M.S. THESIS

Generating synthetic data with Inferential Wasserstein Generative Adversarial Network

Inferential Wasserstein Generative Adversarial Network를 이용한 합성 데이터 생성

BY

KIM SEUNG-JONG

February 2023

DEPARTMENT OF STATISTICS
COLLEGE OF NATURAL SCIENCE
SEOUL NATIONAL UNIVERSITY

M.S. THESIS

Generating synthetic data with Inferential Wasserstein Generative Adversarial Network

Inferential Wasserstein Generative Adversarial Network를 이용한 합성 데이터 생성

BY

KIM SEUNG-JONG

February 2023

DEPARTMENT OF STATISTICS
COLLEGE OF NATURAL SCIENCE
SEOUL NATIONAL UNIVERSITY

Generating synthetic data with Inferential Wasserstein Generative Adversarial Network

Inferential Wasserstein Generative Adversarial Network를 이용한 합성 데이터 생성

지도교수 박 병 욱

이 논문을 이학석사 학위논문으로 제출함

2022년 10월 서울대학교 대학원

통계학과

김 승 종

김승종의 이학석사 학위 논문을 인준함

2023년 1월

위 원 장:	오희석	(인)
부위원장:	박병욱	(인)
위 원:	박건웅	(인)

Abstract

Today, the importance of generating synthetic data has arisen more than ever. It comes from the fact that although there are a lot of datas these days, regardless of big and small, the risk of privacy leakage also arises from there. Therefore, to achieve the initial goal of analyzing the data while preserving privacy of the person of the origin of the data, generating synthetic data should come into place. For synthetic data generation, many generative models have been used, including Generative Adversarial Network, or GAN. In this paper, we use inferential Wasserstein Generative Adversarial Network, or iWGAN, which is an improvement of GAN, to generate synthetic data and see how it performs.

keywords: Deep Learning, Generative Adversarial Network(GAN), WGAN, iWGAN, Synthetic data, Data generation

student number: 2021-20687

Contents

List of Tables

List of Figures

Chapter 1

INTRODUCTION

1.1 Introduction

Synthetic data is an artificially generated data which mimics the data from the real world. The concept of the synthetic data started from Rubin (1993), and have developed in variety as the progressions were made in theory and technology.

There are a lot of points indicating that generating synthetic data is beneficial. First of all, as datasets of sensitive origin have been collected more and more, possible risk of privacy disclosure came into an issue. To avoid this, instead of directly utilizing the original data, using synthetically generated data would be a remedy. Also, if the data is biased, generating additional data and augmenting it to the original data would give the better result. For example, if the dataset consists of male and female with female numbers significantly bigger than the male, then it would make sense to generate more male data to make more accurate comparison within the male and female.

Including reasons mentioned above, the need for good synthetic data has been increasing nowadays. From here, the 'good' synthetic data indicates the data

which avoids or augments the possible downside of using the original data, but meanwhile also is able to give the same, or at least similar result of analysis with working with the original data.

To generate synthetic data by deep-learning based models, generative models have been used, such as Generative Adversarial Network (GAN) or Variational Autoencoder (VAE). In particular, GAN and its variations are mostly utilized this days. In this paper, we seek to utilize iWGAN, recently proposed augmented version of GAN and VAE simultaneosly by Chen et al., to generate the synthetic data.

In Chapter 2, there will be an introduction about GAN and Wasserstein GAN, or WGAN, which is an upgrade of GAN and a base of iWGAN. Additionally, explanation about Autoencoder GAN going to be provided. In Chapter 3, there will be a description about iWGAN. In Chapter 4, there will be an application of the main algorithm iWGAN to practical synthetic data generation. Summary and discussion will be given in Chapter 5.

