Asian Journal of Research in Computer Science

Volume 16, Issue 1, Page 15-27, 2023; Article no.AJRCOS.99992
ISSN: 2581-8260

Generation and Evaluation of Tabular Data in Different Domains Using Gans

Persevearance Marecha [a*] and Lu Ye [a]

[a] *School of Information and Electronic Engineering, Zhejiang University of Science and Technology, Hangzhou, China.*

Authors' contributions

This work was carried out in collaboration between both authors. Both authors read and approved the final manuscript.

Article Information

DOI: 10.9734/AJRCOS/2023/v16i1331

Open Peer Review History:

This journal follows the Advanced Open Peer Review policy. Identity of the Reviewers, Editor(s) and additional Reviewers, peer review comments, different versions of the manuscript, comments of the editors, etc are available here: https://www.sdiarticle5.com/review-history/99992

Received: 12/03/2023
Accepted: 16/05/2023
Published: 24/05/2023

Original Research Article

ABSTRACT

Deep learning techniques like Generative Adversarial Networks (GANs) provide solutions in many domains where real data needs to be kept private. Synthesizing tabular data is difficult because of its high complexity. Tabular data usually contains a mixture of discrete and continuous data, which is not an easy model to build. The contributions made in this paper include training and generating data with the original Vanilla Gan, then CGan and WGan-Gp and WCGan-Gp which performs better than the former. The Adult Income Census dataset mainly focuses on predicting whether income exceeds 50,000 per year based on census data, then comparing the accuracy of machine learning models and calculating the F1 scores. Then the use of TimeGan on the stock dataset, comparing synthetic data vs real data. This paper will explore the use of GANs for generating and evaluating tabular data in different domains.

Keywords: Generative adversarial networks; tabular data; synthetic.

**Corresponding author: E-mail: persevearancem@yahoo.com;*

1 INTRODUCTION

In recent years, generative adversarial networks (GANs) have shown to be an effective method for generating complex data such as images, speech, and text. However, generating and evaluating tabular data can be more challenging due to the structured nature and high dimensionality of this type of data. GANs are a class of artificial neural networks used for generating new data that is similar to a given input dataset. GANs are made up of two neural networks: a generator and a discriminator. The generator creates new data that is meant to be similar to the input dataset, while the discriminator tries to distinguish between the generated data and the real data; the training process is called adversarial training. Data must be stored in an organized manner, just like any other resource. Using tables is a very direct and effective method. Rows and columns make up data structures like Excel spreadsheets. The most prevalent type of data that is stored in this format is known as "tabular data." Finding qualitative tabular data is difficult since it belongs to private sectors that restrict and/or monetize access to it due to its value. However, if the data is made available to the public, it will undoubtedly be subject to strict privacy regulations that may severely limit its utility. Creating qualitative synthetic tabular data is not easy, due to the wide variety of data types in different columns. Categorical data is difficult to handle due to its nature, and numerical data can be difficult to process if it follows complex, non-Gaussian distributions. As a result, there is still no universally accepted generative model for synthesizing tabular data since different GAN models can be used for training data in different domains due to their advantages in each domain [1]. Generative adversarial networks (GANs) have literally conquered the field of image, data and text generation since 2014, when they were introduced by I. J. Good Fellow et al [2].

1.1 Purpose of the Study

In many different fields, tabular data is used a lot, and it has become an important part of predicting what might be needed. When we open YouTube each day, our preferred videos are already in the queue and can be viewed with just a single click, eliminating the need for time-consuming searches. People can get life and medical advice from data that can be used to predict disease risk. Data can also assist businesses and governments in making decisions. In the business sector, we take for example the stock market predicting dataset where businesses can check previous stock exchange with the use of technologies like Fourier transforms which eliminate a lot of noise and create approximations of the real stock movement. These trend approximations can help businesses to pick the prediction trends more accurately. In the case of credit cards, Gans models are used to generate realistic data which is used in fraud detection cases. The availability of massive amounts of data motivates individuals to investigate various applications, resulting in exponential growth of the field. GANs models create new data instances that resemble your training data.

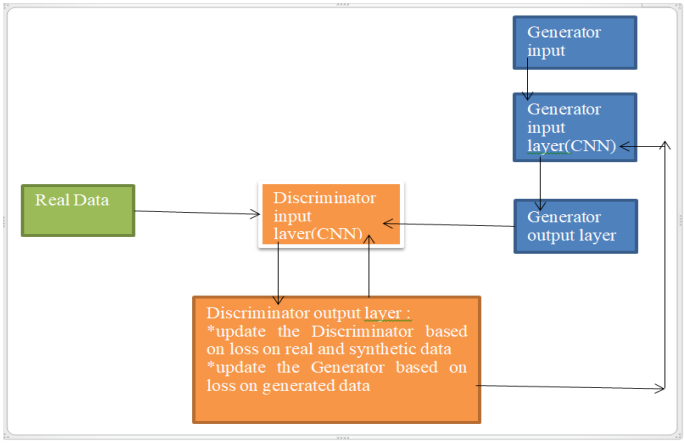

Fig. 1. GAN architecture

The above illustration shows how I best understand the operation of a GAN model. It is comprised of two networks: the generator, which is the neural network that learns to generate plausible data that becomes the training examples for the discriminator.The other network is the discriminator, which learns to distinguish the generator's fake data from real data. The discriminator penalizes the generator for producing implausible results. When training goes well, the discriminator won't be able to tell the difference between real and fake. It starts to classify fake data as real. The training process is shown in the GAN architecture figure above.

