Generation of Synthetic Continuous Numerical Data Using Generative Adversarial Networks

A H Azira¹, N A Setiawan² and I Soesanti³

¹,²,³Department of Electrical Engineering and Information Technology, Faculty of Engineering, Universitas Gadjah Mada, Yogyakarta, Indonesia

Abstract. Continuous numerical is a type of data which often used for unsupervised learning such as clustering. However, this valuable data often provided in a small amount because it is hard to obtain, expensive, required an expert to collect them, or not available because it contains confidential information that cannot be published. These limited data situations can be an obstacle for processing and analyzing data or restrain clustering related research in general. Therefore, there is a need to be an alternative that can replace or increase the amount of data. The proposed method is generating synthetic continuous numerical data using Generative Adversarial Networks (GANs). This study used two GAN architectures (GAN and CGAN) and focused on unlabeled continuous numerical data to provide replacement or additional data for the clustering task. The Quality of synthetic data was measured using the accuracy of the xgboost algorithm in classifying real and synthetic data. When the xgboost accuracy of perfectly realistic data is 50%, synthetic data based on CGAN achieving 63%. The result of this study shows that GAN can generate data similar enough and not significantly different from the real data.

Keywords. Clustering, continuous numerical, generative adversarial networks, synthetic data.

1. Introduction

In artificial intelligence and machine learning approaches, good quality data are always needed. Real data can be costly, challenging to obtain or contains personal, private, and confidential information that a company, programmer, software creator or research project team may not want to be published. Continuous numerical is a type of data that can consist of numbers other than whole numbers, like decimals and fractions. This type of data often found in measurement and test results, such as person height, IQ, blood pressure, temperature, prices, cost, machine lifetime and so forth. This type of data can be used for various supervised and unsupervised learning, in this case, clustering, to solve different types of problems such as predicting the lifetime of the machine for production, analyzing significant problems such as climate change, health, economic deprivation, et cetera. In business analytic aspect, this type of data often used for analyzing sales patterns, product sales, also for predicting the best product based on measurement of its components. Unfortunately, these data not always available as often related to privacy or usually provided in small numbers because it’s expensive, need a lot of time and required an expert to collect them.

These limited data situations can be an obstacle for processing and analyzing data or restrain clustering related research in general. Therefore, synthetic data can replace the real data for the purposes of processing and analysis. Synthetic data can be used as simulation data to test different types of algorithms or systems. It can augment or increase the amount of data for training machine learning model, and for generating rare data that are difficult to record such as equipment malfunctions, rare weather, and rare disease symptoms.
One promising approach in synthesizing data is Generative Adversarial Networks (GANs). Goodfellow et al. [1] were first introduced GAN as an impressive model to generate new data for various image applications. There are two essential components in GAN, generator and discriminator. The generator trained to make realistic fake data, while discriminator will try to differentiate real data distribution from the fake data distribution. Antoniou, Storkey, and Edwards [2] observe that GAN has the potential to produce a more extensive set of synthetic data for augmentation compared to standard data augmentation methods.

GAN has been proven to work very well as a generative model for image synthesis, using constrained or not, for image editing applications, even for video generation [3]. GAN was also applied to several other types of data, Nie, Narodytska, and Patel [4] proposed relational generative adversarial networks for text generation (ReGAN), a new GAN architecture for text creation, and able to outperform the latest models in terms of sample quality and diversity. Research on GAN is also implemented on data with time-series type, combination of recurrent neural networks (RNNs) with GAN architecture, applied to generate realistic, multi-dimensional, time-series medical data [5]. Olof Mogren [6] tested the generative adversarial model for continuous sequential data type and evaluated them using classical music in freely available MIDI files. C-RNN-GAN architecture (Continuous RNN-GAN) is also a combination of GAN with RNN. The results of this study [6] indicate that generative adversarial training can be carried out to train networks that model the distribution through continuous data sequences and have potential to model various other continuous sequential data types.

Using GAN for generating continuous numerical data is still relatively new. Park, Mohammadi, and Gorde [7] proposed a method called table-GAN. This method uses GAN to synthesize fake tables that are statistically similar to the real table but do not cause information leakage. In this study [7], the authors used GAN to generate numerical data and show that table-GAN exhibits the best trade-off between privacy level and model compatibility. However, the table that they used contains not only continuous but also categorical and discrete data. Previous studies in GAN for generating synthetic numerical data more focus on supervised learning such as classification and using datasets with binary classes as in [8][9], the researches focus on using GAN as an oversampling method for addressing imbalanced dataset issues.

