E-ISSN: 2715-2731

Integrating Machine Learning Utility in Tabular Data Synthesizer Training using Loss Function Learning

Muhammad Rizqi Nur[1,*], Rarasmaya Indraswari[2]

Department of Information Systems, Institut Teknologi Sepuluh Nopember, Indonesia

[1]rizqinur2010@gmail.com; [2] raras@its.ac.id

*corresponding author

ARTICLE INFO

Article History
Received: 9 July 2024
Revised: 5 August 2024
Published: 30 August 2024

Keywords
Loss Learning
Set Processing
Tabular Data Synthesis
Transformer

ABSTRACT

Machine learning (ML) utility has been the main evaluation metrics for data synthesizers. However, because ML utility cannot be simply calculated, none of the previous synthesizers were trained to reach the same level of ML utility as a training objective. This study aims to integrate ML utility into data synthesizer training using a transformer-based model as a learned loss function. The transformer was trained to estimate ML utility of synthetic datasets, then it's integrated by backpropagating the difference between estimated and expected value. The integration has significantly improved the average ML utility of LCT-GAN and Realtabformer. The ML utility of LCT-GAN improved by 0.0158 for Contraceptive dataset, 0.031 for Insurance dataset, and 0.0561 for Treatment dataset. The ML utility of Realtabformer improved by 0.02 for Contraceptive dataset and 0.0024 for Insurance dataset. The increase affects the dataset distribution, correlation between features, and privacy, but the direction varies. Correlation coefficients indicate that synthetic data distribution gets closer to real data as ML utility improves. In addition to ML utility integration, this study has also shown that patterns between rows in a dataset can be learned, so better synthesizers can be developed based on them.

INTRODUCTION

Research using a company's data is often done collaboratively with a third party [1]. However, this became hard to do since General Data Protection Regulation (GDPR) came into existence. GDPR requires companies to protect their customers' or users' data privacy and may only share their data with their consent as the rightful owner. GDPR only applies to the European Union, but since 2022, Indonesia also has private data protection law [2]. Therefore, companies must be careful in sharing data that concerns their customers or users. Even when personal information has been removed, the information owner may still be identified by connecting it to other information, such as from the internet [3]. Data synthesis can be an alternative to share data without risking user privacy, or at least without breaking the law. However, despite having smaller dimensions than other kinds of data, such as image data, tabular data is not easy to synthesize. Tabular data may have many categorical variables [4]. Even the numerical variable may have multimode or long tail distribution, or even be a mixed type [3]–[5].

Several data synthesizers based on deep learning have been proposed [3]–[8]. Machine learning utility (ML utility / MLU) – such as the accuracy and F1-score for classification task [3] – has been the main and most common evaluation metrics because the main concern with synthetic data is whether it can lead to similar inference as the real data [9], [10]. ML utility is a measure of data synthesizer performance measured by performance of a model

trained with synthetic data and tested with real data [3], [6], [7]. However, none of the previous models has integrated ML utility in training as one of the training objectives. This may be because ML utility cannot be simply calculated. This study aims to integrate ML utility into data synthesizer training using a transformer-based model as a learned loss function. [11]–[13]. Within the authors' knowledge, none of the previous studies have used ML utility as a training objective. The closest one was TabDDPM which used it as tuning objective [6].

The learned loss model is based on transformer [14] with modifications from set transformer [15]. Transformer was used because it is necessary to recognize patterns between rows within a dataset, in which each row consists of multiple values, the row order must not matter, and each row contributes differently to the prediction performance (ML utility). Transformer is quite slow and costly, but for now, the main concern is to see whether it is doable. The transformer was trained on synthetic datasets to estimate their ML utility. From this point onwards, the model will be called estimator. ML utility is measured using a fine-tuned Catboost model, because it was found to be more robust than simpler models [6]. The estimations are then compared to the expected ML utility value and the error is backpropagated into the synthesizer. The result of the integration is evaluated by comparing metrics before and after integration: ML utility, distribution, correlation between features, and privacy. The rest of this paper is structured as follows. The second part describes the research method used in this study. The third part describes and discusses the result. The last part concludes this paper.

METHODS

Synthesizer Selection

A literature review was done for tabular data synthesizers based on deep learning starting from 2019. For each type of synthesizer found, a synthesizer that stated advantage from the previous models is chosen to have its ML utility estimated. Four synthesizers were chosen: TVAE [5], LCT-GAN [16], TabDDPM [6], and ReaLTabFormer [17]. TVAE is a VAE-based tabular data synthesizer which was introduced as competition for CT-GAN evaluation [5]. LCT-GAN is a GAN-based tabular data synthesizer which tries to mitigate dimension explosion caused by one-hot encoding by synthesizing data in latent space, resulting in faster convergence [16]. TabDDPM is the first DDPM-based tabular data synthesizer which claimed superior performance to the previous GAN-based models [6]. Realtabformer is a transformer-based tabular data synthesizer for relational data, but it can also be used for singular tables and claims to have decent performance without requiring fine-tuning [17]. These four synthesizers are all tabular data synthesizers based on deep learning. They do not have the same level of complexity or performance, but they are not meant to be compared against each other. Instead, this study compares each synthesizer before and after the ML utility integration.

