Self-supervised Autoencoder for Correlation-Preserving in Tabular GANs

Siddarth Ramesh [*1] **Surgan Jandial** [*1] **Gauri Gupta** [*2] **Piyush Gupta** [*1] **Balaji Krishnamurthy** [1]

Abstract

Preserving relationships and interactions between columns(or variables) is crucial for any synthetic tabular data generation approach. Despite their effectiveness, existing generative adversarial network (GAN)-based approaches place little emphasis on this aspect. In this work, we propose VSA+GAN, a framework that augments with the existing GANs to capture and learn inter-variable interactions with a self-supervised autoencoder trained on a pretext task. We show that the method is versatile and is applicable to any variation of Tabular Generative Adversarial Network implementations, and empirically show that our framework significantly improves their performance in terms of data similarity, pair-wise correlation and machine-learning utility metrics.

1. Introduction

Raw tabular data collected from multiple sources tends to be incomplete, noisy, under-balanced (Sánchez-Morales et al., 2020) and often carry the privacy risks (Wang et al., 2017). As a solution, recent approaches propose to leverage synthetic data (Song et al., 2020), which statistically resembles real data and can comply with data privacy laws. In addition, this also reduces data preparation costs and improves data quality.

The existing synthetic tabular data generators typically deploy Generative Adversarial Networks (GANs) to model the tabular distribution (Bourou et al., 2021). While these methods optimize well for single variable distributions, they have the potential to further enhance their capabilities by explicitly accounting for inter-variable interactions, which is crucial for learning latent representations from datasets where we know *a priori* correlations exist. Preserving inter-variable correlations is crucial to use the generated synthetic data as a proxy of the original data for training ML models, which becomes challenging as these dependencies are rather complex and irregular (Bahri et al., 2022). This is in contrast to images and text, which either hold spatial correlations between pixels or sequential correlations between words. The aforementioned motivates us to ask if designing specialized tasks and networks that capture the correlations among features has significant advantages.

In this work, we propose VSA+GAN, a new approach which supplements the existing generative models for an improvised synthesis of tabular data. Our training procedure involves two key steps and commences with a self-supervised pretraining phase of an autoencoder. The autoencoder is trained on the pretext task of recovering the original samples from their corrupted counterparts. By learning to reconstruct the original samples, the denoising autoencoder captures the inter-variable relationships in the data. We then train a generative network using a reconstruction loss in conjunction with the standard adversarial principle. For each mini-batch sampled from the generator, we construct a corrupted version using the same scheme and feed it to the pretrained autoencoder.

Since the pretrained autoencoder has already learnt the inter-column correlations from the original data distribution, it reconstructs the data, filling in the corrupted variables with values that would correlate with the un-corrupted values in the original data. This reconstruction error is passed onto the generator which inherits the learnt implicit inter-variable relations from the autoencoder. This helps to preserve inter-feature correlation, providing additional supervision to improve its utility by penalizing synthesized samples where the combination of values are semantically incorrect.

The proposed method is versatile enough to be applied to any tabular generative model and significantly improves the utility of the synthesized data. Our experiments over widely used tabular datasets and metrics demonstrate that combining the pretrained autoencoder framework with the existing tabular generative adversarial models improves them in terms of data similarity and utility.

2. Proposed Method

Self-supervised learning for tabular data. In this step, we train an autoencoder (encoder E, decoder D), which learns to recover an input data point $\boldsymbol{x}$ from its corrupted version

[*]Equal contribution [1]Media and Data Science Research Lab, Adobe [2]Camera Culture Lab, MIT. Correspondence to: Siddarth Ramesh <sir@adobe.com>.

$\boldsymbol{x'}$. To create an $\boldsymbol{x'}$ for each $\boldsymbol{x}$, we randomly sample an $\overline{\boldsymbol{x}}$ from the current batch, and then use a binary mask vector $\boldsymbol{m} = [m_1, \cdots, m_d] \in \{0, 1\}^d$ as:

$$\boldsymbol{x'} = p_{\boldsymbol{m}}(\boldsymbol{x}) = \boldsymbol{m} \odot \boldsymbol{x} + (\boldsymbol{1} - \boldsymbol{m}) \odot \overline{\boldsymbol{x}} \tag{1}$$

where m_i is sampled from a Bernoulli distribution with probability p_i. In contrast to the standard corruption methods like zeroing features, or adding gaussian noise, Eq.1 ensures that the corrupted sample $\boldsymbol{x'}$ is syntactically similar to other samples in the dataset. We provide a similar ablation in the appendix.

