GANBLR++: Incorporating Capacity to Generate Numeric Attributes and Leveraging Unrestricted Bayesian Networks

Yishuo Zhang*† Nayyar Zaidi*† Jiahui Zhou‡ Gang Li*§

Abstract

Generative Adversarial Networks (GAN) models have led to a major breakthrough in data generation of various sorts. Over the years, we have seen several applications of GAN-based learning for tabular data generation as well. Very recently, GAN-based learning by incorporating Bayesian Networks (BN) as generator and discriminator – GANBLR, has shown to lead to state-of-the-art (SOTA) results for tabular data generation. Despite the impressive performance, GANBLR has an inherent weakness that it can only generate data with categorical attributes. Additionally, the model is trained and tested only with a restricted Bayesian Network. In this work, we have proposed an extension over GANBLR framework – GANBLR++, that has the capacity to generate numeric attributes, by leveraging Dirichlet Mixture Model. We also leverage unrestricted BN in GANBLR framework, and discuss how the use of unrestricted BN can lead to better quality data, as well as more interpretable model. We evaluate the effectiveness of GANBLR++ on wide range of datasets by demonstrating that it generates data of better quality as compared to existing SOTA models for tabular (numeric and categorical) data generation such as CTGAN, MedGAN and TableGAN.

1 Introduction

Generative Adversarial Networks (GAN) models have led to a massive breakthrough in data generation, especially in computer vision where data is mostly images [6]. Their application to other domains such as text mining and natural language processing where data takes the form of text and speech has resulted in generating excellent quality data as well. GAN-based models leverage deep Artificial Neural Networks (ANN) such as Convolutional Neural Networks (CNN) to exploit an apparent structure in the data to learn to generate artificial (or fake) data. However, such structure is not present in tabular datasets, hindering the application of models like CNN [1] in GAN-based models. Another challenge brought by tabular dataset is that they constitute of varying types of attributes such as numeric, ordinal, categorical, etc. Dealing with these different data types is not trivial, as each attribute type has its own peculiar traits.

Despite these challenges, the application of GAN-based models to tabular data has seen a lot of interest over the past few years, and in fact, has led to some state-of-the-art (SOTA) results for tabular data generation tasks with models such as CTGAN [9], TableGAN [11], MedGAN [8], etc. Very recently another breakthrough has been achieved where authors utilized GAN-based learning framework for training Bayesian Networks as both generator and discriminator, leading to framework known as GANBLR [13]. It was shown that GANBLR leads to massive improvements over previous state-of-the-art methods, as summarized in Table 1. Despite impressive results, GANBLR has some inherent weaknesses. First, the model can only generate categorical (nominal) attributes. Note, CTGAN can generate both categorical and continuous (numeric) attributes. How to incorporate the capacity for numeric attributes generation in GANBLR has been the motivation of this work. Secondly, despite being proposed as a general model that can incorporate any Bayesian Network (shown in Figure 1), GANBLR models is only trained and tested with restricted Bayesian Network only. The efficacy of the model with a full unrestricted Bayesian Network is still to be tested. This has been the secondary motivation for this work. In this work, we propose an extension of GANBLR model known as GANBLR++, which incorporates the capacity of generating the numerical feature through a BN, by using Dirichlet Mixture Model in the generation process. Secondly, instead of using the restricted Bayesian model such as K-Dependence Bayesian Network (KDB), we use an unrestricted BN to leverage the full benefit of using a Bayesian Network.

Numeric attributes are prevalent in tabular datasets. GANBLR handles numeric datasets by discretizing them, and then learns to produce a discretized value (bin number) instead of the actual numeric value. Though discretization offers a simple strategy to handle numeric attributes in data generation algorithm that can only process discrete features, there are still scenarios where a numeric value is expected. A typical discretization process used in GANBLR is shown in Figure 2. GANBLR

*School of I.T., Deakin University, Burwood,VIC,3125
†Equal Contribution: Yishuo Zhang and Nayyar Zaidi
‡College of Computer Science, Xi'an shiyou Univeristy
§Corresponding Author: Gang Li

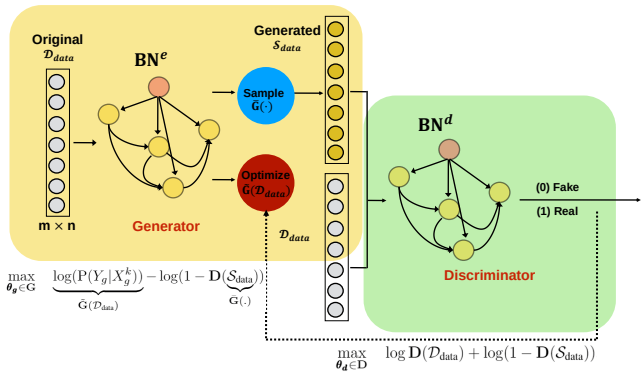

Figure 1: GANBLR Framework [13].

discretizes the data beforehand, that means that if we have a feature X, it gets allocated a value from 1 to 6 if we chose to have 6 bins. GANBLR learns to sample the values from 1 to 6. Sure, one can use the bin ranges that is α_i and α_{i+1} to determine a numeric range in which the newly generated discrete data falls, but this range can be arbitrary large in some case. One simple strategy can be to take the mean/median/mode of the values in the bin in pursuit of generating a numeric value in GANBLR, but it will result in duplicating values being produced. Clearly, we need an elegant way to produce numeric data in GANBLR framework. The trouble with typical discretization is that it offers very little capacity to generate the numeric values back once it is discretized [1]. In this work, we make use of Dirichlet Mixture Model (DMM). We first identify various modes to model a numeric feature distribution, and then represent any numeric value based on its contribution to each mode. During sampling, the discrete value of the numerical feature could be directly sampled from the distribution of the particular mode.

Bayesian Networks (BN) are an excellent framework for incorporating auxiliary information and they lead to models that are interpretable. Though, the original GANBLR framework proposed the use of Bayesian Networks for generator and discriminator – the authors

[1] Information is lost except the location of boundaries.