Previous researches have shown that GAN is a method that initially focused on image data. With the potential of GAN and development of research, GAN has been used to generate various other types of data. Although there are some similarities in techniques, there are differences in implementation for every type of data. This study will use GAN to generate synthetic data in a table that only contains continuous numerical data, which have characteristics such as dimensions, arrangement, and properties that are different from other types of data. Therefore, an adjustment in GAN architecture is necessary for every different data type.

Implementation of GAN for synthetic numerical data generation, more focus on supervised learning such as improve classification performance and not so much on unsupervised learning such as clustering, which focuses on finding all kinds of unknown patterns in data. Many problems that were using clustering as a solving method, also need synthetic data for better analysis. This study will focus on unlabeled data to evaluate how GAN can generate synthetic data that can be used for an unsupervised machine learning task, in this case, clustering. With this research, the ability of GAN to generate this type of data with similar data structures can be evaluated. The obtained results can contribute to the development of data augmentation to produce high-quality synthetic data.

2. Generative Adversarial Networks

The generator generates fake data from the real data distribution. Fake data, then forwarded to a discriminator to further determine whether the data belong to the real data or not. The purpose of a generator is to deceive discriminator, while the discriminator tried to classify fake and real data accurately. Mathematical details about the relationship between these two networks formulated with cross-entropy, as shown in Equation (1).

\[
\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{data}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_z(z)} [\log (1 - D(G(z)))]
\]  

(1)
Generator G generates data from real data x with random noise variable z as input. This process return G(z) as generated data that should follow the real data distribution. Where E indicates the expectation, D(x) is the probability that discriminator D detects and classifies x as real data. D(G(z)) is the probability that D classifies the data as generated data by G.

The generator and discriminator have the opposite goal. D wants to maximize this error by making D(G(z)) approach 0, that indicates D can distinguish fake data from real data accurately, while G try to minimize it. The basic design of GAN can be seen in Figure 1.

Both generator and discriminator try to improve their functioning as the training progresses. GAN uses the cross-entropy loss from the discriminator to train the networks. Loss of real and fake data was used to update the discriminator while updating the generator need loss of fake data.

Patel, Pandya, and Shah [10] reviewed various types of GAN that have been used for generating synthetic data, such as (1) Conditional GAN (CGAN) [11], which is a GAN with additional information based on certain conditions in the system. In CGAN, generators learn to produce fake data with certain conditions or characteristics (such as labels associated with more detailed images or tags) rather than generic data from unknown noise distributions. (2) Deep Convolutional GAN (DCGAN) [12] typically contained multiple convolutional layers without max pooling or fully connected layers. In this type of GAN, convolutional neural networks (CNN) are used instead of a multilayer perceptron with some additional restrictions. (3) Laplacian Pyramid GAN (LAPGAN) [13] is used to produce images using a convolutional network cascade within the framework of the Laplacian pyramid. (4) Generative Recurrent Adversarial Networks (GRAN) [14] consisting of two main entities: encoder (convolutional neural network) for extract images of the current canvas and decoder to decide whether to update the canvas or not.

3. Methodology
3.1. Network architecture
We used two different architectures of GAN to generate synthetic data. The original GAN that used data as it is and a conditional version of the original GAN (CGAN) that adds labels to the data as additional information. Adding labels to the dataset allowing GAN to learn more specific data. Because the data is unlabeled, in order to add a label, this research uses the K-Means method to cluster the data into two classes, and add the labels based on the class distribution.

3.1.1. Generator
The generator is a network with fully connected layers that contains multiple layers, in this case, three hidden layers, inspired by the work of Cody Nash [8]. In this type of network, the neurons connected to every input and output of the layer, it makes the network can learn the relationships among its features. The goal of the generator is to deceive the discriminator and it can be trained with the prediction result of the discriminator. Therefore, the training process can be very efficiently implemented by back-propagation. We used a rectified linear unit (ReLU) as an activation function for the generator.
3.1.2 Discriminator
Discriminator in this study, also a fully connected neural network with three hidden layers, it has n input units and one output unit. This network trained to classify the synthetic data as fake and the data in the real table as real. We use the rectified linear unit (ReLU) as an activation function in the network for each layer, except for the last layer. The last layer uses a sigmoid activation to generate the probability of data being real or fake.

