



**FCT-GAN: Fourier Neural Operator for Global Relation Enhancement in Tabular
Data Synthesizing using Generative Adversarial Networks**

Ethan Keller¹

Supervisor(s): Zilong Zhao¹, Lydia Y. Chen¹

¹EEMCS, Delft University of Technology, The Netherlands

23-6-2022

**A Dissertation Submitted to EEMCS faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering**

Abstract

Since the regularization of data privacy (e.g., GDPR), the effectiveness of data sharing has decreased. A promising technique to circumvent this problem is tabular data synthesis (i.e., the generation of fake tabular data that statistically resembles the original data). However, the state-of-the-art tabular data synthesis model, CTAB-GAN, fails at robustly imitating global data dependencies and underperforms when column orders get permuted. CTAB-GAN internally uses Convolutional Neural Networks (CNN) which limits the model’s performance due to a strictly non-global data perspective during iterative training phases. To address this limitation, this paper proposes FCT-GAN¹ which leverages the Fourier Neural Operator to learn global dependencies in the frequency domain. Specifically, it enhances CTAB-GAN by replacing the CNN of the discriminator with a four-layered two-dimensional Fourier Neural Operator. As a consequence of FCT-GAN’s global nature and cross-column relation robustness, it outperforms CTAB-GAN and additionally offers the column permutation invariant property. The evaluation of FCT-GAN on five datasets shows that the generated data, remarkably resembles the real data and reveals an increase in accuracy, by up to 19% for five machine learning algorithms independent of data column order, compared to CTAB-GAN.

Keywords GAN; Data synthesis; Tabular data; Fourier space; Global dependencies

1 Introduction

Data-driven organizations nowadays acquire valuable business insights through data sharing. However, data privacy constraints and regulations such as the European GDPR often lead to hindrance and limit the development of knowledge. Zhao *et al.* (2021) explains how “the big knowledge behind big data often impedes personal privacy and leads to unjustified analysis” (p. 1). Hence, these limitations oblige data-driven organizations to explore new data handling strategies. Nevertheless, while respecting the privacy regulations, these approaches should additionally preserve the potential of big discovery.

An arising approach is to leverage synthetic data [17]. This strategy entails generating data that statistically resembles real data. Operating on the synthetic data naturally complies with the data privacy regulations, and thus allows data sharing.

The Generative Adversarial Network (GAN) [8] is a leading deep learning model for data synthesizing. It is trained on real datasets to subsequently generate synthetic data. The GAN has successfully been applied to many tasks including image synthesis [11], image super-resolution [14] and music synthesis [6]. Recently, the GAN has also found its purpose

in tabular data generation [19]. This is of great essence since big industrial datasets are formatted in tabular fashion.

FCT-GAN integrates the Fourier Neural Operator [15] (FNO), which is introduced in subsection 1.2, into its GAN architecture to address the limitations of state-of-the-art tabular data synthesizers listed in subsection 1.1.

1.1 State-of-the-art

Currently, the state-of-the-art GAN tabular data synthesizers include CTGAN [26], TableGAN [20], CWGAN [30], MedGAN [2], and CTAB-GAN [33]. These models have all shown to be effective in tabular data synthesis. However, while other models don’t, CTAB-GAN specifically takes data with large tail distributions, multi-modal distributions, and mixed variables (i.e., simultaneously categorical and continuous variables) into consideration. As a consequence, CTAB-GAN outperforms the other models as concluded in Zhao *et al.* (2021). The newly proposed model: FCT-GAN, will thus be evaluated against the CTAB-GAN baseline. By transitivity, outperforming CTAB-GAN implies a new state-of-the-art model.

Limitations

While CTAB-GAN shows promising results, it still faces some obstacles. The CTAB-GAN architecture is based on the Convolutional Neural Network (CNN) model. It operates by spatially shifting a specialized kernel over the whole feature space. Because of this local kernel, the CNN only captures local, regional dependencies. This is problematic since global dependency information could contribute to increased data synthesis effectiveness. Section 3 details and visualizes the underlying problem of CTAB-GAN’s missing global relations.

Aside from this first limitation, another one is identified when permuting columns. When the columns of a dataset get reordered, the results of a tabular data synthesizer should not differ since the semantics of the data do not change. This transformation invariant is called the column permutation invariant and is described and clarified in section 3. However, CTAB-GAN does not seem to have the column permutation invariant property. As seen in Table 3, CTAB-GAN shows some instability when columns get permuted, with up to a 2.351% difference between accuracy on distinct permutations.

The limitations of CTAB-GAN can be summarized as: (1) ineffectively capturing global cross-column relations, and (2) missing robustness against column permutations. FCT-GAN addresses these limitations through the use of global frequency information explained in subsection 1.2.

1.2 Global Frequency Leverage

To address the limitations mentioned in subsection 1.1, FCT-GAN applies the concept of a Fourier Neural Operator (FNO) [15]. The FNO is a neural operator [13] that projects the whole feature space in the frequency domain (i.e., Fourier space), multiplies the frequencies with global weights, and then projects the frequencies back to the spatial domain (further explained in section subsection 4.1). This sequence of

¹<https://github.com/ethan-keller/FCT-GAN>

operations allows the FNO to learn global weights in the frequency domain rather than local weights in the spatial domain. However, the FNO comes with its own challenges such as finding the optimal data dimensionalities for the Fourier projection, or ensuring that the frequency-treated data is usable in other components of a model. The FNO has been applied to a few fields including partial differential equations [15] and image classification [21], showing impressive results.

FCT-GAN integrates an FNO into its architecture to address CTAB-GAN’s limitations. Firstly, since the FNO operates on the whole feature space at once, it naturally profits from global information which can be used to model global column dependencies. Hereby, in the context of tabular data synthesis, the FNO solution addresses CTAB-GAN’s limitation of ineffectively capturing global cross-column relations. Secondly, the FNO learns its global weights in Fourier space. Since column permutations do not semantically affect the tabular data, they do not alter frequency signals in the data. This implies, by transitivity, that the column permutation transformation does not affect the Fourier projection. Hereby, in the context of tabular data synthesis, the FNO solution also addresses CTAB-GAN’s missing robustness against column permutations. Since the FNO addresses all the mentioned limitations about CTAB-GAN, FCT-GAN adopts an enforced version of the FNO in its architecture.

2 Related Studies

Since this paper combines the tabular data synthesis domain with the Fourier computation domain, the fields are first examined individually. The related studies are thus divided into two parts: (1) tabular data synthesizing studies, and (2) global frequency deep learning studies.