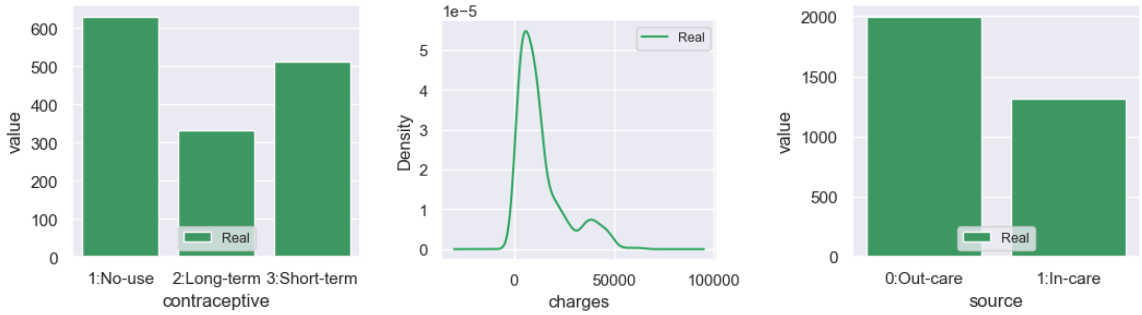
Data Collection

This study used publicly available datasets which are usually private, such as health and business datasets, because they tend to contain sensitive private information [18], [19]. The datasets are shown in Table 1. The contraceptive dataset [20] task is to predict whether a non-pregnant married woman uses long term, short term, or no contraceptives (multiclass classification task). The insurance dataset [22] task is to predict the insurance medical cost (regression task). The treatment dataset [21] task is to predict whether a person should be an in-care or out-care patient (binary classification task). Each dataset is not balanced (

Figure 1); synthesizers are expected to reproduce the distributions, including imbalanced ones, though it is not the concern of this study.

Table 1. Datasets

Dataset	Task	Rows	Attributes
Contraceptive Prevalence Survey [20]	Classification (Multiclass)	1.473	10
Insurance [22]	Regression	1.338	7
Patient **Treatment** [21]	Classification (Binary)	3.309	11



Contraceptive Insurance Treatment

Figure 1. Distribution of the Datasets' Target Variables

Data Preparation

The data was split randomly into 5 folds of training and test sets with ratio of 80:20. To train the estimator, for each model and fold (out of the 5 folds of datasets), 6 synthetic datasets were generated during synthesizer training every 10% epoch or step. This is done to produce training datasets with wider ML utility value range by purposefully synthesizing progressively worse datasets. This resulted in 1200 synthetic datasets, which were then split into 800 training datasets, 200 validation datasets, and 200 testing datasets. The 5 folds of real datasets were then added to training datasets so the estimator would have also seen the optimal dataset. All of them had their ML utility measured by R^2 or F1-score of a fine-tuned Catboost model [6]. Preprocessing is done according to each synthesizer as shown in Table 2. Realtabformer [17] preprocessing is treated specially by adding an embedding layer. During estimator training, the embedding layer is from a trained model, while during inference, the embedding layer is from the model being trained.

Table 2. Preprocessing Methods of the Synthesizers

Model	Preprocessing variable				
	Continuous			Categorical	
	Simple	Multimode	Long tail	Small	Big
TVAE [5]	Mode-specific normalization			One-Hot Encoding	
LCT-GAN [16]	Min-max normalization [-1, 1]	Mode-specific normalization + Mode Vector	Log	One-Hot Encoding	Integer + Min-max normalization [-1, 1]
TabDDPM [6]	Gaussian Quantile Transformation			One-Hot Encoding	Integer Encoding
REaLTabFormer [17]	Rounding + String Conversion + Split			String Conversion	
	Tokenization and embedding				

Estimator Model Development

An estimator model is trained to estimate the ML utility of a dataset. The estimator is made by modifying transformer encoder [14] to use Induced Set Attention Block (ISAB) and Pooling by Multihead Attention (PMA) from set transformer [15]. Transformer [14] is a sequence-to-sequence NLP model that primarily uses attention mechanism, which works as if it calculates weights for each input token based on input tokens. Meanwhile the Set

Transformer is a variant that was proposed for set to set processing [15]. The architecture is shown on Figure 2. Every multi-head attention from the original transformer encoder is replaced by ISAB. ISAB is necessary to reduce the attention computation complexity by doing the computation twice, but within smaller dimensions [15]. This is done using a vector called induction point (Figure 3a). Transformer encoder produces a sequence of output, or in this case a set. It will be fed into dense head layers, so it needs to be pooled. The pooling is done using PMA, which weighs each row differently by attention [15]. PMA works similarly to ISAB using a vector called seed (Figure 3b). Positional encoding is removed because the position of samples in a dataset is meaningless [15]. Embedding layer is changed into dense encoder layer. The synthetic dataset takes the role of training set while the test set is real data (Figure 4a). They're both processed by the same encoder and the results are combined by subtraction and flattened into one vector to be processed by dense head layer which will output the estimation [23]. For regression, R^2 score can be negative so the model output, the estimated value, is transformed to $(-\infty, 1]$ range using Eq. (1. The main loss function used is Mean Squared Error (MSE) between the real loss and the estimated loss [12].