Chapter 2

BACKGROUND

2.1 Generative Adversarial Network

Generative Adversarial Network, or GAN, is the deep generative model first proposed by Goodfellow et al.(2014) The idea of GAN is to train the two neural networks simulatenously. Discriminator network D is trained to discriminate whether the sample x is from the real data or the generator network G. Also, the generator is trained to create the sample which is closer to the real data. This process could be explained as the two-player minimax game by D and G with the value function V(G,D):

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

From this process, the idea of a data generation comes into the place. As the generator gets better by training, it would be able for the generator to create the sample which is close to the real data. Also, since the generator generates the data without having an direct interaction with the real dataset, such dataset wouldn't have an possible risk that the original data might have, such as privacy risk. However, there are some downsides of vanilla GAN. According to

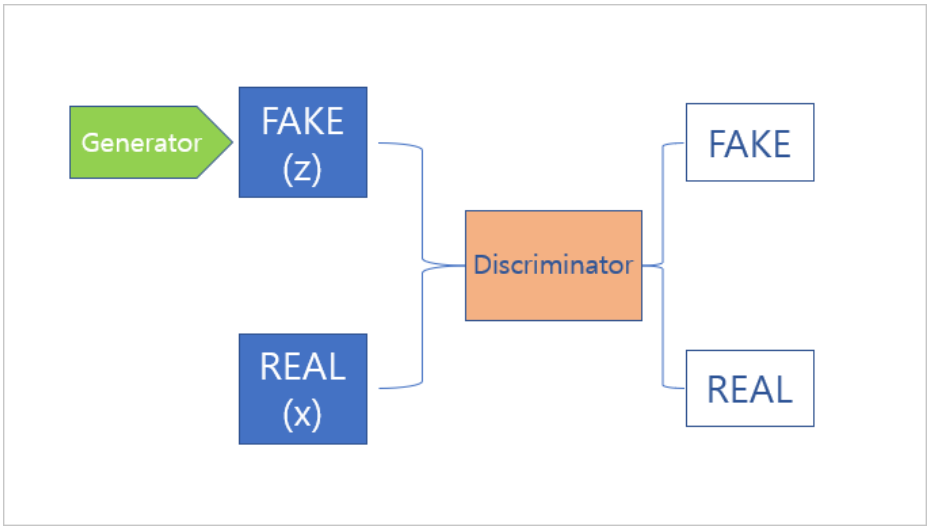

Figure 2.1: Illustration of GAN

Arjovsky and Bottou (2017), the vanilla GAN tends to have worse updates and massively unstable optimizations.

There have been many variations of GANs proposed, such as Conditional GAN (Mirza et al., 2014), CycleGAN (Zhu et al., 2017), Pix2pix (Isola et al., 2017), StyleGAN2 (Karras et al., 2019). and WGAN (Arjovski et al., 2017) Since the main topic in this paper is iWGAN, brief explanation about WGAN is going to be provided first.

2.1.1 Wasserstein GAN (WGAN)

Wasserstein GAN, or WGAN, is the deep-learning model first proposed by Arjovsky et al.(2017). WGAN is the most popular method, which uses Wasserstein distance metric to optimize the generating distribution, which is defined as

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \sim \Pi(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(x,y)\sim\gamma}[\|x - y\|],$$

where $\Pi(\mathbb{P}_r, \mathbb{P}_g)$ denotes the set of all joint distributions $\gamma(x, y)$ whose marginals are respectively $\mathbb{P}_r$ and $\mathbb{P}_g$. Such distance is alternatively defined as the Earth-Mover (EM) distance.

Since the infimum defined above is highly intractable, at Arjovky et al, the alternate form

$$W(\mathbb{P}_r, \mathbb{P}_g) = \sup_{\|f\|_L v \leq 1} \mathbb{E}_{x \sim \mathbb{P}_r}[f(x)] - \mathbb{E}_{x \sim \mathbb{P}_\theta}[f(x)],$$

by using Kantorovich-Rubinstein duality. Such alternation is also used in defining iWGAN model.

Wasserstein distance is proposed to substitute other used distances, such as Total Variation (TV), Kullback-Leibler (KL), and Jensen-Shannon (JS) divergence.

Wasserstein distance possesses the characteristic of continuity and differentiability almost everywhere, by the Theorem 1 from Arjovsky et al. From such property, it is possible to train the WGAN critic until optimality. The critic does not saturate and converges to a linear function that gives clean gradients everywhere. Also, theoretically the mode collapsing from vanilla GAN does not happen because the critic could be trained until optimality. Also, compared with vanilla GAN, WGAN has a better stability. However, mode collapse problem still occurs in WGAN in practice, and there is no metric clearly defined to detect the convergence.