2.1 Tabular Data Synthesis

Many studies make use of the Generative Adversarial Network (GAN) framework for tabular data synthesis. MedGAN [2] integrates a GAN and an auto-encoder to generate continuous or discrete variables for synthetic electronic health records (EHR). CrGAN-Cnet [17] combines the Cramer Distance [3] with the Cross-Net [24] network to generate airline passenger name records. CrGAN-Cnet is additionally capable of handling missing data by filling data gaps with new variables. TableGAN [20] adopts an auxiliary classifier. It uses the Convolutional Neural Network (CNN) model for its generator, discriminator and classifier and introduces information loss for semantics preservation. Aside from the actual data synthesis, privacy is of great importance too. It is the underlying reason for tabular data synthesis. PATE-GAN [28] is not particularly built for tabular data generation, but it offers differential privacy guarantees.

Although these models can synthesize tabular data (except PATE-GAN), they cannot selectively generate tabular data. I.e., they cannot generate data samples from a predetermined class. E.g., it is not possible to generate an airline passenger record for male passengers.

Some studies extend the GAN framework to a conditional GAN framework which allows selective data generation. This is extremely useful when the data is skewed

or limited. Selectively generated data can then be used to re-balance the data distribution. For example when preparing starting datasets of online learning scenarios [31; 32; 29]. CW-GAN [30] introduces the Wasserstein Distance [1] and uses a conditional vector to re-balance data distributions by oversampling minority classes. CTGAN [26] integrates a training strategy called training-by-sampling which leverages a conditional vector to overcome the imbalanced categorical variable problem. It structures its discriminator as described in PacGAN [16], and uses a generator loss, WGAN [1] loss, and gradient penalty [10] for GAN training. Finally, CTAB-GAN [33] not only focuses on modelling continuous and categorical variables, but also mixed data types (i.e., variables that contain both categorical and continuous values, or even missing values). This is done through a combination of the aforementioned models’ strengths such as an auxiliary classifier, information loss, generation loss, conditional vectors and effective encoding. Furthermore, CTAB-GAN also addresses the challenge of long tail variable distributions and proposes a new conditional vector structure to better deal with imbalanced datasets. However, since CTAB-GAN is CNN-based, it falls short in capturing long-range column dependencies. FCT-GAN improves CTAB-GAN’s performance by replacing the CNN discriminator with an adapted Fourier Neural Operator framework.

2.2 Global Frequency Deep Learning

Until recently, Fourier transformations were often used for the acceleration of Deep Learning model training [4; 7]. The gain in efficiency is a direct consequence of the Convolutional Theorem which states that the Fourier transform of a convolution of signals in the temporal domain equals a point-wise multiplication in the frequency domain (further explained in subsection 4.1).

There are not many studies that leverage the Fourier Neural Operator (FNO) [15]. The FNO is a relatively new concept which combines the global expressiveness of Fourier manipulations with the performance and flexibility of the neural operator framework [13]. It has mostly been used in analytical mathematics and physics related fields.

Originally, the FNO was introduced for quantitative physics [15; 25; 22; 12; 9] where models try to solve partial differential equations that mimic physical processes. These models learn weights in Fourier space and offer superior accuracy to models without an FNO enhancement. Nonetheless, the FNO has recently made its first mark in image classification. The Global Filter Network (GFNet) [21] substitutes the self-attention layer in the Vision Transformer (ViT) [5] for an Fourier layer. The Fourier layer consists of a two-dimensional Fast Fourier Transform (FFT), a global filter which learns complex weights, and an Inverse Fourier Transform (IFFT). This allows the model to perceive global features and learn long-term spatial dependencies in the frequency domain with log-linear complexity. It concatenates a Multilayer Perceptron (MLP) to the Fourier layer to classify images based on the FNO’s output. GFNet attains impressive results compared to mainstream CNN image classification, and demonstrates increased efficiency.

It was previously unclear whether the FNO framework was

applicable to tabular GANs. FCT-GAN is thus the first FNO-enhanced GAN-based tabular data synthesizer. It integrates GFNet’s idea of concatenating an MLP to the Fourier Layer, but adds regularization procedures and a binary classification adaptation (further explained in subsection 4.3).

3 Global Dependencies Problem

Tabular data synthesizers must output data that statistically resembles the original data. This implies that relations among columns should be preserved. This section dives deeper into the global dependencies problem underlying CTAB-GAN’s limitations listed in subsection 1.1. Firstly, it separately describes (1) the CNN’s regional nature, and (2) the column permutation invariant. Then, the research question and sub-questions supporting this paper are derived from the global dependencies problem.

3.1 CNN local relations

The Convolutional Neural Network (CNN) framework works by sliding windows, called kernels, over the feature space in a step-wise manner (as demonstrated in Figure 1). These kernels learn localized dependencies and aggregate them to form knowledge about the dataset. However, these kernels only repeatedly see regional parts of the feature space (i.e., they never have a global perception of the feature space). Consequently, a CNN is limited to aggregated local/regional information. For tasks such as image classification, CNN models perform well [23] since recognizing localized objects, animals, humans, etc. in images go well with the regional nature of the CNN.

Unfortunately, for tabular data synthesis, being unable to capture global dependencies can damage synthesis performance. An effective tabular data synthesizer should mimic all relations in the original data, including global relations. Since the CNN framework only captures local information, it only learns relations between proximal columns. Kernels cannot leverage cross-column information if the related columns are too far apart. In that case, the CNN kernel is physically incapable of spatially capturing the relational information. A visual example of this problem instance is depicted in the top portion of Figure 1 where two related columns $C1$ and $C6$ cannot be observed by the same CNN kernel. This leads to weak global column correlations and thus suboptimal synthesis performance. In practice, the CNN might end up capturing the distant column relation if the aggregated and non-linearly activated data ends up in the same kernel after a few iterations. However, this is not guaranteed since the dependency information can get obfuscated by aggregations, activation functions, and pooling layers.