$$f(x) = 1 - x \quad (1)$$

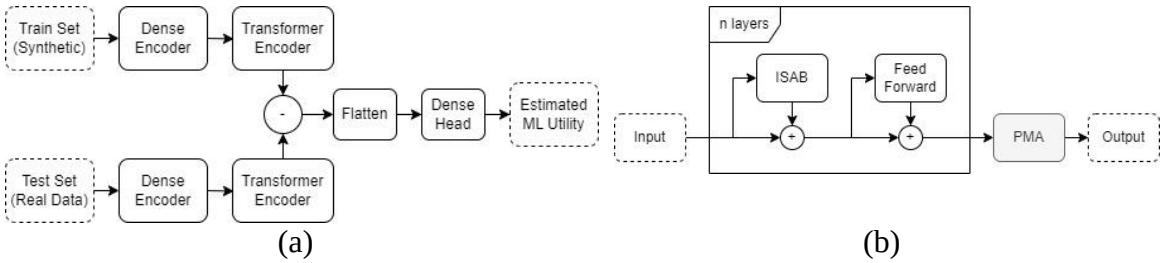


Figure 2 (a) ML Utility Estimator Architecture; and (b) its Transformer Encoder Architecture

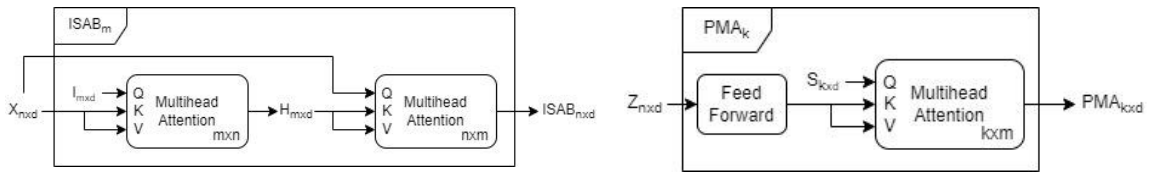


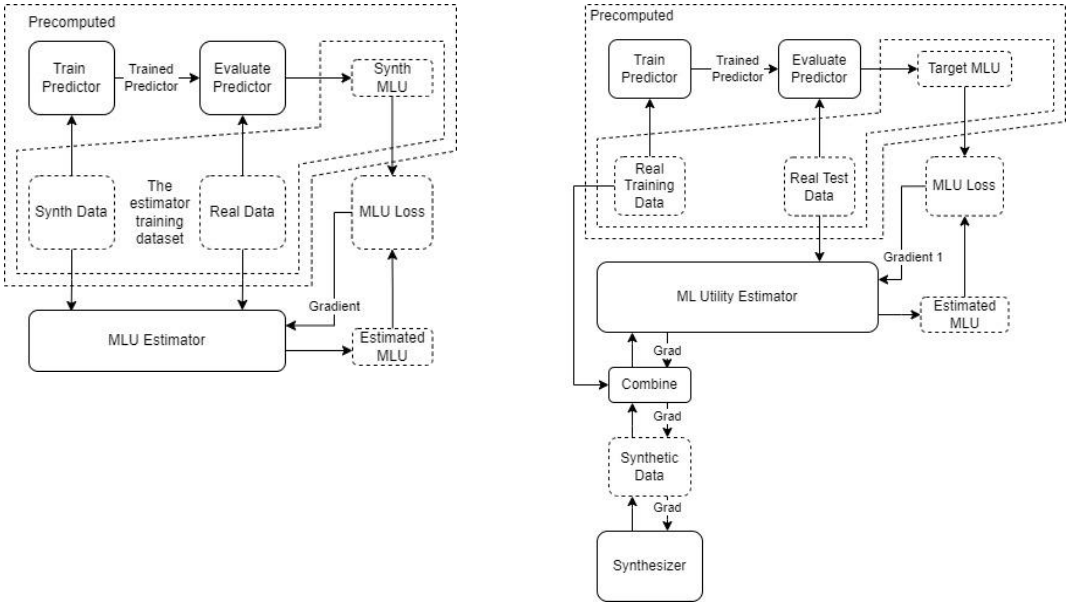
Figure 3 (a) *Induced Set Attention Block* (ISAB); and (b) *Pooling by Multihead Attention* (PMA)

A good loss function should scale smoothly with the loss [24], [25]. Gradient penalty [26] is used to shape the gradient to have magnitude that scales with the mean absolute error (MAE) of the estimation with certain scaling (hyperparameter). The evaluation of the ML utility estimator is done for each dataset and synthesizer pair using the previously separated test set. The evaluation is repeated 5 times with index as seed. The average of RMSE (Root Mean Square Error) between the estimated and actual ML utility are reported.

Estimator Integration

The ML utility estimator is then integrated into the training of data synthesizers as a learned loss function (Figure 4b). It is done by backpropagating the "error" between estimated ML utility and expected ML utility every few epochs or steps. This is done by generating a certain number of synthetic data, combining it with sampled real data to make full-sized dataset, then having its ML utility estimated. The estimation is compared to the expected value and the "error" is propagated back to the synthesizer. This process can be repeated a few times and done every few epochs or steps. The frequency, repetition, generation sample size, and target value are hyperparameters chosen through

hyperparameter tuning. The synthetic dataset is not generated with full size due to hardware limitation, because it may take a very long time and the graph for backpropagation may require very large amount of memory. The synthetic data are assumed to be unable to surpass the ML utility of the real data, so if they're estimated to surpass the target value, it is regarded as estimation error and will not be backpropagated.