Datasets	GANBLR	CTGAN	Improvement
Intrusion	0.9378	0.9102	3.1%
Pokerhand	0.5783	0.4673	23%
Adult	0.8429	0.8012	5.0%
Chess	0.8478	0.7870	7.2%

Table 1: Percentage improvement of GANBLR over CTGAN in terms of Machine Learning Utility based on Train-on-Synthetic and Test-on-Real, on 4 datasets.

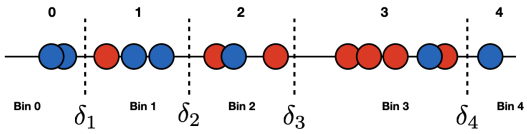

Figure 2: Example discretization of a numeric attribute.

only tested the model with restricted BN such as k-Dependence Bayesian Network (KDB). A KDB is advantaged as the structure learning is very quick and simple, as it is based on sufficient statistics such as Mutual Information and Conditional-Mutual-Information. However, this leads to a structure that is hard to interpret, and also overly complicated. E.g., for a KDB model with $k = 2$, each attribute must take two other attributes as its parents. Full unrestricted BN, on the other hand, can have a arbitrary number of parents for each attribute, but can be much simpler than a high-order KDB. In this work, we have proposed a new formulation GANBLR++, which relies on building an unrestricted Bayesian Network and employs DMM to generate numerical data as well. We claim that GANBLR++ is a significant improvement over GANBLR which we demonstrate by a variety of experiments on a wide range of datasets. We formalize the contributions of this work as:

- We propose a novel model of tabular data generation, namely GANBLR++ that can generate both categorical and numeric data. We demonstrate that it can generate data of much better quality than existing SOTA models of numeric data generation.
- GANBLR++ can be utilized with both restricted BN such as KDB as well as unrestricted BN. Additionally, we claim to be the first work that has exploited the discriminative training of an unrestricted BN (that is used as a discriminator component in GAN formulation).
- To the best of our information, we claim to be the first work that utilizes DMM in the context of BN. Traditionally, the practice is to discretize numeric attributes for BN with typically discretization methods.
- We compare GANBLR++ to existing SOTA GAN models on 15 public tabular datasets. The results demonstrate that GANBLR++ can outperform in terms of **Machine Learning Utility** [2] and other statistical similarity measures.

[2] Machine learning utility means that the synthetic data is used to train the model and then use real data to test the performance.

The rest of this paper is organized as follows. We will discuss preliminary and related work in Section 2. We discuss our proposed formulation GANBLR++ in Section 3, discussing the strategy to generate numeric data as well as the use of unrestricted BN. The experimental evaluation of our proposed framework GANBLR++ is given in Section 4. We conclude in Section 5, with pointers to future works.

2 Preliminaries and Background Work

In this section, let us discuss the preliminary work GANBLR. Later we will delve into some state-of-the-art models for tabular numeric data generation.

2.1 GANBLR Framework Like typical GAN-based learning, GANBLR utilizes a game theoretic approach. It is based on adversarial training between two component – a **generator** model (denoted as $\mathbf{G}$), which tries to generate the synthetic data and the **discriminator** model (denoted as $\mathbf{D}$), which tries to discern the difference between the synthetic and the real data [4]. A typical tabular dataset with m instances and n features is denoted as: $\mathcal{D}_{\text{data}} = [(X^{(k)}, Y]$, which of course is considered as the (real) original dataset for adversarial training. Here, $X^{(k)} = [\mathbf{x}^1, \mathbf{x}^2.., \mathbf{x}^m]$ and $\mathbf{x}^i \in \mathcal{R}^n$. Correspondingly, the class label are denoted as: $Y_g = [y^1, y^2.., y^m]$. In any standard tabular dataset $\mathcal{D}_{\text{data}}$, the features can have k-order feature interactions, which represent the maximum level of interaction among the features. This is represented by the super-script k in $X^{(k)}$.

Notably, in vanilla and conditional GAN models, the generator model uses the random noise to produce the adversarial samples. That is, we assume that the real dataset $\mathcal{D}_{\text{data}} = [(X^{(k)}, Y)]$ belongs to the distribution $\mathrm{P}_{\text{data}}(\cdot)$, and the generator model $\mathbf{G}$ is trained to approximate the $\mathrm{P}_{\text{data}}(\cdot)$ with random noise, which is sampled from the distribution $\mathrm{P}_{\mathcal{Z}}(.)$. The GAN formulation leads to the following objective function:

$$\max_{\mathbf{D}} \min_{\mathbf{G}} \quad \mathbf{E}_{\mathcal{D}\sim \mathrm{P}_{\text{data}}(.)}[\log \mathbf{D}(\mathcal{D}_{\text{data}})] + \mathbf{E}_{\mathcal{Z}\sim \mathrm{P}_{\mathcal{Z}}(.)}[\log(1 - \mathbf{D}(\underbrace{\mathbf{G}(\mathcal{Z})}_{\mathcal{S}_{\text{data}}}))]. \tag{2.1}$$

Here, the synthetic dataset $\mathcal{S}_{\text{data}} = [(\bar{X}^{(k)}, \bar{Y})]$, where $\bar{X}^{(k)} = [\bar{\mathbf{x}}^1, \bar{\mathbf{x}}^2.., \bar{\mathbf{x}}^m]$ and $\bar{\mathbf{x}}^i \in \mathcal{R}^n$. $\mathcal{S}_{\text{data}}$ contains the same order (k) of the feature interactions as original dataset $\mathcal{D}_{\text{data}}$. It can be seen that the main task for the discriminator model $\mathbf{D}$ is to distinguish between $\mathcal{D}_{\text{data}}$ and $\mathcal{S}_{\text{data}}$ and back-propagate the discrimination loss to the generator model for producing better $\mathcal{S}_{\text{data}}$. To do the training for discriminator model, the auxiliary label $Y_d = 1$ and $Y_d = 0$ is added to $\mathcal{D}_{\text{data}}$ and $\mathcal{S}_{\text{data}}$.