2.1.2 Autoencoder GAN

Autoencoder is a process such that it passes through the input data into the encoder network to bottled hidden layer, or latent variable. Then, the latent variable is passed through the decoder network once again to get an output. During the process, the difference between the input data and the output data is the loss value.

The difference between vanilla GAN and Autoencoder GAN is that besides with the generator G, there also exists the encoder $Q; \mathcal{X} \rightarrow \mathcal{Z}$ which sends the data $x \in \mathcal{X}$ into the latent space $\mathcal{Z}$. Also, in Autoencoder GAN, the discriminator work as the decoder from the autoencoder network. The Wasserstein Autoencoder, which is a foundation for the iWGAN, proposes an encoder Q which minimizes the reconstruction error:

$$\inf_{Q \in \mathcal{Q}} \mathbb{E}_X \| X - G(Q(X)) \|$$

According to Chen et al., autoencoder generative model must satisfy 3 conditions simultaneously. First one is the good generator condition, which indicates that the fake generated data $G(Z)$ has the similar distribution with P_X. Second one is the meaningful encoding condition, which is that the $Q(X)$ has the similar distribution with the latent variable Z. And the final one is the small reconstruction error condition, which indicates that the original data X and $G(Q(X))$ has the small difference.

Chapter 3

ALGORITHM

3.1 iWGAN

Inferential Wasserstein GAN, or iWGAN (Chen et al., 2021) is a variant of the WGAN, which takes an advantage of both the WGAN and WAE. iWGAN jointly learns an encoder network which maps the samples from the data space to the latent space, and a generator network which maps the latent variables to the data space. The iWGAN defines the divergence between P_X and $P_{G(Z)}$ by

$$\overline{W}_1(P_X, P_{G(Z)}) = \inf_{Q \in \mathcal{Q}} \sup_{f \in \mathcal{F}} [\mathbb{E}_X \|X - G(Q(X))\| + \mathbb{E}_X \{f(G(Q(X))\} - \mathbb{E}_Z \{f(G(Z))\}]$$

where $\mathcal{F}$ is a set of all bounded 1-Lipschitz functions. According to the Theorem 1 of Chen et al., the encoder Q which satisfies the conditions for autoencoder generative model which was mentioned from previous chapter exists, and such $Q(X)$ follows multivariate standard normal distribution.

The objective is to find (G, Q, f) which minimizes $\overline{W}_1(P_X, P_{G(Z)})$. In practice, $\widehat{\overline{W}}_1(P_X, P_{G(Z)})$, the empirical version of $\overline{W}_1(P_X, P_{G(Z)})$ is minimized, where the expectations are substituted by the empirical average on the observed

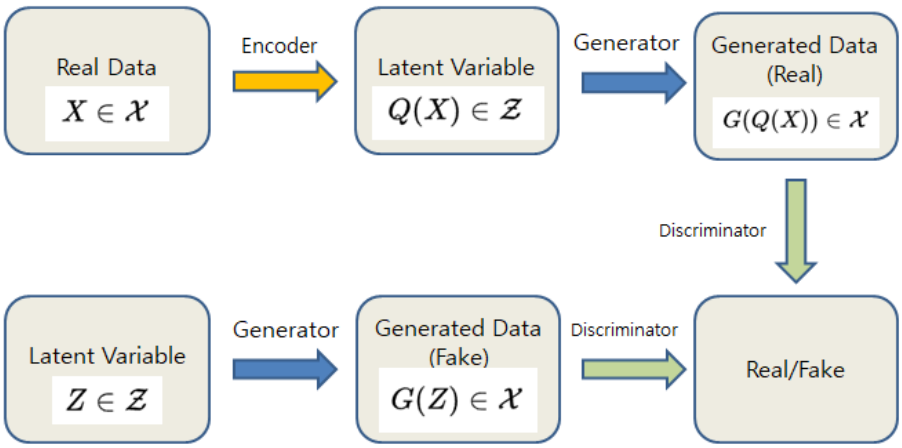

Figure 3.1: Illustration of iWGAN

data for X, and a random sample of standard normal random variables for Z. According to the Theorem 3 of Chen et al., 1-Wasserstein distance between P_X and $P_{G(Z)}$ can be upper bounded by $\widehat{\overline{W}}_1(P_X, P_{G(Z)})$, and Rademacher complexity of $\mathcal{F}$.