(a) (b)

Figure 4 (a) The Proposed ML Utility Estimator Training; and (b) The Proposed ML Utility Estimator Integration into Synthesizer Training

Evaluation

The effect of ML utility integration is evaluated by comparing the average ML utility of each synthesizer before and after the integration. The evaluation is done 5 times for each synthesizer-dataset pair; once for each fold out of the 5 folds of a dataset. In addition, the effect towards distribution, correlation between features, and privacy are also measured and compared similarly. The evaluation is done with metrics commonly used in previous studies (Table 3). F1-score is used to measure ML Utility for classification datasets, while R^2 score is used to measure ML utility for regression datasets. Their optimal value is 1, but R^2 can be negative. Wasserstein Distance (WD) is used to measure the distribution similarity for numerical variables, while Jensen-Shannon Divergence (JSD) is used for categorical variables. They are distance/divergence metrics, so the optimal value for similarity is 0. Diff. Corr. is calculated by Pearson correlation for continuous features and Theil uncertainty coefficient for categorical features [3]. Correlation ranges from -1 to 1, so the difference ranges from 0 to 2, and the optimal value is 0. DCR is the average of 5% smallest distance between a synthetic data and real data closest to it [4], [6]. Low DCR means synthetic data is individually similar to real data, which means privacy level is low, so the larger the value the better the privacy.

Table 3. Evaluation Metrics and Previous Studies which Used The Metrics

Criterion		Metric	Range		Direction	Previous Studies
			Min	Max		
ML Utility / Efficacy	Regression	R^2	-∞	**1**	Maximize	[5], [6], [17]
	Classification	F1-score	0	**1**	Maximize	[4]–[6], [16], [17]
Distribution Similarity	Numerical	Wasserstein Distance (WD)	**0**	∞	Minimize	[3], [4], [16]
	Categorical	Jensen-Shannon Divergence (JSD)	**0**	1	Minimize	
Correlation between features		Difference in Pairwise Correlation (Diff. Corr.)	**0**	2	Minimize	[3], [4], [6], [16]
Privacy		Distance to Closest Record (DCR)	0	∞	Maximize	[4], [6]

RESULTS AND DISCUSSIONS

Dataset Preparation

To train the estimator, for each model and fold (out of the 5 folds of datasets), 6 synthetic datasets were generated during synthesizer training every 10% epoch or step. This resulted in 1200 synthetic datasets. All of them had their ML utility scored by a fine-tuned Catboost model. The bad synthetic datasets generally came from LCT-GAN, so this might result in a bias (Table 4). The datasets were then split into 800 training datasets, 200 validation datasets, and 200 testing datasets. Sample synthetic data is shown in Table 5. Although they look almost real individually, they might not have similar distribution, correlation, or predictive performance (Table 4).

Table 4. Descriptive Statistics of the ML Utility of Synthetic Datasets per Synthesizer for Estimator Training

Synthesizer	Count	ML Utility					
		Contraceptive		Insurance		Treatment	
		F1		R^2		F1	
		Mean	Stdev	Mean	Stdev	Mean	Stdev
LCT-GAN	300	**0.2667**	0.0789	**-0.1622**	0.2006	**0.1443**	0.1936
Realtabformer	300	0.4533	0.0513	0.1317	0.0104	0.6008	0.0448
TabDDPM	300	0.4684	0.0436	0.1301	0.0095	0.5895	0.0373
TVAE	300	0.4283	0.0695	0.0534	0.1012	0.5513	0.0622

Table 5. Sample of real and synthetic insurance data (additional trailing zeros marked by bold)

Model	Data						
	Age	Sex	Bmi	Children	Smoker	Region	Charges
LCT-GAN	21	female	29.809109	0	no	southwest	2467.160142
Realtabformer	51	female	36.385**000**	3	no	northwest	11436.73815
TabDDPM	45	male	39.730782	0	no	northeast	7448.40395**0**
TVAE	21	female	34.795737	0	no	southeast	2157.665097
Real	46	male	24.795**000**	3	no	northeast	9500.57305**0**

Estimator Model Development

For each model and each fold (out of the 5 folds of datasets), an estimator was trained for 1 hour maximum and the checkpoint with smallest validation loss was loaded for evaluation. The error rates are shown in Table 6. The RMSE might be small, but it's also because the target value range, ML utility, is small. However, the TabDDPM estimator for insurance dataset has a high error rate. The high error rate was caused by its inability to predict large negative values (Figure 5).

Table 6. RMSE of the Estimator

Synthesizer	Contraceptive	Insurance	Treatment
LCT-GAN	0.0419	0.0156	0.0430
Realtabformer	0.0415	0.0161	0.0443
TabDDPM	0.0396	**0.1506**	0.0386
TVAE	0.0396	0.0153	0.0424
Mean	0.0406	0.0494	0.0421

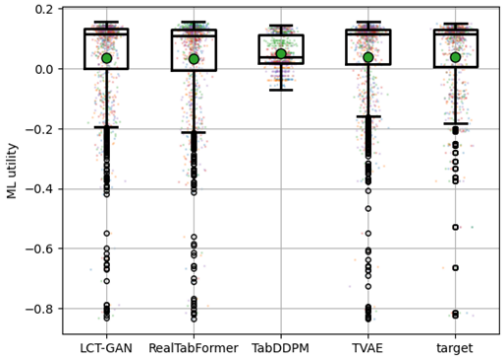


Figure 5. Estimator Test Prediction Box Plot for Insurance Dataset

Estimator Integration

The integration is done by generating synthetic data, estimating its ML utility, comparing the estimated and expected ML utility, then backpropagating the loss and its gradients (Figure 6). This is injected at the end of training iteration code. To enable backpropagation from the estimated ML utility, the sampling (data generation) code was modified to provide the option to skip postprocessing and return synthetic data in raw GPU tensor with backpropagation graph intact. Some functions had to have their *no_grad* decorator removed for this to work. LCT-GAN training is done in two separate phases, the autoencoder training and the GAN training. The ML utility backpropagation was done in both phases. In autoencoder training, instead of sampling synthetic data, the real data was *autoencoded* and had its ML utility estimated. Realtabformer was based on GPT-2 from *huggingface* library, so it uses the *huggingface* Trainer. ML utility backpropagation for Realtabformer was implemented through *TrainerCallback*.