3.2. Loss functions
Neural networks trained by calculating the loss of the networks using loss function. It uses the gradient descent method to update its parameters (weights) to minimize the loss. In this study, the loss was calculated using the cross-entropy function by matching the discriminator output and the actual labels (real or fake) to measure how accurate the discriminator identified real and synthetic data.

3.3. Experimental environments
3.3.1 Dataset
The dataset obtained from public data with the unlabeled continuous numerical data type. This study used The WornBlade002 dataset that contains data of a worn cutting blade for Vega shrink-wrap machine from OCME [19]. The company used the data for monitoring the cutting blades’ degradation to enhance the machine's reliability and reduce unexpected downtime caused by failed cuts. This dataset has 2048 numbers of samples with eight features (such as Timestamp, pCut Motor: Torque, pCut CTRL Position controller: Lag error, pCut CTRL Position controller: Actual position, et cetera). However, we only used 20% of the data (410 samples) chosen randomly, in order to apply the GAN and CGAN architecture to limited data situations.

3.3.2 Data preprocessing
The preprocessing step includes the data cleaning process to investigate the missing value. Also, scaling and centering data by adjusting the means of all the features to 0 and the standard deviations to 1 for a better learning process of the neural network.

3.3.3 Computing environments
GAN and CGAN implemented based on TensorFlow and Keras. GAN and CGAN trained in Google Colaboratory, a cloud service based on Jupyter Notebooks with Graphics Processing Unit (GPU) and Python 3.6. Method for model compatibility test programmed using the popular sci-kit-learn machine learning library.

3.3.4 Training parameter setups
GAN and CGAN trained using Adam optimizer for 15,000 steps. 128 set as a batch-size for training, 1e-4 as learning rate, and the number of data dimensions set as the number of random dimensions. The number of each parameter obtained thorough hyperparameter tuning. The best model of training will be used to generate synthetic data.

3.3.5 Generate synthetic data
Figure 2 shows the steps to generate synthetic data.

![Figure 2. Process generation of synthetic data](image_url)
- Set Network Parameter: Create a local network with parameters such as random seed for generator, data and label dimension.
- Define Network Model: Choose the best model from training results to generate the data. The model can be from GAN or CGAN.
- Set Input Parameter for Generator: Set input parameters for generator, such as real data distribution to learned, random noise, and size of synthetic data. The generated synthetic table have the same number of data as the training dataset.

3.4 Evaluation setup

3.4.1 Qualitative quality measurement
To qualitatively evaluate the quality of the generated data, we used distribution similarity to compare the distribution between real and synthetic data. The pairwise relationships of the numerical features in the dataset for all synthetic data were plotted, observed, and compared to the relationships in the real data.

3.4.2 Quantitative quality measurement
As proposed in [15], we trained a classifier to discriminate between real and synthetic data. Xgboost, a gradient-boosted decision tree algorithm, used to evaluate how realistic the synthetic data quantitatively.
Xgboost classifier trained using half of the training data and an equal number of synthetic data. The other half of the training data and a different set of synthetic data used for testing. This orthogonal method can give some indication of how successful the generator is in generating realistic data. For a better quality of synthetic data, the xgboost algorithm should achieve an accuracy approaching 0.50 (50%), when 50% indicate the perfectly realistic data. Therefore, higher accuracy represents the data is not realistic enough as the algorithm can easily differentiate the synthetic and real data. We used the confusion matrix to calculate the accuracy, which can be seen in Equation (2).

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \times 100\%$$ \hspace{1cm} (2)

3.4.3 Hypothesis test
The purpose of this study is to generate synthetic data that can replace or represent real data. Therefore, adapted from [16], the hypothesis tested in this study is that there is no significant difference between the means of real and synthetic data. The dataset is independent and does not have a normal distribution. Therefore, the Mann-Whitney U test with a confidence interval ($\alpha = 0.05$) is applied to investigate how similar both data statistically [18]. Confidence interval means that there is 95% confidence that the conclusion of this test will be valid.

$$H_0: \text{mean of the real data} = \text{mean of synthetic data}$$

$$H_a: \text{mean of the real data} \neq \text{mean of synthetic data}$$

4. Result
Figure 3 shows the distribution of real and synthetic data from GAN and CGAN as training progresses. The data divided into 2 K-Means classes and plotted with the two features that best discriminate these two classes. GAN that does not make use of class information has their generated output all as one class, while CGAN shows their generated data by class.
The best model is a model that has xgboost accuracy closer to 50%. The best model for GAN and CGAN showed in Table 1. CGAN with 14.720 steps achieves the best accuracy (55%).