3.2 Column Permutation Invariant

Column permutation is a tabular data transformation that does not change the data semantically. This can be better understood with the analogous example of a two-dimensional picture. Rotating or mirroring the image does not change the semantics of the picture. This is an example of a semantics-preserving data transformation. In the context of tabular data, a column permutation is a semantics-preserving transformation, which should not alter synthesis results. As seen in the

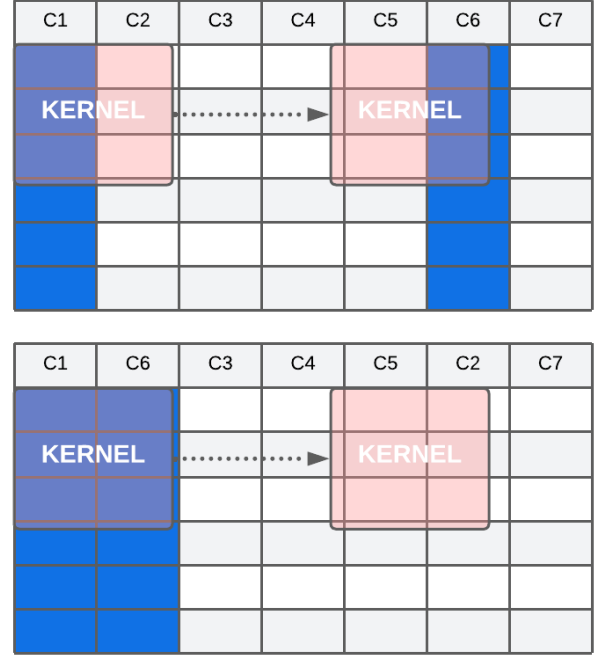


Figure 1: **Top:** The sliding CNN kernel cannot capture information from two related columns $C1$ and $C6$ simultaneously. **Bottom:** After a permutation, the sliding CNN kernel can capture information from two related columns $C1$ and $C6$ simultaneously.

bottom portion of Figure 1, a column permutation led two related columns $C1$ and $C6$ to be considered at the same time by the CNN kernel. In the top portion of Figure 1, the kernel was unable to do that since $C1$ and $C6$ were too far apart. This shows that the CNN captures a column relation on one permutation of the dataset, but not on another. Thus, using the CNN framework will block the tabular data synthesizer from having the column permutation invariant property.

Table 3 shows CTAB-GAN’s results on the evaluation metrics defined in subsection 5.2. CTAB-GAN shows instability facing permutation transformations. This missing column permutation robustness is likely due to the CNN nature of CTAB-GAN.

3.3 Research Questions

The proposed model FCT-GAN has the ability to tackle the global relation problem since it substitutes the CNN framework for a Fourier Neural Operator. Thus, it does not suffer from the CNN kernel’s limited reach. On the contrary, FCT-GAN’s reach is maximal since it observes the data globally in the frequency domain. This leads to the following research question and sub-questions:

”How can the Fourier Neural Operator improve global relation resemblance in tabular data synthesis using Generative Adversarial Networks?”

Sub-questions:

1. Which FNO dimensionality is optimal for tabular data synthesis?

2. In which component of a tabular GAN does the FNO offer the most performance gain?
3. Does an FNO-enhanced synthesizer offer the column permutation invariant property?

4 FCT-GAN

FCT-GAN is an FNO-enhanced tabular data generator which overcomes the limitations of the state-of-the-art tabular data synthesizer: CTAB-GAN, described in subsection 1.1. FCT-GAN extends CTAB-GAN by introducing the Fourier Neural Operator (FNO) into its architecture. The FNO allows FCT-GAN to capture global data dependencies which CTAB-GAN cannot do. As a consequence, FCT-GAN exhibits the column permutation invariant property, and outperforms all the state-of-the-art tabular data synthesizers.

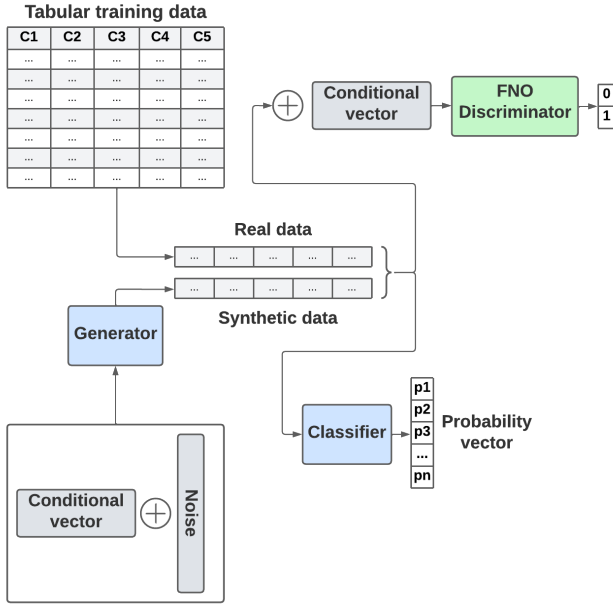


Figure 2: Synthetic tabular data generation with FCT-GAN.

4.1 Technical Background

To better understand FCT-GAN’s architecture and design motivation, this subsection deepens into the technical aspects of its two main building blocks. Specifically, the (1) Generative Adversarial Network (GAN) used in CTAB-GAN, and the (2) Fourier Neural Operator (FNO).

Generative Adversarial Networks for Tabular Data

The Generative Adversarial Network [8] is a generative deep learning framework wherein two adversarial neural networks called the generator and the discriminator compete in a zero-sum game (i.e., one adversary’s gain is the other’s equal loss). The generator tries to synthesize realistic samples while the discriminator tries to distinguish real samples from the synthetic ones. The better the discriminator understands relations within the data (i.e., what makes data realistic), the more realistic the synthesized samples will be.

$$x = [x_0, \dots, x_i, \dots, x_{N-1}] \wedge x_i \in \mathbb{C} \quad (1)$$

$$F(x)_k = \sum_{n=0}^{N-1} x_n \cdot e^{-i2\pi kn/N} \wedge k \in [0, N-1] \quad (2)$$

Equation 1 & 2: The discrete Fourier transform.

$$X = [X_0, \dots, X_i, \dots, X_{N-1}] \wedge X_i \in \mathbb{C}$$

$$F^{-1}(X)_k = \frac{1}{N} \sum_{n=0}^{N-1} X_n \cdot e^{i2\pi kn/N} \wedge k \in [0, N-1] \quad (3)$$

Equation 3: The inverse discrete Fourier transform.

