Decision-Making in the Face of Uncertainty: Harnessing GANs for Probabilistic Forecasting

Thesis approved by
the Department of Computer Science
University of Kaiserslautern-Landau
for the award of the Doctoral Degree
Doctor of Engineering (Dr.-Ing.)

to

Alireza Koochali

Date of Defense: 9 August 2024
Dean: Prof. Dr. Christoph Garth
Reviewer: Prof. Dr. Prof. h.c. Andreas Dengel
Reviewer: Prof. Dr. Koichi Kise

DE-386

I would like to dedicate this thesis to Zahra and Kian.

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation consists of 11 chapters and contains around 35,000 words including, bibliography, footnotes, tables and equations.

Alireza Koochali
9 August 2024

Acknowledgements

As I stand at the culmination of my doctoral journey, it is with a deep sense of gratitude that I pen these acknowledgments. This journey, filled with learning and discovery, would not have been possible without the support and encouragement of many. At the forefront of this journey has been my esteemed professor, Prof. Dr. Prof. h.c. Andreas Dengel. I am immensely grateful for his unwavering scientific guidance, which has been a beacon of light throughout my PhD. His relentless efforts in cultivating a conducive research environment and providing an excellent platform teeming with intellectual minds and state-of-the-art infrastructure have been instrumental in my growth as a researcher. I extend my heartfelt thanks to Prof. Dengel for his constant support, which not only fueled my ambition but also provided a stress-free atmosphere that was essential for my academic pursuits. His presence has been a source of inspiration, steering me through the challenges of research with remarkable confidence.

Continuing on this journey of gratitude, I must express my profound appreciation for my supervisor, Dr. Sheraz Ahmed. His presence and guidance have been a cornerstone of my doctoral voyage. Dr. Ahmed has been more than a supervisor; he has been a scientific companion and a mentor whose insights and advice have been invaluable. His brotherly guidance has touched every aspect of my work, providing clarity and direction when I needed it the most. His open-door policy meant that I always had a place to turn to, whether for academic discussions or personal guidance. His unwavering support and belief in my abilities have played a pivotal role in my ability to navigate and complete this challenging yet rewarding journey. His contribution to my success cannot be overstated, and for that, I am eternally grateful.

I extend my deepest gratitude to the members of the FLaP lab, who have been instrumental in the development of my research. Special thanks go to Ludger van Elst and Dr. Peter Schichtel, whose support and enlightening scientific discussions were pivotal in shaping my ideas and transforming them into tangible results. Their insights and feedback have significantly contributed to the depth and breadth of my research, providing me with the tools

and knowledge necessary to navigate the complexities of my field.

I would like to express my heartfelt appreciation to all the students and colleagues at the FLaP lab and DFKI. Collaborating and discussing various topics with them has been an enriching experience, offering diverse perspectives and enhancing the research environment. The opportunity to work alongside such talented individuals and to share in their unique insights has been a privilege, and I am profoundly thankful for their contributions to my academic journey.

I must express my sincere gratitude to IAV GmbH for their generous funding and support of my PhD journey. Their financial backing created a stable and secure environment, allowing me to concentrate fully on my research and scientific endeavors. This support has been invaluable and greatly appreciated. Moreover, I extend my thanks to my colleagues at IAV. The opportunity to engage in scientific discussions with them has been both enlightening and stimulating. Their perspectives and insights have contributed to broadening my understanding and enhancing my approach to research. The collaborative and intellectually rich environment at IAV has been a significant factor in my academic growth and achievements.

Finally, and most importantly, I extend my deepest gratitude to my family. To my parents, brother, and sister, whose unwavering emotional support was my guiding light through this challenging journey. Their love, encouragement, and belief in my abilities have been the pillars of my strength and resilience. I reserve a special thanks to my wife, Zahra Zavari, who has been my steadfast partner in every sense. Sharing in every hardship and triumph, her support, understanding, and love have been the cornerstones of my success. Zahra, your presence in my life has been a source of continuous inspiration and motivation. And to my newborn son, Kian, although your arrival into this world is recent, your presence has brought immense joy and an unparalleled strength that empowered me to cross the finish line of this PhD journey. Your arrival has been a beautiful reminder of the wonders and possibilities that lie ahead. I look forward to the journey we have yet to embark on together as a family.