Now, we learn our autoencoder to reconstruct $\boldsymbol{x}$, by which it implicitly learns the inter-variable relations between different features. Formally, we reconstruct the original input from the corrupted $\boldsymbol{x'}$ as $\boldsymbol{D}(\boldsymbol{E}(\boldsymbol{x'}))$ and minimize the loss:

$$\mathcal{L}_R = \sum_{i=1}^{m} \sum_{j=1}^{n} [\mathcal{L}_j(\boldsymbol{D}(\boldsymbol{E}(\boldsymbol{x'}))_j, x_i)] \tag{2}$$

$\mathcal{L}_j$ is cross-entropy loss or mean squared error depending on the j^{th} feature being categorical or continuous.

GAN Objective. In general, GANs consist of two neural networks: a generator G and a discriminator D which provides feedback on the quality of the generation. Among many GAN variants, WGANs and in particular WGAN-GP are one of the most successful models.

The generator is trained with a reconstruction loss from the pretrained autoencoder alongside the standard adversarial loss. For each mini-batch sample x from the generator, we generate a corrupted version x' using the scheme described in 1 and reconstructed by the autoencoder, leveraging its knowledge of inter-column correlations. The reconstruction error guides the generator, preserving inter-feature correlation and penalizing semantically incorrect value combinations. Combined with the self-supervised loss $\mathcal{L}_R$ from the generated data, the generator then minimizes the final loss

$$\min_G \max_D \mathbb{E}_x[\log D(x)] + \mathbb{E}_z[1 - \log D(G(z))] + \mathcal{L}(\boldsymbol{D}(\boldsymbol{E}(p_m(G(z)))), G(z)) \tag{3}$$

The reconstruction error guides the generator, preserving inter-feature correlation and penalizing semantically incorrect feature combinations. Note that, the weights of the autoencoder are frozen when the generator is being trained to keep the learnt correlations intact.

3. Experimental Analysis

Datasets and Models. We evaluate our approach of synthetic data generation on five commonly used datasets: **Adult** (Kohavi et al., 1996), **Credit** (Dal Pozzolo et al., 2014), **Cabs** (Poongodi et al., 2022), **Online News Popularity** (Hensinger et al., 2013), **Kings** (Wang & Zhao, 2022). Besides, we augment our technique with four GAN-based tabular data generators: CTGAN, TableGAN, CWGAN and MedGAN. Baseline implementations follow their original description, while our autoencoder is trained for 300 epochs on a batch size of 100.

Evaluation Metrics and Results. We evaluate our method on four aspects [1] to check if the synthetic data can be a good proxy of the real data: (1) statistical similarity (2) detection (3) machine learning (ML) utility and (4) correlation-preserving metrics. Entries in the table are average across the datasets, and for each metric higher values mean better performance.

Statistical similarity metrics use a Chi-squared (CS) test and a two-sample Kolmogorov-Smirnov (KS) test to measure statistical similarity between the real and synthetic data across all columns (see Table 1). We report 1 minus the KS Test D statistic for continuous columns and CS Test p-value for discrete columns.

Detection metrics use two standard classifier models: Logistic Regression and Support Vector Classifier to evaluate how hard it is to distinguish synthetic data from real. We report 1 minus the ROC AUC score of the classifier, whose higher value indicates hardness to distinguish between synthetic and real samples.

Method	CS Test		KS Test	
	GAN	VSA+GAN	GAN	VSA+GAN
CTGAN	0.840231	**0.851344**	0.812384	**0.823283**
TableGAN	0.80629	**0.810704**	0.782935	**0.793258**
MedGAN	0.524561	**0.782258**	0.471983	**0.739212**
CW-GAN	0.583926	**0.785138**	0.545365	**0.748275**

Method	Logistic Detection		SVC Detection	
	GAN	VSA+GAN	GAN	VSA+GAN
CTGAN	0.569730	**0.681714**	0.653683	**0.774174**
TableGAN	0.500629	**0.610704**	0.522935	**0.726287**
MedGAN	0.367297	**0.476116**	0.469691	**0.683701**
CW-GAN	0.403902	**0.515739**	0.425532	**0.640632**

Table 1: (i) Statistical Metrics (ii) Detection Metrics

We consistently improve the quality of synthesized data across the models and datasets, often significantly in many cases. Thus we conclude that our proposed framework VSA+GAN, a simple and versatile method for tabular data generative model that leverages self-supervised pretext tasks demonstrated ability to capture and preserve inter-variable correlations.

[1] ML Utility and Correlation Preserving Metrics are attached in appendix

References

Bahri, D., Jiang, H., Tay, Y., and Metzler, D. Scarf: Self-supervised contrastive learning using random feature corruption. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=CuV_qYkmKb3.

Bourou, S., El Saer, A., Velivassaki, T.-H., Voulkidis, A., and Zahariadis, T. A review of tabular data synthesis using gans on an ids dataset. *Information*, 12(09):375, 2021.

Dal Pozzolo, A., Caelen, O., Le Borgne, Y.-A., Waterschoot, S., and Bontempi, G. Learned lessons in credit card fraud detection from a practitioner perspective. *Expert systems with applications*, 41(10):4915–4928, 2014.