CTAB-GAN’s architecture is based on a conditional GAN framework. It uses a conditional vector v which encodes the classes of the categorical variables. Subsequently, v gets fed to the generator and is used to sample subsets from the training data that satisfy the condition encoded in v . Leveraging v thus allows CTAB-GAN to re-sample minority classes, if necessary, which eliminates data imbalances. Additionally, CTAB-GAN attaches an auxiliary classifier in parallel to the discriminator. The concept of auxiliary classifier GANs has been used in image synthesis [18] and recently tabular data generation [20]. Every time the generator outputs a synthetic sample, the classifier predicts a label. CTAB-GAN then uses the mismatch between synthesized classes and predictions to increase semantic integrity of the synthetic data. E.g., a synthetic record containing entries $sex = male$ and $pregnant = true$ is semantically incorrect since a male cannot be pregnant. Thus such a record should not be found in the training data and is not learned by CTAB-GAN’s classifier. FCT-GAN follows CTAB-GAN’s state-of-the-art framework with adaptations for the discriminator to address CTAB-GAN’s limitations laid out in subsection 1.1.

Fourier Neural Operator

The Fourier Neural Operator (FNO) [15] is a neural operator [13] that learns global weights directly in Fourier space. The FNO consists of three main operations. (1) Firstly, data gets mapped to the frequency domain (i.e., Fourier space). This can be done with the Fast Fourier Transform (FFT) algorithm which efficiently computes the Discrete Fourier Transform (DFT) defined in Equation 2.

Where x denotes the temporal input data, F the DFT, and $e^{i2\pi/N}$ the primitive N -th root of unity. For the Fourier mapping, this results in an asymptotic time complexity of $O(N \cdot \log N)$ (log-linear) where N is the size of the data. This is more efficient than the plain DFT algorithm which has an asymptotic time complexity of $O(N^2)$. On top of the efficiency benefit, the FFT algorithm internally discretizes the feature space (see Equation 1) which regularizes the FNO to avoid over-fitting. After the Fourier mapping, (2) a weighted multiplication takes place between a set of global complex weights and the frequency signals. Finally, (3) the Inverse

Fast Fourier Transform (IFFT) is applied to the weight-treated frequency signals to map them back to the temporal feature space. Analogously to the FFT, the IFFT algorithm efficiently computes the Inverse Discrete Fourier Transform (IDFT) as defined in Equation 3. Where X denotes the spectral input data, and F^{-1} the IDFT. The FNO can be applied to any data dimensionality. E.g., a three dimensional data sample can be mapped to Fourier space by applying Equation 2 to the x , y , and z dimension sequentially. The FNO’s three operations happen in a global manner, compared to the limited regional nature of the CNN framework. This gives the FNO the upper hand on tackling the global relation problem described in section 3.

$$\begin{aligned} F(f * g) &= F(f)F(g) \\ f * g &= F^{-1}(F(f)F(g)) \end{aligned} \quad (4)$$

Equation 4: The convolution theorem.

To effectively replace a CNN model, the FNO must be capable of performing the same actions as the CNN (i.e., convolutions). The Convolution Theorem (see Equation 4) states that the Fourier Transform of a temporal signal convolution (i.e., $F(f * g)$) equals a point-wise multiplication in the frequency domain (i.e., $F(f)F(g)$). Since the FNO performs a point-wise multiplication in the frequency domain, by the Convolution Theorem, its actions are equivalent to convolutions in the temporal domain. The FNO methodology can thus be seen as a complex global convolution.

FCT-GAN substitutes CTAB-GAN’s CNN discriminator for a layered FNO. This allows the discriminator to learn from global information in the data. Additionally, it feeds the FNO output to a Multilayer Perceptron (MLP) to enable binary discrimination which is the discriminator’s fundamental task.

4.2 Design

FCT-GAN’s architecture (see Figure 2) can be subdivided in three main components: Generator G , Discriminator D , and Classifier C . FCT-GAN only integrates the FNO framework into D , and leaves G and C identical to their CTAB-GAN equivalents. The reason behind this choice is clarified in subsection 4.3.

4.3 Discriminator FNO

As explained in subsection 4.1, a GAN is trained via a zero-sum game where two adversarial networks (i.e., G and D) compete against each other. One tries to maximize an objective while the other tries to minimize it. *Zhao et al. (2021)* explains how “the game can be seen as a mentor D giving feedback to a student G on the quality of his work” (p. 6). This feedback is quantified through loss functions to allow the GAN to learn.

Losses

On top of the original generator feedback [8], G is regulated by three additional loss functions: (1) information loss, (2) classification loss, and (3) generator loss. Firstly, the **information loss** combines the mean and standard deviation (i.e.,

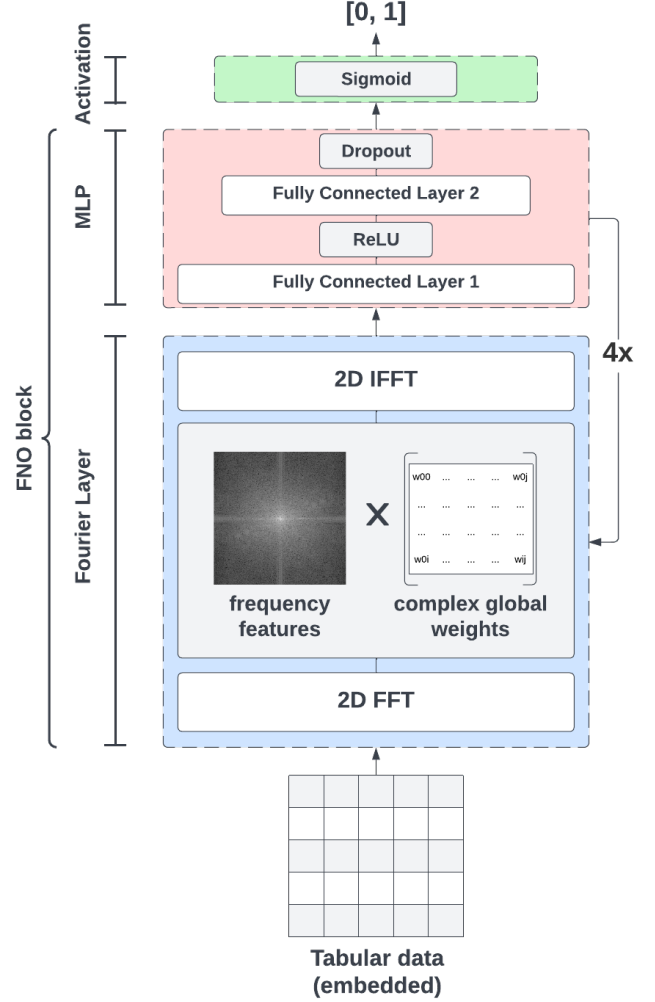


Figure 3: FCT-GAN’s discriminator architecture as four concatenated FNO blocks closed by a Sigmoid activation.

