Detecting fake accounts through Generative adversarial network in online social media

Jinus Bordbar\textsuperscript{1*}, Mohammadreza Mohammadrezaei\textsuperscript{2*}, Saman Ardalan\textsuperscript{3} and Mohammad Ebrahim Shiri\textsuperscript{4}

\textsuperscript{1}Department of Computer, Islamic Azad University, Shiraz Branch, Shiraz, Iran.
\textsuperscript{2}Department of Computer, Islamic Azad University, Ramhormoz Branch, Ramhormoz, Iran.
\textsuperscript{3}Institute for Medical Informatics and Statistics, University of Kiel, Kiel, Germany.
\textsuperscript{4}Department of Mathematics and Computer Science, Amirkabir University of Technology, Tehran, Iran.

*Corresponding author(s). E-mail(s): Jinus.Bordbar@gmail.com; Mohammadrezaei.m.reza@gmail.com;
Contributing authors: Ardalan@medinfo.uni-kiel.de; 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 fake user accounts in the Twitter dataset. 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.

Keywords: Fake accounts, Online social networks, Generative adversarial network, Machine learning algorithms
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 [1]. 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, this phenomenon has become ubiquitous, with more than 3.8 billion users utilizing this environment [2]. However, the widespread usage of social media has also led to its improper use by some profit-seeking and hostile individuals [3]. 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 [4].

Most methods and approaches used to identify fake accounts have been either user-level or graph-based [5, 6]. 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. 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 have a high error rate based on their performance [7].

This method attempts to integrate 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, and therefore use those connections to help identify fake accounts. Essentially, the method tries to leverage information about real connections between accounts to better identify fake accounts that seem to be similar to real connections. (2) The method uses feature extraction methods (Principal Component Analysis) due to the high dispersion of important features in a matrix, to reduce the number of variables that are used to separate fake and real accounts. 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. By using deep learning, the method can better identify fake accounts, even when they are very similar to real accounts. In this proposed method, the social network was mapped into a graph, 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. The Twitter data which were used is fully shown in Table 1.

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

| Table 1 The Dataset |
|---------------------|
| Vertices | Links | Date | Labels |
| Twitter | 5,384,160 | 16,011,443 | 2012 | yes |

2 Related Work

Yousefi et al. [8] 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.

Lee et al. [9] 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 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 [10] presented two innovations for detecting fake accounts in social networks. Firstly, real-time data on the behavior of fake accounts were 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.
Mohammadrezaei et al. [7] 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. [11] 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. [12] 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. [13] 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 links 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. [14] 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 generate 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. [15] 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 949%.

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 [16] 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 ten columns with the highest variance are used to achieve better classification. At this stage, PCA is used to pass each similarity criterion once through the PCA stage [17][18]. 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 number of dimension, one formal group method is Kaiser’s rule [19], 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 consists 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 structure of nodes [20]. 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.

4 Proposed Method

The steps involved in this method are demonstrated in Figure 1. The GitHub repository contains the Twitter data used to support the findings of this study:
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, and were then mapped into a graph. By adjacency matrix the graph was converted into an matrix to analyze the relations between each node [21]. In order to find relations between all nodes and better separation in fake classes, similarity measures were calculated. All of these similarities are explained in detail in the following part. Ten similarity measures were applied to each matrix. Due to having large and scattered features, all matrices of similarities were passed through PCA. By the help of Kaiser’s rule dimension of matrix reduce to 10, it means ten most important features were adopted. Thus, a matrix of 1,000,000 * 100 dimensions remained. The Min-max normalization algorithm was then implemented [22]. The data were 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. The minority class of the training data was an obstacle in [7] synthetic minority over sampling technique (SMOTE) were used while, SMOTE create nodes based on K-nearest neighbors and these nodes did not contain accurate information. 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.

