MeTGAN: Memory efficient Tabular GAN for high cardinality categorical datasets

Shreyansh Singh, Kanishka Kayathwal, Hardik Wadhwa, and Gaurav Dhama

AI Garage, Mastercard, India
{shreyansh.singh,kanishka.kayathwal,hardik.wadhwa,gaurav.dhama}@mastercard.com

Abstract. Generative Adversarial Networks (GANs) have seen their use for generating synthetic data expand, from unstructured data like images to structured tabular data. One of the recently proposed models in the field of tabular data generation, CTGAN, demonstrated state-of-the-art performance on this task even in the presence of a high class imbalance in categorical columns or multiple modes in continuous columns. Many of the recently proposed methods have also derived ideas from CTGAN. However, training CTGAN requires a high memory footprint while dealing with high cardinality categorical columns in the dataset. In this paper, we propose MeTGAN, a memory-efficient version of CTGAN, which reduces memory usage by roughly 80%, with a minimal effect on performance. MeTGAN uses sparse linear layers to overcome the memory bottlenecks of CTGAN. We compare the performance of MeTGAN with the other models on publicly available datasets. Quality of data generation, memory requirements, and the privacy guarantees of the models are the metrics considered in this study. The goal of this paper is also to draw the attention of the research community on the issue of the computational footprint of tabular data generation methods to enable them on larger datasets especially ones with high cardinality categorical variables.

Keywords: Synthetic Data Generation · Generative Adversarial Networks · Tabular Data · Privacy

1 Introduction

Despite the growing interest in areas like natural language processing and computer vision, the type of datasets have diversified considerably. However, tabular datasets remain prominent for a number of scenarios like patient records, banking records, census data, etc. Often, due to privacy concerns, these datasets cannot be used to train machine learning models as that would make it easy for attackers to extract details about the original data. Furthermore, the lack of sufficient real data for training the machine learning models also creates a need for synthetic data generation methods.

Tabular data usually consists of both numerical and categorical columns. Real-world numerical data is typically multi-modal and categorical columns are often imbalanced. These two issues make synthetic tabular data generation a difficult

task. GANs have shown a promising ability to model arbitrary distributions in the data for a variety of datasets. In this paper, we focus primarily on the GAN-based methods for tabular data generation. Some of the recently proposed and well known models in the synthetic tabular data generation domain include medGAN [2], tableGAN [10], TGAN [20], CTGAN [19] and cWGAN [4]. We compare the performance of these models on two publicly available datasets from the UCI Machine Learning repository namely the Adult [7] (classification data) and News [15] (regression data) datasets. CTGAN was found to be the best performer among these models. The Adult and News datasets although widely used in synthetic data generation papers like [19], [10], do no have high cardinality categorical columns and hence are not very close to what large datasets would look like. CTGAN creates a one-hot encoded representation for all the categorical columns in the conditional vector and uses a residual connection in the first layer of the generator. This creates a memory bottleneck, mainly while dealing with a large number of categories in the dataset.

As demonstrated in the experiments section, it was observed that for the Loan [18] dataset, CTGAN has a very high memory footprint. To overcome this, we propose a new architecture MeTGAN, wherein we use a sparse linear layer for the conditional vector in both the generator and discriminator. Additionally, we remove residual connections in the first layer of the generator. This helps to get an 80% reduction in memory usage that makes it easy to generate synthetic data for large datasets. The proposed model performs at par with CTGAN across all the datasets. To summarise, the paper's contributions are:

- Propose a memory-efficient tabular GAN architecture to handle datasets with high cardinality categorical columns with at par performance as compared to the current state of the art algorithms
- A thorough comparison of the current existing GAN-based tabular data generation methods with MeTGAN on several metrics, machine learning efficacy, distribution similarity, privacy guarantees, and the memory requirements.
- Draw the attention of the research community to the computational requirements of such models and motivate researchers to work towards solutions.
- Additional details regarding the paper are available at: `https://github.com/shreyansh26/MeTGAN-Paper`.