first- and second-order statistics) to form feedback about statistical similarity between synthetic and original data. As a consequence, the synthesized samples statistically resemble the original training data. This loss is formally defined in Equation 5 where $\mathbb{E}$ indicates mean, $\mathbb{SD}$ denotes standard deviation, and $p_{data}(x)$ & $p(z)$ indicate prior distributions for respectively original data, and latent variables. The **classification loss** compares class correlations with the other variable values. This helps to ensure semantic integrity in synthetic records and penalizes G if synthesized records are semantically incorrect (see example in subsection 4.1). This loss is written out in Equation 6 where, for a given row x , $l(\cdot)$ returns the target label and $r(\cdot)$ returns the input features. Lastly, the **generator loss** computes the cross-entropy between the conditional vector and synthesized output classes. It enforces that G generates samples that respect the condition set in the conditional vector. Equation 7 formalizes this loss, where m_i and $\hat{m}_i$ denote the original and synthetic conditional vector bits that correspond to column i . The cross-

$$\mathbb{L}_{info}^G = \left\| \mathbb{E}[f_x]_{x \sim p_{data}(x)} - \mathbb{E}[f_{g(z)}]_{z \sim p(z)} \right\|_2 \quad (5)$$

$$+ \left\| \mathbb{SD}[f_x]_{x \sim p_{data}(x)} - \mathbb{SD}[f_{g(z)}]_{z \sim p(z)} \right\|_2$$

$$\mathbb{L}_{class}^G = \mathbb{E}[l(G(z)) - C(r(G(z)))]_{z \sim p(z)} \quad (6)$$

$$\mathbb{L}_{gen}^G = H(m_i, \hat{m}_i) \quad (7)$$

$$\mathbb{L}^G = \mathbb{L}_{info}^G + \mathbb{L}_{class}^G + \mathbb{L}_{gen}^G + \mathbb{L}_{orig}^G \quad (8)$$

$$\mathbb{L}_{class}^C = \mathbb{E}[l(x) - C(r(x)))]_{x \sim p_{data}(x)} \quad (9)$$

Equation 5, 6, 7, 8 & 9: FCT-GAN loss functions: information loss, classification loss, generator loss, total G loss, and classifier loss.

entropy loss is computed by $H(\cdot)$. The three aforementioned losses are added to the original generator loss (i.e., $\mathbb{L}_{orig}^G$) to form the complete training objective for G (Equation 8). The training objective for D remains the same as in the original GAN framework [8]. Lastly, the classifier loss for the training of the auxiliary classifier component is denoted in Equation 9, and strongly resembles the classification loss for the generator (i.e., $\mathbb{L}_{class}^G$).

As depicted in the discriminator’s architecture in Figure 3, the discriminator’s input data must go through three main sub-components: (1) a Fourier layer, (2) an MLP layer, and (3) an activation layer. The **Fourier layer** is the FNO inspired, but slightly adapted sub-component. Firstly, it performs a global two-dimensional Fast Fourier Transform (explained in subsection 4.1). Secondly, it computes a point-wise multiplication between all the frequency signals and a set of complex global weights. The original FNO [15] cuts out part of the signals for regularization purposes while FCT-GAN uses all frequency information to effectively learn global relations. To compensate for this loss in regularization, FCT-GAN adds dropouts in the attached MLP layer. Lastly, it projects the signals back to feature space by performing an Inverse Fast Fourier Transform. The **MLP layer** consists of two fully connected linear layers separated by the Rectified Linear Unit (ReLU) activation function, and is closed by a dropout regularization. The second fully connected layer reduces the size of the of the latent space by 80% to $\lfloor 0.8 \cdot n_{in} \rfloor$, where n_{in} denotes the size of the input space. One concatenation of a Fourier layer and an MLP layer is called an FNO block and is intercalated by a batch normalization which is omitted from Figure 3 for visibility purposes. FCT-GAN chains four FNO blocks together. Since each Fourier block reduces the size of the latent space by 80%, the last Fourier block outputs latent data of size $\lfloor 0.8^4 \cdot n \rfloor \approx \lfloor 0.41 \cdot n \rfloor$. This size reduction mimics the size reduction in CNNs where kernels aggregate regional parts of the data to form smaller latent spaces. Finally, the **activation layer** consisting of a Sigmoid activation maps the FNO block output to a continuous value between 0 and 1, quantifying the binary discrimination. Values close to 0 mean that the discriminator believes the sample is fake, while values closer to 1 mean the discriminator supposes the sample is real.

FCT-GAN maintains CTAB-GAN’s generator and classifier

structure. The reason for only implementing the FNO framework in the discriminator is relatively intuitive, and can be understood via a process of elimination. CTAB-GAN’s architecture contains three main components that could be substituted for FNOs: Generator, Discriminator, and Classifier. The classifier does not make use of a CNN structure, but rather an MLP. It is made up of four hidden fully connected 256-neuron layers. Since the classifier does not suffer from the CNN kernel’s limited reach (see Figure 1), it does not need an FNO to address this limitation. However, the generator and discriminator do use a CNN. As explained in subsection 4.1, the generator should generate data as realistically as possible. The discriminator helps the generator understand if the synthesized samples are realistic or not. Thus, during training, the discriminator is the component that should understand all data relations in order to effectively discriminate and consequently give better feedback to the generator. Therefore, FCT-GAN integrates the FNO framework into the discriminator, and neither of the remaining components (i.e., the generator and classifier). However, since this reasoning is informal, it must be empirically verified. Subsection 5.5 shows the decrease in synthesis performance when the generator is FNO-enhanced, and when both the generator and discriminator are FNO-enhanced.

4.4 FNO Dimensionality

FCT-GAN follows a two-dimensional FNO framework as seen in Figure 3. The FNO uses a Fourier transform to map data to the frequency domain (see subsection 4.1). However, the Fourier transform can be applied to any data dimensionality (e.g., one-dimensional vectors, two-dimensional pictures, three-dimensional volumes, etc.). Since CTAB-GAN makes use of one-dimensional conditional vectors, a one-dimensional FNO discriminator could also be integrated. However, CTAB-GAN internally embeds the tabular data in a two-dimensional fashion. Therefore, FCT-GAN’s discriminator is implemented as a two-dimensional discriminator. This goes paired with CTAB-GAN’s CNN implementation where convolutions are performed in two-dimensional space. Since this thought process is informal, it is empirically validated in subsection 5.4.