With the regularization term added, the optimization problem becomes

$$\min_{G\in\mathcal{G},Q\in\mathcal{Q}} \max_{f\in\mathcal{F}} [\hat{\mathbb{E}}_{obs}\|x - G(Q(x))\| + \hat{\mathbb{E}}_{obs}\{f(G(Q(x)))\} - \hat{\mathbb{E}}_z\{f(G(z))\}$$

$$-\lambda_1 J_1(f) + \lambda_2 J_2(Q)]$$

with J_1 and J_2 be regularization term for f and Q respectively. Since assumption is given that f is 1-Lipschitz, in Chen et al, the gradient penalty is defined as $J_1(f) = \mathbb{E}_{\hat{x}}\left\{(\hat{\nabla}_{\hat{x}} f(\hat{x})\|\|_2 - 1)^2\right\}$ as given in Gulrajani et al. Also, from the normality of $Q(X)$, the maximum mean discrepancy (MMD) proposed by Gretton et al., is used. It is defined as $J_2(Q) = \text{MMD}_k(P_{Q(X)}, P_Z)$

$$= \frac{1}{n(n-1)}\sum_{l\neq j} k(z_l^i, z_j^i) + \frac{1}{n(n-1)}\sum_{l\neq j} k(Q(x_l^i), Q(x_j^i)) - \frac{2}{n^2}\sum_{l,j} k(z_l^i, Q(x_j^i))$$

Algorithm 1 The training algorithm of iWGAN (Chen et al., 2021)

Require: The regularization coefficients λ_1 and λ_2, tolerance for duality gap ϵ_1, tolerance for loss ϵ_2, and running steps n

initialization $(G^0.Q^0, f^0)$

while DualGap$(G^i.Q^i, f^i) > \epsilon_1$ or $L(G^i.Q^i, f^i) > \epsilon_2$ **do**

for $t = 1, ..., n$ **do**

Sample real data $\{x_k^i\}_{k=1}^n \sim P_X$, latent variable $\{z_k^i\}_{k=1}^n \sim P_Z$, and $\{\epsilon_k^i\}_{k=1}^n \sim U[0,1]$

Set $\hat{x}_k^i \leftarrow \epsilon_k x_k^i + (1-\epsilon_k) G^i(z_k^i)$, $i = 1, ..., n$ for the calculation of gradient penalty.

Calculate: $L^i = L(G^i, Q^i, f^i)$, $J_1(f^i) = (\|\nabla_{\hat{x}^i} f^i(\hat{x}^i)\|_2 - 1)^2$ and

$$-\nabla_f L^i = \nabla_f \left[\frac{1}{n} \sum_{k=1}^{n} \left(f^i(G^i(z_k^i)) - f^i(G^i(Q^i(x_k^i))) + \lambda_1 J_1(f^i) \right) \right]$$

Update f by Adam: $f^{i+1} \leftarrow f^i + Adam(-\nabla_f L^i)$

end for

for $t = 1, ..., n$ **do**

Sample real data $\{x_k^i\}_{k=1}^n \sim P_X$, latent variable $\{z_k^i\}_{k=1}^n \sim P_Z$

Calculate: $L'^i = L(G^i, Q^i, f^{i+1})$, $J_2(Q^i)$ and

$$\nabla_{G,Q} L'^i = \nabla_{G,Q} \left[\frac{1}{n} \sum_{k=1}^{n} \left(\|x_k^i - G^i(Q^i(x_k^i))\| + f^{i+1}(G^i(Q^i(x_k^i))) - f^{i+1}(G^i(z_k^i)) + \lambda_2 J_2(Q^i) \right) \right]$$

Update G, Q by Adam: $(G^{i+1}, Q^{i+1}) \leftarrow (G^i, Q^i) + Adam(\nabla_{G,Q} L'^i)$

end for

DualGap$(G^{i+1}.Q^{i+1}, f^{i+1}) = L(G^i, Q^i, f^{i+1}) - L(G^{i+1}, Q^{i+1}, f^{i+1})$

$i \leftarrow i + 1$

end while

We might suggest an alternative loss for the optimization problem, by applying Jenson-Shannon divergence once again. This will give an alternate algorithm as given. As we look through real data examples at section 4, training at Jenson-Shannon divergence type loss show similar, or better performance. For later studies, we will demonstrate this theoretically.