2 Related Work

Initial contributions in the field of tabular data generation treated each column in the table as a distinct random variable and their distributions were modeled separately. [14] used decision trees to model the categorical variables while [21] used Bayesian networks. In [3] spatial data was modeled using trees. Continuous variables that were non-linearly related were modeled using copulas in [11].

Later, the use of GANs [5] started to emerge in the domain of synthetic tabular data generation. medGAN [2] uses an autoencoder with a GAN model to generate medical patient records with heterogeneous non-time-series continuous and/or binary data. tableGAN [10] uses a DCGAN [13] based architecture for

synthetic data generation. TGAN [20] uses an LSTM based generator and an MLP based discriminator while CTGAN [19] uses a conditional generator and training-by-sampling to deal with the imbalanced categorical columns. CTGAN also uses mode-specific normalization to handle the non-Gaussian and multi-modal distribution of continuous data. The architecture integrates PacGAN [8] and uses Wasserstein loss with gradient penalty [6]. cWGAN [4] models the data using self-conditioning in the generator and uses crosslayers for both the generator and the discriminator to explicitly calculate the feature interactions. Additionally embedding layers are used for dimensionality reduction and better representative power. In [19], in addition to CTGAN, the authors also propose a variational autoencoder-based approach, TVAE for synthetic tabular data generation. CrGAN-Cnet [9] uses GAN to perform synthetic airline passenger name record generation. CTAB-GAN [22] is a very recent paper that can effectively model columns that have a mix of continuous and categorical data. It introduces the information loss and classification loss to the conditional GAN. We were not able to reproduce the CTAB-GAN paper ourselves correctly and hence we do not include the results in our paper. However, from their paper, we understand that, architecturally, CTAB-GAN will not be more memory efficient than CTGAN, so not including the model in this paper will not affect the goal of our study.

3 Proposed Model

In this section, we present the MeTGAN architecture, which is designed to overcome the shortcomings of CTGAN on datasets with high cardinality. The conditional vector in CTGAN is sparse, most of its values are zero, ones being only at the places corresponding to the conditioned columns of the real data. This conditional vector is concatenated with a noise vector before passing it as an input to the generator. Similarly, in the case of the discriminator, the input is the concatenation of the conditional vector with either the generated data or with the real data. For datasets with high cardinality, this concatenated vector becomes large and leads to a memory bottleneck. Moreover, in the generator, an addition of a residual layer over this large vector further escalates the memory requirements.

In MeTGAN (architecture shown in Figure 1), we address the above-mentioned issues as follows: First, we do not concatenate the conditional vector, neither with the noise vector in the case of the generator nor with real/generated data in the case of the discriminator. Second, we remove the first residual layer of the generator. In the case of the generator, the noise vector is passed through a linear layer and the conditional vector is passed through a separate sparse linear layer. The outputs from these layers are concatenated before passing it to the next layer. For discriminator, the real/generated data is passed through a fully connected layer with LeakyReLU activation and Dropout, whereas for the conditional vector the fully connected layer is replaced by a sparse linear layer. The outputs of these layers are also concatenated before passing it to the next layer.

