edge. Depending on the definition of the relationship in the social network, the graph could be either directional or non-directional.

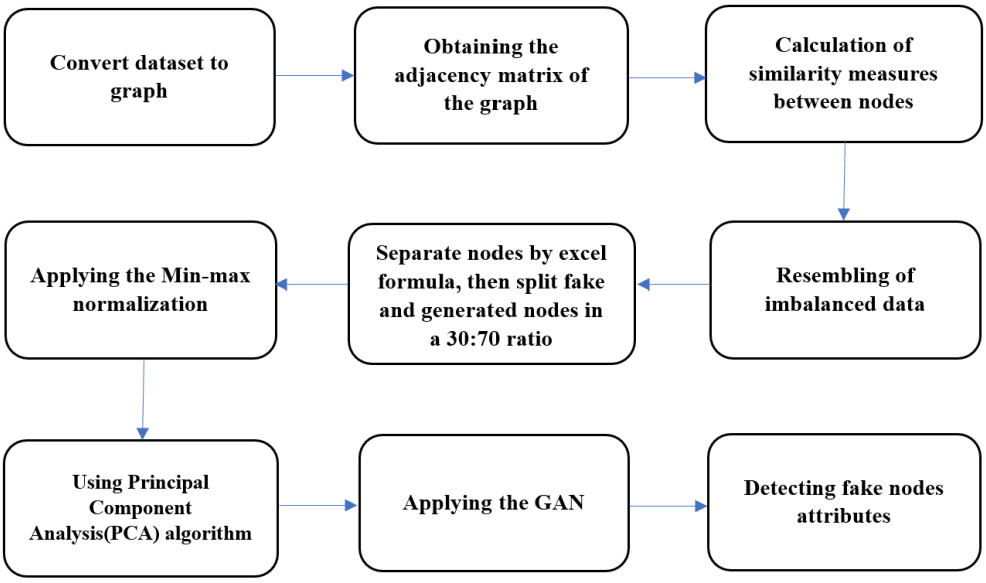

Figure 1: Steps of proposed method

4.2 Calculation of Network Graph Adjacency Matrix

The 1 million nodes were obtained by combining the source and destination columns using Excel formulas such as =UNIQUE(VSTACK(Source:Source, Destination:Destination)). We will clarify this step in the revised manuscript. For a graph with N users, a square matrix of size N*N was created [1]. If there was an edge from vertex i to vertex j, the element in the i-th row and j-th column, as well as the element in the j-th row and i-th column, was set to one; otherwise, it was set to zero.

4.3 Calculation of Different Similarity Measures Between Nodes

The information obtained from reviewed papers led to the conclusion that no single feature by itself was capable of distinguishing between users in a network. Therefore, several features were used in this method to increase the accuracy of detecting fake accounts. The purpose of defining similarity measures was to optimize and improve the quality of features extracted from the user's network. As stated in various papers, similarity measures were used to reduce the complexity of graph analysis problems. These measures are defined in the following paragraph.

Friendship Graph

In the social network, graph G is defined as a friendship graph because all nodes are directly connected to particular nodes, and they can also have no connections with each other [1].

$$FG(v).N = \{v\} \cup \{n \in G.N | \, n \neq v, \exists \, e \in G.E, e = \langle v, n \rangle\} \tag{1}$$

$$FG(v).E = \{\langle v, n' \rangle \in \ G.E |\ n \ \in FG(v).N\} \cup \{\langle n, n' \rangle \in \ G > E | n, n' \in FG(v).N\} \tag{2}$$

Common Friends

All vertices that are at a distance of 2 from both nodes are common friends of those two nodes [30]. FG(v).N represents the set of all nodes (including v itself) that are directly connected to node v — i.e., its immediate friends. The concept of "common friends" refers to the number of shared neighbors between two nodes in the graph.[10] :

$$CF(u, v) = |FG(v).N \cap FG(u).N| \tag{3}$$

Total Friends

The number of different friends between two nodes was displayed as follows: [10]. The term "total friends" typically refers to the degree of a node in the graph. T

$$Total\ Friends(v, u) = |FG(v).N \cup FG(u).N| \tag{4}$$

Jaccard Similarity

Jaccard coefficient calculates the ratio of mutual friends of two neighbor's nodes to their total friends [30]. The Jaccard similarity is a measure used to quantify the similarity or overlap between two sets of nodes based on their neighborhood relationships in the graph.

$$Jaccard\, F(v, u) = \frac{\left|FG(u).N \cap FG(v).N\right|}{\left|FG(u).N \cup FG(v).N\right|} \tag{5}$$

Adamic Adar Similarity

This similarity measure was originally used to find strong connections between web pages and was related to the number of common features that two pages shared. In the link prediction, this common link was the common neighbor of two vertices. The degree of similarity between two vertices in this method was obtained from the following relation. In this relation, Z was the common neighbor of 2 vertices U and V [20]. The Adamic-Adar similarity is a measure used to quantify the similarity or closeness between two nodes based on the common neighbors they share in the graph.