Algorithm 2 The training algorithm of iWGAN (Chen et al., 2021)

Require: The regularization coefficients λ_1 and λ_2, tolerance for duality gap ϵ_1, tolerance for loss ϵ_2, and running steps n

initialization $(G^0.Q^0, f^0)$

while DualGap$(G^i.Q^i, f^i) > \epsilon_1$ or $L(G^i.Q^i, f^i) > \epsilon_2$ **do**

for $t = 1, ..., n$ **do**

Sample real data $\{x_k^i\}_{k=1}^n \sim P_X$, latent variable $\{z_k^i\}_{k=1}^n \sim P_Z$, and $\{\epsilon_k^i\}_{k=1}^n \sim U[0,1]$

Set $\hat{x}_k^i \leftarrow \epsilon_k x_k^i + (1-\epsilon_k) G^i(z_k^i)$, $i = 1, ..., n$ for the calculation of gradient penalty.

Calculate: $L^i = L(G^i, Q^i, f^i)$, $J_1(f^i) = (\|\nabla_{\hat{x}^i} f^i(\hat{x}^i)\|_2 - 1)^2$ and

$$-\nabla_f L^i = \nabla_f \left[\frac{1}{n} \sum_{k=1}^{n} \left(\log \left\{ f^i(G^i(z_k^i)) \right\} + \log \left\{ 1 - f^i(G^i(Q^i(x_k^i))) \right\} + \lambda_1 J_1(f^i) \right) \right]$$

Update f by Adam: $f^{i+1} \leftarrow f^i + Adam(-\nabla_f L^i)$

end for

for $t = 1, ..., n$ **do**

Sample real data $\{x_k^i\}_{k=1}^n \sim P_X$, latent variable $\{z_k^i\}_{k=1}^n \sim P_Z$

Calculate: $L'^i = L(G^i, Q^i, f^{i+1})$, $J_2(Q^i)$ and

$$\begin{aligned} \nabla_{G,Q} L'^i = & \nabla_{G,Q} [\frac{1}{n} \sum_{k=1}^{n} (\|x_k^i - G^i(Q^i(x_k^i))\| + \log \left\{ f^{i+1}(G^i(Q^i(x_k^i))) \right\} \\ & + \log \left\{ 1 - f^{i+1}(G^i(z_k^i)) \right\} + \lambda_2 J_2(Q^i))] \end{aligned}$$

Update G, Q by Adam: $(G^{i+1}, Q^{i+1}) \leftarrow (G^i, Q^i) + Adam(\nabla_{G,Q} L'^i)$

end for

DualGap$(G^{i+1}.Q^{i+1}, f^{i+1}) = L(G^i, Q^i, f^{i+1}) - L(G^{i+1}, Q^{i+1}, f^{i+1})$

$i \leftarrow i + 1$

end while

iWGAN also has an advantage in the perspective of maximum likelihood estimation (MLE). Although MLE is a fundamental framework for learning models from data, it is hard to compute MLE for complex models. iWGAN is advantageous in a perspective that it provides an easier way to compute MLE. From Chen et al., surrogate log-likelihood is given as

$$\mathcal{L}(\theta;\widetilde{\theta}) = -\hat{\mathbb{E}}_{\text{obs}}\|x - G_\theta(Q_\theta(x))\| + \mathbb{E}_{\widetilde{\theta}}\|x - G_\theta(Q_\theta(x))\| - H(p(x|\widetilde{\theta})).$$

From Theorem 4 of Chen et al., this surrogate log-likelihood is an upper bound for (real) log-likelihood, and if an algorithm which is updated by maximizing the surrogate log-likelihood converges, then the solution for it is the MLE.

Chapter 4

SIMULATION

In Chen et al., application has been conducted only for the image data. In this paper, we would like to see in addition that iWGAN also makes a good generation on general table data. In this chapter, IWGAN algorithm is applicated to the real world table data, whether without any modification on the algorithm, or giving an additional shift or conditioning to the algorithm.