Abstract

Accurately anticipating future events is essential in decision-making domains such as healthcare, finance, and security, where managing risk is crucial. Probabilistic forecasting is key to understanding potential risks and their impacts. This thesis advances probabilistic forecasting through three primary contributions: (i) innovative forecasting models using Generative Adversarial Networks (GANs), (ii) GAN-based methods for time series data augmentation to address data scarcity, and (iii) a new, unbiased framework for evaluating generative models in time series domain.

Key contributions include the introduction of ForGAN, a GAN-based model for probabilistic forecasting, VAEneu, a Variational Auto-Encoder-based probabilistic forecaster, and CRPS Loss, a loss function optimized for these models. CRPS-ForGAN, which integrates CRPS Loss with ForGAN, consistently outperforms baseline models in one-step ahead forecasting tasks across multiple datasets. For multi-step forecasting, auto-regression and attention-based seq2seq models are introduced, with proposed models leading in performance in an extensive experimental setting. These models also generate forecasts efficiently, suitable for real-time applications.

To tackle data scarcity, the thesis proposes a GAN-based model for time series data augmentation, particularly for rare extreme events, improving forecasting accuracy by 8.6% in real-world wastewater management applications. Additionally, the Structured Noise Space GAN (SNS-GAN) is presented for class conditional time series generation, achieving up to a 64% improvement in synthetic data quality over baseline models in time series data generation study.

Finally, the thesis introduces the Fréchet Inception Time Distance (FITD) and Inception-Time Score (ITS) as novel metrics for evaluating generative models in time series. Extensive experimentation across diverse datasets confirms their effectiveness, setting a new standard for model evaluation in this field.

Table of contents

List of figures

List of tables

Part I

Preamble

Chapter 1

Introduction

As we continue to navigate the complex landscape of life, it becomes increasingly evident that our decision-making processes pivot on our ability to foresee and understand the future. The knowledge of the future illuminates our choices today, enabling us to maneuver the labyrinth of life with conviction and foresight. Despite its inherent uncertainty and ambiguity, the quest for the pursuit of knowledge and the desire to discern the patterns of the future has been a fundamental human endeavor throughout history. As our ancestors sought foresight in divination and prophecy, we, too, crave that same certainty but with the powerful tools of the contemporary era. *Forecasting*, a methodology that combines historical data, statistical algorithms, and machine learning techniques, enables businesses, governments, and individuals to make projections about future events and trends. Forecasting models reveal potential outcomes and their associated probabilities to decision-makers, empowering them to assess trade-offs, allocate resources efficiently, and formulate policies that are reactive and proactive.

Forecasting models have ubiquitous integration in decision-making processes across different spheres of life, from daily life decisions to strategic government and business plans. For instance, weather forecasting aids us in planning our activities for the day, from deciding what to wear to choosing the best time for outdoor events. Traffic forecasting also plays a significant role in our daily commute, assisting in determining the quickest routes and the optimal time to travel, thereby minimizing delays and increasing efficiency. In the realm of business, forecasting is employed to anticipate future sales, identify emerging trends, and plan for production and inventory needs. Governments use forecasting to project revenue and expenses, which helps them create and manage budgets. Accurate weather forecasting is essential for emergency preparedness, which results in saving lives and preventing property damage. In healthcare, organizations employ forecasting to estimate future demand for healthcare services, patient volumes, bed occupancy rates, and staffing needs. They use