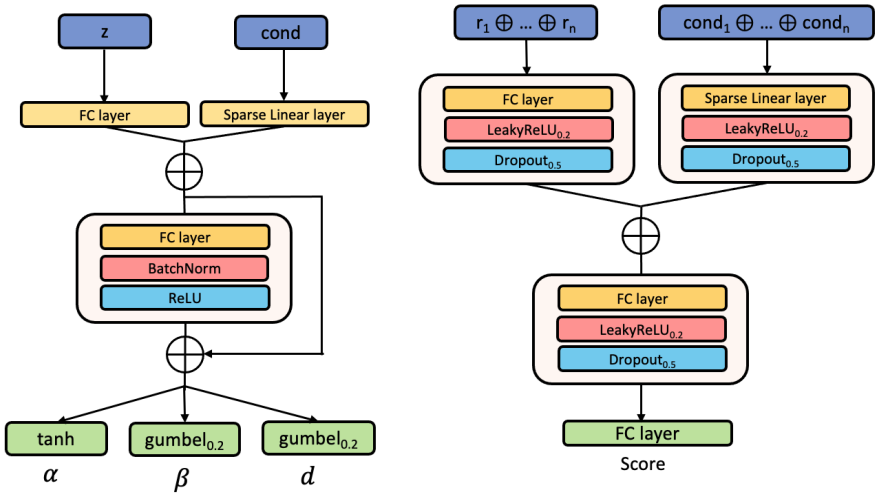

Fig. 1: Architecture of the generator (left) and discriminator (right) of MeTGAN. $\bigoplus$ indicates concatenation. The input z to the generator is the noise vector sampled from a standard multivariate normal distribution. *cond* is the conditional vector that represents the column name and value that the generator conditions upon. For the discriminator, a PacGAN [8] framework with n samples in each pac is used. $r_1, ..., r_n$ is the real or generated data which is passed through the discriminator, and $cond_1, ..., cond_n$ is used for the conditional vector. The outputs β, α indicate the one-hot vector of the mode and the value within the mode for continuous columns and d is the one-hot vector of the discrete values for categorical columns. The *Score* from the discriminator is the critic score indicating the realness of the input data given the conditional vector.

4 Datasets

For the comparison of the models, two publicly available datasets were considered. The Adult dataset and the News dataset. The objective of taking these two datasets is that 1) They are widely used in synthetic tabular data literature [19], [20], [10] and 2) the Adult dataset poses a classification problem and the News dataset presents a regression problem, which can help to better analyze the performance of the models on the two major facets of machine learning on tabular data - regression and classification. The statistics associated with the datasets are shown in Table 1.

To compare the performance of CTGAN and MeTGAN on a dataset with high cardinality, we used the Loan dataset. A sample size of 150k data points was selected from the Loan dataset such that it had a sufficiently large number of total distinct categories across columns (21k) to test the memory efficacy of MeTGAN. The statistics of the Loan dataset are also shown in Table 1.

Name	Train/Test	#N	#C	#Categories
Adult	26k/6.5k	6	9	104
News	31k/8k	45	14	28
Loan	120k/30k	11	10	21k

Table 1: Dataset statistics. #N and #C represent the number of continuous (numerical) and categorical columns in the dataset respectively. #Categories is the total number of categories across all the categorical columns.

Additional details of the features from each dataset used in our experiments and the target feature used for the Machine Learning Efficacy tests are described in our Github repository.

5 Metrics

5.1 Non-Privacy Metrics

Memory Requirements - The GPU memory requirements of the MeTGAN and CTGAN models on the Loan dataset is one of the key metrics.

Machine Learning Efficacy - As defined in [19]. For the Adult dataset, we used Logistic Regression (LR) and Random Forest Classifier (RFC) and for News and Loan we used Random Forest Regressor (RFR) and ElasticNet (ENet). In both cases, we reported the difference (ΔF1 or ΔMSE) of the ML model. The smaller the difference, the better the ML efficacy.

Statistical Similarity - The statistical closeness between the two datasets is shown using the Chi-squared (CS) test and the Kolmogorov–Smirnov (KS) test.

5.2 Privacy Metrics

DCR - [10] introduced a metric - Distance to Closest Record (DCR) which we use in this paper as well to measure privacy.

CategoricalCAP (CCAP) - This is based on the Correct Attribution Probability privacy method. It requires a set of key columns and a sensitive column for a dataset. We use the [1] package for the implemtation. The closer the CCAP score is to one, the better is the privacy. CCAP was used for the Adult data.

NumericalLR (NLR) - This method also requires a set of key columns and a sensitive column. We use the [1] package for the implemtation. Again, here higher the score (closer to 1), the better is the model performance. NumericalLR was used for the News and Loan datasets.

6 Results and Discussion

We study five GAN baselines, on Adult and News datasets, to compare the performance of the proposed MeTGAN model. Additionally, MeTGAN and CTGAN were studied on the Loan dataset keeping a focus on memory usage. We used an NVIDIA Quadro RTX 6000 for all our experiments. The training parameters for all the models and the key and sensitive columns required for CCAP and NLR metrics are mentioned in our Github repository.