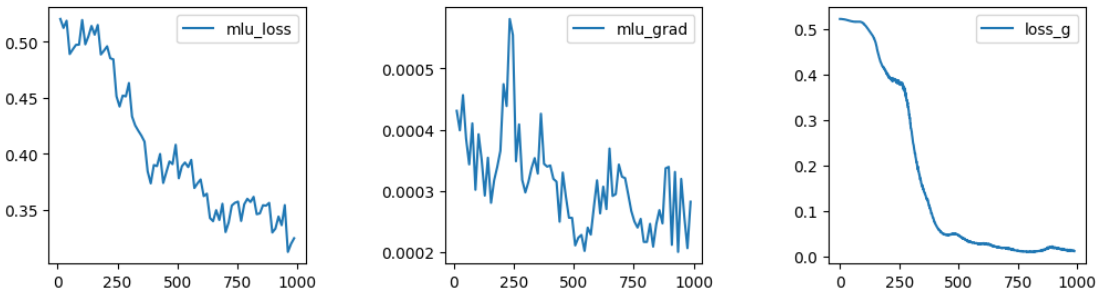


Figure 6. Example of ML Utility Loss (Left) and its Gradient (Center) for LCT-GAN Generator for Insurance Dataset, in Addition to the Generator's Own Loss (Right)

Evaluation

The ML utility of real dataset and baseline synthetic datasets are shown in Table 7. The synthetic datasets were produced by training each synthesizer with each fold of the 5 folds of real datasets then generating 30 datasets from each model, resulting in 150 datasets for each synthesizer. Realtabformer synthetic treatment dataset has equal ML utility to the real dataset. Aside from that, LCT-GAN synthetic datasets are generally bad compared to the

other synthesizers, especially for insurance dataset with negative ML utility score. The score for insurance dataset can be negative because the metric is R^2.

The changes in average ML utility of synthetic datasets after ML utility integration are shown in Table 8. The values were calculated by subtracting the average ML utility of synthetic datasets with integration and the baseline values. Positive change means the integration has successfully improved the average ML utility. The integration has improved the average ML utility of 7 out of 12 synthesizer-dataset pairs. One-way T-Test was done to test for significance with 5% alpha value. The significant improvements is marked by bold (Table 8). Only 5 out of the 12 pairs have improved significantly, which are LCT-GAN for all three datasets and Realtabformer for Contraceptive and Insurance datasets. Recall that originally Realtabformer had achieved equal ML utility to the real dataset for Treatment dataset (Table 7).

Table 7. Average ML Utility of Real Datasets and Baseline Synthetic Datasets

Dataset Type	**Synthesizer**	**Average ML Utility**		
		Contraceptive	**Insurance**	**Treatment**
		F1	**R^2**	**F1**
Synthetic Baseline	LCT-GAN	0.3442	**-0.0932**	0.3943
	Realtabformer	0.4454	0.1274	**0.6224**
	TabDDPM	0.4710	0.1321	0.5983
	TVAE	0.4790	0.1127	0.5533
Real		0.5120	0.1390	**0.6220**

Table 8. Changes in Average ML Utility of Synthetic Datasets after ML Utility Integration

Synthesizer	**Change in Average ML Utility**		
	Contraceptive	**Insurance**	**Treatment**
	F1	**R^2**	**F1**
LCT-GAN	**0.0158**	**0.0310**	**0.0561**
Realtabformer	**0.0200**	**0.0024**	-0.0136
TabDDPM	0.0033	0.0000	-0.2211
TVAE	-0.0048	-0.0153	-0.0038

For the other metrics, only the pairs with significantly improved ML utility will be discussed (Table 8). The average values of the other metrics for synthesizer-dataset pairs without integration are shown in Table 9. Jensen-Shannon Divergence (JSD), Wasserstein Distance (WD), and Difference in Pairwise Correlation (Diff. Corr.) are metrics where smaller values mean better models, while Distance to Closest Record (DCR) are metrics where larger values mean better models. Synthetic datasets generated by Realtabformer have rather small DCR values, which means individual data rows are similar to the real data. Synthetic datasets generated by LCT-GAN have rather large Diff. Corr. values, which means the generated datasets have quite a different correlation between features compared to the real datasets.

The changes in average values for synthetic datasets after ML utility estimator integration are shown in Table 10. The changes were calculated by subtracting the average values of each metric value of the synthetic datasets with ML utility estimator integration with the baseline values. JSD, WD, and Diff. Corr. are metrics where smaller values mean better models, so negative change means the value has improved. Meanwhile DCR is a metric where larger values mean better models, so positive change means the value has improved.

The four other metrics have changed in inconsistent directions. Two-way T-test was done to test for significance with 5% alpha value. The significant changes is marked by bold (Table 10). Almost every change is significant, even though the direction varies.

Additionally, the correlation between the change in ML utility and the change in the other metrics are measured with Pearson Correlation (Table 11). JSD and WD always correlate negatively with ML utility, which means ML utility improves along with improvement in distribution which becomes closer to the real data. Meanwhile for Diff. Corr. and DCR, the direction of the correlation still varies, and some are even the opposite of the direction of change in average value.