In GANBLR, the generator model $\mathbf{G}$ plays two roles:

- Learning the parameter of the BN (denoted as BN^e) by optimizing the discriminative objective function on conditional log-likelihood $\log(\mathrm{P}(Y|X^{(k)}))$. Here the BN^e is learned by constraining the parameter which could be actual conditional probabilities according to the work of [10].
- Sampling the synthetic data from BN and inputting the synthetic dataset $\mathcal{S}_{\text{data}}$ with original dataset to discriminator for minimizing the loss on $\log(1 - \mathbf{D}(\mathbf{G}(.))$ or maximizing the $-\log(1 - \mathbf{D}(\mathcal{S}_{\text{data}}))$. In this role, the $\mathbf{G}$ is performing in generative mode to sample data from learned BN and there is no input needed for this sampling.

The generator model $\mathbf{G}$ of GANBLR can seamlessly perform the two roles. That is, the more accurately trained BN could provide better synthetic dataset $\mathcal{S}_{\text{data}}$ and therefore, the smaller loss of $\log(1 - \mathbf{D}(\mathbf{G}(.)))$ on discriminative model.

The discriminator $\mathbf{D}$ in GANBLR, is again a BN (denoted as BN^d) with the same structure as that of the BN of the generator model. However, the parameter of BN^d is not constrained and the training of $\mathbf{D}$ is based on maximizing the discriminative loss, i.e., $\log \mathbf{D}(\mathcal{D}_{\text{data}}) + \log(1 - \mathbf{D}(\mathcal{S}_{\text{data}}))$.

Training in GANBLR framework follows the game theoretic approach, where the generator model $\mathbf{G}$ tries to produce better synthetic data by minimizing the $-\log(\mathrm{P}(Y|X^{(k)}))$ and $\log(1 - \mathbf{D}(\mathcal{S}_{\text{data}}))$; the discriminator $\mathbf{D}$ maximizing the $\log \mathbf{D}(\mathcal{D}_{\text{data}}) + \log(1 - \mathbf{D}(\mathcal{S}_{\text{data}}))$, leading to following objective function:

$$\max_{\boldsymbol{\theta_d}} \min_{\boldsymbol{\theta_g}} \quad \log \mathbf{D}(\mathcal{D}_{\text{data}}) + \log(1 - \mathbf{D}(\mathcal{S}_{\text{data}})) - \log(\mathrm{P}(Y|X^{(k)})). \tag{2.2}$$

It can be seen that minimizing the negative conditional log-likelihood $-\log(\mathrm{P}(Y_g|X_g^{(k)}))$ plays an important role in GANBLR's strategy to produce better synthetic data.

2.2 Related Work Other than GANBLR, there are two kinds of adversarial-trained tabular data generator: vanilla GAN-based tabular generator and conditional GAN based tabular generator. Several famous models fall in the category of vanilla GAN-based models, such as MedGAN [8], TableGAN [11], PATEGAN [5]. MedGAN utilizes the auto-encoder architecture and can generate both categorical and numerical attributes. TableGAN deploys via the convolutional neural network architecture. PATEGAN is particularly designed to prevent the privacy attacks by adding the differential privacy. The most critical drawback on vanilla GAN-based tabular generators is these models are not able to generate the synthetic data with a particular (or specified) feature-value. In contrast,

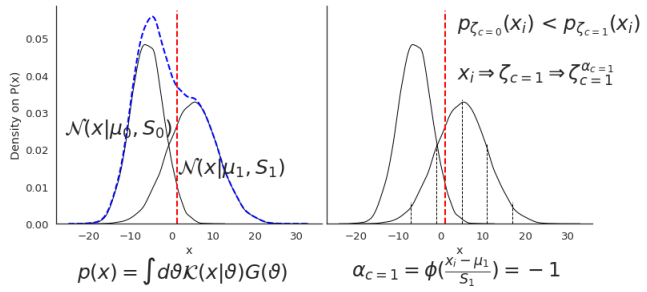

Figure 3: Illustration of numerical sampling from **G** in GANBLR++.

conditional GAN-based tabular data generator uses the conditional-vector to specify the particular feature value or class label to generate. The most successful work is the well-known CTGAN. CTGAN deploys the mode-specific normalization with training-by-sampling to better generate both categorical and numeric features.

3 Methodology

Let us discuss our proposed GANBLR++ algorithm in this section. We will start by explaining the two novel components, i.e., its strategy to a) generate numeric data and b) use of unrestricted BN.

3.1 Component I – Numerical Sampling Inspired from Conditional-GAN [9], we propose to use the numerical sampling in the **G** component of GANBLR, based on the Dirichlet Process Gaussian Mixture Model – DP-GMM. Bayesian non-parametric models are the models where the number of parameters grow freely with the amount of data provided [12]. So, instead of tuning the parameter of cluster number as is done in VGM and standard GMM, the cluster number in DP-GMM is allowed to grow as more data is observed during the training of DP-GMM. We have proposed two steps to implement the numerical sampling in this work: Dirichlet Mode Discretization and Specific Mode Sampling. Let us discuss these two steps in detail in the following.

3.1.1 Dirichlet Mode Discretization Any given numeric attribute can have non-Gaussian or multi-modal distribution. The critical question is how to determine the number of modes. As forehand mentioned, we have made use of non-parametric DP-GMM to automatically determine the number of modes. The Dirichlet Process firstly samples some unbounded points (atom) to represent the cluster centers (mean values in the distribution) for any given numeric attribute. Once the cluster centers are drawn, the Gaussian distributions can be created around the corresponding cluster centers. Finally, the observed data point is assigned to each cluster during the inference to maximize the likelihood for obtaining the optimized number of clusters. The following equation can be used to define a density function of DP-GMM: $\mathrm{P}(\mathbf{x}) = \int d\vartheta \mathcal{K}(\mathbf{x}|\vartheta)G(\vartheta)$. Here $\mathrm{P}(\mathbf{x})$ represents the probability of input $\mathbf{x}$ belonging to a mode. $\mathcal{K}(\mathbf{x}|\vartheta) = \mathcal{N}(\mathbf{x}|\mu, S)$ represents a Gaussian distribution with parameters – $[\mu, S]$ and $G(\vartheta)$ represents a prior distribution over parameter ϑ, which is the mean value of the Gaussian (or the cluster center).