The performance of the models on the Adult dataset is reported in Table 2. In terms of ML Efficacy, except medGAN, all models have a reasonably good performance. The MeTGAN model stands out in this test among all the models. TGAN performs well particularly in the CS and KS tests while MeTGAN and CTGAN have the next best performance. MeTGAN has a good CategoricalCAP score and is comparable to CTGAN. Overall, CTGAN and MeTGAN are the most balanced methods across all the metrics for synthetic data generation on this dataset, with MeTGAN performing slightly better on ML Efficacy than CTGAN.

Table 3 shows the performance of the models on the News dataset. On this dataset, CTGAN has the best ML Efficacy performance. MeTGAN and medGAN are next with reasonably good performance as well. cWGAN does not support regression tasks so it is not evaluated on the News dataset. medGAN has a good KS test score on this dataset. MeTGAN has a higher DCR-mean, and the relatively good NumericalLR score confirms that the higher DCR-mean is a positive sign. Although medGAN has a high KS test score, the low NumericalLR score indicates that it somehow leaks information. Overall, again both CTGAN and MeTGAN have a stable performance.

Table 4 shows the results of CTGAN and MeTGAN on the Loan dataset. It can be seen from the ΔMSE score that both CTGAN and MeTGAN do not perform well on this dataset in terms of ML Efficacy score. In the ML Efficacy metrics, MeTGAN outperforms CTGAN on the Random Forest regressor test and marginally loses out on the ElasticNet test. CTGAN has a slightly better performance on the CS and KS tests. In the privacy metrics, the performance of CTGAN and MeTGAN are very close to each other.

The main thing to note however is that for this high cardinality categorical dataset, the MeTGAN model requires 80% less memory (from Table 5) to give a similar performance as CTGAN. The use of the sparse layers for the model resulted in memory-efficient matrix operations over the sparse conditional vector. Additionally, removing the first residual layer from the generator helped to reduce the size of the input to the next layer. These two reasons primarily caused the reduction in GPU memory usage while not hampering the performance of the model significantly. Overall, MeTGAN performed at par or even marginally better than other models across different metrics on all the datasets.

Dataset	Model	ΔF1-LR	ΔF1-RFC	CS test	KS test	DCR-mean	DCR-std	CCAP
Adult	MeTGAN	**0.013**	**0.028**	0.987	0.813	1.711	0.706	0.300
	CTGAN	0.030	0.035	0.970	0.773	1.782	0.732	**0.310**
	TGAN	0.085	0.087	**0.990**	**0.917**	1.742	0.727	0.304
	cWGAN	0.051	0.030	0.906	0.596	1.890	0.619	0.232
	medGAN	0.719	0.725	0.872	0.109	**2.246**	0.975	0.035
	tableGAN	0.036	0.052	0.976	0.610	1.826	**0.537**	0.303

Table 2: Results on the Adult dataset

Dataset	Model	ΔMSE-RFR	ΔMSE-ENet	CS test	KS test	DCR-mean	DCR-std	NLR
News	MeTGAN	154	125	NA	0.915	**3.041**	5.433	0.263
	CTGAN	**87**	**106**	NA	0.910	2.903	5.430	**0.279**
	TGAN	408	2491	NA	0.902	2.787	5.389	0.267
	medGAN	128	126	NA	**0.965**	2.026	5.436	0.182
	tableGAN	540	262	NA	0.585	2.741	**5.235**	0.217

Table 3: Results on the News dataset

Dataset	Model	ΔMSE-RFR	ΔMSE-ENet	CS test	KS test	DCR-mean	DCR-std	NLR
Loan	MeTGAN	**64015**	23320	0.632	0.716	**3.007**	**1.234**	0.093
	CTGAN	74323	**23082**	**0.656**	**0.755**	2.746	2.626	**0.115**

