INFORMS JOURNAL ON DATA SCIENCE
Vol. 00, No. 0, Xxxxx 0000, pp. 000–000
ISSN 2694-4022 | EISSN 2694-4030 | 00 | 0000 | 0001

INFORMS
DOI 10.1287/xxxx.0000.0000
© 0000 INFORMS

Authors are encouraged to submit new papers to INFORMS journals by means of a style file template, which includes the journal title. However, use of a template does not certify that the paper has been accepted for publication in the named journal. INFORMS journal templates are for the exclusive purpose of submitting to an INFORMS journal and should not be used to distribute the papers in print or online or to submit the papers to another publication.

Time series modeling for synthetic relational database generation

Philippe Beliveau
HEC Montreal

Supervisor: Maryam Daryalal
HEC Montreal

Supervisor: Yazid Attabi
Croesus Lab

This paper introduces an attempt to generate synthetic temporal data to capture temporal dynamics while maintaining data quality. We extend the capabilities of the Row-Conditional Tabular Generative Adversarial Network (RCTGAN) to incorporate temporal information by leveraging mini-batch generation and the power spectrum of Fast Fourier Transform (FFT). Our approach aims to address the limitations of RNNs base architecture in generating long sequences by employing a mini-batch strategy and parallel generation within each batch. To enhance the capture of long-term dependencies, we incorporate the mini-batch approach of the DoppelGANger model and test a novel method using FFT features into the generator. We evaluate the performance of our proposed method and show the difference in the generated data when dealing with time series data compared to the RCTGAN.

1. Introduction

Synthetic data generation, particularly using Generative Adversarial Networks (GANs), has emerged as a powerful solution to address data scarcity and privacy concerns. While GANs have shown remarkable success in various domains, their application to generating synthetic relational databases with temporal dependencies remains an under-explored area, especially within a deep learning context.

Applying GANs to relational-sequential data offers numerous benefits, including data augmentation, imputation, denoising, and enhanced privacy protection. However, GANs also face challenges such as training instability (non-convergence, vanishing gradients, mode collapse) and the lack of established evaluation metrics for time-series GANs.

This work focuses on advancing the generation of synthetic relational-sequential data using GANs. We define time series as sequences of vectors dependent on time, which can be continuous or discrete, and their values can be either continuous or discrete. The time series are *measurements/children* tables e.g sales of a store, associated with multi-dimensional *metadata/parents* tables e.g store characteristics.

By tackling the unique challenges associated with this type of data, we aim to contribute to the development of robust and effective methods for generating synthetic relational-sequential datasets.

1.1. Literature review

The incorporation of time series modeling into the generation of synthetic relational databases presents unique challenges that necessitate the development of robust and effective methodologies.

Several approaches have been proposed to tackle the problem of generating synthetic time series data, but very few papers have been tackling the problem of generating synthetic relational-sequential tabular data. On one hand, the generation of relational tables has been researched by the RCTGAN paper, which employs conditional information to generate synthetic rows within a tabular format and model the relationship between the table, but it doesn't inherently model the temporal aspect and lacks comprehensive evaluation of temporal correlations Gueye et al. (2022). On the other hand, numerous papers have been working on generating synthetic time series but without modeling the relationship between the *measurements metadata* of the time series. TIMEGAN leverages a supervisor to encourage the generator to learn temporal dynamics, but it doesn't explicitly consider conditional aspects, such as the relation between tables or handle specific challenges found when generating tabular data Yoon et al. (2019). TTS-GAN utilizes a pure transformer encoder-decoder architecture for time series generation Li et al. (2022), while Time-series Transformer Generative Adversarial Networks leverage, similarly to Time-GAN, use a supervisor that indirectly incentives the generator to generate synthetic data coherent with the learned conditional embedding distribution of the real data Srinivasan and Knottenbelt (2022). One important paper in the literature that effectively addresses the generation of *measurements* conditional to its *metadata* is the DoppelGANger (DG) paper, which introduces key innovations like dual networks for modeling more effectively the relational between entities and to generate more realistic time series, it also makes use of mini-batched generation, and auto-normalization to successfully capture of long-term auto-correlations and to tackle mode collapse Lin et al. (2020).

This paper aims to contribute to this evolving landscape by extending the capabilities of RCTGAN to incorporate temporal information by extending the RCTGAN capabilities to not only model the relation between tables, but also the temporal dimensions. This is by incorporating the approach from DoppelGANger in employing a mini-batch strategy and, by proposing a novel method utilizing Fast Fourier transform within the generator to enhance the capture of long-term dependencies.

1.2. Motivations and contributions

The objective of this research is to extend the capabilities of the original RCTGAN model to effectively handle and generate time-series data. This overarching goal has guided the development and refinement of our approach through a series of studies, each motivated by specific challenges and opportunities.

Model 1: Incorporating Temporal Modeling: Our initial investigation aimed to directly integrate temporal modeling into the RCTGAN framework. Recognizing the inherent sequential nature of time-series data, we introduced LSTM or GRU as the generator and discriminator. This approach sought to leverage the recurrent architecture of these models to capture dependencies across time steps. While maintaining key functionalities of CTGAN, such as conditional vector sampling, to address common challenges in synthetic tabular data generation.

Model 2: Addressing Long Sequence Generation: A recognized limitation of RNN-based architectures is their difficulty in capturing long-term dependencies. To overcome this challenge, we adopted a mini-batch approach proposed by the DoppelGANger paper for parent-child data generation Lin et al. (2020). By breaking down the generation process into smaller, manageable batches, we reduced the computational burden on the RNN, enabling it to generate longer sequences more effectively and replicating the autocorrelation at longer time steps. The generation is done by doing parallel generation of children within mini-batches using fully connected layers, more details in section **??**. This approach allowed us to model the conditional distribution between mini-batches and the joint distribution within a mini-batch while retaining some essential features of CTGAN.

Model 3: Adding temporal information: Following the mini-batch approach, we shifted our focus to enhance the model's ability to capture long-term dependencies within time-series data from mini-batch to Fast Fourier Transform (FFT). To achieve this, we employed FFT to extract relevant frequency information from continuous columns. These FFT representations served as additional features for the generator, enriching its understanding of temporal patterns. We systematically tuned the number of relevant FFT spectrum components to optimize feature representation and explored various combinations of vector length and relevant spectrum selections.

The paper is organized as follows. Section 2 formally describes the problem and details the models. Section 3 outlines and explains each model architecture. Section 4 describes the datasets and metrics used to perform the evaluation. Section 5 delivers a numerical study to evaluate the impact of the change made to RCTGAN to capture the temporal aspect of the data. Finally, Section 6 concludes the paper with reflections on our findings and directions for future research.