5 Experimental Analysis

FCT-GAN is evaluated on five commonly used machine learning (ML) datasets, and compared against the state-of-the-art tabular data synthesizer CTAB-GAN. The synthesis effectiveness is evaluated in terms of ML utility, statistical similarity to the original data, and distance-based privacy. Additionally, the column permutation invariant property and FNO dimensionality effect are analyzed using the aforementioned metrics.

5.1 Experimental Setup

This subsection elaborates on the setup of the experiments. Specifically it elucidates the five datasets that were used, the baselines for comparison, and the specs of the computational environment.

Datasets

FCT-GAN is evaluated on five commonly used ML datasets: **Adult**, **Coverttype**, **Intrusion**, **Credit**, and **Loan**. The first three datasets are from the UCI ML repository² while the last two originate from Kaggle³. Table 1 shows information about the data types in each of the datasets. It also indicates the target variable for each dataset against which the rest of the variables are used to perform classification. Due to limitations in computational resources, 50K rows are randomly sampled from the three biggest datasets: Coverttype, Credit, and Intrusion, in a stratified manner with respect to their target variables. As seen in Table 1, the Loan dataset is significantly smaller than the others. Therefore it is trained for 300 epochs since it needs more time to converge. All other datasets are trained for 150 epochs.

Baseline

FCT-GAN is mainly compared to CTAB-GAN, the current state-of-the-art GAN-based tabular data synthesizer. However, FCT-GAN is also compared to three modifications of its own model: (1) FCT-GAN-gd where both the generator and discriminator are implemented as two-dimensional FNOs, (2) FCT-GAN-g where only the generator is implemented as a two-dimensional FNO, and (3) FCT-GAN-1D where the discriminator is a one-dimensional FNO. To allow fair comparison, all models are fully implemented in PyTorch. Additionally, for each model, the experiments are performed three times.

Computational Environment

The experiments are performed on a machine with Ubuntu 20.04.1, 32GB of memory, an Intel(R) Core(TM) i9 CPU with 10 cores, and a GeForce RTX 2080 Ti GPU.

5.2 Evaluation Metrics

The evaluation metrics are identical to the ones used by *Zhao et al. (2021)* to evaluate CTAB-GAN [33]. This allows for a fair comparative evaluation between FCT-GAN and CTAB-GAN. The evaluation metrics follow three aspects: (1) Machine Learning utility, (2) Statistical Similarity, and (3) Privacy Preservation. The first two metrics evaluate how well the synthetic data represents the real data, while the third metric quantifies the possibility of a privacy breach.

Machine Learning Utility

With ML utility, the goal is to compare the performance of an ML model on the real data versus the synthetic data. To do this, five commonly used ML algorithms: decision tree classifier, linear support-vector-machine, random forest classifier, multinomial logistic regression, and MLP, are trained on real and synthetic data. Accuracy, area under curve (AUC), and F1-score represent the ML results. If the synthetic data closely resembles the original data, the difference between their ML results should be small.

²<http://archive.ics.uci.edu/ml/datasets>

³<https://www.kaggle.com/mlg/creditcardfraud;itsmesunil/bank-loan-modelling>

Statistical Similarity

Three metrics are used to quantify statistical similarity. Firstly, the **Jensen-Shannon Divergence (JSD)** is used to quantify the difference between probability mass distributions of categorical variables in real and synthetic data. Secondly, the **Wasserstein Distance (WD)** measures how well the distributions of continuous/mixed variables in the synthetic data resemble the ones in the real data. Lastly, the **Difference in Pair-Wise Correlation (Diff. Corr.)** evaluates how correlations between features are preserved in the synthetic data. To quantify this difference the following operations are performed on the real and synthetic data. Firstly, the Pearson correlation coefficients are computed between continuous variables. Secondly, the Theil uncertainty coefficient is computed between categorical variables. Lastly, the difference between these correlations in the real and synthetic data is computed. Ideally, these metrics should be as low as possible to show statistical similarity between real and synthetic data.

Privacy Preservability

To quantify privacy preservability, two distance-based metrics are used to evaluate privacy risk. Firstly, the **Distance to Closest Record (DCR)** measures the Euclidean distance between every synthetic record and its closest corresponding real record neighbor. The 5th percentile is used to provide a more robust measure. The higher the DCR, the lesser the risk of privacy breach. Secondly, the **Nearest Neighbor Distance Ratio (NNDR)** measures the ratio between the Euclidean distance from the closest and second closest real record to all corresponding synthetic records. The 5th percentile is computed here as well. NNDR is mostly used in Computer Vision [27] and was introduced by Platzer⁴ to evaluate privacy in synthetic data. Low NNDR values between real and synthetic data indicate that the synthetic records could reveal sensitive information from the real records. DCR and NNDR are both computed between real and synthetic data, within real data, and within synthetic data.

5.3 CTAB-GAN comparison

Table 2 shows the results from the experiments performed on FCT-GAN compared to the state-of-the-art CTAB-GAN model. In terms of average ML utility, lower values indicate that the synthetic data is more likely a better representative for the real data. FCT-GAN outperforms CTAB-GAN in accuracy difference by significantly reducing it from 10.59% (CTAB-GAN) to 7.46%. It also reveals better results for the AUC metric. However, CTAB-GAN seems to have the upper hand on F1-score. While accuracy is the most used classification metric, F1-score and AUC are often used in the case of imbalanced datasets. FCT-GAN might thus lose some effectiveness when run on heavily imbalanced datasets.

As seen in Table 2, FCT-GAN fully outperforms CTAB-GAN across all statistical similarity metrics. The difference in pair-wise correlation decreased by 14% compared to CTAB-GAN. This confirms that the FNO approach indeed captures cross-column relations in a more effective way than CTAB-GAN.

⁴<https://github.com/mostly-ai/virtualdatalab>

Lastly, for privacy preservation, it is important to first understand the relativity of this metric. If the distance between real and synthetic records is too small, there is a risk of privacy breach. On the other hand, if the distance is too large, the quality of the synthetic data could be poor. FCT-GAN results in higher DCR and NNDR than CTAB-GAN while increasing synthetic data quality as shown by the ML utility and statistical similarity metrics. Therefore, FCT-GAN slightly decreased the risk of privacy breach which is an underlying goal of tabular data synthesis (explained in section 1).