Table 9. Baseline Average ML Utility and Other Metrics of Synthetic Datasets

Dataset	Synthesizer	ML Utility		Distribution		Diff. Corr.	Privacy
				JSD (Cat.)	WD (Num.)		DCR
		(+)		(-)	(-)	(-)	(+)
Contraceptive	LCT-GAN	F1	0.344	0.243	0.079	2.311	1.098
Insurance	LCT-GAN	R^2	-0.093	0.191	0.110	1.883	0.405
Treatment	LCT-GAN	F1	0.394	0.087	0.058	2.836	0.867
Contraceptive	RealTabFormer	F1	0.445	0.065	0.021	0.421	0.008
Insurance	RealTabFormer	R^2	0.127	0.067	0.025	0.229	0.068

Table 10. Changes in Average ML Utility and Other Metrics of Synthetic Datasets after ML Utility Integration

Dataset	Synthesizer	Changes in Values					
		ML Utility		Distribution		Diff. Corr.	Privacy
				JSD (Cat.)	WD (Num.)		DCR
		(+)		(-)	(-)	(-)	(+)
Contraceptive	LCT-GAN	F1	0.016	0.008	**0.010**	**-0.187**	**-0.539**
Insurance	LCT-GAN	R^2	0.031	**0.014**	**-0.013**	**-0.205**	**-0.024**
Treatment	LCT-GAN	F1	0.056	**-0.008**	**-0.003**	**0.182**	**0.028**
Contraceptive	RealTabFormer	F1	0.020	**0.010**	**0.003**	**0.027**	0.005
Insurance	RealTabFormer	R^2	0.002	**-0.010**	**-0.003**	**-0.014**	**-0.007**

Table 11 Pearson Correlation Values Between the Changes in the Other Metrics and the Changes in ML Utility

Dataset	Synthesizer	Correlation with Changes in ML Utility			
		JSD (Cat.)	WD (Num.)	Diff. Corr.	DCR
Contraceptive	LCT-GAN	**-0.643**	**-0.327**	-0.167	**-0.791**
Insurance	LCT-GAN	**-0.450**	-0.093	-0.248	**0.536**
Treatment	LCT-GAN	**-0.724**	**-0.519**	**-0.716**	**-0.794**
Contraceptive	RealTabFormer	-0.129	-0.100	0.126	-0.122
Insurance	RealTabFormer	**-0.886**	**-0.876**	**-0.448**	**0.641**

Sample

To show the result of the experiment, a pair of synthetic Insurance datasets by LCT-GAN, before (Synth Baseline) and after ML utility integration (Synth MLU), was selected from the same fold and seed. Figure 7 shows the distribution of charges, which is the target variable, and the distribution of smoker, which the Catboost model deemed as the most important variable (75.81%). The distribution of Synth MLU is closer to the real distribution, specially for smoker variable where Synth Baseline has no “yes” class at all. Table 12 shows the difference in correlation (Diff. Corr.) of features toward the target variable “charges” from the pair of datasets. These are differences between the correlation of synthetic data and real data, so smaller values are better. Overall, the Synth MLU dataset has smaller Diff. Corr., thus more similar correlation to real dataset.

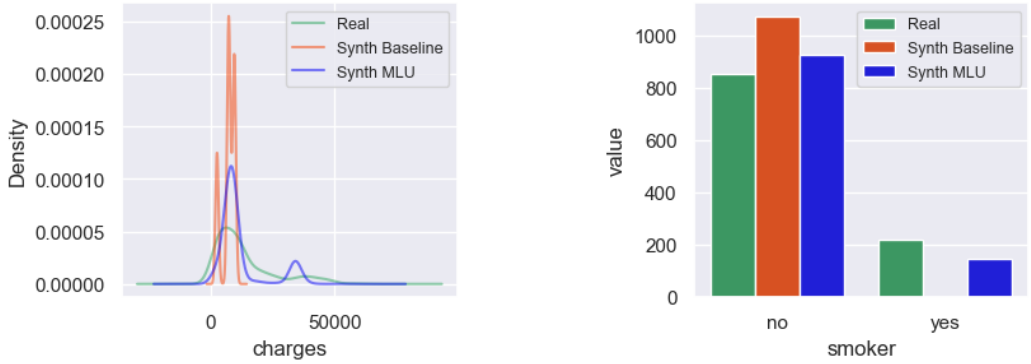


Figure 7 Distribution of Charges (Left) and Smoker (Right) Variable of the Sample Synthetic LCT-GAN Insurance Dataset Pair

Table 12 Difference in Correlation of Features Toward Target Variable from the Sample Synthetic LCT-GAN Insurance Dataset Pair

Model	Difference in Feature Correlation towards Target						Sum	Magnitude
	Age	Sex	BMI	Children	Smoker	Region		
Synth Baseline	0.037	0.307	0.208	0.477	0.018	0.225	1.273	0.646
Synth MLU	0.094	0.248	0.134	0.161	0.146	0.119	0.902	0.387
Difference	0.056	-0.059	-0.074	-0.316	0.128	-0.106	-0.371	-0.259

Table 13 and Table 14 shows a pair of most similar real and synthetic Insurance data row from the dataset pair. There is not much difference to see, except that the distance is closer after the integration. The DCR value decreased by 0.073, which is not much but still a degradation. This is consistent with previous studies which stated that there is trade-off between information or prediction performance of synthetic datasets and its privacy [3], [27]–[29]. The ML utility improved with privacy as a cost, although privacy is still kept at some level.