2. Problem Description

Notations

Symbols	Description
z_{ij}	**Noise vector** for child j from parent i
z_{ic}	**Noise vector** for mini-batch c from parent i
x_{ij}	**Real data** for child j from parent i
$\hat{x}_{ij}$	**Synthetic data** for child j from parent i
$\hat{x}_i^{(c)}$	**Synthetic data** at mini batch c from parent i
y_i	**Features of parent** i
$cond_{ij}$	**Conditional vector** (see CTGAN) for child j from parent i
$cond_i^{(c)}$	**Conditional vector** for mini-batch c from parent i
fft_{ij}	**Power spectrum vector** for child j from parent i
$\mathcal{G}_1$	Generator of classical RCTGAN
$\mathcal{D}_1$	Discriminator of classical RCTGAN
$\mathcal{G}_2$	Generator of Sequential RCTGAN
$\mathcal{D}_2$	Discriminator of Sequential RCTGAN
$\mathcal{G}_3$	Generator of Mini-Batch RCTGAN
$\mathcal{D}_3$	Discriminator of Mini-Batch RCTGAN
$\mathcal{G}_4$	Generator of FFT RCTGAN
$\mathcal{D}_4$	Discriminator of FFT RCTGAN
$\mathcal{L}_{G_1}$	Loss generator for classical RCTGAN
$\mathcal{L}_{D_1}$	Loss discriminator for classical RCTGAN
$\mathcal{L}_{G_2}$	Loss generator for Sequential RCTGAN
$\mathcal{L}_{D_2}$	Loss discriminator for Sequential RCTGAN
$\mathcal{L}_{G_3}$	Loss generator for Mini-Batch RCTGAN
$\mathcal{L}_{D_3}$	Loss discriminator for Mini-Batch RCTGAN
$\mathcal{L}_{G_4}$	Loss generator for FFT RCTGAN
$\mathcal{L}_{D_4}$	Loss discriminator for FFT RCTGAN

2.1. Classical RCTGAN

We will describe the model by the example of child j from parent i.

2.1.1. Generator

$$\begin{cases} h_0 = z_{ij} \oplus cond_{ij} \oplus y_i \\ h_1 = h_0 \oplus \text{ReLU}\left(\text{BN}\left(\text{FC}(h_0)\right)\right) \\ h_2 = h_1 \oplus \text{ReLU}\left(\text{BN}\left(\text{FC}(h_1)\right)\right) \end{cases}$$

where $\oplus$ is a concatenation

$\mathcal{G}_1\left(z_{ij}, cond_{ij}, y_i\right) = \text{Activation}\left(h_2\right) = \hat{x}_{ij}$

$\mathcal{L}_{G_1} = -\mathbb{E}\left[\mathcal{D}_1\left(\hat{x}_{ij}\right)\right]$

2.1.2. Discriminator $\mathcal{D}_1\left(x_{ij}, cond_{ij}, y_i\right) = c_{ij} \in \mathbb{R}$

$\mathcal{D}_1\left(\hat{x}_{ij}, cond_{ij}, y_i\right) = \hat{c}_{ij} \in \mathbb{R}$

$\mathcal{L}_{D_1} = -\left[\mathbb{E}\left[x_{ij}\right] - \mathbb{E}\left[\hat{x}_{ij}\right]\right] + \lambda\text{GP}$

2.2. Model 1: Sequential RCTGAN

We will describe the model by the example of children $1, \ldots, l$ from parent i.

2.2.1. Generator For $j = 1, \ldots, l$: $(\hat{x}_{ij}, h_j) = \Phi_G\left(h_{j-1}, z_{ij} \oplus cond_{ij} \oplus y_i\right)$ where Φ_G is the fully connected layer associated to the Generator LSTM model.

$\mathcal{G}_2\left(t_{i1}, \ldots, t_{il}\right) = \left(\hat{x}_{i1}, \ldots, \hat{x}_{il}\right)$, where $t_{ij} = z_{ij} \oplus cond_{ij} \oplus y_i$.

$\mathcal{L}_{G_2} = -\mathbb{E}\left[\mathcal{D}_2\left(\hat{s}_{i1}, \ldots, \hat{s}_{il}\right)\right]$ where $\hat{s}_{ij} = \hat{x}_{ij} \oplus cond_{ij} \oplus y_i$

2.2.2. Discriminator For $j = 1, \ldots, l$:

$$\begin{cases} (\hat{c}_{ij}, h_j) = \Phi_D\left(h_{j-1}, \hat{x}_{ij} \oplus cond_{ij} \oplus y_i\right) \\ (c_{ij}, h_j) = \Phi_D\left(h_{j-1}, x_{ij} \oplus cond_{ij} \oplus y_i\right) \end{cases}$$

where Φ_D is the fully connected layer associated to the Discriminator LSTM model.

$$\begin{cases} \mathcal{D}_2\left(s_{i1}, \ldots, s_{il}\right) = \frac{1}{l}\sum_{j=1}^{l} c_{ij} \in \mathbb{R} \\ \mathcal{D}_2\left(\hat{s}_{i1}, \ldots, \hat{s}_{il}\right) = \frac{1}{l}\sum_{j=1}^{l} \hat{c}_{ij} \in \mathbb{R} \end{cases}$$

$\mathcal{L}_{D_2} = -\left[\mathbb{E}\left[\mathcal{D}_2\left(s_{i1}, \ldots, s_{il}\right)\right] - \left[\mathbb{E}\left[\mathcal{D}_2\left(\hat{s}_{i1}, \ldots, \hat{s}_{il}\right)\right]\right]\right]$

where $\hat{s}_{ij} = \hat{x}_{ij} \oplus cond_{ij} \oplus y_i$ and $s_{ij} = x_{ij} \oplus cond_{ij} \oplus y_i$

2.3. Model 2: Mini-batch

2.3.1. Generator We will describe the model by the example of children $1,\dots,l$ from parent i.

We will describe the model describe $1\dots M$

For $c=1,\dots,m$: $(\hat{x}_{isc},h_c)=\Phi_G\left(h_{c-1},z_{ic}\oplus cond_{ic}\oplus y_i\right)$ where Φ_G is the Generator LSTM model.