5.4 FNO Dimensionality Analysis

As outlined in subsection 4.4, multiple dimensionalities are possible for the implementation of the FNO framework in FCT-GAN. The informal explanation for the two-dimensional superiority given in subsection 4.4 is confirmed with an empirical analysis. For the sake of this analysis, a new model: **FCT-GAN-1D**, is created. FCT-GAN-1D is a modification of FCT-GAN where the FNO framework is one-dimensional one. All other aspects such as the amount of Fourier blocks (four) is unaltered. Table 2 shows that FCT-GAN-1D heavily underperforms compared to FCT-GAN and CTAB-GAN across all evaluation metrics. This confirms that the one-dimensional FNO variant is less effective for tabular data synthesis. The flattening of the inherently two-dimensional tabular data, resulted in a loss of spatial signal which can consequently not be captured by the FNO anymore. Hence, resulting in less effective tabular data synthesis, which answers research sub-question 1 in subsection 3.3.

5.5 Different FNO Component Analysis

FCT-GAN only integrates the FNO framework into its discriminator. The other components (Generator and Classifier) are identical to the ones in CTAB-GAN. Subsection 4.3 provides an informal hypothesis stating that only the discriminator should have a global understanding of the data. This is confirmed by an empirical analysis comparing different FNO configurations against FCT-GAN. Since the Classifier does not suffer from the CNN kernel’s limited reach (see Figure 1), it is not considered for the experiment. For the sake of this analysis, two new models: **FCT-GAN-g** and **FCT-GAN-gd** are created. FCT-GAN-g only integrates the FNO framework into the generator, while FCT-GAN-gd integrates it into both the generator and discriminator. Both models identically implement the FNO like FCT-GAN as seen in Figure 3. When compared to FCT-GAN in Table 2, FCT-GAN-g and FCT-GAN-gd both thoroughly underperform across all metrics. This confirms the hypothesis and explanation given in subsection 4.3, suggesting to only integrate the FNO framework into the discriminator. This answers research sub-question 2 in subsection 3.3.

5.6 Column Permutation Invariant Analysis

To evaluate if FCT-GAN possesses the column permutation invariant property, the model is tested in a permutation experiment. For this experiment, the Adult and Loan dataset are used. On one side, these datasets are used because they are both small, which is appropriate considering the limited computational resources. On the other side, these datasets con-

tain a healthy amount of continuous, categorical variables, and mixed-type variables (see Table 1) compared to Credit (30 continuous, 1 categorical) or Intrusion (0 mixed-type). Firstly, FCT-GAN is run on the original column order. Secondly, the columns are permuted such that all continuous variables are moved to the left, followed by the mixed type columns, and finally the categorical variables. This permutation allows for reproducibility and fair comparison across datasets. Thirdly, FCT-GAN is run on the permuted dataset. Lastly, the absolute difference between all evaluation metrics is computed (see Table 3). This experiment gets repeated three times and is also done for CTAB-GAN. The results in Table 3 show that FCT-GAN shows significantly lower metric differences than CTAB-GAN across all metrics except F1-score (CTAB-GAN: 0.040 versus FCT-GAN: 0.041). Especially accuracy (CTAB-GAN: 2.351% versus FCT-GAN: 0.520%) and WD (CTAB-GAN: 466.093 versus FCT-GAN: 0.031) significantly improve robustness against permutations. Hence, FCT-GAN is more robust to column permutations than CTAB-GAN because it captures all the data at once, instead of partially like CTAB-GAN. This answers research sub-question 3 in subsection 3.3.

6 Responsible Research

As outlined in section 1, one of the goals of tabular data synthesis is to enable private data-sharing. Contributing to this cause helps the data industry at preserving privacy as enforced by governmental data regulations such as the European GDPR. Additionally, to help promote reproducibility of this study, the full source code of FCT-GAN has been made public in a GitHub repository⁵.

7 Conclusion

Motivated by the significance of data sharing and the satisfaction of governmental regularization, this paper proposes FCT-GAN, the first conditional Generative Adversarial Network (GAN) that integrates a Fourier Neural Operator (FNO) to synthesize tabular data. It extends the state-of-the-art CTAB-GAN synthesizer by upgrading its discriminator component to a four-layered FNO model. Additionally, it concatenates a Multilayer Perceptron to the Fourier Layer, and adds regularization procedures and a binary classification adaptation. FCT-GAN advances beyond the state-of-the-art by capturing global data correlations in the frequency domain. Furthermore, FCT-GAN shows a remarkable resemblance between its results on permuted versions of same datasets. This indicates the presence of the column permutation invariant property, which is essential for an effective tabular data generator. The results show that synthetic data generated by FCT-GAN achieve higher machine learning utility, higher statistical similarities to real data, and superior privacy guarantees compared to the state-of-the-art. FCT-GAN offers an improvement of up to 19% in accuracy on five commonly used datasets, compared to CTAB-GAN. These results demonstrate its potential for a wide range of applications that greatly benefit from data sharing, such as banking, insurance, and manufacturing.

⁵<https://github.com/ethan-keller/FCT-GAN>

A Dataset descriptions

Dataset	Train/Test split	Target variable	Continuous	Binary	Multi-class	Mixed-type
Adult	39k / 9k	[Income]	3	2	7	2
Covertypes	45k / 5k	[Cover_Type]	10	44	1	0
Credit	40k / 10k	[Class]	30	1	0	0
Intrusion	45k / 5k	[Class]	22	6	14	0
Loan	4k / 1k	[PersonalLoan]	5	5	2	1

Table 1: The description of five known ML datasets used for evaluation of FCT-GAN.

B Evaluation results of FCT-GAN, CTAB-GAN and FCT-GAN modifications

Method	ML Utility Difference			Statistical Similarity			Privacy Preservation					
							DCR			NNDR		
	Accuracy	F1-score	AUC	Avg JSD	Avg WD	Diff. Corr.	R&S	R	S	R&S	R	S
FCT-GAN	7.46%	0.243	0.141	0.031	518	2.06	1.225	0.383	0.877	0.723	0.439	0.599
CTAB-GAN	10.59%	0.184	0.147	0.074	1344	2.39	1.034	0.383	0.828	0.739	0.439	0.578
FCT-GAN-gd	13.33%	0.294	0.250	0.129	4586	2.98	1.238	0.383	0.785	0.739	0.439	0.363
FCT-GAN-g	15.26%	0.302	0.211	0.256	13808	3.60	1.386	0.383	0.735	0.722	0.439	0.321
FCT-GAN-ID	14.89%	0.252	0.246	0.091	2005	3.01	1.391	0.383	0.828	0.682	0.439	0.477

Table 2: Evaluation results of FCT-GAN and its modifications against the state-of-the-art CTAB-GAN model on ML utility, statistical similarity, and privacy.