$$score(v, u) = \sum_{z \epsilon \Gamma(x) \cap \Gamma(y)} \frac{cosine(x, y)}{log(|\Gamma|)} \tag{6}$$

Correlation Similarity

This similarity is used to summarize the strength of the linear relation between two data samples. [3]. It is more relevant to the similarity between nodes based on their topological relationships within the graph.

$$Pearson's\ correlation\ coef(v.u) = \frac{covariance(v, u)}{x}(stdv(v) * stdv(u)) \tag{7}$$

Euclidean Similarity

Euclidean similarity (or distance) measures the straight-line distance between two points in a multi-dimensional space. In the context of social networks, each user can be represented as a feature vector capturing their network behavior or connections. By computing the Euclidean distance between users, we can estimate how structurally similar or different they are. This helps identify outliers, such as fake accounts, whose connectivity patterns deviate significantly from normal users.

$$D(x, y) = \sqrt{\sum_{n-1}^{i} (x_i - y_i)^2} \tag{8}$$

Cosine Similarity

the expression cosine(x, y) refers to the cosine similarity between two vectors x and y.

$x \cdot y$ is the dot product of the vectors.

$\|x\|$ and $\|y\|$ are their Euclidean (L2) norms.

The result ranges from –1 (opposite direction) to +1 (same direction); for non-negative data (e.g., text frequencies) the range is typically [0,1]

$$Cos(v.u) = \frac{|FG(v).N \cap FG(u).N|}{\sqrt{|FG(v).N|.|FG(u).N|}} \tag{9}$$

L1_Norm Similarity

In an attribute-less graph dataset, where nodes lack explicit attributes, a possible interpretation of L1 norm similarity could involve considering the pairwise differences in the number of common neighbors between nodes.[16]

$$L1\_norm(v.u) = \frac{|FG(v).N \cap FG(u).N|}{|FG(v).N|.|FG(u).N|} \tag{10}$$

Edge Weight Measure

Edge Weight similarity was first calculated as two separate features for each of the two vectors [7]. In an attribute-less graph dataset, an "edge weight measure" refers to the quantification or assignment of weights to the edges of the graph. Edge weights provide additional information about the relationships between nodes beyond the presence or absence of connections. Edge weights can be used to convey various characteristics or attributes associated with the connections in the graph. These weights can represent factors such as the strength of the relationship, the importance of the connection, or the frequency of interaction between nodes.[18]

$$w(v) = \frac{1}{\sqrt{1 + FG(v).N}} \tag{11}$$

$$w(u) = \frac{1}{\sqrt{1 + FG(u).N}} \tag{12}$$

Now the Edge Weight between two vertices U and V can be calculated in the following two ways.

Summation of the weights: The summation of the weights was equal to the two weights of u and v which are added together:

$$W(v, u) = w(v) + w(u) \tag{13}$$

Coefficient of weights: The coefficient of weights was equal to the product of the two weights. It was defined as below :

$$W(v, u) = w(v) * w(u) \tag{14}$$

In this section, for each similarity measure, a matrix with a diagonal of zero will be defined.

4.4 Resampling of Imbalance Data

To address the computational challenges of processing the full similarity matrices—each originally sized at 1,000,000×1,000,000 — we applied a dimensionality reduction strategy to extract meaningful node-level features. Each similarity matrix (e.g., Common Friends, Jaccard, Adamic-Adar) encodes pairwise similarity scores between nodes, with rows and columns representing individual nodes. Processing such large matrices directly was infeasible, so each was transformed into a more compact feature matrix of size 1,000,000×5. For every node (i.e., row), we extracted five features: (1) the diagonal value, representing the node's own score (e.g., centrality or self-similarity), (2–4) three row-wise statistics—the mean, maximum, and variance of similarity scores—capturing how a node relates to others, and (5) the column-wise sum, representing total incoming influence or connectivity from other nodes. As a result, we generated 10 matrices of size 1,000,000 × 5 , one for each similarity measure, and concatenated them into a combined matrix of size 1,000,000×50. This matrix was then processed using Principal Component Analysis (PCA) to reduce dimensionality and eliminate redundancy. Based on Kaiser's rule (retaining components with eigenvalues greater than

1), the first 20 principal components were selected, resulting in a final matrix of size 1,000,000 × 20. This dataset was then split into training and test sets, with the test data reserved for evaluation. Notably, the dataset was highly imbalanced, with approximately 90% of the instances corresponding to real users. This imbalance caused the minority class (fake users) to be treated as outliers during training. To mitigate this issue, approximately 10,000 fake nodes from the training set were identified and used to train a GAN model, allowing the network to generate realistic fake-like samples and improve classification performance.[23][34]

4.5 Training GAN and Detection of Fake Accounts

Researchers faced a significant challenge when applying the Synthetic Minority Oversampling Technique (SMOTE) to generate synthetic fake nodes. SMOTE often resulted in a high error rate, as the synthetic samples it produced failed to realistically capture the structure and distribution of fake nodes [6]. This limitation made it unsuitable for our task, where understanding subtle differences between real and fake nodes was crucial. To address this, we employed Generative Adversarial Networks (GANs), which are well-suited to learning complex data distributions and generating high-fidelity synthetic data.