4.1 Numerical Data Case

We first conduct an experiment on a table which contains only numerical variables. We use the classic Iris dataset for this case, proposed by Fisher, 1936. We only use the first 4 rows of the dataset, which is Sepal.Length, Sepal.Width, Petal.Length, and Petal.Width. We have set the iteration number as 5000, and learning rate as 1e-4.

We use the R package synthpop, which is designed by Nowok et al., to compare the quality of synthesized data. We first see the similarity of the distributions of each columns of the data. From figure 1, we can see that for each columns, the distribution of the observed, or original data and the synthetic data is similar.

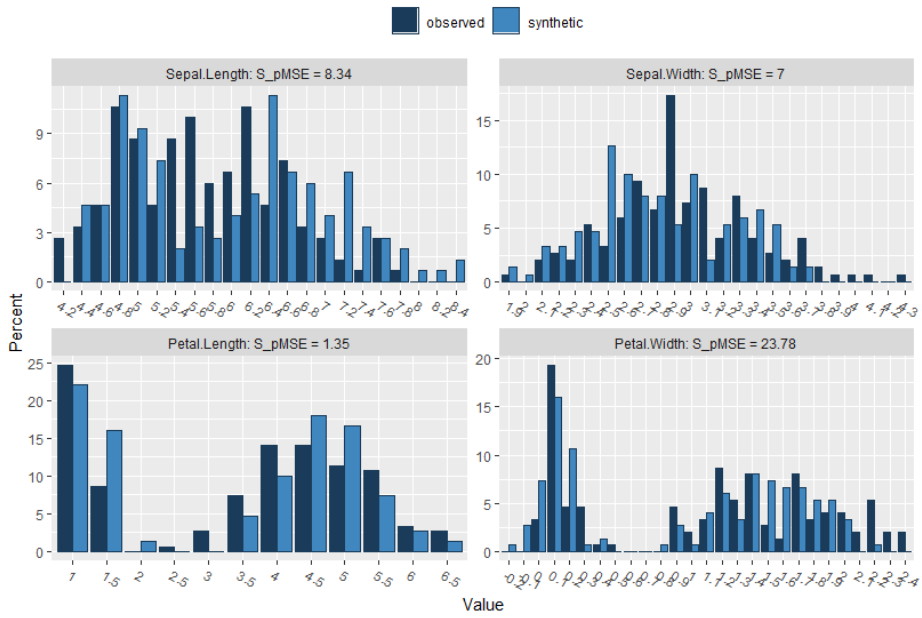

Figure 4.1: Comparison of the original iris dataset and the data synthesized by iWGAN

Also, we check pMSE, or propensity score mean square error of the synthesized data. pMSE is proposed by Woo et al., 2009, which is defined as

$$U_p = \frac{1}{N}\sum_{i=1}^{N}[\hat{p}_i - c]^2$$

where $N = n_1 + n_2$ is the total number of rows of original, n_1, and synthesized dataset, n_2, $\hat{p}_i$ is the estimated propensity score for unit i, and c is the proportion of units with synthetic data and the total merged data, or simply $c = \frac{n_2}{N}$. pMSE of this synthetic data is proposed at table 4.1. We can see that the numbers of pMSE are generally low, so we can consider that the synthetic data is well generated.

In addition, as for slight alteration, we have changed the objective by shifting

Variables	pMSE	SpMSE	df
Sepal.Length	0.013906	8.343512	4
Sepal.Width	0.011667	7.000000	4
Petal.Length	0.002248	1.348622	4
Petal.Width	0.039627	23.776323	4

Table 4.1: pMSE of synthetic data

Variables	pMSE	SpMSE	df
Sepal.Length	0.000805	0.482843	4
Sepal.Width	0.006667	4.000000	4
Petal.Length	0.005257	3.153970	4
Petal.Width	0.050512	30.307323	4

Table 4.2: pMSE of synthetic data (Jenson-Shannon type loss)

the EM distance as Jenson-Shannon divergence version, as

$$\min_{G\in\mathcal{G},Q\in\mathcal{Q}} \max_{f\in\mathcal{F}} [\hat{\mathbb{E}}_{obs}\|x-G(Q(x))\|+\log(\hat{\mathbb{E}}_{obs}\{f(G(Q(x)))\})+\log(1-\hat{\mathbb{E}}_{z}\{f(G(z))\})$$

$$-\lambda_1 J_1(f) + \lambda_2 J_2(Q)]$$

, and did the simulation under same setting. The results are shown in the figure 4.2 and the table 4.2. As we can see, the simulation worked well, or even in fact improved. Further studies are to be conducted to give theoretical background to this result.