Table 13 A Pair of Most Similar Real and Synthetic Insurance Data Row from the Baseline LCT-GAN Dataset

Type	Age	Sex	BMI	Children	Smoker	Region	Charges	Distance
Real	47	female	29.55	1	no	northwest	8930.93	0.092
Synth Baseline	46	female	29.81	1	no	northwest	9390.71	

Table 14 A Pair of Most Similar Real and Synthetic Insurance Data Row from the MLU LCT-GAN Dataset

Type	Age	Sex	BMI	Children	Smoker	Region	Charges	Distance
Real	47	female	29.55	1	no	northwest	8930.93	0.089
Synth MLU	46	female	29.81	1	no	northwest	8571.73	

CONCLUSION

The ML utility integration has significantly improved the average ML utility of LCT-GAN and Realtabformer. The ML utility of LCT-GAN improved by 0.0158 for Contraceptive dataset, 0.031 for Insurance dataset, and 0.0561 for Treatment dataset. The ML utility of Realtabformer improved by 0.02 for Contraceptive dataset and 0.0024 for Insurance dataset. ML utility of Realtabformer for Treatment dataset did not improve, but it had achieved ML utility equal to real data from the beginning. The improvement of ML utility affects distribution, correlation between features, and the privacy of the synthetic datasets, but the direction of the change varies. However, Pearson correlation between the change in ML utility and the change in Jensen-Shannon Divergence (JSD) and Wasserstein Distance (WD) are always negative, indicating JSD and WD decreases (distribution becomes closer to real data) as ML utility increases (improves). This method of integrating ML utility

can require quite a long time for the sampling and a large amount of memory for backpropagation, depending on the synthesizer. The estimator itself requires a lot of parameters to learn the patterns in a dataset, synthesizer, and the target model used to evaluate ML utility, so it is hard to use for large datasets. Therefore, scalability and efficiency are the main issues of this method. Future research can be done to address these issues.

This study has shown that pattern between rows in a dataset can be learned by a transformer model with modifications from set transformer. This study has also shown that the estimated ML utility from the transformer model can be propagated back to the synthesizers to improve the value like a loss function does. The findings in this study can be used as basis to develop a synthesizer that also learns the pattern between data rows in a dataset and produces a set of data. Previous models, including ones used in this study, learns data individually and produces data individually too. The integration of pattern between data rows into a synthesizer might be the missing piece of data synthesizers to achieve even closer predictive performance to real data. As opposed to Realtabformer which processes a data row as a string of text tokens, the transformer in this study processes rows of data as a set.

The improvement of ML utility in this study may cause decrease in privacy. This is a known tradeoff, so users should pay attention to this, especially for sensitive medical records. Synthesizers that can generate data with certain privacy levels have been developed previously, but within the writers' knowledge, there hasn't been a study to find ways to optimize the tradeoff between ML utility and privacy. Perhaps loss learning with privacy objective can also be applied. Aside from that, data rows with high similarity to real data can be detected and removed, but it will affect the ML utility, distribution, and correlation in the dataset. Future research can be done to develop a method to determine which rows are safe to remove and which rows are important in a dataset. If such a method or model exists, the filtering can be automated during the synthesizing, or even integrated into the synthesizer training.

ACKNOWLEDGEMENT

This research was funded by the Ministry of Education, Culture, Research and Technology, Master's Thesis Research Scheme with grant number 112/E5/PG.02.00.PL/2023 (Master Contract) or 2006/PKS/ITS/2023 (Researcher Contract).

REFERENCES

[1] V. Ayala-Rivera, P. McDonagh, T. Cerqueus, L. Murphy, dan C. Thorpe, "Enhancing the utility of anonymized data by improving the quality of generalization hierarchies," *Trans. Data Priv.*, vol. 10, no. 1, hal. 27–59, 2017, doi: 10.5555/3121409.3121411.

[2] Republik Indonesia, *Undang-Undang Republik Indonesia Nomor 27 Tahun 2022 tentang Pelindungan Data Pribadi*, no. 016999. 2022, hal. 1–50.

[3] Z. Zhao, A. Kunar, R. Birke, dan L. Y. Chen, "CTAB-GAN+: Enhancing Tabular Data Synthesis," Apr 2022, [Daring]. Tersedia pada: http://arxiv.org/abs/2204.00401

[4] Z. Zhao, A. Kunar, R. Birke, L. Y. Chen Lydiaychen, V. N. Balasubramanian, dan I. Tsang, "CTAB-GAN: Effective Table Data Synthesizing," 2021.

[5] L. Xu, M. Skoularidou, A. Cuesta-Infante, dan K. Veeramachaneni, "Modeling tabular data using conditional GAN," in *Advances in Neural Information Processing Systems*, 2019, vol. 32. [Daring]. Tersedia pada: https://proceedings.neurips.cc/paper/2019/hash/254ed7d2de3b23ab10936522dd547b78-Abstract.html

[6] A. Kotelnikov, D. Baranchuk, I. Rubachev, dan A. Babenko, "TabDDPM: Modelling Tabular Data with Diffusion Models," in *International Conference on Machine Learning*, 2023. [Daring]. Tersedia pada: https://proceedings.mlr.press/v202/kotelnikov23a.html

[7] J. Kim, J. Jeon, J. Lee, J. Hyeong, dan N. Park, "OCT-GAN: Neural ODE-based conditional tabular

GANs," in *The Web Conference 2021 - Proceedings of the World Wide Web Conference, WWW 2021*, Apr 2021, hal. 1506–1515. doi: 10.1145/3442381.3449999.