Table 4: Results on the Loan dataset

Dataset	Model	GPU Mem.
Loan	MeTGAN	**3.2 GB**
	CTGAN	16.3 GB

Table 5: GPU Memory usage on the Loan dataset

7 Conclusion and Future Work

In this paper, we propose MeTGAN, a memory-efficient approach of synthetic tabular data generation, that performs at par with other recent GAN approaches. In addition, for large datasets, MeTGAN considerably reduces the memory consumption without any significant degradation in performance. Reducing memory footprint of models has recently gained popularity, like [16] for large pre-trained language models, [12] for speech recognition, [17] for computer vision, but not much work has been done in the domain of synthetic tabular data generation. As per our knowledge, MeTGAN is the first work in this direction. Additionally, for future work, it can be seen that on a high cardinality categorical dataset like the Loan dataset, both CTGAN and MeTGAN models have limited efficacy. With the current architecture of these models, they are not yet capable of capturing the distribution well when there are a large number of columns and categories. Certain architectural changes are required that can better capture the nuances in variance. Future work directions could include the intersection of these aspects i.e., improving the memory footprint of such algorithms and/or making more real-world usable models.

References

1. Data to AI Lab, M.: Sdmetrics (2020), https://github.com/sdv-dev/SDMetrics
2. Choi, E., Biswal, S., Malin, B., Duke, J., Stewart, W.F., Sun, J.: Generating multi-label discrete patient records using generative adversarial networks. In: Proceedings of the 2nd Machine Learning for Healthcare Conference. vol. 68. PMLR (2017)
3. Cormode, G., Procopiuc, C., Srivastava, D., Shen, E., Yu, T.: Differentially private spatial decompositions. In: 2012 IEEE 28th International Conference on Data Engineering. pp. 20–31 (2012). https://doi.org/10.1109/ICDE.2012.16
4. Engelmann, J., Lessmann, S.: Conditional wasserstein gan-based oversampling of tabular data for imbalanced learning. Expert Systems with Applications **174** (2021). https://doi.org/10.1016/j.eswa.2021.114582
5. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial networks (2014)
6. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.: Improved training of wasserstein gans. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. p. 5769–5779. NIPS'17, Curran Associates Inc., Red Hook, NY, USA (2017)
7. Kohavi, R., Becker, B.: Adult data set (May 1996), https://bit.ly/3v3VDIj
8. Lin, Z., Khetan, A., Fanti, G., Oh, S.: Pacgan: The power of two samples in generative adversarial networks. IEEE Journal on Selected Areas in Information Theory **1**, 324–335 (2020)
9. Mottini, A., Lheritier, A., Acuna-Agost, R.: Airline passenger name record generation using generative adversarial networks. CoRR **abs/1807.06657** (2018)
10. Park, N., Mohammadi, M., Gorde, K., Jajodia, S., Park, H., Kim, Y.: Data synthesis based on generative adversarial networks. Proc. VLDB Endow. **11**(10), 1071–1083 (Jun 2018). https://doi.org/10.14778/3231751.3231757
11. Patki, N., Wedge, R., Veeramachaneni, K.: The synthetic data vault. pp. 399–410 (10 2016). https://doi.org/10.1109/DSAA.2016.49
12. Peng, Z., Budhkar, A., Tuil, I., Levy, J., Sobhani, P., Cohen, R., Nassour, J.: Shrinking bigfoot: Reducing wav2vec 2.0 footprint (2021)
13. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks (2016)
14. Reiter, J.: Using cart to generate partially synthetic, public use microdata. Journal of Official Statistics **21** (01 2005)
15. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: A proactive intelligent decision support system for predicting the popularity of online news. Proceedings of the 17th EPIA 2015, Coimbra, Portugal (2015)
16. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. ArXiv **abs/1910.01108** (2019)
17. Tan, M., Le, Q.V.: Efficientnetv2: Smaller models and faster training (2021)
18. Toktogaraev, M.: Should this loan be approved or denied?, https://bit.ly/3AptJaW
19. Xu, L., Skoularidou, M., Cuesta-Infante, A., Veeramachaneni, K.: Modeling tabular data using conditional gan. In: NIPS (2019)
20. Xu, L., Veeramachaneni, K.: Synthesizing tabular data using generative adversarial networks. arXiv preprint arXiv:1811.11264 (2018)
21. Zhang, J., Cormode, G., Procopiuc, C.M., Srivastava, D., Xiao, X.: Privbayes: Private data release via bayesian networks
22. Zhao, Z., Kunar, A., der Scheer, H.V., Birke, R., Chen, L.Y.: Ctab-gan: Effective table data synthesizing (2021)