In Dirichlet Process formulation, we use $G(\vartheta)$ to sample parameters $[\mu, S]$ for a Gaussian distribution. At the time of inference, we have a finite number of clusters, and their index is represented as ζ_c. We use notation $P_{\zeta_c}(x_i)$ to represent the probability of x_i belonging to cluster ζ_c. Figure 3, illustrates a case where two modes (clusters) $\zeta_{c=0}$ and $\zeta_{c=1}$ have been identified. Since, we know the probability of each mode (P_ζ) – for a numeric value, x_i, we can determine the mode (cluster) of x_i by comparing the $\mathrm{P}_{\zeta_{c=0}}(x_i)$ with $\mathrm{P}_{\zeta_{c=1}}(x_i)$. Next, as depicted in the R.H.S of Figure 3, the distance d_1 between value x_i and its mode center μ_1 is computed after normalization as: $\left(\frac{x_i - \mu_1}{S_1}\right)$. Now, the issue is that GANBLR relies on discrete model BN to learn its parameters, therefore, we must find a way to map the numeric values (d_1) to discretized values. We have established a simple strategy to covert d_i into a discrete value. We make use of the cluster to which x_i belongs and make the following transformation for each x_i and its related d_i:

$$\zeta_c^{\alpha_c} = \zeta_c \oplus \mathbb{1}_{\tau=1}\alpha_c. \tag{3.3}$$

where the α_c is defined as:

(3.4)

$$\alpha_c = \phi(d_c),$$
$$d_c = \left(\frac{x_i - \mu_c}{S_c}\right),$$
$$\phi(d_c) = \begin{cases} +1|-1 & |d_c| \leq S_c \wedge d_c > 0 | d_c < 0; \\ +2|-2 & S_c < |d_c| \leq 2S_c \wedge d_c > 0 | d_c < 0; \\ +3|-3 & 2S_c < |d_c| \leq 3S_c \wedge d_c > 0 | d_c < 0; \\ \infty|-\infty & |d_c| \geq 3S_c \wedge d_c > 0 | d_c < 0. \end{cases}.$$

It can be seen that we have designed two scenarios through the use of indicator function. Well, if $\tau \neq 1$, the final discretized value for x_i is just its cluster center's index – ζ_c. Otherwise, the value x_i is discretized based on Equation 3.4 to $\zeta_c^{\alpha_c}$. We will discuss the difference of ζ_c and $\zeta_c^{\alpha_c}$ in the ablation study in experiment. We found that using the discretization of Equation 3.4 can lead to better performance.

3.1.2 Mode-Specific Sampling When the generator **G** in GANBLR starts to sample numerical values say $\bar{\mathbf{x}}$ for synthetic dataset $\mathcal{S}_{\text{data}}$, the discrete value $\zeta_c^{\alpha_c}$ (or ζ_c) will be sampled first for any continuous attribute. Afterwards, the $\bar{\mathbf{x}}$ will be sampled based on $\zeta_c^{\alpha_c}$. For instance, if $\zeta_c^{\alpha_c}$ has an associated Gaussian distributions – $\mathcal{N}(\bar{\mathbf{x}}|\mu_c, S_c)$, the continuous value $\bar{\mathbf{x}}$ can be sampled, with following condition fulfilled: $\alpha_c < |\frac{\bar{\mathbf{x}}-\mu_c}{S_c}| \leq \alpha_c + 1$. Finally, the $\bar{\mathbf{x}}$ is obtained by checking whether $(\bar{\mathbf{x}} - \mu_c) > 0$ holds. The following equation summarizes our Mode Specific Sampling as:

$$\bar{\mathbf{x}} \sim \mathcal{N}(\bar{\mathbf{x}}|\mu_c, S_c) \quad \text{Such that} \quad \alpha_c < |\frac{\bar{\mathbf{x}} - \mu_c}{S_c}| \leq \alpha_c + 1, \quad \text{sign}(\bar{\mathbf{x}} - \mu_c) = \text{sign}(\alpha_c) \tag{3.5}$$

One can also directly use the discrete value $\zeta_c^{\alpha_c}$ for synthetic dataset $\mathcal{S}_{\text{data}}$. Therefore, GANBLR++ has the flexibility to work on both discrete and continuous data types.

3.2 Component II – Unrestricted BN The structure learning for BN is the process of learning the structure of the directed acyclic graph (DAG) from the given dataset ($\mathcal{D}_{\text{data}}$). The structure of BN indicates the feature interactions and is highly interpretable. In typical restricted mode, such as KDB, the feature interaction are limited to a maximum level, which means each attribute can only connect to a specified number of other attributes (parents). In the proposed GANBLR++, we have aimed to use unrestricted structure learning. There are two steps within the structure learning of the proposed GANBLR++:

- Scoring the current structure $\mathcal{G}$ given $\mathcal{D}_{\text{data}}$.
- Updating (adding, deleting, reversing connection) the structure $\mathcal{G}$ until convergence.

The scoring step can be specified as:

$$\text{Score}(\mathcal{G} : \mathcal{D}_{\text{data}}) = \text{LL}(\mathcal{G} : \mathcal{D}_{\text{data}}) - \lambda(|\mathcal{D}|)|\mathcal{G}| \tag{3.6}$$

where $\text{LL}(\mathcal{G} : \mathcal{D}_{\text{data}})$ is the log-likelihood score to indicate how well the BN with structure $\mathcal{G}$ can fit the data, and $\lambda(|\mathcal{D}|)|\mathcal{G}|$ is the regularization term to avoid over-fitting.. Note, the function $\lambda(\cdot)$ here is $\log(\cdot)/2$ and, therefore, the criterion is Bayesian Information Criterion.