The dataset was divided into a training set and a test set. Within the training set, fake and real nodes were separated. The GAN model was trained using only the fake nodes, with the goal of learning their structural distribution [31]. GANs consist of two competing neural networks: the generator and the discriminator. The generator aims to create fake nodes that resemble actual fake samples, while the discriminator tries to distinguish between real fake nodes and generated samples.

The generator received as input a 100-dimensional noise vector sampled from a standard distribution. This noise vector was passed through six fully connected (Dense) layers. The first five layers contained 64, 128, 256, 256, and 512 nodes, respectively. The final layer outputted a 100-dimensional vector, representing the generated fake node sample. All layers used the Tanh activation function. Therefore, the generator model comprised six Dense layers in total.

The discriminator, on the other hand, was designed to take as input a 100-dimensional vector, which could either be a real fake node from the training set or a generated sample from the generator. It processed this input through four Dense layers with 256, 128, 128, and 128 nodes, respectively, followed by a final Dense layer with a single node to output the classification probability. Each of the Dense layers in the discriminator was followed by a ReLU activation function and a Dropout layer with a rate of 0.2, to prevent overfitting. The final output layer used a Sigmoid activation function to produce a probability score indicating the likelihood that the input was real. A summary can be seen in table 1.

Table 1: Summary of Generator and Discriminator Architectures

Attribute	**Generator**	**Discriminator**
Input	100-dimensional noise vector	100-dimensional vector (real or generated)
Number of Layers	6 Dense layers	4 Dense layers + 1 output layer
Layer Sizes	64, 128, 256, 256, 512, 100	256, 128, 128, 128, 1
Activation Functions	Tanh (all layers)	ReLU (all hidden layers), Sigmoid (output)
Dropout	None	0.2 after each hidden layer
Output	100-dimensional fake node representation	Probability score [0, 1] (Fake vs. Generated)
Loss Function	Binary Cross-Entropy (via $\log D(G(z))$)	Binary Cross-Entropy (generated vs. fake)

The GAN was trained using the binary cross-entropy loss function, which is appropriate for binary classification problems. The discriminator loss function was defined as:

$$\mathcal{L}_D = -\mathbb{E}_{x\sim p_{\text{data}}}[\log D(x)] - \mathbb{E}_{z\sim p_z}[\log(1 - D(G(z)))]$$

where $D(x)$ is the probability that the discriminator assigns to a real sample x, and $G(z)$ is a sample generated from a random noise vector z. The generator, in turn, was optimized to minimize:

$$\mathcal{L}_G = -\mathbb{E}_{z\sim p_z}[\log D(G(z))]$$

These adversarial objectives ensured that the generator improved by producing samples that the discriminator could not easily distinguish from real ones.

During training, the generator attempted to generate fake-like data that could deceive the discriminator. In parallel, the discriminator adapted to detect even subtle differences between real and synthetic fake nodes. This adversarial process continued for several epochs, during which both models improved iteratively. As the generator learned to create increasingly realistic samples, the discriminator's ability to understand the structural characteristics of fake nodes improved as well.

Unlike typical GAN-based classifiers that incorporate both real and fake samples during training, our approach focuses exclusively on fake nodes. This design reflects the nature of our problem as an anomaly detection task, where the minority class (fake users) is underrepresented and more critical to model. By training the GAN solely on fake instances, the discriminator learns the fine-grained structure of fake profiles and acts as a detector for anomalous behavior at inference time. This setup allows the model to generalize to real samples without needing them explicitly during training.

It is important to emphasize that the generator was used only during training and was discarded during the inference phase. The trained discriminator became the final model used for fake node classification. During prediction, all nodes (both real and fake) were passed through the discriminator. In addition to fake and generated nodes, a subset of real (non-fake) nodes was also included in the test phase to evaluate the discriminator's generalization ability. These real nodes were passed through the trained discriminator to assess its capability to distinguish genuine user behavior from fake profiles. This step ensures a more realistic evaluation setting aligned with real-world deployment. Metrics such as accuracy and AUC reflect this combined test set. The output of the discriminator $D(x)$ was interpreted as the likelihood that the node was fake. A high score (close to 1.0) indicated a fake node, whereas a low score (close to 0.0) indicated a real one.