$$\mathcal{G}_3\left(t_i^{(1)},\dots,t_i^{(m)}\right)=\left(\hat{x}_i^{(1)},\dots,\hat{x}_i^{(m)}\right),\text{ where } t_i^{(c)}=z_i^{(c)}\oplus cond_i^{(c)}\oplus y_i$$
$$\mathcal{L}_{G_3}=-\mathbb{E}\left[\mathcal{D}_3\left(\hat{s}_i^{(1)},\dots,\hat{s}_i^{(m)}\right)\right]\text{ where } \hat{s}_i^{(c)}=\hat{x}_i^{(c)}\oplus cond_{ic}\oplus y_i$$

2.3.2. Discriminator For $j=1,\dots,l$:

$$\begin{cases}(\hat{c}_i^{(m)})=\Phi_D\left(\hat{x}_{ij}\oplus cond_{ij}\oplus y_i\right)\\(c_i^{(m)})=\Phi_D\left(x_{ij}\oplus cond_{ij}\oplus y_i\right)\end{cases}$$

where Φ_D is the fully connected layer associated to the Discriminator Fully connected layer model.

$$\begin{cases}\mathcal{D}_2\left(s_i^{(1)},\dots,s_i^{(m)}\right)=c_i\in\mathbb{R}\\\mathcal{D}_2\left(\hat{s}_i^{(1)},\dots,\hat{s}_i^{(m)}\right)=\hat{c}_i\in\mathbb{R}\end{cases}$$
$$\mathcal{L}_{D_3}=-\left[\mathbb{E}\left[\mathcal{D}_3\left(s_i^{(1)},\dots,s_i^{(m)}\right)\right]-\left[\mathbb{E}\left[\mathcal{D}_3\left(\hat{s}_i^{(1)},\dots,\hat{s}_i^{(m)}\right)\right]\right]\right]$$

2.4. Model 3: Fast Fourier Transform

2.4.1. Generator For $j=1,\dots,l$: $(\hat{x}_{ij},h_j)=\Phi_G\left(h_{j-1},z_{ij}\oplus cond_{ij}\oplus fft_{ij}\oplus y_i\right)$ where Φ_G is the fully connected layer associated to the Generator LSTM model.

$$\mathcal{G}_4\left(t_{i1},\dots,t_{il}\right)=\left(\hat{x}_{i1},\dots,\hat{x}_{il}\right),\text{ where } t_{ij}=z_{ij}\oplus cond_{ij}\oplus fft_{ij}\oplus y_i.$$
$$\mathcal{L}_{G_4}=-\mathbb{E}\left[\mathcal{D}_4\left(\hat{s}_{i1},\dots,\hat{s}_{il}\right)\right]\text{ where } \hat{s}_{ij}=\hat{x}_{ij}\oplus cond_{ij}\oplus fft_{ij}\oplus y_i$$

2.4.2. Discriminator For $j=1,\dots,l$:

$$\begin{cases}(\hat{c}_{ij},h_j)=\Phi_D\left(h_{j-1},\hat{x}_{ij}\oplus cond_{ij}\oplus fft_{ij}\oplus y_i\right)\\(c_{ij},h_j)=\Phi_D\left(h_{j-1},x_{ij}\oplus cond_{ij}\oplus fft_{ij}\oplus y_i\right)\end{cases}$$

where Φ_D is the fully connected layer associated to the Discriminator LSTM model.

$$\begin{cases}\mathcal{D}_4\left(s_{i1},\dots,s_{il}\right)=\frac{1}{l}\sum\limits_{j=1}^{l}c_{ij}\in\mathbb{R}\\\mathcal{D}_4\left(\hat{s}_{i1},\dots,\hat{s}_{il}\right)=\frac{1}{l}\sum\limits_{j=1}^{l}\hat{c}_{ij}\in\mathbb{R}\end{cases}$$

$$\mathcal{L}_{D_4} = -\left[\mathbb{E}\left[\mathcal{D}_4\left(s_{i1},\ldots,s_{il}\right)\right] - \left[\mathbb{E}\left[\mathcal{D}_4\left(\hat{s}_{i1},\ldots,\hat{s}_{il}\right)\right]\right]\right]$$

where $\hat{s}_{ij} = \hat{x}_{ij} \oplus cond_{ij} \oplus fft_{ij} \oplus y_i$ and $s_{ij} = x_{ij} \oplus cond_{ij} \oplus fft_{ij} \oplus y_i$

3. Architecture

3.1. Model 1: Sequential RCTGAN

Figure 1 illustrates a proposed architecture for generating synthetic sequential tabular data that respects both inter-table relationships and temporal patterns. This architecture builds upon the Recurrent Conditional Tabular Generative Adversarial Network (RCT-GAN) framework, incorporating sequence modeling capabilities.

The Generator employs a Long Short-Term Memory (LSTM) or Gated Recurrent Unit (GRU), to generate sequences of ”children” data conditioned on their corresponding ”parent” information and additional conditional vectors. This enables the model to capture sequential dependencies and generate data that aligns with the underlying parent-child relationships.

The Critic also implemented as an RNN (LSTM or GRU), acts as the discriminator in the adversarial framework. It assesses the authenticity of both real and generated sequences, taking into account the parent information, and the conditional vectors. The critics output a score for each row, after passing through an RNN and then a fully connected layer, then the scores of all children for a single parent are aggregated into a mean score to give a Wasserstein score on the whole sequence, which guides the training of both the generator and the critic.

3.2. Model 2: Mini-Batch

In our implementation of the DoppelGANger (DG) model, we employ an approach to tackle the challenge of generating long, temporally coherent sequences. The generator utilizes an LSTM network to process the input noise sequentially, capturing the temporal dependencies inherent in the data. However, to mitigate the issue of LSTMs struggling to retain long-term correlations over extended sequences, we use the mini-batch mechanism of DoppelGANger (see figure 2).

The input noise is divided into chunks, each representing a segment of the final sequence. The LSTM processes these chunks sequentially, but only the final hidden state of each chunk is retained. This final hidden state serves as a condensed representation of the temporal information learned from that chunk.