3.3 GANBLR++ Algorithm The proposed GANBLR++ framework is based on the training of GANBLR, but with the two modifications as explained in the previous two sub-sections. The generator model **G** (parameterized by $\boldsymbol{\theta_g}$) and the discriminator model **D** (parameterized by $\boldsymbol{\theta_d}$) are trained in turns for updating each model's parameters. $\mathcal{D}_{\text{data}}$ is first processed by the Dirichlet Mode Discretization method as discussed in Section 3.1.1. It is later used to learn the structure $\mathcal{G}$ of BN, and lastly it is employed to train the generator model **G** and the discriminator model **D** based on eq. (2.2). The complete algorithm of GANBLR++ is given in algorithm 1.

Algorithm 1 Algorithm GANBLR++

Input: Original data - $X^{(k)}, Y$
Output: Synthetic data - $\bar{X}^{(k)}, \bar{Y}$

Sample m instances $(X^{(k)}, Y) \sim \text{P}_{\text{data}}(.)$
Obtain $\mathcal{G}$ by optimizing eq. (3.6)
for iteration $t \subset T$ in training **do**
  if $\mathbf{x} \subset X^{(k)}$ is numerical **then**
    Obtain ζ_c, μ_c, S_c
    Discretize $\mathbf{x}$ to $\zeta_c^{\alpha_c}$ via eq. (3.4)
    Obtain $X^{(k)}, Y$
  end if
  Obtain $\boldsymbol{\theta_g}$ by optimizing **G** with $\mathcal{G}$ via eq. (2.2).
  Sample $\mathcal{S}_{\text{data}}$
  for step $q \subset Q$ in discriminator **D** **do**
    Obtain $\boldsymbol{\theta_d}$ by optimizing **D** via eq. (2.2)
  end for
end for
if $\bar{\mathbf{x}} \subset \bar{X}^{(k)}$ is numerical **then**
  Mode Specific Sampling $\bar{\mathbf{x}}$ via eq. (3.5)
  Obtain $\bar{X}^{(k)}, \bar{Y}_g$
end if
return $\mathcal{S}_{\text{data}} \equiv \bar{X}^{(k)}, \bar{Y}$

3.4 GANBLR++ (Disc) Variant It can be seen from algorithm 1, that we have the capacity to generate both numeric and categorical data. For a better comparison, and study its effectiveness, we have identified following two variants:

- GANBLR++ (Disc) – is the version where we use DMM for discretization but do not use Mode-Specific Sampling. This version should be compared against standard GANBLR model, which handles numeric attributes through a pre-step discretization.
- GANBLR++ – is the version where we use both DMM as well as mode-specific sampling. This version should be tested with other SOTA numeric tabular data generation models such as CTGAN, TableGAN and MedGAN.

The GANBLR++ and GANBLR++ (Disc) are both evaluated in Section 4.

Dataset	m	n	C	Size
Intrusion	4M	41	2	Large
Pokerhand	1M	11	10	Large
Covtype	581012	55	7	Large
Shuttle	58000	9	7	Large
Connect-4	67557	42	3	Large
Credit	50000	31	2	Medium
Adult	50000	14	2	Medium
Chess	28056	6	7	Medium
letter_rocog	20000	16	26	Medium
Magic	19020	11	2	Medium
Nursery	12960	8	5	Small
Sign	6500	8	3	Small
Satellite	6435	36	7	Small
Loan	5000	13	2	Small
Car	1728	6	2	Small

M denotes a million. m, n, C denote the number of instance, attributes and number of classes, respectively.

Table 2: Data Statistics.

4 Experiment

Let us start by discussing our experimental set-up first before we delve into empirical evaluation of our proposed framework GANBLR++.

4.1 Experiment setup

4.1.1 Datasets There are a total of 15 datasets that we have used in the experiment – 13 datasets are from UCI datasets and 2 from Kaggle repository (Credit and Loan). Each dataset has the target attribute and a set of independent attributes, and are used for classification task. Among all datasets, 5 datasets have more than 50K instances and are denoted as Large, whereas, 5 datasets have less than 50K instances and more than 15K instances, and are denoted as Medium. Finally, the remaining 5 datasets have less than 15K instances and are denoted as Small. The statistics of the data are summarized in Table 2

4.1.2 Baselines and Evaluation Metric We compare the following methods in our experiments – GANBLR++, GANBLR++ (Disc) (described in Section 3.4), GANBLR [13], and CTGAN, TableGAN, MedGAN as described in Section 2.

All baseline models are trained with 150 epochs for 5 Large datasets, and 100 epochs for the Medium and the Small datasets. The experiment is repeated 3 times with 2 fold cross validation and the average results are reported. As, many datasets in Table 2 are multi-class, therefore the accuracy is used in the experiment.

4.1.3 Configuration and Running Environment Unlike GANBLR, we do not have to specify k for BN in GANBLR++, as it is based on unrestricted BN. The default value τ is set to 1 for indicator function $\mathbb{1}$ in eq. (3.4). GANBLR++ is coded in Python 3.7 in Tensorflow 2.5 framework, with 8 core Intel i8 CPU machine with 32 GB RAM memory.

4.1.4 Machine Learning Utility There are two commonly used strategies to evaluate the machine learning utility for synthetic data: TSTR: Training on Synthetic data, Testing on Real data; TRTR: Training on Real data, and Testing on Real data. In order to obtain the TSTR and TRTR performance of GANBLR++ and the other baseline models, 4 commonly used machine learning classification algorithms are used here, i.e., Logistics Regression (LR), Muti-layer-Perceptron (MLP), Random Forest (RF), and Extreme Gradient Boosting Tree (XGBT).

For obtaining TSTR, the real dataset is firstly divided into real training and real testing datasets: the real training dataset is used to train the GANBLR++ and baseline models. Once training is completed, the synthetic datasets are obtained. Then the synthetic datasets are used to train the above 4 machine learning classification algorithms, which will be evaluated on the real testing datasets. In short, the TSTR can best qualify *whether the synthetic dataset can be used as the substitute of the real dataset in machine learning tasks.* In contrast, the TRTR mainly serves as a baseline, which indicates what sort of performance we should expect for TSTR.

4.1.5 Statistical Similarity To compare the quality of the synthetic datasets, particularly for synthetic datasets under numerical sampling, it is important to also measure the statistical similarity between the real and the synthetic datasets. Similar to work of [13], two metrics are used to measure the statistical similarity: Jensen-Shannon Divergence (JSD) and Wasserstein Distance (WD).