4.2 General Case

Now, we conduct an experiment on a table with the both numerical variables and the categorical variables. We use the automobile data, from 1985 Ward's

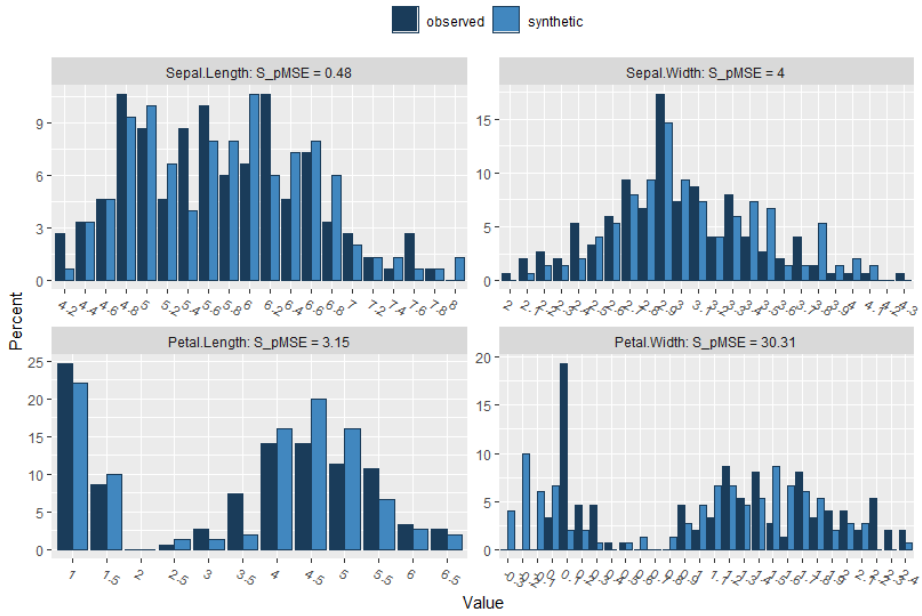

Figure 4.2: Comparison of the original iris dataset and the data synthesized by iWGAN (Jenson-Shannon type loss)

Automotive Yearbook. It has total 26 variables, with 16 numerical variables and 10 categorical variables. For the categorical variables, we have applied the Reversible Data Transforms, or RDT to transform it to numerical values. For all the categorical values, FrequencyEncoder has been applied. It transforms the data into decimals in range $[0, 1]$. This range is broken up into separate intervals for each category, where popular categories take up larger intervals. The rest of the setting is the same with numerical data case.

After the data is transformed, we generate the numerical data and transformed categorical data separately, and reunite it after generation. For the categorical variables, we have applied cross-entropy loss at generation. Then, we applied the same Synthpop package for comparison between the original data and the synthetic data. Table 4.3 is the result of the comparison.

As we can see, for most of the cases, numerical variables were generated nicely, but categorical variables are generated quite not good. It would be our goal to find out the reason that the transformed categorical data were not generated well.

Variables	pMSE	SpMSE	df
symboling.value	0.040208	32.970228	4
normalized_losses.value	0.026602	21.813405	4
make.value	0.008507	6.975610	4
fuel_type.value	0.132900	145.304426	4
aspiration.value	0.095317	104.213541	3
num_doors.value	0.167067	136.994943	4
body_style.value	0.064116	52.574906	4
drive_wheels.value	0.202063	165.691262	4
engine_location.value	0.221936	242.649713	3
wheel_base.value	0.020826	17.077354	4
length.value	0.005532	4.536585	4
width.value	0.012415	10.180592	4
height.value	0.024340	19.959130	4
curb_weight.value	0.011207	9.189905	4
engine_type.value	0.116664	127.553055	3
num_cylinders.value	0.142677	155.993044	3
engine_size.value	0.007615	6.243902	4
fuel_system.value	0.043301	35.506911	4
bore.value	0.016637	13.642011	4
stroke.value	0.009726	7.975418	4
compression_ratio.value	0.22386	18.341463	4
horsepower.value	0.012196	10.000465	4
peak_rpm.value	0.029873	24.496101	4
city_mpg.value	0.021027	17.241801	4
highway_mpg.value	0.025402	20.829268	4
price.value	0.006960	5.707317	4