4.1 Mapping Social Network’s Data to Graph

To analyze the Twitter dataset, the data was first converted into a graph using similarity measures [21]. 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.
4.2 Calculation of Network Graph Adjacency Matrix

For a graph with N users, a square matrix of size N*N was created [23]. 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 [23].

\[
FG(v).N = \{v\} \cup \{n \in G.N \mid n \neq v, \exists e \in G.E, e = \langle v, n \rangle\}
\]

\[
FG(v).E = \{(v, n') \in G.E \mid n \in FG(v).N \} \cup \{(n, n') \in G \mid n, n' \in FG(v).N\}
\]

**Common Friends**

All vertices that are at a distance of 2 from both nodes are common friends of those two nodes [24]. The concept of “common friends” refers to the number of shared neighbors between two nodes in the graph [25]:

\[
CF(u, v) = |FG(v).N \cap FG(u).N|
\]

**Total Friends**

The number of different friends between two nodes was displayed as follows: [25]. The term “total friends” typically refers to the degree of a node in the graph.

\[
Total\ Friends(v, u) = |FG(v).N \cup FG(u).N|
\]

**Jaccard Similarity**

Jaccard coefficient calculates the ratio of mutual friends of two neighbor’s node to their total friends [24]. 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.

\[
Jaccard \ F(v, u) = \frac{|FG(u).N \cap FG(v).N|}{|FG(u).N \cup FG(v).N|} \quad (5)
\]

### 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 [26]. 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.

\[
\text{score}(v, u) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{\text{cosine}(x, y)}{\log(|\Gamma|)} \quad (6)
\]

### Correlation Similarity

This similarity is used to summarize the strength of the linear relation between two data samples. [27]. It is more relevant to similarity between nodes based on their topological relationships within the graph.

\[
\text{Pearson’s correlation coef}(v, u) = \frac{\text{covariance}(v, u)}{x \cdot \text{stdv}(v) \cdot \text{stdv}(u)} \quad (7)
\]

### Euclidean Similarity

The Euclidean Similarity formula was simple and uses the Pythagorean theorem to calculate the distance between two points in Cartesian coordinates, which can be seen in Formula 8. Euclidean distance finds the shortest path between two points as the distance [16].

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

### Cosine Similarity

Another measure of similarity between graph nodes was Cosine Similarity. Cosine Similarity actually calculates the similarity between two product vectors [23] and in an attribute-less graph dataset, cosine similarity is a measure used to quantify the similarity between two nodes based on their neighborhood relationships in the graph.

\[
\text{Cos}(v, u) = \frac{|FG(v).N \cap FG(u).N|}{\sqrt{|FG(v).N| \cdot |FG(u).N|}} \quad (9)
\]
**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.\[28\]

\[ L_{1\text{norm}}(v,u) = \frac{|FG(v).N \cap FG(u).N|}{|FG(v).N||FG(u).N|} \quad (10) \]

**Edge Weight Measure**

Edge Weight similarity was first calculated as two separate features for each of the two vectors \[29\]. 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.

\[ w(v) = \frac{1}{\sqrt{1 + FG(v).N}} \quad (11) \]

\[ w(u) = \frac{1}{\sqrt{1 + FG(u).N}} \quad (12) \]

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:

\[ W(v, u) = w(v) + w(u) \quad (13) \]

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

\[ W(v, u) = w(v) \times w(u) \quad (14) \]

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

**4.4 Resampling of Imbalance Data**

Each of the criteria was passed once through the PCA stage. The first ten columns (as mentioned in section 3) of every matrix had the most important features. Afterward, these ten columns of ten main features were concatenated together, which became a matrix of 1,000,000*100. The data was divided into two splits, test and train. Next, the Min-max normalization algorithm was applied to the test and train datasets so that the data range would place between zero and one, next the test data was set aside for
evaluation. In this case, data was not balanced and about 90% of the data belonged to one class (real users), however, applying such a data set made the minority class (fake users) be ignored and considered as an outlier. To solve this problem, About 10,000 fake nodes were identified in the training set and were fed to the GAN algorithm to address the problem of the minority class. [30]

4.5 Training GAN and Detection of Fake Accounts

Researchers faced a challenge with the high error percentage of producing fake nodes in the synthetic minority over sampling technique (SMOTE). [31] To address this, the use of algorithms like GAN was necessary to learn the structure of fake nodes. The dataset was divided into a training set and a test set. In the training set, fake nodes were separated from real ones, and GAN was trained based on the fake nodes. [32]. During the training process, the generator attempts to generate data that can deceive the discriminator, ultimately leading to improve discriminator’s ability to detect fake nodes, therefore, The GAN learns the structure of the fake nodes. GAN consists of two neural network models: the generator and the discriminator model. The generator model attempted to produce fake nodes that resembled the actual fake nodes. At first, The input of the generator is a 100-dimensional noise vector, the generator actually has 5 Dense layers with 64, 128, 256, 256, and 512 nodes, respectively, followed by a final Dense layer with 100 nodes. So in total, this generator has 6 Dense layers and the Tanh activation function in each layer. While, the discriminator model tried to find fake nodes in the dataset in such a way that the input of the discriminator is a 100-dimensional vector representing a generated sample, and a 100-dimensional vector representing a real sample from the training dataset. The discriminator has 4 Dense layers with 256, 128, 128, and 128 nodes, respectively, followed by a final Dense layer with just 1 node. Each Dense layer is followed by a ReLU activation function and a Dropout layer with a rate of 0.2, except for the final Dense layer, which is followed by a Sigmoid activation function. It is worth noting that the training process involves optimizing the Binary crossentropy loss function and adjusting various hyperparameters, such as the learning rate to 0.001, batch size to 16, and the epochs number to 50.