4.2 Machine Learning Utility Results Analysis Table 3 reports the detailed results from GANBLR++ (Disc), GANBLR++ and all other baseline models. It could be seen that, the GANBLR++ (Disc) and GANBLR++ outperform other methods on almost all the datasets. Note, if all features in a dataset are categorical, we have omitted the results of GANBLR++ on these datasets (namely – Poker-hand, Connect, Chess, Nursery, Car). GANBLR++ (Disc) demonstrates the effectiveness of using unrestricted BN over standard GANBLR models on these datasets, as it wins on

all four out of five datasets (only loss is on car dataset).

4.2.1 Results of Machine Learning Utility without numerical sampling Table 4 provides the averaged accuracy on all 15 Large, Medium and Small datasets. It is clear that GANBLR++ (Disc) outperforms all other baseline methods which includes the current SOTA model GANBLR in terms of TSTR performance. Particularly on small and medium datasets, GANBLR++ (Disc) and GANBLR have significantly better performance than other baseline methods. But GANBLR++ (Disc) provides a much better results on Large, Medium and Small datasets. It is interesting to see that for GANBLR++ (Disc) and GANBLR, the TSTR performances are close to TRTR performance, while none of the other baseline methods have the TSTR performance close to TRTR.

4.2.2 Results of Machine Learning Utility with numerical sampling To better evaluate the numerical sampling capability of GANBLR++, the comparison has been made between GANBLR++ and selected baseline models. Note, the baseline models are selected which have the capability of providing numerical attributes in synthetic datasets [3]. Table 5 shows the comparison results. It can be seen from the results, that GANBLR++ outperforms over all selected baseline models on Large, Medium and Small datasets.

4.3 Statistical Similarity Results Analysis To obtain JSD and WD results – for a dataset, each attribute from synthetic dataset is compared against the same attribute in real dataset in terms of JSD and WD. We repeat the process for all attributes, and for all datasets, and then, report the averaged result separately with or without numerical sampling.

4.3.1 Results of Statistical Similarity without numerical sampling It can be seen from Table 6 that GANBLR++ (Disc) outperforms all baselines on Larger and Small datasets. It is also interesting to mention that the GANBLR++ (Disc) is better than GANBLR. This highlights that GANBLR++ (Disc) produces dataset of superior quality. On Medium datasets, the results between GANBLR++ (Disc) and GANBLR are very close and CTGAN wins on the JSD distance while TableGAN wins on the WD distance. However, the difference between GANBLR++ (Disc) and the baselines based on the statistical similarity from Medium datasets is quite minor. One reason for the results on Medium datasets –

[3]Only datasets with numeric features are used, the results represents average over 10 datasets.

Dataset	Model	GANBLR++ (Disc)	GANBLR++	GANBLR	CTGAN	TableGAN	MedGAN
Intrusion	LR	**0.9402**	0.9278	0.9316	0.9232	0.792	0.826
	MLP	0.9219	0.9017	0.901	**0.926**	0.739	0.803
	RF	**0.9563**	0.9381	0.9398	0.893	0.768	0.812
	XGBT	**0.9581**	0.9290	0.9318	0.918	0.721	0.831
Pokerhand	LR	**0.769**		0.533	0.469	0.471	0.371
	MLP	**0.778**		0.5518	0.435	0.39	0.382
	RF	**0.781**		0.572	0.502	0.41	0.39
	XGBT	**0.783**		0.5837	0.511	0.423	0.402
Covtype	LR	**0.729**	0.725	0.653	0.671	0.57	0.472
	MLP	**0.721**	0.711	0.673	0.6961	0.602	0.455
	RF	**0.733**	0.731	0.703	0.705	0.582	0.466
	XGBT	**0.740**	0.735	0.691	0.6827	0.591	0.497
Shuttle	LR	**0.998**	0.991	0.996	0.951	0.927	0.988
	MLP	0.991	0.989	**0.993**	0.972	0.912	0.982
	RF	**0.995**	0.993	0.9931	0.967	0.929	0.992
	XGBT	**0.997**	0.990	0.996	0.9801	0.92	0.991
Connect	LR	**0.7543**		0.7517	0.702	0.663	0.692
	MLP	**0.789**		0.783	0.727	0.652	0.701
	RF	**0.803**		0.791	0.682	0.673	0.691
	XGBT	0.812		0.803	0.709	0.656	0.712
Credit	LR	0.9998	0.9998	**0.9998**	0.9979	0.9981	0.998
	MLP	**0.9999**	0.9998	0.9997	0.999	0.9972	0.9982
	RF	0.9994	0.9992	**0.9996**	0.9981	0.9987	0.999
	XGBT	0.9993	0.9995	**0.9998**	0.9976	0.998	0.998
Adult	LR	**0.79**	0.7713	0.74	0.787	0.757	0.737
	MLP	**0.853**	0.836	0.842	0.831	0.761	0.739
	RF	**0.821**	0.819	0.81	0.792	0.783	0.721
	XGBT	0.825	0.827	**0.851**	0.839	0.775	0.733
Chess	LR	**0.8617**		0.8478	0.693	0.722	0.577
	MLP	**0.889**		0.877	0.673	0.735	0.552
	RF	**0.903**		0.893	0.701	0.714	0.56
	XGBT	**0.901**		0.882	0.723	0.751	0.557
Letter_recog	LR	**0.722**	0.702	0.689	0.531	0.462	0.47
	MLP	**0.753**	0.715	0.739	0.601	0.392	0.462
	RF	**0.732**	0.710	0.713	0.566	0.452	0.47
	XGBT	**0.740**	0.720	0.736	0.539	0.478	0.483
Magic	LR	**0.803**	0.797	0.78	0.656	0.641	0.648
	MLP	**0.819**	0.802	0.789	0.672	0.632	0.668
	RF	**0.833**	0.813	0.803	0.698	0.665	0.671
	XGBT	**0.845**	0.822	0.812	0.681	0.632	0.676
Nursery	LR	**0.907**		0.8819	0.793	0.678	0.569
	MLP	**0.922**		0.902	0.839	0.681	0.575
	RF	**0.927**		0.919	0.802	0.676	0.582
	XGBT	**0.931**		0.93	0.836	0.688	0.585
Sign	LR	**0.652**	0.633	0.61	0.472	0.482	0.389
	MLP	**0.673**	0.662	0.649	0.479	0.473	0.401
	RF	0.681	**0.689**	0.61	0.4563	0.491	0.412
	XGBT	**0.689**	0.688	0.636	0.489	0.499	0.418
Satellite	LR	**0.853**	0.837	0.85	0.496	0.502	0.405
	MLP	0.876	0.870	**0.887**	0.502	0.492	0.388
	RF	**0.880**	0.863	0.879	0.51	0.511	0.392
	XGBT	**0.898**	0.859	0.89	0.499	0.502	0.41
Loan	LR	**0.7725**	0.7572	0.772	0.706	0.756	0.386
	MLP	**0.828**	0.803	0.801	0.683	0.762	0.392
	RF	**0.831**	0.820	0.816	0.731	0.757	0.406
	XGBT	**0.840**	0.819	0.822	0.739	0.759	0.403
Car	LR	0.81		**0.85**	0.713	0.62	0.611
	MLP	0.822		**0.865**	0.686	0.59	0.582
	RF	0.841		**0.873**	0.719	0.621	0.575
	XGBT	0.837		**0.891**	0.721	0.633	0.591
Average		0.8418	0.8298	0.8151	0.7145	0.6651	0.6108