this information to manage resources more effectively and efficiently and plan for capacity expansion or contraction, ensuring that they can meet the needs of their patients. Furthermore, the forecast model can predict disease outbreaks, which helps healthcare organizations respond to potential epidemics or pandemics in time and decide on public health interventions, such as vaccination campaigns or quarantine measures. Forecasting models also pave the way for personalized medicine. With forecasts on disease risk in individuals based on their genetics, lifestyle, and environmental factors, healthcare providers can develop personalized prevention and treatment plans, improving patient outcomes. In the automotive industry, forecasting enables predictive maintenance by predicting when a component will fail or require maintenance. Furthermore, by monitoring and forecasting the engine load, fuel consumption, and other factors that impact fuel efficiency, we can optimize the vehicle's performance, resulting in better fuel economy. Also, autonomous driving relies heavily upon the forecast of other vehicles' and pedestrians' trajectories to make decisions about acceleration, braking, and steering.

The importance of these forecasts lies in their ability to model uncertainty about the future in the form of predictive distribution, providing a basis for strategic planning and risk management. Despite the extensive research on the forecasting task, the industry is dominated by deterministic models, which offer a constrained view of potential future occurrences and are incapable of specifying the uncertainty tied to the forecast. Furthermore, the current prominent probabilistic models either provide a property of the predictive distribution, such as quantiles or prediction intervals, or rely upon assumptions about the type and form of the predictive distribution. This dissertation introduces a broad range of novel methodologies to leverage the state-of-the-art generative models to architect potent probabilistic forecasters (ForGAN, VAEneu, and ProbCast), which are capable of modeling the predictive distribution without relying on any extra information and assumptions. Moreover, this thesis suggests various methods (FITD and ITS) for the discriminative assessment of probabilistic models in the time series domain. Furthermore, it proposes a novel approach to harness the power of GANs (SNS-GAN), constructing a generative model for time series data. This method effectively mitigates the data scarcity challenge, enabling the training of probabilistic forecasters to their fullest potential.

1.1 Motivation and Problem Statement

This dissertation ambitiously aims to elevate the decision-making process and risk assessment by refining the probabilistic forecasting pipeline, a crucial endeavor focused on capturing the

inherent uncertainty of future events. The inherent unpredictability of the future stands as a primary source of this uncertainty, yet it's further compounded by factors such as suboptimal model selection and the prevalent issue of data scarcity. Given that the core objective of probabilistic forecasting lies in accurately reflecting the uncertainty inherent in data, it becomes imperative to mitigate the impact of these additional sources of uncertainty. In pursuit of this goal, this thesis sets forth two pivotal objectives: the development of precise and reliable probabilistic forecasting models and the effective reduction of data scarcity.

The modeling of the predictive distribution for probabilistic forecasting is a challenging task. Given a historical window, a probabilistic forecaster must encapsulate the uncertainty of forthcoming time steps while concurrently modeling the temporal dependencies between these steps. This task involves handling a high-dimensional conditional probability distribution with a complex dependency structure. Current dominant approaches are either heavily reliant on expert knowledge or restrictively designed to model specific properties of the predictive distribution. However, decision-makers need a comprehensive understanding of future events for accurate risk assessment and optimal decision-making. With the complex predictive distribution scenarios encountered in real-world situations, expert knowledge availability is limited, underlining the importance of developing a generic framework that reduces reliance on such expertise.

Moreover, the complexity of the predictive distribution makes the assessment of probabilistic forecasters a daunting task. An effective evaluation metric must accurately quantify the alignment between the predictive and real uncertainty distributions. The failure of an evaluation method can lead to misinterpretation of the probabilistic forecaster's performance, leading to sub-optimal model selection, poor decision-making, and potential financial or human life losses in sensitive domains.

Furthermore, while the era of information technology provides abundant data, time series data's nature—requiring consistent recording over time—perpetuates the issue of data scarcity. We cannot expedite data collection for time series data with a long resolution period (quarterly, yearly, etc.). As abundant data forms the backbone of modern machine learning methods' performance, data scarcity impedes these models' full potential. Therefore, generative models for time series data, designed to mitigate data scarcity, gain paramount importance.