[8] J. Kim, C. Lee, dan N. Park, "STaSy: Score-based Tabular data Synthesis," *arXiv Prepr. arXiv2210.04018*, 2022, [Daring]. Tersedia pada: http://arxiv.org/abs/2210.04018

[9] B. Nowok, G. M. Raab, dan C. Dibben, "Synthpop: Bespoke creation of synthetic data in R," *J. Stat. Softw.*, vol. 74, no. 11, 2016, doi: 10.18637/jss.v074.i11.

[10] T. E. Raghunathan, J. P. Reiter, dan D. B. Rubin, "Multiple imputation for statistical disclosure limitation," *J. Off. Stat.*, vol. 19, no. 1, hal. 1–16, 2003, [Daring]. Tersedia pada: http://hbanaszak.mjr.uw.edu.pl/TempTxt/RaghunathanEtAl_2003_Multiple Imputation for Statistical Disclosure Limitation.pdf

[11] J. H. Moltz *et al.*, "Learning a Loss Function for Segmentation: A Feasibility Study," in *Proceedings - International Symposium on Biomedical Imaging*, Apr 2020, vol. 2020-April, hal. 957–960. doi: 10.1109/ISBI45749.2020.9098557.

[12] S. W. Fu, C. F. Liao, dan Y. Tsao, "Learning with Learned Loss Function: Speech Enhancement with Quality-Net to Improve Perceptual Evaluation of Speech Quality," *IEEE Signal Process. Lett.*, vol. 27, hal. 26–30, 2020, doi: 10.1109/LSP.2019.2953810.

[13] D. Yoo dan I. S. Kweon, "Learning Loss for Active Learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, hal. 93–102.

[14] A. Vaswani *et al.*, "Attention Is All You Need," *31st Conf. Neural Inf. Process. Syst. (NIPS 2017)*, no. Nips, 2017.

[15] J. Lee, Y. Lee, J. Kim, A. R. Kosiorek, S. Choi, dan Y. W. Teh, "Set Transformer: A Framework for Attention-based Permutation-Invariant Neural Networks," in *International conference on machine learning*, 2019. [Daring]. Tersedia pada: http://arxiv.org/abs/1810.00825

[16] V. Velev, "LCT-GAN Improving the efficiency of tabular data synthesis via latent embeddings," TU Delft, 2022. [Daring]. Tersedia pada: http://resolver.tudelft.nl/uuid:e6b008b4-7fdb-49b3-a3c1-d5d22f96ea14

[17] A. V. Solatorio dan O. Dupriez, "REaLTabFormer: Generating Realistic Relational and Tabular Data using Transformers," *arXiv Prepr.*, 2023, [Daring]. Tersedia pada: http://arxiv.org/abs/2302.02041

[18] S. A. Assefa, D. Dervovic, M. Mahfouz, R. E. Tillman, P. Reddy, dan M. Veloso, "Generating synthetic data in finance: Opportunities, challenges and pitfalls," *ICAIF 2020 - 1st ACM Int. Conf. AI Financ.*, 2020, doi: 10.1145/3383455.3422554.

[19] I. E. Olatunji, J. Rauch, M. Katzensteiner, dan M. Khosla, "A Review of Anonymization for Healthcare Data," *Big Data*, hal. 1–14, 2022, doi: 10.1089/big.2021.0169.

[20] T.-S. Lim, "Contraceptive Prevalence Survey," *UCI Machine Learning Repository*, 1997. https://www.kaggle.com/datasets/joelzcharia/contraceptive-prevalence-survey (diakses 11 September 2023).

[21] M. Sadikin, "EHR Dataset for Patient Treatment Classification," *Mendeley Data*, 2020. https://www.kaggle.com/datasets/manishkc06/patient-treatment-classification (diakses 11 September 2023).

[22] Z. Stednick, "Data for Machine Learning with R," *Github*, 2017. https://www.kaggle.com/datasets/mirichoi0218/insurance (diakses 11 September 2023).

[23] P. Pokhrel, E. Ioup, J. Simeonov, M. T. Hoque, dan M. Abdelguerfi, "A Transformer-Based Regression Scheme for Forecasting Significant Wave Heights in Oceans," *IEEE J. Ocean. Eng.*, vol. 47, no. 4, hal. 1010–1023, 2022, doi: 10.1109/JOE.2022.3173454.

[24] M. Arjovsky, S. Chintala, dan L. Bottou, "Wasserstein generative adversarial networks," *34th Int. Conf. Mach. Learn. ICML 2017*, vol. 1, hal. 298–321, 2017.

[25] S. Yoa, J. Park, dan H. J. Kim, "Learning Non-Parametric Surrogate Losses with Correlated Gradients," *IEEE Access*, vol. 9, hal. 141199–141209, 2021, doi: 10.1109/ACCESS.2021.3120092.

[26] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, dan A. Courville, "Improved training of wasserstein GANs," *Adv. Neural Inf. Process. Syst.*, vol. 2017-Decem, hal. 5768–5778, 2017.

[27] T. Carvalho, N. Moniz, P. Faria, dan L. Antunes, "Towards a data privacy-predictive performance trade-off," *Expert Syst. Appl.*, vol. 223, hal. 119785, 2023, doi: 10.1016/j.eswa.2023.119785.

[28] T. Li dan N. Li, "On the tradeoff between privacy and utility in data publishing," *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, hal. 517–525, 2009, doi: 10.1145/1557019.1557079.

[29] J. Brickell dan V. Shmatikov, "The cost of privacy: Destruction of data-mining utility in anonymized data publishing," *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, hal. 70–78, 2008, doi: 10.1145/1401890.1401904.