To determine the decision threshold, we analyzed the ROC curve and selected the point with the minimum Euclidean distance to the perfect classifier (i.e., TPR $= 1$, FPR $= 0$). This approach ensured that the classifier achieved high recall for fake nodes while minimizing false positives.

Finally, it should be noted that all components of this architecture — including layer sizes, activation functions, and training hyperparameters (learning rate = 0.0001, batch size = 128, epochs = 4000) — were selected based on extensive experimentation and empirical validation. This setup enabled the model to generate high-quality fake-like samples and train a robust discriminator capable of accurate fake node detection.

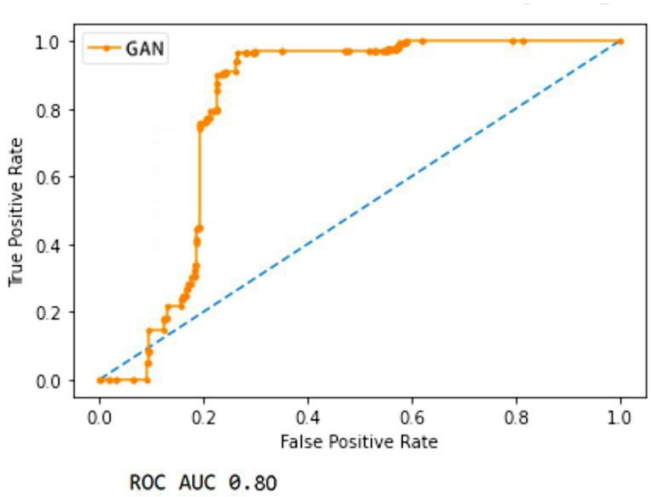

Figure 2: AUC of GAN in proposed method

Table 2: Overview of the Twitter Dataset

	Vertices	Links	Date	Labels
Twitter	5,384,160	16,011,443	2012	yes

5 Evaluation

In general, in deep learning algorithms, there are several criteria for evaluating and checking the performance of algorithms. One of the methods is the confusion matrix, a square matrix whose dimensions are equal to the number of classification classes [36]. The complete form of the square matrix of the data can be seen in Figure 3. In fact, the normal labels in this figure refer to fake nodes. The number of fake nodes, identified as fake, was 1,784, and the number of nodes generated by the generator and predicted as generated was 486. In addition, 339 nodes were generated but identified as fake, and 391 nodes were fake and predicted to be generated. Another criterion that shows the classification performance graphically is the ROC curve [9]. AUC is the area under the ROC curve, and if the AUC is one, the classification would be as accurate as possible. The horizontal axis of this graph indicates the false positive rate for negative classes, and the vertical axis shows the true positive rate of the positive classes. Figure 2 represents the AUC diagram of the proposed method in this Twitter dataset.[21]

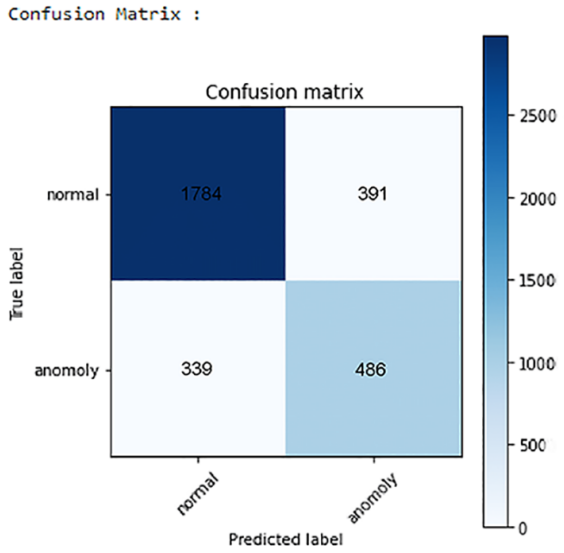

Figure 3: Confusion matrix of the proposed method

Although the generator is not used in the final prediction, we evaluated the quality of its generated fake nodes after training. This is important because good generated samples help the discriminator learn better. We compared the generated fake nodes with real fake ones using statistical features (mean, standard deviation) in table 3. The results show that the generated nodes are very similar to real fake nodes. The discriminator also assigned similar scores to both. These results suggest that the generator was able to produce realistic fake samples during training, which contributed to improving the discriminator's ability to distinguish fake accounts and enhanced overall model performance. Table 3 shows the comparison.