Similarly to probabilistic forecasting, evaluating generative models entails the assessment of

high-dimensional probability distribution with intricate temporal dependencies. Unlike probabilistic forecasting models, applying generative models to time series data is a novel area of scientific exploration, and a standard assessment method does not exist. Manual inspection of generated sample quality is unfeasible due to the non-intuitive nature of time series data. Thus, a mathematically sound evaluation method is critical to facilitating generative models' application on time series data.

1.2 Scientific Questions and Contributions

The primary contribution of this thesis is developing and implementing a generic framework for constructing a probabilistic forecaster grounded in Generative Adversarial Networks (GANs). The universality of this framework allows its application to various time series data across diverse domains and dynamics, requiring minimal adjustments for specific forecasting tasks. The thesis aims to answer 4 key questions, each of which is an integral challenge in creating an effective probabilistic forecaster. The scientific questions and contributions are as follows:

Question 1: Do currently available assessment methods for probabilistic forecasters accurately measure model performance?
Understanding the accuracy of assessment methods is paramount in the realm of probabilistic forecasting. Accurate measurement tools are essential for the development and validation of new forecasting models, ensuring that they are evaluated based on their true predictive capabilities.

Contributions:

1. A thorough analysis of the discrimination capabilities of established evaluation metrics such as Continuous Ranked Probability Score (CRPS), CRPS-Sum, and Energy score, both from theoretical and empirical perspectives.

2. Identification of critical limitations in these conventional evaluation metrics, providing insights into their shortcomings and potential for misinterpretation.

3. A review of the application of these metrics in contemporary research, illustrating instances where these flaws have led to inaccuracies in interpreting model performance.

4. Essential guidance for interpreting results derived from these evaluation methods, contributing to more accurate and reliable assessment practices in probabilistic forecasting.

Question 2: How can we effectively quantify the performance of generative models in the time series domain?
Developing an effective means of evaluating generative models in the time series domain is crucial due to the unique challenges presented by time series data. Efficient evaluation techniques are essential for advancing the use of generative models in the time series domain, where data collection is inherently time-consuming and challenging.

Contributions:

1. Introduction of the InceptionTime Score (ITS) and the Fréchet InceptionTime Distance (FITD): These novel evaluation metrics, inspired by methods used in the image domain, offer a comprehensive framework for assessing the quality of generative models specifically tailored to time series data.

2. Comprehensive Testing and Validation: The ITS and FITD metrics were thoroughly tested across an extensive collection of 80 datasets, proving their effectiveness in outperforming existing metrics and adeptly detecting common issues prevalent in generative models.

3. Integration with Existing Metrics: The thesis demonstrates that when combined with the existing Train on Synthetic Test on Real (TSTR) metric, the newly proposed ITS and FITD provide a more robust and precise toolkit for performance assessment, thereby enhancing the reliability of evaluations.

4. Formulation of Interpretive Principles: This dissertation offers a set of principles designed to assist both researchers and practitioners in comprehending the subtleties of the ITS and FITD scores. This contributes to a more profound understanding of the strengths and weaknesses of generative models in the context of the time series domain.

Question 3: How can we design a probabilistic forecasting model for univariate time series data that minimizes model uncertainty while facilitating direct interaction with the predictive distribution?
Addressing the complexity of minimizing model uncertainty in probabilistic forecasting is pivotal for developing advanced forecasting models. These models must be capable of comprehensively capturing the uncertainties associated with future events without adhering to strict assumptions about the predictive distribution. Furthermore, enabling an interactive approach with predictive distribution is essential for a deeper understanding and utilization of these models.

Contributions:

1. The development of ForGAN and VAEneu: These state-of-the-art probabilistic forecasting models leverage the power of implicit generative modeling to learn and interpret complex, high-dimensional probability distributions and their dependencies. Furthermore, these models facilitate a nuanced interaction with the predictive distribution, allowing for a more intuitive and insightful understanding of their forecasting outputs.

2. Application to one-step-ahead univariate time series forecasting: Tested across 13 datasets and benchmarked against the well-established DeepAR model, ForGAN and VAEneu demonstrated unmatched performance superiority in 12 of these datasets, with VAEneu particularly standing out.