C Evaluation results of FCT-GAN and CTAB-GAN after a column permutation

Method	ML Utility Difference			Statistical Similarity			Privacy Preservation					
							DCR			NNDR		
	Accuracy	F1-score	AUC	Avg JSD	Avg WD	Diff. Corr.	R&S	R	S	R&S	R	S
FCT-GAN	0.520%	0.041	0.037	0.003	0.031	0.032	0.004	0.0	0.025	0.009	0.0	0.002
CTAB-GAN	2.351%	0.040	0.125	0.010	466.093	0.111	0.030	0.0	0.051	0.036	0.0	0.031

Table 3: Evaluation results of FCT-GAN and the state-of-the-art CTAB-GAN model, after a permutation, on ML Utility, Data Similarity, and Privacy.

References

- [1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan, 2017.
- [2] Karim Armanious, Chenming Jiang, Marc Fischer, Thomas Küstner, Tobias Hepp, Konstantin Nikolaou, Sergios Gatidis, and Bin Yang. MedGAN: Medical image translation using GANs. *Computerized Medical Imaging and Graphics*, 79:101684, jan 2020.
- [3] Marc G. Bellemare, Ivo Danihelka, Will Dabney, Shakir Mohamed, Balaji Lakshminarayanan, Stephan Hoyer, and Rémi Munos. The cramer distance as a solution to biased wasserstein gradients, 2017.
- [4] Kamran Chitsaz, Mohsen Hajabdollahi, Nader Karimi, Shadrokh Samavi, and Shahram Shirani. Acceleration of convolutional neural network using fft-based split convolutions, 2020.
- [5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2020.
- [6] Jesse Engel, Kumar Krishna Agrawal, Shuo Chen, Ishaan Gulrajani, Chris Donahue, and Adam Roberts. GANSynth: Adversarial neural audio synthesis. In *International Conference on Learning Representations*, 2019.
- [7] Chao Fang, Liulu He, Haonan Wang, Jinghe Wei, and Zhongfeng Wang. Accelerating 3d convolutional neural networks using 3d fast fourier transform. pages 1–5, 05 2021.
- [8] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.
- [9] Steven Guan, Ko-Tsung Hsu, and Parag V. Chitnis. Fourier neural operator networks: A fast and general solver for the photoacoustic wave equation, 2021.
- [10] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans, 2017.
- [11] He Huang, Philip S. Yu, and Changhu Wang. An introduction to image synthesis with generative adversarial nets, 2018.
- [12] T. Konuk and J. Shragge. Physics-guided deep learning using fourier neural operators for solving the acoustic vti wave equation. 2021(1):1–5, 2021.
- [13] Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces, 2021.
- [14] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network, 2016.
- [15] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations, 2020.
- [16] Zinan Lin, Ashish Khetan, Giulia Fanti, and Sewoong Oh. Pacgan: The power of two samples in generative adversarial networks, 2017.
- [17] Alejandro Mottini, Alix Lheritier, and Rodrigo Acuna-Agost. Airline passenger name record generation using generative adversarial networks, 2018.
- [18] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans, 2016.
- [19] Noseong Park, Mahmoud Mohammadi, Kshitij Gorde, Sushil Jajodia, Hongkyu Park, and Youngmin Kim. Data synthesis based on generative adversarial networks. *Proceedings of the VLDB Endowment*, 11(10):1071–1083, jun 2018.
- [20] Noseong Park, Mahmoud Mohammadi, Kshitij Gorde, Sushil Jajodia, Hongkyu Park, and Youngmin Kim. Data synthesis based on generative adversarial networks. *Proceedings of the VLDB Endowment*, 11(10):1071–1083, jun 2018.
- [21] Yongming Rao, Wenliang Zhao, Zheng Zhu, Jiwen Lu, and Jie Zhou. Global filter networks for image classification, 2021.
- [22] Chao Song and Yanghua Wang. High-frequency wave-field extrapolation using the fourier neural operator. *Journal of Geophysics and Engineering*, 19:269–282, 04 2022.
- [23] Farhana Sultana, Abu Sufian, and Paramartha Dutta. Advancements in image classification using convolutional neural network. In *2018 Fourth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*. IEEE, nov 2018.
- [24] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. Deep & cross network for ad click predictions, 2017.
- [25] Gege Wen, Zongyi Li, Kamyar Azizzadenesheli, Anima Anandkumar, and Sally M. Benson. U-fno – an enhanced fourier neural operator-based deep-learning model for multiphase flow, 2021.
- [26] Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional gan, 2019.
- [27] Han Yan, Guohua Lv, Xiaoqiang Ren, and Xiangjun Dong. Improved Nearest Neighbor Distance Ratio for Matching Local Image Descriptors: First CCF International Conference, ICAI 2018, Jinan, China, August 9-10, 2018, *Proceedings*, pages 185–197. 08 2018.

- [28] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. PATE-GAN: Generating synthetic data with differential privacy guarantees. In *International Conference on Learning Representations*, 2019.
- [29] Taraneh Younesian, Zilong Zhao, Amirmasoud Ghiassi, Robert Birke, and Lydia Y. Chen. Qactor: On-line active learning for noisy labeled stream data, 2020.
- [30] Ying Yu, Bingying Tang, Ronglai Lin, Shufa Han, Tang Tang, and Ming Chen. Cwgan: Conditional wasserstein generative adversarial nets for fault data generation. In *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 2713–2718, 2019.
- [31] Zilong Zhao, Robert Birke, Rui Han, Bogdan Robu, Sara Bouchenak, Sonia Ben Mokhtar, and Lydia Y. Chen. Enhancing robustness of on-line learning models on highly noisy data. *IEEE Transactions on Dependable and Secure Computing*, pages 1–1, 2021.
- [32] Zilong Zhao, Sophie Cerf, Robert Birke, Bogdan Robu, Sara Bouchenak, Sonia Ben Mokhtar, and Lydia Y. Chen. *Robust Anomaly Detection on Unreliable Data*. 2019.
- [33] Zilong Zhao, Aditya Kunar, Hiek Van der Scheer, Robert Birke, and Lydia Y. Chen. Ctab-gan: Effective table data synthesizing, 2021.