Table 3: Machine Learning Utility on all datasets in terms of TSTR.

Method	L˜Accuracy%	M˜Accuracy%	S˜Accuracy%
	TRTR		
	80.84%	86.80%	84.62%
	TSTR		
GANBLR++ (Disc)	**79.32%**	**86.10%**	**82.56%**
GANBLR	78.85%	84.02%	81.67%
CTGAN	75.27%	74.88%	64.37%
TableGAN	66.95%	71.72%	60.87%
MedGAN	67.28%	68.58%	47.36%

Table 4: Machine Learning Utility comparison of GANBLR++ (Disc).

Method	L˜Accuracy%	M˜Accuracy%	S˜Accuracy%
	TRTR		
	78.90%	86.27%	82.69%
	TSTR		
GANBLR++	**78.88%**	**85.57%**	**81.30%**
CTGAN	77.27%	72.11%	66.77%
TableGAN	63.65%	70.29%	58.29%
MedGAN	65.71%	66.36%	48.29%

Table 5: Machine Learning Utility comparison of GANBLR++

Method	Large		Medium		Small	
	JSD	WD	JSD	WD	JSD	WD
GANBLR++ (Disc)	**0.130**	**0.589**	0.313	0.680	**0.201**	**0.751**
GANBLR	0.136	0.619	0.301	0.662	0.263	0.783
CTGAN	0.133	0.621	**0.287**	0.672	0.330	0.810
TableGAN	0.142	0.701	0.350	**0.613**	0.531	1.220
MedGAN	0.190	0.730	0.392	0.690	0.380	1.021

Table 6: Statistical Similarity Comparison of GANBLR++ (Disc).

we believe that this could be due to GANBLR++ (Disc)'s generator – Bayesian Network's ability to generate some attribute value which are rarely seen in the real dataset. The same reason applies to GANBLR as well.

4.3.2 Results of Statistical Similarity with numerical sampling For the statistical similarity comparison with numerical sampling, as above mentioned, only 10 datasets are used in the comparison. Based on Table 7, it can be seen that the statistical similarity of GANBLR++ stands out from all the other baselines on Large, Medium and Small datasets. It only loses on JSD from Medium datasets. The results are encouraging and show that the distribution of numerical attributes from the synthetic dataset is almost the same as the real datasets.

To better illustrate the quality of the numerical sampling from GANBLR++, three attributes Age, Education-Num and Hours-per-week are selected from the dataset Adult, then the actual and estimated densities (distributions) are presented in Figure 4. Notably, the distributions with red and green color are the sampled synthetic distributions from GANBLR++ and CTGAN respectively, and the black represents the real data distribution. It can be seen that the distribution from sampled synthetic dataset obtained from GANBLR++ and real dataset matches for all the three attributes. Particularly, for attribute Education-Num, the difference can be seen as really minor. However, the distribution from sampled synthetic dataset obtained from CTGAN and real dataset are not well matched. E.g., for attribute of Hours-per-week, the CTGAN failed to find the correct number of modes.

As GANBLR++ make use of BN to learn the feature interaction among attributes, the joint distribution of sampled synthetic dataset on those attributes that interacted with each other should also be preserved and should lead to better quality data as compared to the CTGAN model, which does not have an explicit feature interaction capability. Figure 5 illustrate the joint distribution on Class:Age, Class:Education-Num and Class:Hours-per-week from both sampled synthetic dataset and real dataset. The blue color represent the joint distribution for real dataset, and orange color represent the joint distribution for sampled synthetic dataset from either GANBLR++ or CTGAN. It is clear to see that the difference of joint distribution between GANBLR++ sampled synthetic dataset and real dataset is really minor. In contrast, the difference of joint distribution from comparison between CTGAN sampled synthetic dataset and real dataset is obvious. The above interesting patterns can also support the TSTR results in previous experiment – as the synthetic dataset from GANBLR++ is not only similar enough to the real dataset but also better at preserving the feature interactions.

Method	Large		Medium		Small	
	JSD	WD	JSD	WD	JSD	WD
GANBLR++	**0.117**	**0.520**	0.297	**0.533**	**0.129**	**0.570**
CTGAN	0.128	0.571	**0.257**	0.562	0.270	0.603
TableGAN	0.130	0.621	0.327	0.593	0.341	1.031
MedGAN	0.177	0.703	0.356	0.637	0.353	1.021

Table 7: Statistical Similarity Comparison of GANBLR++

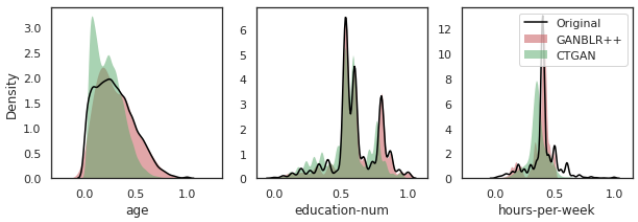

Figure 4: Selected attributes of Age, Education-Num, Hours-per-week's distributions estimated by GANBLR++ and CTGAN.