Table 3: Comparison Between Real Fake Nodes and GAN-Generated Samples

Metric	**Real Fake Nodes**	**Generated Samples**	**Observation**
Mean (PCA features)	0.523	0.517	Very close
Standard Deviation	0.176	0.181	Very close
Discriminator Output ($D(x)$)	Similar scores		Generator fooled D well

5.1 Results Comparison

In this paper [24], three types of machine learning algorithms were chosen: Gaussian support vector machine (Gaussian SVM), linear support vector machine (linear SVM), and logistic regression. The paper suggests that Gaussian SVM is superior to linear SVM and logistic regression in identifying fake accounts. As previously mentioned, a dataset of 1,000 nodes from the Twitter dataset was selected for this study. However, this separated dataset had only 10 fake nodes,

which was not adequate for classification. The researchers attempted to use the SMOTE method to compensate for the shortage of fake nodes, but the percentage of errors to create nodes was high. In the proposed method, 1,000,000 data points were set aside, and the number of fake data points was increased to around 10,000. The 10,000 data points were tested with classical machine learning algorithms too, and the following results were obtained. The results are shown in Figure 4. When the data increased, based on Figure 2 and Figure 5, the AUC of GAN was equal to 80% while the AUC of Linear Regression, SVM, and Gaussian SVM were 39.5%,53%, and 62%. Also, the efficiency of classical algorithms decreases in all metrics and does not perform data classification well. Figure 5 demonstrates another view of the three classical machine learning algorithms by examining 10,000 data points on AUC, with cross-validation set to K=10. As the amount of data increases, classical methods lose their efficiency in classification and performance. By comparing Figure 5 with Figure 2, which illustrates AUC for four distinct algorithms, it can be concluded that deep learning performs better in classifying enormous amounts of data and has a higher performance than the other three algorithms. However, it is important to note that the proposed method has some limitations, such as the need for a large amount of labeled data and computational resources. Further research is necessary to explore the effectiveness of other state-of-the-art techniques and address these limitations.

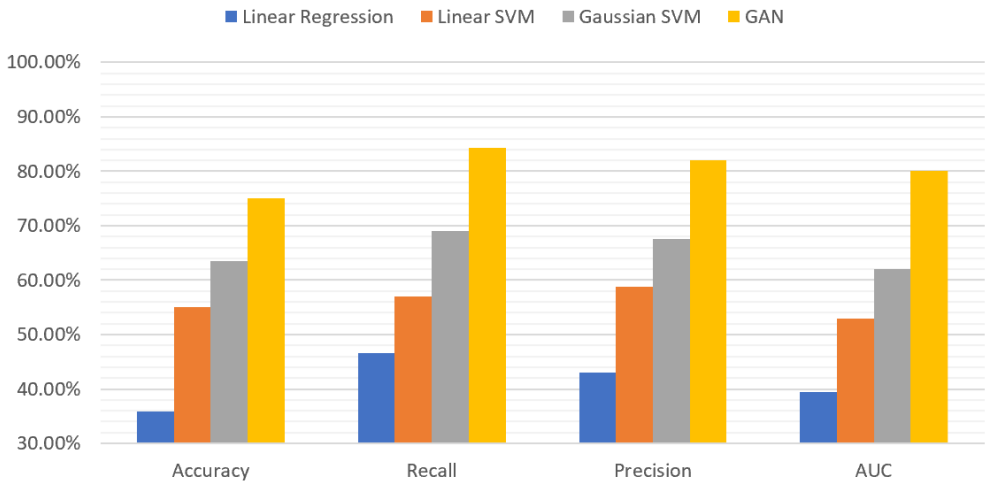

Figure 4: Comparison of accuracy, precision, recall and AUC for proposed method via logistic regression, SVM and Gaussian SVM by SMOTE resampling [24] over 10,000 data

Deep neural networks' multiple layers enable models to become more effective at learning complex features and carrying out more demanding computational tasks, this is because deep learning algorithms can ultimately learn from their own mistakes. One of the reasons that the GAN model performed better was the presence of an imbalanced dataset, meaning that there may be significantly more observations in one class than the other. Logistic regression can struggle to handle imbalanced data, as it tends to be biased towards the majority class. On the other hand, SVM is a powerful algorithm; it can be sensitive to outliers, which can significantly affect the classification performance of severely imbalanced datasets. Gaussian SVM is a type of traditional machine learning algorithm that may struggle with large and diverse datasets. On the other hand, GANs can be trained on large and diverse datasets and can even generate synthetic data to increase the size and diversity of the training set, which can improve their performance. As can be seen in Table 2, the proposed method has successfully classified two classes, while with the increase in the number of data, classical machine learning methods have shown their weakness in percentages.