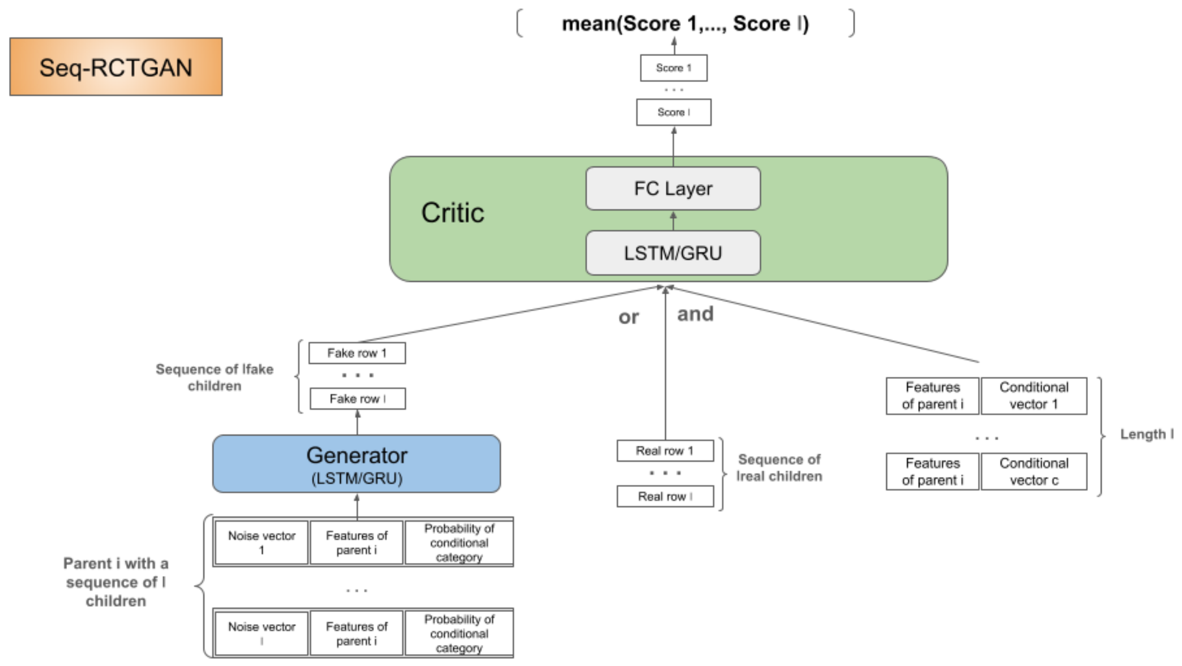

Figure 1 **Sequential RCTGAN Architecture**

Subsequently, a set of OutputDecoder modules, whose quantity is determined by the sample length parameter (representing the time series batch size), receives these final hidden states. Each OutputDecoder is responsible for generating the output for all features at a specific time step within a chunk. The OutputDecoders themselves consist of simple linear layers, they do not implicitly learn their positional roles within a chunk through their weights for example, thus the Output decoders are not specialized in generating the output for their assigned time step. This entails that we are not modeling the conditional distribution within a chunk, but effectively the joint distribution of the chunk. The shared final hidden state from the LSTM for each chunk provides a common context, ensuring coherence between the outputs of different OutputDecoders within that chunk.

Furthermore, we extended the DG architecture to include the conditional vector from the CTGAN model. Each output decoder, instead of only receiving the final hidden state of the last chunk, it receives aditionnally a conditional vector corresponding to a specific category to condition the generation on.

Finally, a Merger module combines the outputs from all OutputDecoders for a given chunk, concatenating them along the time dimension to produce a complete chunk of the generated sequence. This process is repeated for all chunks, and the generator then assembles these chunks to construct the final sequence.

The important parameter here is the sample length (S) parameter that defines the size of the generated mini-batch at each chunk.

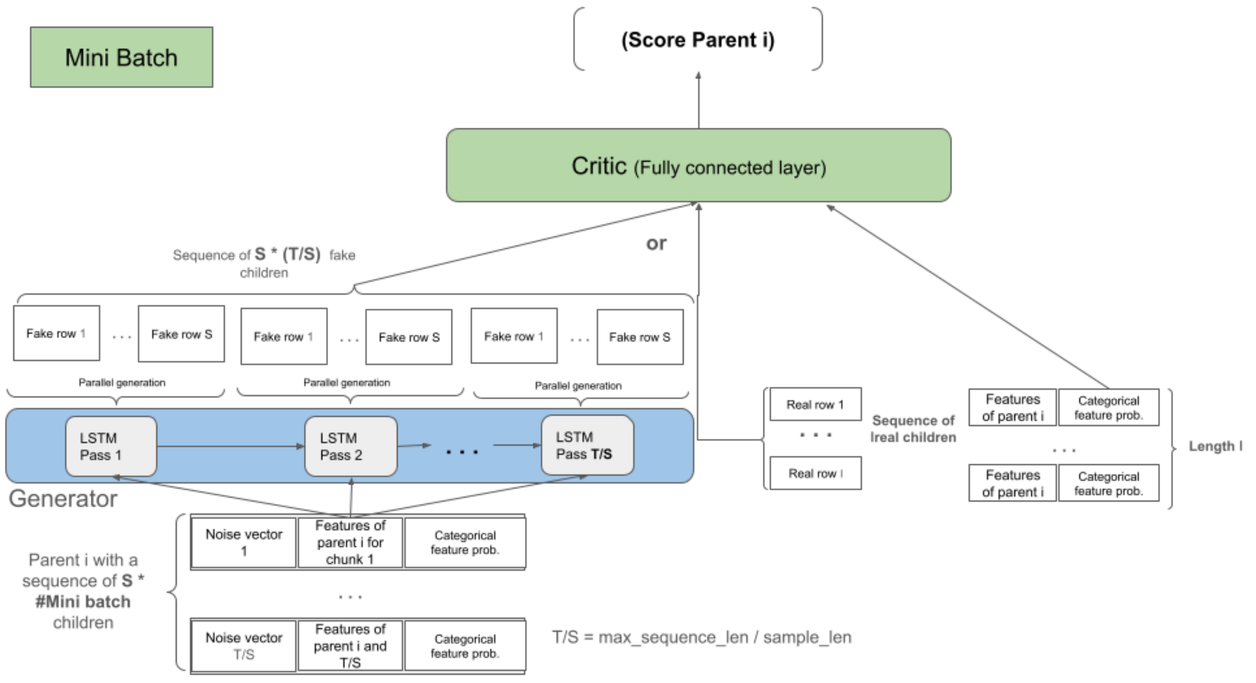
Mini Batch
(Score Parent i)
Critic (Fully connected layer)
or
Sequence of S * (T/S) fake children
Fake row 1
Fake row S
Parallel generation
LSTM Pass 1
LSTM Pass 2
LSTM Pass T/S
Generator
Real row 1
Real row l
Sequence of lreal children
Features of parent i
Categorical feature prob.
Length l
Parent i with a sequence of S * #Mini batch children
Noise vector 1
Features of parent i for chunk 1
Categorical feature prob.
Noise vector T/S
Features of parent i and T/S
T/S = max_sequence_len / sample_len

Figure 2 **Mini-Batch Architecture**

3.3. Model 3: Fast Fourier Transform

In order to get around the problem of replicating the temporal dynamics present in the data, we implemented a novel FFT-enhanced GAN within the RCTGAN framework (see figure 5. This enhancement was motivated by the need to explicitly model periodic patterns and long-term dependencies often present in time-series data. This attempt was to replace the model 2.