For more explanation, the generator takes a random noise vector as input and generates a sample that is meant to resemble the actual fake data. The generator tries to create synthetic data that is as close to the actual fake data as possible. The discriminator takes both fake data samples and synthetic data samples as input and tries to classify them as fake nodes which were made or fake nodes that existed in the dataset. During training, the discriminator is trained on a combination of fake and synthetic data samples and is updated to better distinguish between fake and synthetic data. The generator is updated based on how well it was able to fool the discriminator. The process of training the generator and discriminator is repeated until the discriminator perfectly differentiates the classes, therefore, the discriminator has fully understood the fake data structure. The generator is not required in the subsequent steps, it contributes to improving the discriminator’s performance. During prediction, the trained discriminator is employed to classify all fake instances and whatever is not fake consider real. It is noteworthy that all the aforementioned results have been obtained through numerous and repeated testing experiences.
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 [33]. 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, this method identified as fake were 1,784, and the number of nodes that were generated by generator and predicted as generated was 486. Moreover, 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 [34]. 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 on this Twitter dataset.

5.1 Results Comparison

In this paper [7], 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 10,000. The 10,000 data 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.
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 3, 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.

Fig. 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.
| Modules                             | Accuracy | Recall | Precision | AUC   |
|------------------------------------|----------|--------|-----------|-------|
| Generative Adversarial Networks    | %75      | %84    | %82       | %80   |
| 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   |

### 6 Conclusion & Further Work

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 3 reveals a summary of the performance of the algorithms and a comparison of different metrics. The superiority of this method was not only 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.

### References

[1] Boshmaf, Y., Logothetis, D., Siganos, G., Lería, J., Lorenzo, J., Ripeanu, M., Beznosov, K.: Integro: leveraging victim prediction for robust fake account detection in osns. In: NDSS, vol. 15, pp. 8–11 (2015)

[2] Turna, G.B.: Impact of social media on tourism, hospitality and events. In: Handbook on Tourism and Social Media. Edward Elgar Publishing, ??? (2022)

[3] Erşahin, B., Aktaş, Ö., Kılınç, D., Akyol, C.: Twitter fake account detection. In: 2017 International Conference on Computer Science and Engineering (UBMK), pp. 388–392 (2017). IEEE

[4] Bilge, L., Strufe, T., Balzarotti, D., Kirda, E.: All your contacts are belong to us: automated identity theft attacks on social networks. In: Proceedings of the 18th
[5] Tan, C., Lee, L., Tang, J., Jiang, L., Zhou, M., Li, P.: User-level sentiment analysis incorporating social networks. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1397–1405 (2011)

[6] Al-Qurishi, M., Rahman, S.M.M., Alamri, A., Mostafa, M.A., Al-Rubaian, M., Hossain, M.S., Gupta, B.B.: Sybiltrap: A graph-based semi-supervised sybil defense scheme for online social networks. Concurrency and Computation: Practice and Experience 30(5), 4276 (2018)

[7] Mohammadrezaei, M., Shirif, M.E., Rahmani, A.M.: Identifying fake accounts on social networks based on graph analysis and classification algorithms. Security and Communication Networks 2018, 173–183 (2018)

[8] Yousefi Kharaji, M., Salehi Rizi, F., Khayyambashi, M.R.: A new approach for finding cloned profiles in online social networks. arXiv e-prints 11, 1406 (2014)

[9] Lee, S., Kim, J.: Early filtering of ephemeral malicious accounts on twitter. Computer communications 54, 48–57 (2014)

[10] Yang, Z., Wilson, C., Wang, X., Gao, T., Zhao, B.Y., Dai, Y.: Uncovering social network sybils in the wild. ACM Transactions on Knowledge Discovery from Data (TKDD) 8(1), 1–29 (2014)