Table 4: Final result

Modules	Accuracy	Recall	Precision	AUC
Logistic Regression	%35.8	%46.6	%43	%39.5
Support Vector Machine	%55	%57	%58.7	%53
Gaussian Support Vector Machine	%63.5	%69	%67.6	%62
Generative Adversarial Networks (Proposed Method)	%75	%84	%82	%80

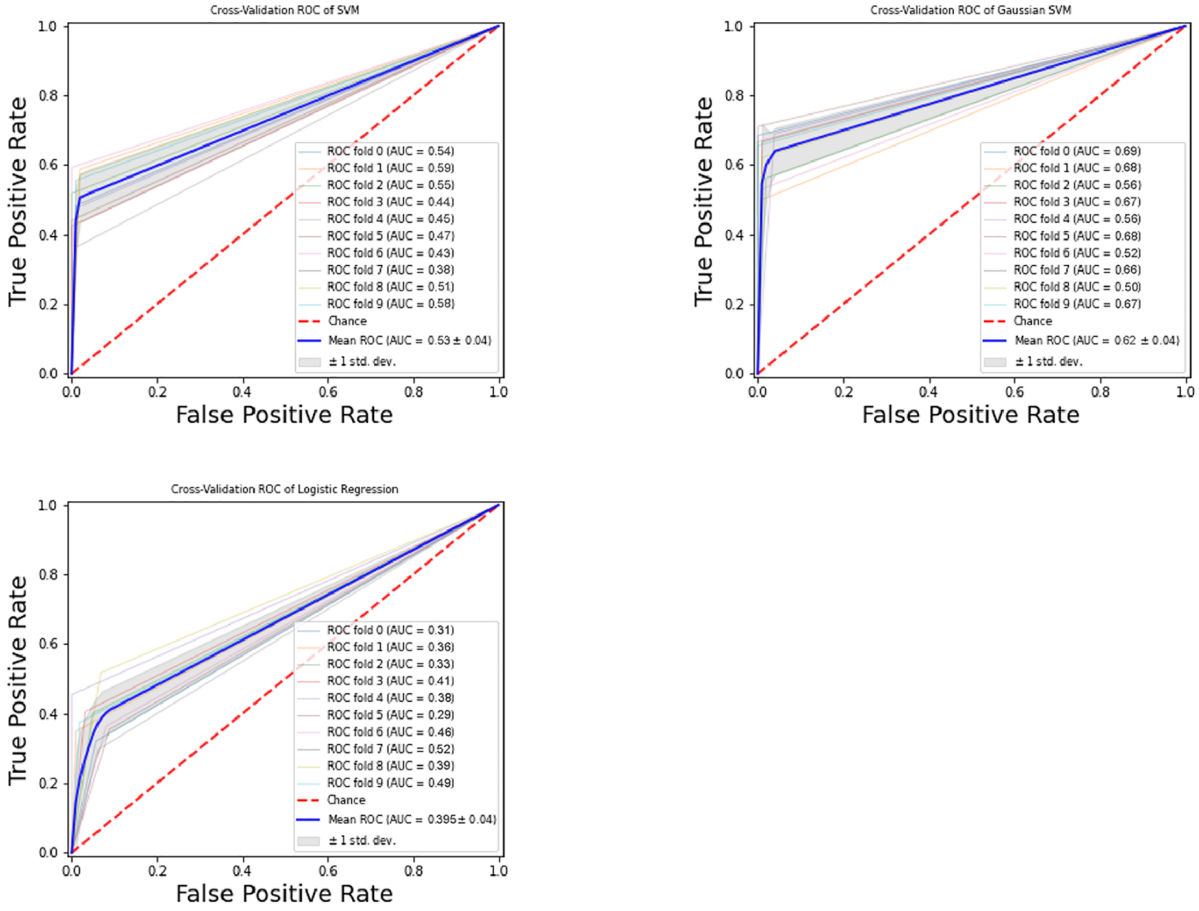

Figure 5: Classifying fake and real nodes in classical machine learning algorithms on 10,000 data. The top right image shows the Gaussian SVM algorithm which has the best performance among the other two algorithms, and the top left photo is the SVM algorithm and the photo Below is the logistic regression algorithm. According to pictures, these 3 algorithms will lose their effectiveness if the number of data increases.

6 Conclusion & Further Work

One known limitation of our training setup is the potential distribution shift between training and inference phases. Since the GAN-based discriminator was trained exclusively on generated and real fake nodes, it has not explicitly seen real user profiles during training. This could result in reduced generalization performance when encountering real nodes at test time. However, our approach frames this task as anomaly detection, focusing on modeling the fake class, and the discriminator's performance on real data suggests effective separation.

Also, node classification in social media is a significant method for detecting anomalies in a network. The current method demonstrates a generative deep-learning classifier that attempts to eliminate the challenge of covering big data problems and imbalanced classes. In generative deep learning, the generator develops fake data, and the discriminator learns the structure of nodes. Table 2 reveals a summary of the performance of the algorithms and a comparison of different metrics. The superiority of this method was not only in being flexible in dealing with all amounts of data but also in handling minority classes. Based on the results of this experiment, this method can achieve high accuracy and an AUC score equal to 80%. Thus, it can be concluded that GAN can be a promising solution to handle minority classes and data. However, the lack of sufficient data on the fake class is still a major challenge. To overcome this problem, applying other pre-trained models could affect the performance and accuracy. Moreover, different similarity criteria from what was introduced, and experimenting with other state-of-the-art generative algorithms such as generative transformers, can be investigated in future studies.