2 FUNDAMENTALS

2.1 Machine Learning

Machine learning (ML) is a field of inquiry devoted to understanding and building methods that learn, that is, methods that leverage data to improve performance on some set of tasks. It is seen as a part of artificial intelligence. Machine learning algorithms build a model based on sample data, known as "training data," in order to make predictions or decisions without being explicitly programmed to do so [3]. Machine learning algorithms are used in a wide variety of applications, such as medicine, email filtering, speech recognition, agriculture, and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed tasks.

2.1.1 Supervised learning

Supervised learning is the type of machine learning in which machines are trained using well-labeled training data, and on the basis of that data, machines predict the output. In supervised learning, the training data is provided to the machines, and it is the supervisor that teaches the machines to predict the output correctly. Supervised learning algorithms learn a function that maps feature vectors to labels based on example input-output pairs. The optimal scenario allows the algorithm to correctly determine the class labels for unseen instances in a "reasonable" way. The aim of a supervised learning algorithm is to find a mapping function to map the input variable (x) with the output variable (y). In supervised learning, models are trained using labeled datasets, where the model learns about each type of data. Once the training process is completed, the model is tested on the basis of test data (a subset of the training set), and then it predicts the output.

2.1.2 Unsupervised learning

Unsupervised learning is a type of algorithm that learns patterns from untagged data. The distinction between labeled and unlabeled datasets is the key difference between supervised and unsupervised learning. Supervised learning models are far more precise than their counterparts' approaches, but they require humans to be involved in the data processing procedure to ensure the labels on the information are appropriate. Unsupervised learning models work without human intervention and arrive at a structure of sorts using unlabeled data. The only human help needed is for the validation of output variables. Unsupervised learning is about getting new insights from massive amounts of new data, whereas supervised learning is about predicting outcomes[3]. Models created from supervised learning are ideal for spam detection and sentiment analysis, while unsupervised learning is perfect for looking for anomalies.

2.1.3 Clustering algorithms

K-Means Clustering is an unsupervised learning algorithm that groups unlabeled datasets into different clusters. According to AndreyBu, K-means groups similar data points together to discover underlying patterns. K-means clustering is a method of vector quantization that aims to partition n observations into k clusters, where each observation belongs to the cluster with the nearest mean (cluster center or cluster centroid). This results in a partitioning of the data space into Voronoi cells. A cluster is a collection of data points aggregated together because of certain similarities. The K-means algorithm identifies k centroids and allocates every data point to the nearest cluster. K-means clustering aims to partition observations into k (n) sets so as to minimize the within-cluster sum of squares (WCSS). This algorithm clusters data into different groups and allows us to discover the categories of groups in the unlabeled dataset on our own without any training [4]. XGBoost is a popular machine learning algorithm that can be used for classification and regression problems alike. XGBoost is built on top of a gradient boosting framework. It optimizes the model when making predictions, using weak learners.

XGBoost started as a research project by Tianqi Chen as part of the Distributed (Deep) Machine Learning Community (DMLC) group. It was soon integrated with Scikit-learn for Python users and with the caret package for R users, and is also available on OpenCL for FPGAs.

2.2 Deep Learning

Deep learning is a modern variation of machine learning that uses multiple layers and a no polynomial activation function [5]. It is permitted to deviate widely from biologically informed connectionist models. Deep learning uses multiple layers to progressively extract higher-level features from the raw input. Deep learning achieves recognition accuracy at higher levels than ever before, helping consumer electronics meet user expectations and improving safety-critical applications. A neural network with at least three layers that divides problems into data levels and then solves them is replaced by DL algorithms. The main difference is that deep learning models don't need data with a bunch of relevant features. All they need is raw data, so the algorithm can figure out what's important on its own. With more and more data being used for training, DL models are getting better.

2.2.1 Deep generative models

Generative models aim to learn the true data distribution of a training set and then generate new data points with some variations. Neural networks are used to approximate the true data distribution [5]. Variational autoencoders (VAE) and Generative Adversarial Networks (GAN) are two commonly used and efficient approaches to deep learning. Autoencoders are feedforward neural networks that compress the input into a lower-dimensional code and reconstruct the output from this representation. Autoencoders leverage neural networks for the task of representation learning, by forcing the input through a bottleneck that compresses the knowledge representation of the original input. Autoencoders use convolutional blocks followed by pooling modules to compress an input image into a compact section called the bottleneck [6]. Variational autoencoders are probabilistic generative models that use neural networks only as a part of their overall structure. They are typically trained together with the reparameterization trick. A variational autoencoder is a generative model with a prior and noise distribution, respectively, that is usually trained using the Expectation-Maximization meta-algorithm. However, a variational autoencoder uses a neural network as an amortized approach to jointly optimize across data points.