[11] Yuan, F., Shang, Y., Liu, Y., Cao, Y., Tan, J.: Data augmentation for insider threat detection with gan. In: 2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI), pp. 632–638 (2020). IEEE

[12] Sahoo, S.R., Gupta, B.B.: Multiple features based approach for automatic fake news detection on social networks using deep learning. Applied Soft Computing 100, 106983 (2021)

[13] Wanda, P., Jie, H.J.: Deepfriend: finding abnormal nodes in online social networks using dynamic deep learning. Social Network Analysis and Mining 11(1), 1–12 (2021)

[14] Shafqat, W., Byun, Y.-C.: A hybrid gan-based approach to solve imbalanced data problem in recommendation systems. IEEE Access 10, 11036–11047 (2022)

[15] Sharma, A., Singh, P.K., Chandra, R.: Snotified-gan for class imbalanced pattern classification problems. Ieee Access 10, 30655–30665 (2022)

[16] Elmore, K.L., Richman, M.B.: Euclidean distance as a similarity metric for principal component analysis. Monthly weather review 129(3), 540–549 (2001)

[17] Sadowski, L., Nikoo, M., Nikoo, M.: Principal component analysis combined
with a self organization feature map to determine the pull-off adhesion between concrete layers. Construction and Building Materials 78, 386–396 (2015)

[18] Mohammadrezaei, M.: Detecting fake accounts in social networks using principal components analysis and kernel density estimation algorithm (a case study on the twitter social network). Electronic and Cyber Defense 9(3), 109–123 (2021)

[19] Kaufman, J.D., Dunlap, W.P.: Determining the number of factors to retain: Q windows-based fortran-imsl program for parallel analysis. Behavior Research Methods, Instruments, & Computers 32(3), 389–395 (2000)

[20] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial networks. Communications of the ACM 63(11), 139–144 (2020)

[21] Jouili, S., Tabbone, S., Valveny, E.: Comparing graph similarity measures for graphical recognition. In: International Workshop on Graphics Recognition, pp. 37–48 (2009). Springer

[22] Gajera, V., Gupta, R., Jana, P.K., et al.: An effective multi-objective task scheduling algorithm using min-max normalization in cloud computing. In: 2016 2nd International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT), pp. 812–816 (2016). IEEE

[23] Akcora, C.G., Carminati, B., Ferrari, E.: User similarities on social networks. Social Network Analysis and Mining 3(3), 475–495 (2013)

[24] Santisteban, J., Tejada-Cárcamo, J.: Unilateral jaccard similarity coefficient. In: GSBD@ SIGIR, pp. 23–27 (2015)

[25] Dong, L., Li, Y., Yin, H., Le, H., Rui, M.: The algorithm of link prediction on social network. Mathematical problems in engineering 2013, 173–183 (2013)

[26] Lü, L., Zhou, T.: Link prediction in weighted networks: The role of weak ties. EPL (Europhysics Letters) 89(1), 18001 (2010)

[27] Benesty, J., Chen, J., Huang, Y., Cohen, I.: Pearson correlation coefficient. In: Noise Reduction in Speech Processing, pp. 1–4. Springer, ??? (2009)

[28] Kwak, N.: Principal component analysis based on l1-norm maximization. IEEE transactions on pattern analysis and machine intelligence 30(9), 1672–1680 (2008)

[29] Cukierski, W., Hamner, B., Yang, B.: Graph-based features for supervised link prediction. In: The 2011 International Joint Conference on Neural Networks, pp. 1237–1244 (2011). IEEE

[30] Mohammadrezaei, M., Mohammad, E.S., Rahmani, A.M.: Detection of fake
accounts in social networks based on one class classification. THE ISC INTERNATIONAL JOURNAL OF INFORMATION SECURITY 11, 173–183 (2019)

[31] Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: Smote: synthetic minority over-sampling technique. Journal of artificial intelligence research 16, 321–357 (2002)

[32] Schlegl, T., Seeböck, P., Waldstein, S.M., Schmidt-Erfurth, U., Langs, G.: Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In: International Conference on Information Processing in Medical Imaging, pp. 146–157 (2017). Springer

[33] Stehman, S.V.: Selecting and interpreting measures of thematic classification accuracy. Remote sensing of Environment 62(1), 77–89 (1997)

[34] Davis, J., Goadrich, M.: The relationship between precision-recall and roc curves. In: Proceedings of the 23rd International Conference on Machine Learning, pp. 233–240 (2006)