Hensinger, E., Flaounas, I., and Cristianini, N. Modelling and predicting news popularity. *Pattern Analysis and Applications*, 16(4):623–635, 2013.

Kohavi, R. et al. Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. In *Kdd*, volume 96, pp. 202–207, 1996.

Poongodi, M., Malviya, M., Kumar, C., Hamdi, M., Vijayakumar, V., Nebhen, J., and Alyamani, H. New york city taxi trip duration prediction using mlp and xgboost. *International Journal of System Assurance Engineering and Management*, 13(1):16–27, 2022.

Sánchez-Morales, A., Sancho-Gómez, J.-L., Martínez-García, J.-A., and Figueiras-Vidal, A. R. Improving deep learning performance with missing values via deletion and compensation. *Neural Computing and Applications*, 32(17):13233–13244, 2020.

Song, K., Tan, X., Qin, T., Lu, J., and Liu, T.-Y. Mpnet: Masked and permuted pre-training for language understanding. *Advances in Neural Information Processing Systems*, 33:16857–16867, 2020.

Wang, X., Chou, J.-K., Chen, W., Guan, H., Chen, W., Lao, T., and Ma, K.-L. A utility-aware visual approach for anonymizing multi-attribute tabular data. *IEEE transactions on visualization and computer graphics*, 24(1): 351–360, 2017.

Wang, Y. and Zhao, Q. House price prediction based on machine learning: A case of king county. In *2022 7th International Conference on Financial Innovation and Economic Development (ICFIED 2022)*, pp. 1547–1555. Atlantis Press, 2022.

A1. Additional Results: ML Utility

ML utility measures how well the synthetic data can be the proxy of the target data in the machine learning tasks. It is measured as the performance on original test datasets of machine learning models which are trained on the generated data. The original and synthetic data are evaluated by 5 widely used machine learning algorithms: decision tree classifier, linear support-vector-machine (SVM), random forest classifier, multinomial logistic regression and multi-layerperceptron (MLP). The value of the metric is determined not only by the quality of our synthetic data, but also by the difficulty of the machine learning problem that we are attempting to solve. Synthetic data generated from our method does 14.52% better than that of the baseline

A2. Additoinal Results : Correlation Metrics

Correlation metrics evaluate how well inter-feature interactions are preserved in the generated records. Pair-wise correlation matrices for columns within real and synthetic datasets are first computed individually. Pearson and Theil uncertainty coefficients are used to measure the correlation between numerical and categorical columns respectively. The L_2-norm of the difference between these correlation matrices for the real and synthetic datasets is considered the final metric. Our method captures correlations a lot better in both numerical and categorical columns.

Table A1: ML Utility

Method	Continuous		Categorical	
	GAN	VSA+GAN	GAN	VSA+GAN
CTGAN	0.677608	**0.743092**	0.472855	**0.506979**
TableGAN	0.396333	**0.435885**	**0.350966**	0.327833
MedGAN	0.424052	**0.458852**	0.346336	**0.482111**
CW-GAN	0.652449	**0.697192**	0.427002	**0.633291**

Table A2: Correlation Difference Metrics

Method	Pearson Correlation		Theil Correlation	
	GAN	VSA+GAN	GAN	VSA+GAN
CTGAN	2.43	**2.26**	2.30	**2.12**
TableGAN	2.26	**1.99**	2.32	**1.95**
MedGAN	5.52	**4.37**	4.72	**4.66**
CW-GAN	6.24	**5.83**	5.89	**5.83**

A3. Ablation Study

To understand the efficiency of each component in VSA+GAN, we implement an ablation study with machine learning utility metrics. Table 3 shows the results from the ablation study. We try to understand VSA+GAN by ablating the masking and corruption technique we proposed, replacing it with the following promising strategies, while keeping all else fixed.

Settings	Performance
No corruption	-53.26%
Feature Dropout	-32.12%
Mean corruption	-12.56%
Gaussian Noise	-48.21%

Table A3: Ablation study results on different masking and corruption schemes. The performance (measured as ML utility) changes on real-world datasets are reported.

1. *No corruption*. We do not apply any corruption - i.e. $\boldsymbol{x'} = \boldsymbol{x}$, in Equation 1. In this case, the autoencoder is not imputing missing values but simply reconstructing data samples. It ends up learning weaker correlation structures between features and we see that with no corruption the performance of the generator is not improved.

2. *Feature dropout.* The features selected to corrupt are zeroed-out.

3. *Mean corruption.* The selected features are replaced with the empirical marginal distribution's mean.

4. *Additive Gaussian noise.* We add i.i.d $N(0, 0.25)$ noise to features

Our masking and feature resampling method outperforms the other standard corruption strategies. We also observed that our corruption strategy is fairly insensitive to the corruption rate, seeing a stable performance when the rate is in the range 30-60%. The pretext task is also not sensitive to batch-size