Our FFT implementation involved several key steps:

Power Spectrum Computation: At each time step during the generation process, we computed the Fast Fourier Transform (FFT) on the sequence generated up to that point. Subsequently, we calculated the power spectrum by taking the absolute value of the FFT results. This power spectrum encapsulates the frequency distribution of the generated sequence, highlighting dominant periodicities.

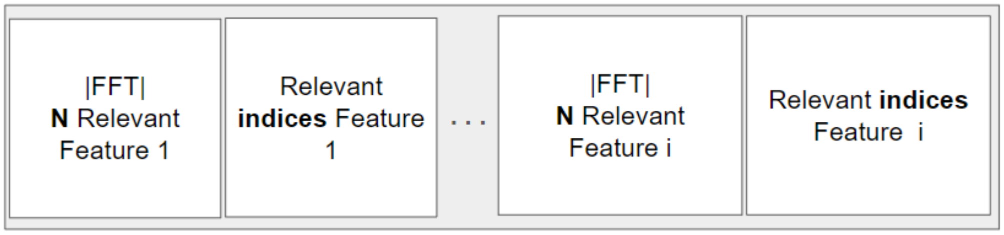

Figure 3 **Most relevant spectrum**

Selective Dimensionality Reduction: Recognizing that not all frequencies in the power spectrum contribute equally to the data's temporal characteristics, we implemented two mechanisms for dimensionality reduction described in figure **??** and **??**. In the first mechanism, we sorted the power spectrum for each continuous column in descending order and retained only the top 'n' most prominent frequencies, where 'n' is a hyperparameter controlling the level of dimensionality reduction. This approach prioritizes the most informative frequency components while discarding less significant ones, thereby improving computational efficiency without sacrificing essential temporal information.

Index Retention: To maintain the association between the extracted spectral components and their original frequencies, we carefully preserved the indices corresponding to the selected top 'n' frequencies. This index retention is crucial because the power spectra for different columns are sorted in a descending format.

Concatenation and Input Augmentation: The selected spectral components and their indices were concatenated to form an augmented input vector. This augmented vector, combining both the noise vector, the conditional vector and the extracted FFT features, was then fed into the LSTM or GRU layer of the generator at each time step. This integration allowed the model to leverage the spectral information to guide the generation of subsequent data points.

Padding and Shape Consistency: To ensure compatibility with the LSTM/GRU input dimensions, we padded the power spectra to maintain a consistent shape across all time steps. This padding accounted for the varying lengths of the generated sequences at different time steps.

Shown in figure 4 is the second mechanism tried, multiple approaches were tested:

- Adding the entire FFT vector without sorting or index vectors (for all variables)
- Adding the lower half of the FFT vector without sorting or index vectors (for all variables)

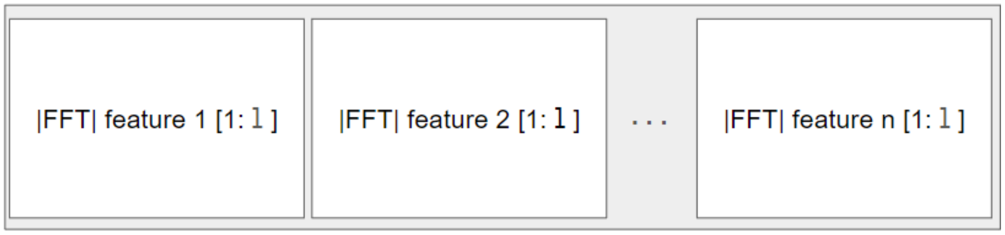

Figure 4 **Power spectrum not sorted**

- Adding the lower third of the FFT vector without sorting (for all variables)
- Adding the lower quarter of the FFT vector without sorting (for all variables)

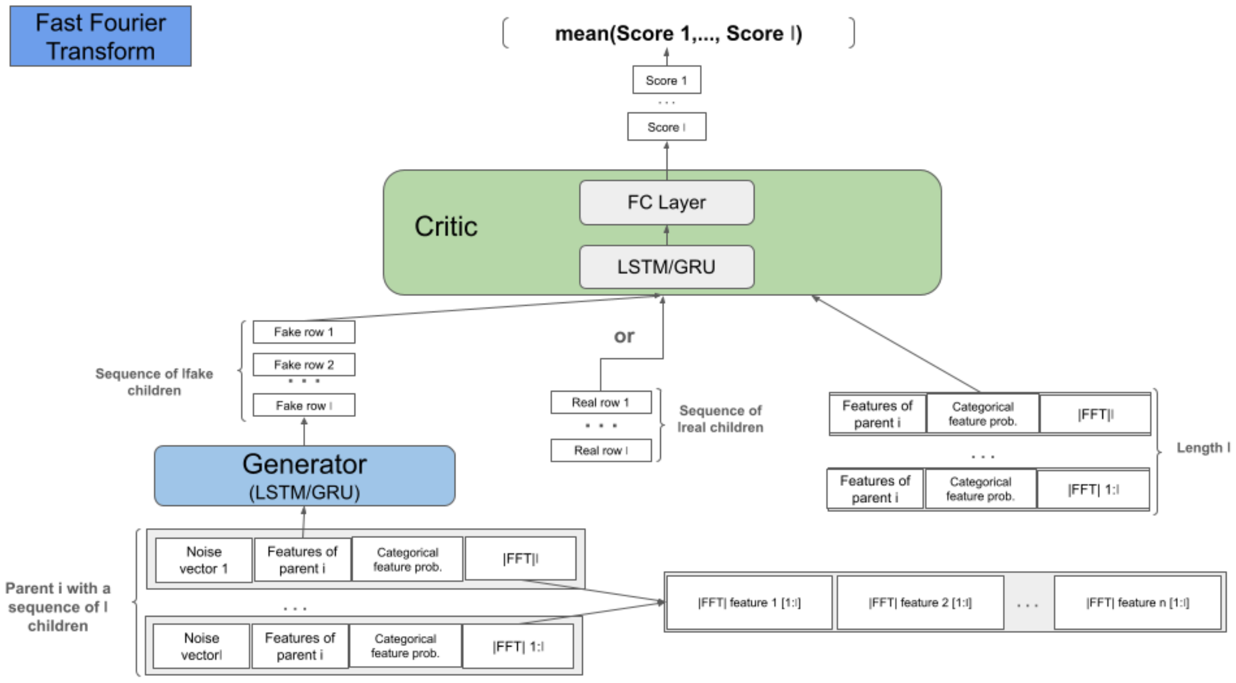

Figure 5 **Fast Fourier Transform Architecture**

4. Solution Methodology

4.1. Setup

4.1.1. Datasets . The dataset was chosen to exhibit different combinations of challenges: (1) correlations within time series and metadata, (2) multi-dimensional measurements, and/or (3) variable measurement lengths. Because of computation and infrastructure constraints, we were not able to get comprehensive results in multiple datasets, thus our analysis is limited to only one dataset.