7 Acknowledgement

We acknowledged there is a preprint version of this manuscript that exists in arxiv.org and assets.researchsquare.com [4].

8 Conflict of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper. No financial, personal, or professional relationships have influenced the research, analysis, or findings presented in this study. Furthermore, the authors would like to confirm that all data, methods, and findings were presented transparently and objectively. The research has been conducted independently, and no external entity has unduly influenced the outcomes of this work.

References

[1] Cuneyt Gurcan Akcora, Barbara Carminati, and Elena Ferrari. "User similarities on social networks". In: *Social Network Analysis and Mining* 3.3 (2013), pp. 475–495.

[2] Morteza Behniafar, Alireza Nowroozi, and Hamid Reza Shahriari. "A Survey of Anomaly Detection Approaches in Internet of Things." In: *ISeCure* 10.2 (2018).

[3] Jacob Benesty et al. "Pearson correlation coefficient". In: *Noise reduction in speech processing*. Springer, 2009, pp. 1–4.

[4] J. Bordbar et al. "Detecting fake accounts through generative adversarial network in online social media". In: (2023). DOI: 10.21203/rs.3.rs-3710452/v1.

[5] Yazan Boshmaf et al. "Integro: leveraging victim prediction for robust fake account detection in osns." In: *NDSS*. Vol. 15. 2015, pp. 8–11.

[6] Nitesh V Chawla et al. "SMOTE: synthetic minority over-sampling technique". In: *Journal of artificial intelligence research* 16 (2002), pp. 321–357.

[7] William Cukierski, Benjamin Hamner, and Bo Yang. "Graph-based features for supervised link prediction". In: *The 2011 International joint conference on neural networks*. IEEE. 2011, pp. 1237–1244.

[8] Twitter Dataset. *anomalous-vertices-detection*. https://github.com/kagandi/anomalous-vertices-detection/tree/master/data. Accessed: July 1, 2014. 2014.

[9] Jesse Davis and Mark Goadrich. "The relationship between Precision-Recall and ROC curves". In: *Proceedings of the 23rd international conference on Machine learning*. 2006, pp. 233–240.

[10] Liyan Dong et al. "The algorithm of link prediction on social network". In: *Mathematical problems in engineering* 2013 (2013), pp. 173–183.

[11] Kimberly L Elmore and Michael B Richman. "Euclidean distance as a similarity metric for principal component analysis". In: *Monthly weather review* 129.3 (2001), pp. 540–549.

[12] Buket Erşahin et al. "Twitter fake account detection". In: *2017 International Conference on Computer Science and Engineering (UBMK)*. IEEE. 2017, pp. 388–392.

[13] Ian Goodfellow et al. "Generative adversarial networks". In: *Communications of the ACM* 63.11 (2020), pp. 139–144.

[14] Salim Jouili, Salvatore Tabbone, and Ernest Valveny. "Comparing graph similarity measures for graphical recognition". In: *International Workshop on Graphics Recognition*. Springer. 2009, pp. 37–48.

[15] Jennifer D Kaufman and William P Dunlap. "Determining the number of factors to retain: Q windows-based FORTRAN-IMSL program for parallel analysis". In: *Behavior Research Methods, Instruments, & Computers* 32.3 (2000), pp. 389–395.

[16] Nojun Kwak. "Principal component analysis based on L1-norm maximization". In: *IEEE transactions on pattern analysis and machine intelligence* 30.9 (2008), pp. 1672–1680.

[17] Sangho Lee and Jong Kim. "Early filtering of ephemeral malicious accounts on Twitter". In: *Computer communications* 54 (2014), pp. 48–57.

[18] Oleksandr Letychevskyi et al. "Algebraic Matching of Vulnerabilities in a Low-Level Code." In: *ISeCure* 11.3 (2019).

[19] Yansong Liu et al. "An ensemble learning method with GAN-based sampling and consistency check for anomaly detection of imbalanced data streams with concept drift". In: *PLOS ONE* 19.1 (2024), e0292140. DOI: 10.1371/journal.pone.0292140.

[20] Linyuan Lü and Tao Zhou. "Link prediction in weighted networks: The role of weak ties". In: *EPL (Europhysics Letters)* 89.1 (2010), p. 18001.

[21] KA Malyshenko, MM Shafiee, and VA Malyshenko. "Identification of Fake News Using Emotional Profiling as an Approach to Text Analysis." In: *ISeCure* 16.2 (2024).