Table 1. Best GAN model for WornBlade002.

| Step   | Accuracy |
|--------|----------|
| GAN    | 13.210   | 59%      |
|        | 13.000   | 66%      |
| CGAN   | 14.720   | 55%      |
|        | 11.700   | 57%      |

After 30 times of training, it can be assumed that the data distribution is already in the normal distribution based on the central limit theory [17], the average accuracy for GAN is ±59% and ±57% for CGAN.

The accuracy of synthetic data based on GAN and CGAN decreasing and closer to 50% as a step increase. Increasing accuracy indicates that the mode collapse set in (the generator only learns a small subset of the possible realistic modes). Flat accuracy indicates that the network stops the learning process. The accuracy of synthetic data shown in Figure 4.
Figure 4. Accuracy of synthetic data for the WornBlade002 dataset

Synthetic data was generated using the model that has the best accuracy. Because the model is saved for every 100 steps, the model used is CGAN with 11,700 steps, with accuracy 57%. A comparison of the distribution of real and synthetic data can be seen in Figure 5.

Figure 5. Distribution of real and synthetic data

Quantitatively, the quality of synthetic data is measured based on the accuracy of the classifier in distinguishing real data and synthetic. The accuracy of the classifier can be determined through calculations using a confusion matrix, the results can be seen in Table 2.

Table 2. Confusion matrix from classifier

|       | Pred 0 | Pred 1 |
|-------|--------|--------|
| True 0 | TP = 117 | FN = 88 |
| True 1 | FP = 60  | TN = 145 |

Based on the confusion matrix, the accuracy of the classifier to distinguish real data from synthetic data is 63%. The accuracy obtained is not much different from the accuracy of the model trained based on GAN. Therefore, a better model produced more realistic synthetic data.
Mann-Whitney U test used to test the hypothesis. The p-value must be equal or higher than α to accept $H_0$. All the comparison of means in every feature of dataset able to accept $H_0$. Table 3 shows the results of hypothesis test.

| Feature                                         | p-value | $H_0$     |
|-------------------------------------------------|---------|-----------|
| Timestamp                                       | 0.991   | Accepted  |
| pCut Motor: Torque                              | 0.999   | Accepted  |
| pCut CTRL Position controller: Lag error        | 0.856   | Accepted  |
| pCut CTRL Position controller: Actual position  | 0.099   |Accepted   |
| pCut CTRL Position controller: Actual speed     | 0.432   | Accepted  |
| pSvolFilm CTRL Position controller: Actual position | 0.499   | Accepted  |
| pSvolFilm CTRL Position controller: Actual speed | 0.474   | Accepted  |
| pSvolFilm CTRL Position controller: Lag error   | 0.704   | Accepted  |

5. Discussion

5.1. Data distribution

As shown in Figure 3, at step 0, all of the synthetic data show the normal distribution of the random input fed to the generator. As the training step increase, GAN and CGAN architecture start to learn the shape and range of the real data. CGAN architecture faster than GAN at spreading out and approaching the distribution of each class of data. However, CGAN needs more time for the training process. The average time for training CGAN is ± 9 minutes 11 seconds, while GAN is ± 8 minutes 11 seconds. Qualitatively, synthetic data can resemble real data. It was concluded based on the similarity between the distribution of real data and the synthetic data. Figure 5 shows that GAN architectures can generate synthetic unlabeled numerical data that have a similar distribution with the real data.

5.2. Xgboost accuracy

As shown in Figure 4, the xgboost accuracy for GAN and CGAN synthetic data decreasing as soon as the training begins. These architectures do not experience the mode collapse as the accuracy keeps decreasing. As long as the network continues to learn and it does not plateau, then further training may help to generate better data. Table 1 depicts the best accuracy of the GAN and CGAN model from whole training steps and every 100 steps saved. As can be seen in Table 1, CGAN not only achieves more realistic data faster but also better than GAN. It is important to point out that, in general, the models obtained good accuracy as it's not too far from 50%. It indicates that the generated synthetic data possibly have good quality.