Rossmann Daily sales for 1,115 stores located across Germany. Store sales are influenced by many factors, including promotions, competition, school and state holidays, seasonality, and locality. With thousands of individual managers predicting sales based on their unique circumstances, the accuracy of results can be quite varied. https://www.kaggle.com/c/rossmann-store-sales

4.1.2. Baseline This study strives to exceed the baseline performance established by RCTGAN. We focused on enhancing metrics specifically related to the temporal dimension without compromising other aspects. Our goal was to generate synthetic data that were as realistic as the one produced by RCTGAN, while demonstrating superior performance in capturing the autocorrelation structure inherent in the original data.

4.1.3. Metrics The metrics employed for evaluation were twofold:
Visual Inspection: This involved examining graphical representations of the data, including randomly selected samples and aggregated summaries (mean) of the generated data points across all parent entities. This approach allowed for a qualitative assessment of the data's realism and adherence to the desired patterns.
Numerical Measurements: Quantitative metrics were also computed to provide a more objective and precise evaluation. These numerical measurements corresponded to the same aspects assessed visually, offering a way to compare and quantify the performance of different models or techniques.

Temporal correlations. Our primary focus was on capturing the temporal dependencies within the data. We aimed to replicate the autocorrelation structure of the original data as faithfully as possible. To quantify this, we calculated the Mean Absolute Error (MAE) between the average autocorrelation of the synthetic data (generated across all parent entities) and the autocorrelation of the real data.

Metadata distributions. We also sought to maintain the quality of parent generation (metadata) achieved by the original RCTGAN model. Generating realistic parent entities is crucial not only for learning the relationships between measurements and metadata but also for producing plausible synthetic children.

Note: No numerical measurements were collected for this aspect.

Measurement distributions. This metric evaluates how well the distributions of the generated children align with those of the real children.

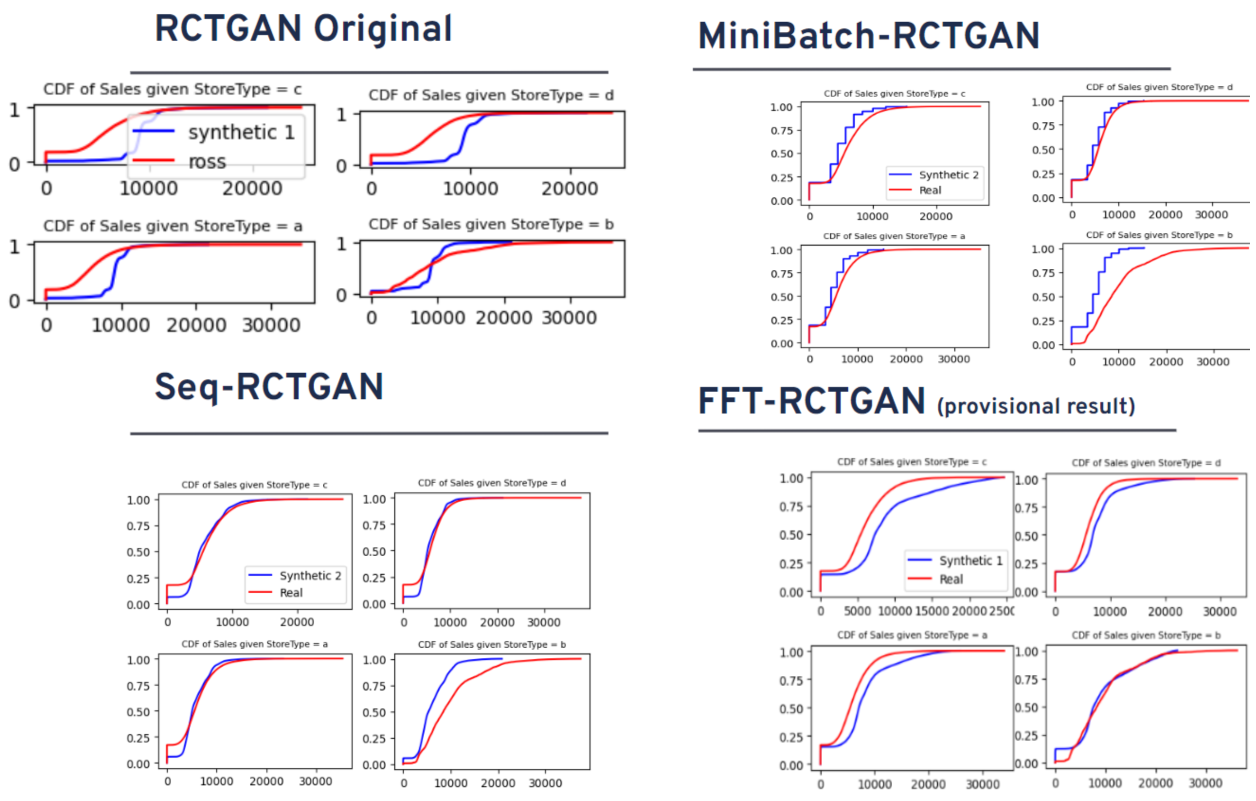

Figure 6 **Parent-Children distribution**

Note: No numerical measurements were collected for this aspect.

Cross-correlation. Preserving the correlations between different measurements within the generated children is another important objective. We calculated the average cross-correlation between the synthetic data and compared it against the real cross-correlation.

Measurement-metadata correlations. Learning the joint metadata-measurement distribution between children/measurements and parents/metadata is a considerably more challenging yet critical task than learning the parent distribution alone. To quantitatively assess this, we employed the Kolmogorov-Smirnov (KS) test. This test compares the distributions of the *measurements* between the synthetic and real datasets, stratified by metadata, allowing us to quantify any distributional discrepancies (see figure 6).