[22] Mohammadreza Mohammadrezaei. "Detecting Fake Accounts in Social Networks Using Principal Components Analysis and Kernel Density Estimation Algorithm (A Case Study on the Twitter Social Network)". In: *Electronic and Cyber Defense* 9.3 (2021), pp. 109–123.

[23] Mohammadreza Mohammadrezaei, Ebrahim Shiri Mohammad, and Amir Masoud Rahmani. "Detection of fake accounts in social networks based on One Class Classification". In: *THE ISC INTERNATIONAL JOURNAL OF INFORMATION SECURITY* 11 (2019), pp. 173–183.

[24] Mohammadreza Mohammadrezaei, Mohammad Ebrahim Shiri, and Amir Masoud Rahmani. "Identifying fake accounts on social networks based on graph analysis and classification algorithms". In: *Security and Communication Networks* 2018 (2018), pp. 173–183.

[25] Shaghayegh Najari et al. "Adversarial Botometer: Adversarial Analysis for Social Bot Detection". In: *Social Network Analysis and Mining* 14.1 (2024), p. 220.

[26] Muhammad Al-Qurishi et al. "SybilTrap: A graph-based semi-supervised Sybil defense scheme for online social networks". In: *Concurrency and Computation: Practice and Experience* 30.5 (2018), e4276.

[27] Shubhangi Rastogi and Divya Bansal. "A review on fake news detection 3T's: typology, time of detection, taxonomies". In: *International Journal of Information Security* 22.1 (2023), pp. 177–212.

[28] Łukasz Sadowski, Mehdi Nikoo, and Mehrdad Nikoo. "Principal component analysis combined with a self organization feature map to determine the pull-off adhesion between concrete layers". In: *Construction and Building Materials* 78 (2015), pp. 386–396.

[29] Somya Ranjan Sahoo and Brij B Gupta. "Multiple features based approach for automatic fake news detection on social networks using deep learning". In: *Applied Soft Computing* 100 (2021), p. 106983.

[30] Julio Santisteban and Javier Tejada-Cárcamo. "Unilateral Jaccard Similarity Coefficient." In: *GSB@ SIGIR*. 2015, pp. 23–27.

[31] Thomas Schlegl et al. "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery". In: *International conference on information processing in medical imaging*. Springer. 2017, pp. 146–157.

[32] Wafa Shafqat and Yung-Cheol Byun. "A Hybrid GAN-Based Approach to Solve Imbalanced Data Problem in Recommendation Systems". In: *IEEE Access* 10 (2022), pp. 11036–11047.

[33] Anuraganand Sharma, Prabhat Kumar Singh, and Rohitash Chandra. "SMOTified-GAN for class imbalanced pattern classification problems". In: *Ieee Access* 10 (2022), pp. 30655–30665.

[34] Taher Al-Shehari and Rakan A Alsowail. "Random resampling algorithms for addressing the imbalanced dataset classes in insider threat detection". In: *International Journal of Information Security* 22.3 (2023), pp. 611–629.

[35] S Sivamohan, SS Sridhar, and S Krishnaveni. "An effective recurrent neural network (RNN) based intrusion detection via bi-directional long short-term memory". In: *2021 international conference on intelligent technologies (CONIT)*. IEEE. 2021, pp. 1–5.

[36] Stephen V Stehman. "Selecting and interpreting measures of thematic classification accuracy". In: *Remote sensing of Environment* 62.1 (1997), pp. 77–89.

[37] Chenhao Tan et al. "User-level sentiment analysis incorporating social networks". In: *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2011, pp. 1397–1405.

[38] Adem Tekerek. "A novel architecture for web-based attack detection using convolutional neural network". In: *Computers & Security* 100 (2021), p. 102096.

[39] Gulcin Bilgin Turna. "Impact of social media on tourism, hospitality and events". In: *Handbook on Tourism and Social Media*. Edward Elgar Publishing, 2022.

[40] Putra Wanda and Huang J Jie. "DeepFriend: finding abnormal nodes in online social networks using dynamic deep learning". In: *Social Network Analysis and Mining* 11.1 (2021), pp. 1–12.

[41] Zhi Yang et al. "Uncovering social network sybils in the wild". In: *ACM Transactions on Knowledge Discovery from Data (TKDD)* 8.1 (2014), pp. 1–29.

[42] Morteza Yousefi Kharaji, Fatemeh Salehi Rizi, and Mohammad Reza Khayyambashi. "A New Approach for Finding Cloned Profiles in Online Social Networks". In: *arXiv e-prints* 11 (2014), arXiv–1406.

[43] Fangfang Yuan et al. "Data augmentation for insider threat detection with GAN". In: *2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE. 2020, pp. 632–638.

[44] Hongfang Zhou et al. “A novel oversampling method based on Wasserstein CGAN for imbalanced classification”. In: *Cybersecurity* 8.1 (2025), p. 7.