For generating image data [3], the convolutional layer often used in GAN architecture, as image data have the spatial structure. However, continuous numerical data doesn’t have any spatial structure among the variables, therefore convolutional networks, not the best choice. Different data type needs different adjustment in architectures in order to generate synthetic data.

5.3. Mann-Whitney U Test Score

From the hypothesis testing, the p-value is higher than 0.05 for entire features, which means $H_0$ is accepted for all of the cases. Therefore, there are no significant differences between the means of real data and synthetic data based on GAN. However, synthetic data generation models do not recreate the real data exactly. Therefore, any analysis of synthetic data needs to be verified on the real dataset.
6. Conclusion and Future Work

GAN has proven to work very well as a generative model for images and other types of data. Based on this study, it can be proven that GAN can also be a suitable method for generating unlabeled continuous numerical data with the similarity between synthetic and real data distribution, and accuracy of synthetic data reaches 63% while the perfect ideal accuracy is 50%. CGAN, with additional information in the dataset, can achieve more realistic data distribution faster and better than original GAN, although with the slower training process. Synthetic data are also capable of supporting the clustering process by providing realistic and not significantly different synthetic data. Therefore it can replace or increase the real data.

Although this study only used one dataset, we can evaluate the ability of GAN to produce this type of data. This study used a dataset that has high dimensions. Therefore an experiment to use this architecture to generate data from another dataset with similar characteristics such as unlabeled numerical data with higher or lower dimensions could be interesting. Another interesting possibility is to experiment with other GAN variations, such as Wasserstein GAN (WGAN).

References

[1] Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, and Bengio Y 2014 Generative adversarial nets. In Advances in neural information processing systems (pp. 2672-2680).
[2] Antoniou A, Storkey A, and Edwards H 2017 Data augmentation generative adversarial networks. arXiv preprint arXiv:1711.04340.
[3] Wu X, Xu K, and Hall P 2017 A survey of image synthesis and editing with generative adversarial networks. Tsinghua Science and Technology, 22(6), pp.660-674.
[4] Nie W, Narodytska N, and Patel A 2018 RelGAN: Relational generative adversarial networks for text generation.
[5] Esteban C, Hyland SL, and Rätsch G 2017 Real-valued (medical) time series generation with recurrent conditional gans. arXiv preprint arXiv:1706.02633.
[6] Mogren O 2016 C-RNN-GAN: Continuous recurrent neural networks with adversarial training. arXiv preprint arXiv:1611.09904.
[7] Park N, Mohammad M, Gorde K, Jajodia S, Park H, and Kim Y 2018 Data synthesis based on generative adversarial networks. Proceedings of the VLDB Endowment, 11(10), pp.1071-1083.
[8] C Nash “Create Data from Random Noise with Generative Adversarial Networks,” Toptal Engineering Blog, 14-Nov-2017. [Online]. Available: https://www.toptal.com/machine-learning/generative-adversarial-networks. [Accessed: 28-Jul-2019].
[9] Tanaka FH and Aranha C 2019 Data Augmentation Using GANs. arXiv preprint arXiv:1904.09135.
[10] Patel MJ, Pandya MS, and Shah V 2018 Review on Generative Adversarial Networks. International Journal of Technical Innovation in Modern Engineering & Science (IJTIMES). 4(7) 1230-1235.
[11] Mirza M and Osindero S 2014 Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784.
[12] Radford A, Metz L, and Chintala S 2015 Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434.
[13] Denton EL, Chintala S, and Fergus R 2015 Deep generative image models using a laplacian pyramid of adversarial networks. Neural Information Processing Systems (pp. 1486-1494).
[14] Im DJ, Kim CD, Jiang H, and Memisevic R 2016 Generating images with recurrent adversarial networks. arXiv preprint arXiv:1602.05110.
[15] Lopez-Paz D and Oquab M 2016 Revisiting classifier two-sample tests. arXiv preprint arXiv:1610.06545.
[16] Chokwitthaya C, Yimin ZH, Mukhopadhyay S, and Jafari A 2019 Applying the Gaussian Mixture Model to Generate Large Synthetic Data from a Small Data Set.
[17] Fischer H 2010 A History of The Central Limit Theorem: From Classical to Modern Probability Theory. Springer Science & Business Media.

[18] Takagi, H. 2015. Statistical Tests for Computational Intelligence Research and Human Subjective Tests. Japan

[19] OCME,. Shrink-wrap packers vega. 2017. URL: http://www.ocme.com/en/our-solutions/secondary-packaging/