3 RELATED WORKS

3.1 Generative Adversarial Networks

According to [7], a random noise vector is used by the generator to construct the fake images. The generator makes fake data to pass to the discriminator, and the discriminator sees real training data and predicts if the data it receives is real or fake. A novel game-theoretic neural architecture with two adversarial "players" competing in a minimax game (1) was presented by [2]. Approximating intractable probabilistic calculations that occur in maximum likelihood estimation with complex functions (such as in the VAE) is one problem that their proposed architecture circumvents. A generative model is pitted against an adversary in the GAN framework: a discriminative model that learns to distinguish between real data and samples from the generative model [8]. Both parties are driven to improve their methods as a result of this competition until the fake data points are indistinguishable from the real data points.

3.2 GANS on Tabular Data

A vast majority of the GANs literature focuses on images, but tabular data are the most common data source, and there are some scientific papers that address the topic of tabular data generation. GANs for tabular data face a number of additional challenges when generating data, including transforming the data to be suitable for GAN use, handling multiple data types at once, and performing a deterministic inverse transformation of the generated data [1]. Thus, this involves a few more steps than discriminative models and GANs for image data. [9] No distinction is made between nominal and ordinal data, so, for instance, gender and education will both have the same encoding despite the fact that education obviously has a hierarchy with higher and lower levels[10].

3.3 Vanilla GAN

The Vanilla GAN is the simplest type of GAN, made up of the generator and discriminator, which uses

multi-layer perceptron to generate and classify images. A generative adversarial network (GAN) is a class of machine learning frameworks that competes with another neural network to generate new data with the same statistics as the training set. It can be used for unsupervised learning, semi-supervised learning, fully supervised learning, and reinforcement learning. The generator's task is to match the reference distribution as closely as possible, and the discriminator's task is to output a value close to 1. Generative modeling is an unsupervised learning task in machine learning that involves automatically discovering and learning the regularities or patterns in input data in such a way that the model can be used to generate or output new examples. GANs consist of a generator and a discriminator model. The generator model attempts to fool the discriminator model, while the discriminator model attempts to identify fakes. The Discriminator is a neural network that identifies real data from fake data created by the Generator [8]. Nevertheless, GANs are difficult to train, and training faces two major problems, namely mode collapse, and non-convergence. One feasible method to make GAN solve these two challenges is to redesign the network architecture to get a more powerful model. The original GAN is implemented in this paper.

$$\min_G \max_D \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_Z(z)}[1 - \log D(G(z))] \tag{3.1}$$

3.4 Wasserstein GAN

The Wasserstein GAN (WGAN) is an adaptation of the vanilla GAN, proposed by Arjovsky et al. [11] in 2017. They identified many of the problems discussed in the above sections with the original architecture and tried to improve it using the loss function, which measures distance, which is the 1-Wasserstein distance. This seemingly small change results in a model that not only trains more stable but also suffers less from mode collapse, and the end result is often better. Wasserstein metric provides smoother gradients, even when the generator provides very poor samples. This is in stark contrast to the gradients provided by the vanilla GAN discriminator, which provides no gradients if the real and fake data distributors are far apart from each other [12].

3.4.1 Wasserstein-1 metric

The earth mover's distance is the minimum 'cost' of turning one pile into another and was first formalized by Gaspard Monge in 1781. One can best explain it as having two piles of dirt, and the cost of turning one pile into the other is parallel to the amount that needs to be moved times the distance needed to travel. The Wasserstein distance was coined by R. L. Dobrushin in 1970, but was first defined by Leonid Kantorovich in 1939 in the context of optimal transport planning of goods and materials. The Wasserstein distance is the minimum work needed to transform one probability distribution into another on a given metric space. The discriminator is mainly used to provide feedback for the generator, and can be defined using a different criterion than Jensen-Shannon divergence. The 1-Wasserstein metric is defined as:

$$W(p_r, p_g) = inf_{\gamma\epsilon\Pi(p_r,p_g)}\mathbb{E}_{(x,y)\sim\gamma}[\| x - y \|] \tag{3.2}$$

where pr and pg denote the distributions of the real and fake data, respectively. Furthermore, $\Pi(p_r p_g)$ indicates all possible joint distributions $\gamma(x, y)$ whose marginals are pr and pg respectively. Intuitively, $\gamma(x, y)$ denotes the amount of mass that needs to be transported from x to y to transform the distribution pr into pg . The earth mover's distance will be the cost associated with the optimal transport plan. Wasserstein distance has its own drawback, which is that the equation is inflexible. For example, when computing the infimum $inf_{\gamma\epsilon\Pi(p_r,p_g)}$ shown in the equation above, you must exhaust all the possible joint distributions. If you apply the Kantorovich-Rubeinstein duality, the function can be deduced to:

$$W(p_r p_g) = sup_{\|f\|L\leq 1)}\mathbb{E}_{x\sim p_r}[f(x)] - \mathbb{E}_{x\sim p_g}[f(x)] \tag{3.3}$$

authors use the weight clipping when training neural networks because this function does not work with the 1-Lipschitz constraint on its own. weight clipping formula: $w \leftarrow clip(w, -c, c)$