4.4 Ablation Analysis Over τ To illustrate the impact of hyper-parameter τ under numerical sampling on GANBLR++, we conducted the ablation studies by switching the value of τ between 0 and 1. Specifically, when $\tau = 1$, the discrete value on given numerical value $\mathbf{x}$ is $\zeta_c^{\alpha_c}$. Whereas, when $\tau = 0$, the discrete value on given numerical value $\mathbf{x}$ is ζ_c, which is just the the cluster index. We compare the TSTR performance between GANBLR++ with $\tau = 1$ and $\tau = 0$. It can be seen from Table 8 that when $\tau = 0$, the GANBLR++ always has worse performance comparing to $\tau = 1$. Particularly, the Large datasets have suffered most. This pattern can be explained as the

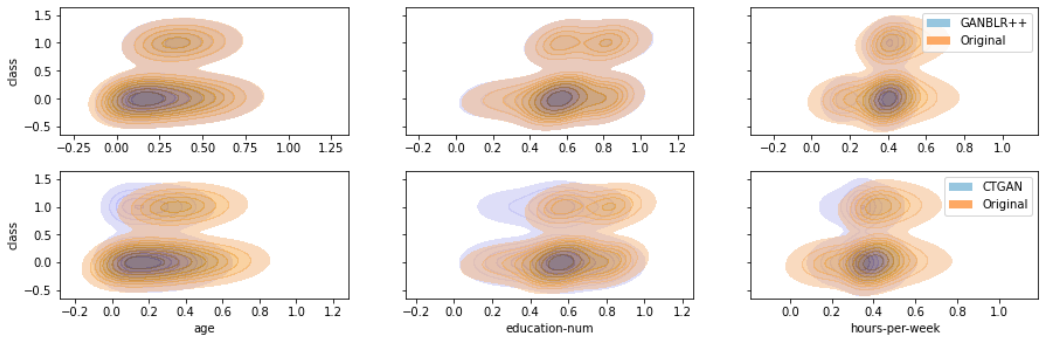

Figure 5: Selected joint attribute tuples of `class:Age, class:Education-Num, class:Hours-per-week`'s distributions estimated by `GANBLR++` and `CTGAN`.

	Large	Medium	Small
`GANBLR++ Performance`	$\tau = 1/\tau = 0$	$\tau = 1/\tau = 0$	$\tau = 1/\tau = 0$
	-8.13%	-5.70%	-7.82%

Table 8: Ablation Analysis Over τ

`Large` datasets requires more detailed feature interaction and when $\tau = 0$, the sampling process without condition is just the ζ_c.

5 Conclusion

In this work, we presented a novel model of tabular data generator `GANBLR++`, which can generate both categorical and numerical attributes. Our proposed `GANBLR++` is based on recently developed `GANBLR` framework, and builds on discriminatively trained unrestricted Bayesian Networks as the generator as well as discriminator, which learns by optimizing a game-theoretic objective function. We showed that `GANBLR++` not only advances the existing `SOTA GAN`-based models including `GANBLR`, but also leads to excellent quality data when numerical sampling is required. The experiment results based on both machine learning utility and statistical similarity, when compared against several `SOTA` baselines, suggests that the `GANBLR++` can provide better quality synthetic datasets. In the future, we would like to extend the work on adding differential privacy to the process of synthetic data generation.

References

[1] Adrian Bulat, Jing Yang, and Georgios Tzimiropoulos. To learn image super-resolution In *Proceedings of the European conference on computer vision (ECCV)*,pages 185–200, 2018.

[2] Ethan Fetaya, Jörn-Henrik Jacobsen, Will Grathwohl, and Richard Zemel. Understanding the limitations of conditional generative models. 2020.

[3] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson et al. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*, pages 7–10, 2016.

[4] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley et al. Generative adversarial networks. In *Communications of the ACM*, 2020.

[5] James Jordon, Jinsung Yoon, and Mihaela Van Der Schaar. Pate-gan: Generating synthetic data with differential privacy guarantees. In *ICLR*, 2018. *arXiv preprint arXiv:1406.2661*, 2014.

[6] Jianmin Bao, Dong Chen, Fang Wen, Houqiang Li, and Gang Hua. Cvae-gan: fine-grained image generation through asymmetric training. In *ICCV*, 2017.

[7] Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, Zhongxia Chen et al. xdeepfm: Combining explicit and implicit feature interactions for recommender systems. In *KDD*, 2018.

[8] Karim Armanious, Chenming Jiang, Marc Fischer, Thomas Küstner, Tobias Hepp et al. Medgan: Medical image translation using gans. *Computerized medical imaging and graphics*, 2020.

[9] Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional gan. *arXiv preprint arXiv:1907.00503*, 2019.

[10] Nayyar A. Zaidi, Geoffrey I. Webb, Mark J. Carman, François Petitjean, Wray L. Buntine, Mike Hynes, and Hans De Sterck. Efficient parameter learning of bayesian network classifiers. *Mach. Learn.*, 2017.

[11] Noseong Park, Mahmoud Mohammadi, Kshitij Gorde, Sushil Jajodia, Hongkyu Park, and Youngmin Kim. Data synthesis based on generative adversarial networks. *arXiv preprint arXiv:1806.03384*, 2018.

[12] Teh, Yee Whye, and Michael I. Jordan Hierarchical Bayesian nonparametric models with applications, *Bayesian nonparametrics*, pages 158–207, 2010.

[13] Yishuo Zhang, Nayya Zaidi, Jiahui Zhou and Gang Li. GANBLR: A Tabular Data Generation Model. *ICDM*, 2021.