5. Numerical Experiments

5.1. Comparison between Baseline Model / Model 1 / Model 2

The table 1 presents a performance comparison of various models against the baseline RCTGAN classique, focusing on their ability to generate synthetic data that accurately reflects the real-world Rossman dataset. The evaluation encompasses metrics related to temporal correlations, cross-correlations, and the joint distribution of measurements and metadata.

Table 1 Rossman

Model	**Sequence length**	**Sample_len**	**MAE Autocorrelation - Sales**	**Average Cross-correlation Real = 0.4566**	**Average KS Statistics (Meta-Measurement distribution)**
RCTGAN	300	-	0.165	0.254	0.328
RCTGAN	500	-	0.17	0.25	0.33
Seq-RCTGAN	100	-	0.165933	0.172976	0.296
Seq-RCTGAN	300	-	0.165301	0.22636	0.426
Seq-RCTGAN*	400	-	0.164	0.261	0.210
Seq-MiniBatch	100	10	0.173	0.328	0.244
Seq-MiniBatch	300	10	0.178263	0.415	0.328
Seq-MiniBatch	500	10	0.151	0.240	0.264
FFT Relevant	100	-	0.166	0.157	0.707
FFT Relevant*	300	-	0.165	0.237	0.348
FFT half vector	100	-	0.165	0.149	0.373
FFT half vector*	300	-	0.166	0.192	0.830

[*] Due to hardware constraints, not many sequence lengths, sample lengths, and FFT experiments were conducted.

Model Performance Analysis:

- **Seq-RCTGAN do not outperform the baseline RCTGAN in capturing cross-correlations.** Seq-RCTGAN with a sequence length of 300 achieves a low average cross-correlation (0.22636), not close to the real data's cross-correlation of 0.4566. While both RCTGAN models achieve 0.254 and 0.25.

  In terms of autocorrelation, quantitatively, Seq-RCTGAN is not better than RCTGAN, but when looking at figure 7, we observe that Seq-RCTGAN effectively tries to capture autocorrelation, while the original RCTGAN doesn't.

  Regarding the average KS statistics (see figure 6, the lower the value, the better the model is at respecting the joint distribution between parent and children data. We can observe that the size of the sequence does affect significantly this metric. We can observe that the Seq-RCTGAN for a sequence length of 400 achieves better results than RCTGAN at 300 and 500.

- **Seq-MiniBatch exhibits a mixed performance.** While a Sequence length of 300 yields a high cross-correlation (0.415), its performance with other sequence lengths is less consistent and sometimes falls short of the baseline RCTGAN. This suggests a potential sensitivity to sequence length selection for this model. Also, this model

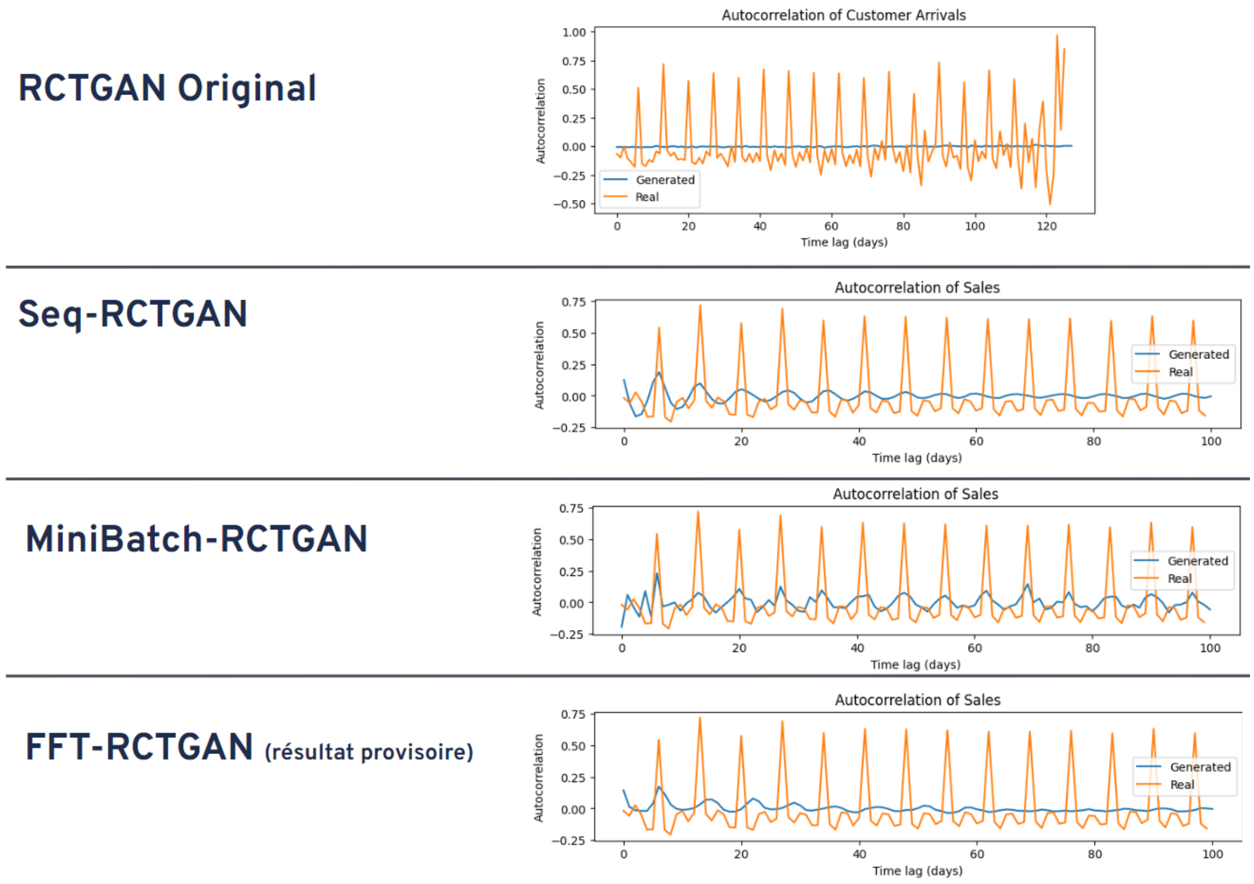

Figure 7 **Autocorrelation comparison**

seems to have difficulty in generating realistic children as we can observe in figure **??**. This difficulty might arise because the conditional vector is the same for each fully connected layer within a mini-batch. In terms of autocorrelation, notably, Seq-MiniBatch with a sequence length of 500 and sample len of 10 achieves the lowest MAE (0.151), suggesting it most accurately replicates the sales autocorrelation structure. We can also observe from the figure 7 that Seq-MiniBatch effectively tries to capture autocorrelations over a long period of time. Where at each peak up until more than 100 days, the synthetic autocorrelations try to follow those peaks. Regarding the ks statistic, the mini-batch performs the best at respecting the meta-measurement distributions of all models.