3.5 WGAN-GP

WGAN-GP stands for Wasserstein GAN with Gradient Penalty, which was introduced by Gulrajani et al. [13] after doing the research for the original WGAN paper. WGAN-GP is a pure artificial intelligence method for risk management systems that allows us to deal with potentially complex financial services data. WGAN-GP replaces the weight clipping method used to enforce Lipschitz continuity on the critic with a constraint on the gradient norm of the critic. This allows for more stable training of the network. WGAN-GP was also introduced because it does not suffer from the exploding or vanishing gradients caused by the interaction between the weight constraint and the loss function when training with WGAN. The loss function is transformed from the WGAN to:

$$L_{d=\underbrace{\mathbb{E}[D(x)]-\mathbb{E}[D(G(z))]}_{\text{Original critic loss}}+\underbrace{\lambda.\mathbb{E}_{\hat{x}\sim p_{\hat{x}}}[(\parallel \nabla_{\hat{x}} D(\hat{x}) \parallel_2 -1)^2]}_{\text{Gradient penalty}}} \tag{3.4}$$

$$\text{With}\hat{x}=\epsilon x+(1-\epsilon)G(z)$$

$$\text{And}\epsilon\sim U[0,1]$$

Where λ is set to 10. The input of λ is effective on many architectures and datasets. Some additional changes, like the introduction of batch normalization, were made to the architecture. Batch normalization is frequently promoted in other research since it stabilizes training. Instead of mapping a single input to a single output, batch normalization transforms the discriminator issue into one that maps a batch of inputs to a batch of outcomes. The gradient penalty would no longer be legitimate because it is calculated with regard to each individual data point rather than the complete batch. The model still trains successfully without batch normalization, according to experiments. From [14].

3.6 Time GAN

TimeGAN is a novel GAN architecture that can model sequential data. TimeGAN is a GAN architecture proposed by Yoon and Jarret in 2019, which is able to generate realistic time-series data in a variety of different domains. [15]. The TimeGAN uses a sequence generator and a sequence discriminator, but they work on the latent space, and the discriminator uses a bidirectional recurrent network with a feedforward output layer. Time-series data is everywhere, from credit card transactions to medical records, but many reasons prevent its use. It introduces the concept of supervised loss and an embedding network to reduce the adversarial learning space's dimensionality. It is able to generate training to handle a mixed-data setting, where both static (attributes) and sequential data (features) are able to be generated at the same time[16]. TimeGAN is a framework to synthesize sequential data composed of four networks: generator, discriminator, recovery, and embedder. They are trained concurrently. It is composed of three losses: reconstruction, supervised, and unsupervised. Training phases include autoencoder training, recovery training, and embedder training. A time-series model should be able to capture the dynamic relationship between temporal variables across time, and not just focus on minimizing the error involved in multi-step sampling[17]. In this paper, the GAN approach is being used to generate time-series data in finance.

3.7 Evaluation Methods

Stavroula Bourou et al., [10] support the idea that recent advances in generative modeling have identified the need for suitable quantitative and qualitative methods to evaluate trainable models. Qualitative measures are those that are not numerical and often involve human subjective evaluation or evaluation via comparison. for example, evaluating mode drop and mode collapse and investigating and visualizing the internals of networks. Quantitative GAN generator evaluation refers to the calculation of specific numerical scores used to summarize the quality of generated images. For example, precision, recall, and F1 score; classification performance Evaluating a GAN model is not a straightforward procedure, since various metrics can lead to different outcomes.

Table 1. Gans and their limitations

Overall Gan models and their limitations		
Gan Model	How it Works	Limitations
VGan	-improved generation method unlike the earlier v.a -uses the Cross Entropy loss to train both models.	-no classes on data -mode collapse
CGan	-adds labels like 'real' or 'fake' to the training model	-doesn't really finish training before the mode collapses
WGan	-introduces a new cost function using the Wasserstein distance which has a smoother gradient. -adds noise to the training model. -provides a better learning signal to the generator.	-might also fail to converge in some cases. -in some cases it generate poor samples. -vanishing gradients.
WGan-Gp	-replaces weight clipping with a constraint on the gradient norm of the critic to enforce Lipschitz continuity. -stable training	

Statistical properties are quantities computed from values in a sample, and are used for a statistical purpose. Quantities are computed from values in a sample, and are used for a statistical purpose. A statistic may be used to estimate a population parameter, describe a sample, or evaluate a hypothesis. Different types of statistical tests can be applied on real and generated tables. Machine learning efficacy looks at how usable data is for cross domain applications. Reducing the amount of computation required to train a particular capability is one way to define algorithmic efficiency. Because they have a clearer measure of task difficulty, efficiency gains on traditional problems like sorting are easier to measure than in ML [18]. However, we can apply the efficiency lens to machine learning by maintaining constant performance. In addition, it sheds light on some relationships that may or may not be as evident in one of the two datasets.