Table 4.3: pMSE of synthetic data (Automobile)

Chapter 5

CONCLUSIONS

In this paper, we have discussed about how to implement recently proposed iWGAN algorithm to synthetic data generation. By theory, iWGAN is an improvement of several previously proposed GANs and VAEs, and we have seen that it has shown efficient, and stable learning compared to them. Also, in chapter 4, we managed to show that iWGAN creates synthetic data well not only for the image cases, but also the cases of tabular data. Indeed, to guarantee if the quality of the generated data is always good, there need to be more application. Also, we need to consider not only the synthesized data resembles the original data, but also need to consider the privacy of the data, which is seeing whether the synthetic data doesn't reveal the sensitive information to defined an individual. If we are able to succesfully combine the iWGAN framework into the privacy protecting methods which have been used before, we would be able to create a decent data synthesizing framework to meet the demand which has gotten bigger than ever.

For further studies, we are going to discuss about the way to improve the iWGAN algorithm with respect of data synthesizing. We will try to find a theoret-

ical foundation to incorporate already existing methods for applying GAN as data synthesizer to iWGAN algorithm, such as CTGAN proposed by Xu et al., and improved it.

Furthermore, there is a problem of heavy computation if the data to be synthesized is too large, but generative models introduced so far doesn't have solutions for it. Zhao et al. (2021) have introduced the method of data condensation to deal with this problem, by synthesizing a small batch of data which gives not only the same optimization result but also a similar path throught the optimization. If we could combine this method into in-theory effective iWGAN, we might be able to create a powerful tool for synthesizing data.

Bibliography

[1] Arjovsky, M., S. Chintala, and L. Bottou (2017), "Wasserstein Generative Adversarial Networks," in *International conference on machine learning*, pp. 214-223. PMLR.

[2] Chen, Y., Gao, Q., and Wang, X. (2022) "Inferential Wasserstein Generative Adversarial Networks," *The Journal of the Royal Statistical Society.*, vol. 84(1), pp. 83-113, February.

[3] Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio (2014). " Generative adversarial nets," in *In Advances in neural information processing systems*pp. 2672–2680.

[4] Gulrajani, I., F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville (2017). "Improved training of wasserstein gans," in *Advances in Neural Information Processing Systems*, pp.5767–5777.

[5] Nowok B, Raab GM, Dibben C (2016). "synthpop: Bespoke Creation of Synthetic Data in Rm," in *Journal of Statistical Software*, 74(11), pp. 1–26. doi:10.18637/jss.v074.i11.

[6] Tolstikhin, I., O. Bousquet, S. Gelly, and B. Schoelkopf (2018). "Wasserstein auto-encoders," in *International Conference on Learning Representations*

[7] Zhao, B., K. Mopuri. H. Bilen (2021). ”Dataset condensation with gradient matching,” in *International Conference on Learning Representations*

초 록

최근에 합성 데이터를 생성하는 것에 대한 중요성이 그 어느 때보다도 올라갔다. 이는 지금 같은 상황에서 크기에 무관하게 많은 데이터가 존재하고, 그로 인해 사생활에 대한 침해의 우려 때문에 발생하는 상황이다. 이에 따라 데이터를 분석하는 최초의 목적을 달성함과 동시에 원 데이터로부터의 사생활 침해 우려를 막기 위해서 합성 데이터 생성이 필요하다. 합성 데이터 생성을 위해 많은 생성 모델이 활용되었는데, 그 중에는 Generative Adversarial Network, 또는 GAN이 있다. 이 논문에서는 GAN의 일종인 새로 제안된 기법 inferential Wasserstein Generative Adversarial Network, 또는 iWGAN을 활용하여 합성 데이터를 생성하고, 생성이 얼마나 잘 되는지를 확인할 것이다.

주요어: Deep Learning, Genarative Adversarial Network(GAN), Wasserstein GAN, Synthetic data, Data generation

학번: 2021-20687