- **FFT-based models generally underperform compared to Seq-RCTGAN and the baseline RCTGAN.** It is important to note that these results are preliminary. Given the results gathered to this date, the FFT model shows cross-correlation values farther from the real average cross-correlations when compared to the other

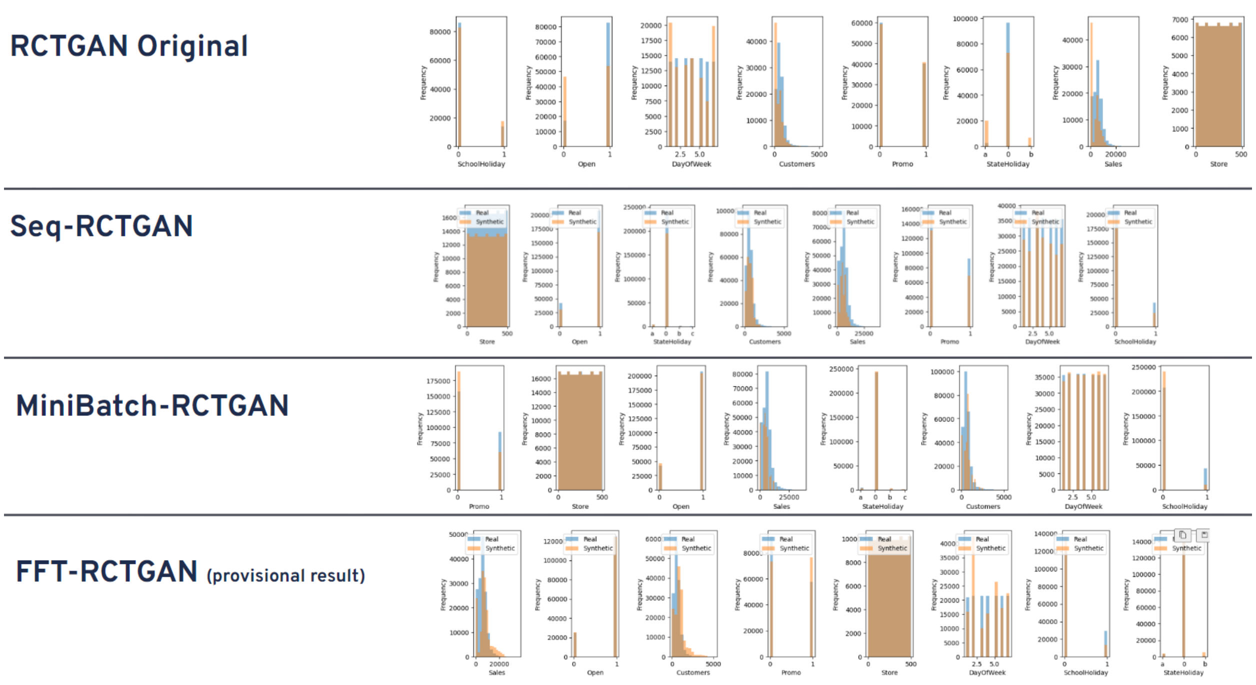

Figure 8 **Children distributions comparison**

model. In terms of autocorrelations, the FFT-based model seems to be performing very closely to the RCTGAN, but when looking at the figure 7, we can again observe that the model tries to capture some signal. Still, those values indicate a potential limitation in capturing the temporal dynamics of the data, despite incorporating frequency domain information. To finish, no promising results are observed for the meta-measurement distribution. But again, these results are not final, further experiments need to be tested regarding the size of the FFT vector, the number of relevant spectra, and the general hyperparameters.

Impact of Sequence length and Sample Length:

- **Sequence length significantly influences the performance of Seq-RCTGAN and Seq-MiniBatch.** Seq-RCTGAN shows a general trend of improved cross-correlation with increasing sequence length, while Seq-MiniBatch exhibits a less predictable pattern. This emphasizes the need for careful sequence length tuning for optimal performance.
- **Sample length (*Sample len*) Lack of experiment to establish the effect of it** Further investigation with varying sample lengths is necessary to draw definitive conclusions about its effect.

6. Conclusion

This study investigated the potential of incorporating frequency domain information, via Fast Fourier Transform (FFT), sequential learning, using lstm base architecture, and mini-batch generation for capturing the temporal dynamics in the data. While the preliminary results show promise in enhancing, in particular, auto-correlation within generated sequences, several challenges and avenues for future research emerged.

Including FFT vectors and sequential components introduced significant hyperparameter sensitivity, making model training and optimization complex. Further investigation into optimal network architectures, including GRU vs. LSTM, bidirectional layers, and varying FFT vector sizes, mini-batch size, size of epochs, seems to have a big effect. Additionally, computational constraints limited the exploration of a wider range of experiments, particularly regarding the size of the mini-batch sizes, the size of the FFT vectors and the length of the time series, which significantly impact performance.

While the proposed models hold the potential to preserve the quality of parent distributions, cross-correlation within children, improved joint parent-child distributions, and autocorrelations, the generation of realistic child distributions remains a challenge. Further analysis is needed to understand this limitation and explore potential solutions.

Despite these challenges, this work demonstrates the potential of integrating frequency domain information and sequential learning in time series generation within the RCTGAN framework.

References

Gueye M, Attabi Y, Dumas M (2022) Row conditional-tgan for generating synthetic relational databases. URL https://arxiv.org/abs/2211.07588.

Li X, Metsis V, Wang H, Ngu AHH (2022) Tts-gan: A transformer-based time-series generative adversarial network.

Lin Z, Jain A, Wang C, Fanti G, Sekar V (2020) Using gans for sharing networked time series data: Challenges, initial promise, and open questions. *Proceedings of the ACM Internet Measurement Conference*, IMC '20 (ACM), URL http://dx.doi.org/10.1145/3419394.3423643.

Srinivasan P, Knottenbelt WJ (2022) Time-series transformer generative adversarial networks.

Yoon J, Jarrett D, van der Schaar M (2019) Time-series generative adversarial networks. Wallach H, Larochelle H, Beygelzimer A, d'Alché-Buc F, Fox E, Garnett R, eds., *Advances in Neural Information Processing Systems*, volume 32 (Curran Associates, Inc.), URL https://proceedings.neurips.cc/paper_files/paper/2019/file/c9efe5f26cd17ba6216bbe2a7d26d490-Paper.pdf.