4 EXPERIMENTATION, RESULTS AND DISCUSSION

This chapter shows synthetic data generation methods to produce substitute data closely resembling real data in data-limited situations, minimizing the need for accessing real data to make informed decisions. It is significant that the discoveries of most GAN models could be effortlessly summed up to other datasets with comparative information structures. The credit card datasets from kaggle which contains transactions made by credit cards in September 2013 by European cardholders will be implemented. And Time series synthetic data generation using TimeGAN Yahoo Stock Data. Lastly, the adult income dataset where predicts whether the income exceeds 50000 using the Barry Becker the census database.

4.1 GAN Models on Credit Card Dataset

Different GANs models were trained on this dataset; Vanilla GAN and CGAN then the WGAN and WCGAN calculating the losses from both the generator and discriminator networks as shown in the tables below. The tables below also shows how we explored discriminator combined losses and generator loss and Xgb loss using learning rate of 5e-4 and 1e-4 at an interval of every 100 training steps. Continuous data involves variables in which there are an infinite number of values between set ranges of possible values. Using the same XGBoost algorithm as for fraud detection, we can assess how realistic the generated data appears. It is quick, powerful, and requires little tuning to operate

off the shelf.The XGBoost algorithm's boosting process to help improve performance of the generated data for unbiased evaluation. A confusion matrix is a technique for summarizing the performance of a classification algorithm. After using the xgb model on real data and getting an accuracy percentage of 100; we then trained it on generated data and accuracy score was 0.981707317073. The accuracy can be visualized in the Fig. 2.

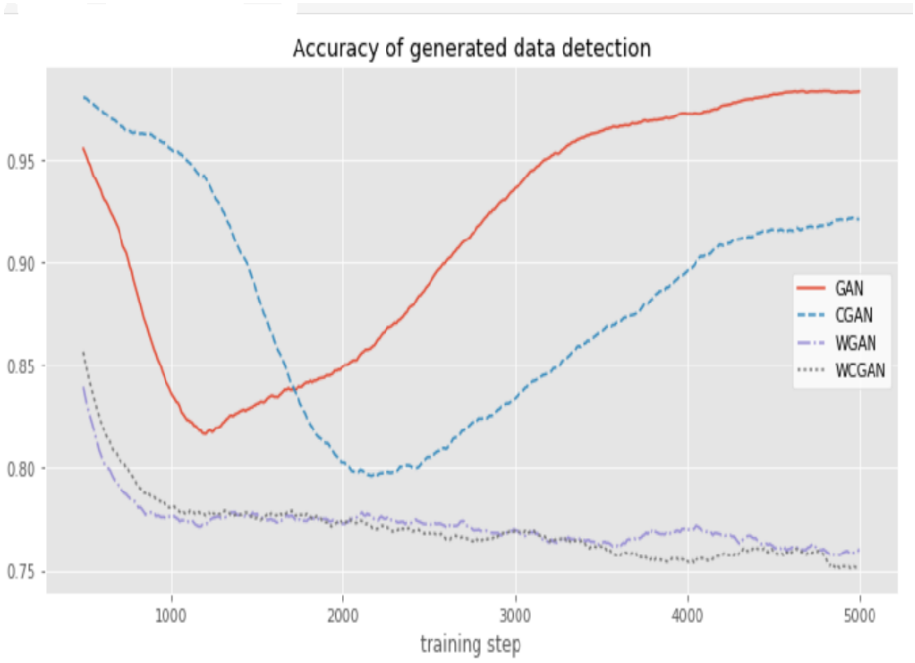

Fig. 2. Accuracy of generated data

Table 2. VGan and CGan losses

GAN Training Losses						
GAN Model	Steps Interval	Disc Real	Generator	Disc Generated	Xgb	Disc Real- Disc Generated
VGAN	100	0.7227	0.9945	0.6006	0.9878	0.1221
CGAN	100	0.3819	1.3992	0.7085	1.0000	−0.3267
VGAN	200	0.5321	0.9677	0.6388	0.9878	−0.1267
CGAN	200	0.8715	1.2793	0.6143	0.9939	0.2572
VGAN	300	0.6853	1.0075	0.7025	0.9817	−0.0172
CGAN	300	0.5973	0.8460	0.8424	0.9533	−0.2451
VGAN	400	0.7477	0.9930	0.6215	0.9472	0.1262
CGAN	400	0.6503	0.9452	0.6505	0.9634	−0.0002
VGAN	500	0.5418	1.6672	0.4173	0.9797	0.1245
CGAN	500	0.5915	1.0164	0.6773	0.9939	−0.0858

Table 3. WGan-Gp and WCGan losses

GAN Training Losses						
GAN Model	Steps Interval	Disc Real	Generator	Disc Generated	Xgb	Disc Real- Disc Generated
WGAN-GP	100	0.0615	0.0207	-0.0268	0.8252	0.0883
WCGAN	100	0.0848	0.0127	-0.0177	0.9065	0.1025
WGAN-GP	200	0.0846	-0.0112	0.0065	0.8150	0.0781
WCGAN	200	0.1311	-0.0332	0.0325	0.8455	0.0987
WGAN-GP	300	0.0841	0.0032	-0.0021	0.8191	0.0861
WCGAN	300	0.1347	-0.0387	0.0367	0.8394	0.9860
WGAN-GP	400	0.0517	0.0190	-0.0187	0.7825	0.0705
WCGAN	400	0.1077	-0.0195	0.0307	0.7866	0.0770
WGAN-GP	500	0.0322	0.0446	-0.0472	0.8110	0.0794
WCGAN	500	0.1061	-0.0059	0.0228	0.8394	0.0833

Fig. 3. Difference Critic Loss.

The main idea of the loss function is effectively measure how close the model is to the distribution, since how you measure the distance directly impacts the convergence success of the model. One can see that because the WGAN and the WCGAN adapts to the Earth Mover distance, it clearly shows their critic networks performing well since they have clean gradients. Once the discriminator reaches the optimal value, it simply provides a loss to the generator that can be trained as a neural network. During training, the Wasserstein loss leads to higher quality of the gradients to train the generator. Then we compare combined losses from both the real data and generated data. WGAN critic loss function algorithm is used to calculate the lowest critic loss and the lowest step saved. For data visualization the curve is plotted using Matplotlib as shown in the Fig. 3.

4.2 GANs on Adult Census Income Dataset

The dataset named Adult Census Income was extracted from the 1994 Census bureau database by Ronny

Kohavi and Barry Beckhavi. Predicting whether income exceeds 50K per year based on census data. The first step was loading the data and shape and print the dataframe head to visualize the categories of dataset seeing whats the values of each row and column. More exploration and data visualization is done per column like work class checking the percentages of workers in each sector, exploring levels of education, marital statuses, race and sex among other things from the adult dataset. From this dataset 25000 earn ¡= 50000 that is 75 percent of the whole data set and 25 percent earns more than 50000. This is better visualized via a histogram plot of the income. More exploring for people having capital losses and capital gain greater than zero also calculating the median value percentage. We used the heatmap plot based on random values generated by numpy using numpy random seed generator, which is a numerical value that generates a new set or repeats pseudo-random numbers. A heatmap is a plot of rectangular data as a color-encoded matrix which sho age and hours per week,Capital gain and hours per week have a strong correlation. Capital loss and age have a moderate correlation, as shown in the Fig. 4.

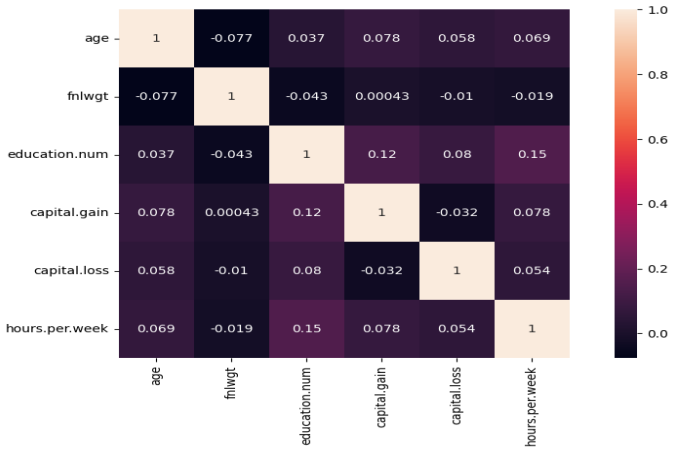

Fig. 4. Heat map plot

We used classifying models like Logistic Regression, Decision Tree ,KNN, Random Forest and Naive and also calculated their accuracy scores as shown below using the F1 classifier. A high area under the precision-recall curve represents both high recall and high precision, where high scores for both show that the classifier is returning accurate results.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN}$$

The results of the accuracy scores shows that the logistic regression and svc has the highest accuracy scores of 84 percent followed by random forest with accuracy percentage of 83. The naive bayes algorithm has the least accuracy score. Logistic regression is a very commonly used for predicting a target label from structured tabular data. Also Logistic regression works better on a dataset that is relatively small as adult dataset.

Table 4. Model accuracy scores

Accuracy scores				
Classification Model	Accuracy Score	Precision	Recall	F1-Score
Log Reg	0.84	0.87	0.93	0.90
Decision Tr	0.80	0.87	0.86	0.87
KNN	0.81	0.85	0.91	0.88
SVC	0.84	0.87	0.94	0.90
Random F	0.83	0.87	0.92	0.90
Naive Bayes	0.30	0.94	0.10	0.18

4.3 GANs on Yahoo Stock Data

The Yahoo Goldman Sachs stock dataset is used to create a complete process for predicting stock price movements. On this stock data we used GANs with LSTM since its good for predicting time series data, TimeGan trains with four models namely Generator, Discriminator, Embedder and Recovery Network For faster convergence we loaded synthetic data on Colab since you can switch the run time to GPU. Accurately predicting the stock markets is a complex task as there are millions of events and pre-conditions for a particular stock to move in a particular direction. One should note that the stock markets are not 100 random and the history of markets repeats, also markets follow people's rational behavior. we will use daily closing price from January 2010 to December 2022 and 3319 events were recorded. The data has 6 variables which are Open, High, Low, Close, Adj Close and Volume. After installing ydata-synthetic we then trained the TimeGAN synthesizer and the Fig. 5 shows the fully synthesized networks after 5000 training steps because training for longer steps like 50000 steps or even lower to 30000 kept on causing the run-time to crush even when we were using GPU run-time.

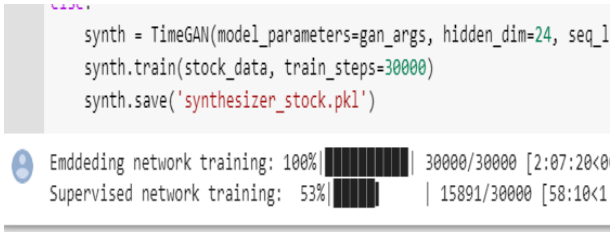

Fig. 5. 30000 training steps data generation.

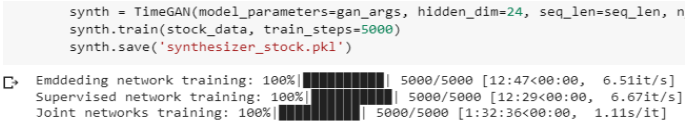

Fig. 6. 5000 training steps data generation.

Let's visualize the stock for the last ten years. The dashed vertical line represents the separation between training and test data as shown in the visualizing picture below.

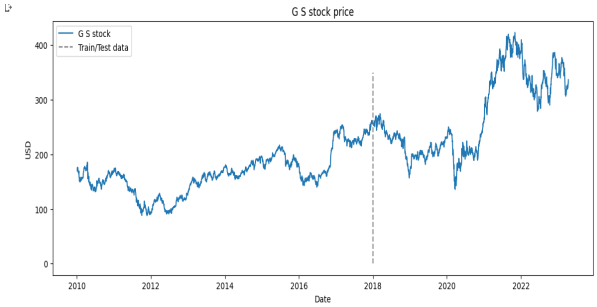

Fig. 7. Ten years data visuals

The Arima model SARIMAX(2, 1, 2) was used as a technique to remove some noise on the stock dataset. After 3343 Observations the number of Log likelihood was -9212.832. We also checked the p-values to see how strong is the effect of the residual error on the estimated parameters before using the model. And then plotted the autocorrelation plot for visualization. We then used the Xgb model to check if there is overfitting using the validation and training errors [19-22].

5 CONCLUSION AND FUTURE WORKS

Taking everything into account, the main goal for this paper was to build a model that is capable of generating high fidelity synthetic tabular data. The procedure is scientifically referred to as data augmentation. The experiments shows how we have successfully trained GAN models on three different datasets with supporting results for the accuracy and critic losses. Machine learning models have also been used to calculate accuracy, precision and F-1 scores for classification. For future improvements the concept of additional data should be considered to get more accurate and meaningful results as it was shown on the earth mover difference's curve plot between the two better models. It should also be put into consideration to try the basic xgboost for regression which is the similarity score which has λ as the regularization parameter which aids in preventing overfitting. In addition, it will be a good improvement if we take use of TGAN skip connections which allows layers to skip layers and connect to further away network residual as this process alleviates the problem of mode collapse. In Time series we should make use of Stacked autoencoders like Gaussian Error to help find new types of features that affect stock movements. And also apply XGBoost to get the best results in both classification and regression problems.

COMPETING INTERESTS

Authors have declared that no competing interests exist.

REFERENCES

[1] Saman Motamed, Patrik Rogalla, Farzad Khalvati,Data augmentation using Generative Adversarial Networks (GANs) for GAN-based detection of Pneumonia and COVID-19 in chest X-ray images,Informatics in Medicine Unlocked. 2021;27:100779.
ISSN 2352-9148
Available: https://doi.org/10.1016/j.imu.2021.100779.

[2] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks. arXiv:1406.2661 [cs, stat]. 2014. arXiv: 1406.2661.

[3] Raschka S, Patterson J, Nolet C. Machine Learning in Python: Main Developments and Technology Trends in Data Science, Machine Learning, and Artificial Intelligence. Information. 2020;11:193.
Available: https://doi.org/10.3390/info11040193

[4] Sinaga KP, Yang MS. Unsupervised K-Means Clustering Algorithm. in IEEE Access. 2020;8:80716-80727. doi:10.1109/ACCESS.2020.2988796.

[5] Dong, Shi, Ping Wang, and Khushnood Abbas. "A survey on deep learning and its applications." Computer Science Review. 2021;40:100379.

[6] Vivek Harsha Vardhan, Stanley KokSynthetic. Tabular Data Generation with Oblivious Variational Autoencoders: Alleviating the Paucity of Personal Tabular Data for Open Research; 2020.

[7] Guanyue Li, Qianfen Jiao, Sheng Qian, Si Wu1, and Hau-San Wong. High Fidelity GAN Inversion via Prior Multi-Subspace Feature Composition. The Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI-21); 2021.

[8] Mohammad Esmaeilpourxy, Nourhene Chaaliay, Adel Abusittaz, Franc¸ois-Xavier Devaillyy, Wissem Maazouny, Patrick Cardinal. RCC-GAN: Regularized Compound Conditional GAN for Large-Scale Tabular Data Synthesis; 2022.

[9] Rick Sauber-Cole, Taghi M. Khoshgoftaar. The use of generative adversarial networks to alleviate class imbalance in tabular data: a survey. Journal of Big Data. 2022;9:Article number: 98.

[10] Aggarwal, Alankrita, Mamta Mittal, and Gopi Battineni. "Generative adversarial network: An overview of theory and applications." International Journal of Information Management Data Insights. 2021;1(1): 100004.

[11] Arjovsky M, Chintala S, Bottou L. Wasserstein GAN; 2017.

[12] Manhar Walia, Brendan Tierney, and Susan McKeever Synthesising Tabular Data using Wasserstein Conditional GANs with Gradient Penalty (WCGAN-GP); 2020.

[13] Massart E. Improving weight clipping in Wasserstein GANs," 2022 26th International Conference on Pattern Recognition (ICPR), Montreal, QC, Canada. 2022; 2286-2292.
DOI: 10.1109/ICPR56361.2022.9956056.

[14] Skandarani Y, Jodoin PM, Lalande A. GANs for Medical Image Synthesis: An Empirical Study. J. Imaging. 2023;9:69.
Available: https://doi.org/10.3390/jimaging9030069

[15] Geiger A, Liu D, Alnegheimish S, Cuesta-Infante A, Veeramachaneni K. Tadgan: Time series anomaly detection using generative adversarial networks. In2020 IEEE International Conference on Big Data (Big Data) 2020 Dec 10 (pp. 33-43). IEEE.
DOI: 10.1109/BigData50022.2020.9378139.

[16] Qing Li, Xinyan Zhang, Tianjiao Ma, Dagui Liu, Heng Wang, Wei Hu. A Multi-step ahead photovoltaic power forecasting model based on Time GAN, Soft DTW-based K-medoids clustering, and a CNN-GRU hybrid neural network.
Available: https://doi.org/10.1016/j.egyr.2022.08.180

[17] Eoin Brophy, Zhengwei Wang, Qi She, Tomás Ward. Generative adversarial networks in time series: a survey and taxonomy; 2021.

[18] Ulrike Faltings, Tobias Bettinger, Swen Barth and Michael Schäfer. Impact on Inference Model Performance for ML Tasks Using Real-Life Training Data and Synthetic Training Data from GANs. Information. 2022;13:9.
Available: https://doi.org/10.3390/info13010009 .2022.

[19] Guoyun Lv, Syed Muhammad Israr, and Shengyong Qi Multi-Style Unsupervised Image Synthesis Using Generative Adversarial Nets .Digital Object Identifier; 2021.
DOI:10.1109/ACCESS.2021.3087665.

[20] Pavitha N, Sugave S. Ensemble Approach with Hyperparameter Tuning for Credit Worthiness Prediction. In2022 IEEE 3rd Global Conference for Advancement in Technology (GCAT). 2022;1-5. IEEE.
DOI: 10.1109/GCAT55367.2022.9971879.

[21] Stavroula Bourou, Andreas El Saer, Terpsichori-Helen Velivassaki, Artemis Voulkidis and Theodore Zahariadis. A Review of Tabular Data Synthesis Using GANs on an IDS Dataset. Information. 2021;12:375.

[22] Pavitha N, Atharva Bakde, Shantanu Avhad, Isha Korate, Shaunak Mahajan, Rudraksha Padole. "Brain Tumor Classification using Machine Learning; 2021.

Peer-review history:
The peer review history for this paper can be accessed here:
https://www.sdiarticle5.com/review-history/99992