3. Expansion to multi-step-ahead forecasting for univariate time series: This involved the introduction of Auto-Regressive and Attention-based variants of the proposed models. In comparative studies with 11 baseline models across 12 datasets, the Auto-Regressive ForGAN variants consistently showcased exceptional forecasting accuracy.

Question 4: How can we effectively create probabilistic forecasting models for complex multivariate time series data while maintaining manageable model complexity?
Multivariate time series forecasting presents a more intricate challenge compared to univariate forecasting, necessitating models that can adeptly manage not only the temporal dependencies but also the intricate interrelationships between various data channels. This complexity demands a strategic approach to model development that balances advanced forecasting capabilities with manageable complexity in model architecture.

Contributions:

1. Introduction of ProbCast, a transformative framework that adeptly extends the GAN-based modeling paradigm to the multivariate time series domain. ProbCast minimizes the added complexity by transitioning a deterministic forecaster to a probabilistic one, ensuring efficient model design.

2. Comprehensive analysis and validation of ProbCast using 2 real-world datasets, demonstrating that ProbCast could not only successfully perform this transformation but also enhance the accuracy of the resulting probabilistic models

Question 5: How can generative models be tailored to effectively fabricate artificial time series data emphasizing on rare extreme patterns?

In time series analysis, extreme patterns, though infrequent, play a pivotal role in various critical applications. Overcoming the challenges posed by data scarcity, especially in representing these patterns, is vital for enhancing the effectiveness of downstream tasks such as forecasting.

Contributions:

1. Introduction of a GAN-based generative model, specifically designed to focus on the generation of time series data that enhance the representation of extreme events.

2. Empirical validation of the model's effectiveness in a real-world dataset, showcasing its ability to address the extreme pattern scarcity issue and enhance the predictive accuracy of a downstream forecasting model.

Question 6: What are the effective strategies for integrating discrete auxiliary information like labels into the time series data generation process?
Class conditional generative models have revolutionized machine learning by allowing data generation based on specific classes. This functionality is particularly advantageous in focused dataset augmentation and enhancing data representation for select categories. However, integrating such discrete elements, notably class labels, into the time series data generation process has remained a challenge due to the limitations of existing methods.

Contributions:

1. Development of SNS-GAN: This thesis introduces the Structured Noise Space GAN (SNS-GAN), a novel model that seamlessly integrates class labels into the time series data generation process. This advancement significantly enhances the model's ability to produce context-specific and high-fidelity samples.

2. Efficacy Validation in the Image Domain: The effectiveness of SNS-GAN, particularly its label integration approach, was rigorously validated using 2 image datasets. The outcomes affirm the model's capacity to generate samples that accurately reflect specified input conditions.

3. Extension to Time Series Generation: Extensive testing of SNS-GAN with 4 time series datasets further validated its superior performance. The successful application in this domain underscores its versatility and marks a significant contribution to the field of time series data generation.

1.3 Dissertation Overview

This dissertation is organized into eleven chapters, structured across four main parts, each offering key contributions to the domain of probabilistic forecasting and generative models.

The 1^{st} part serves as an introductory section. Chapter 2 reviews essential topics related to generative models and probabilistic forecasting, which are necessary for comprehending the content of the subsequent chapters.

The 2^{nd} part of the dissertation provides multiple contributions to probabilistic forecasting. Chapter 3 performs a comprehensive literature review on probabilistic forecasting tasks. Chapter 4 conducts a detailed examination of three conventional assessment methods for probabilistic forecasting. Chapter 5 presents the fundamentals of the proposed ForGAN and VAEneu pipeline and explores their application on univariate one-step-ahead forecasting tasks. This chapter also reviews a practical application of ForGAN in the automotive industry. Chapter 6 is devoted to multi-step-ahead probabilistic forecasting on univariate time series data. It introduced two methods, namely auto-regression, and attention, to extend the ForGAN and VAEneu to multi-step-ahead forecasting. This chapter presents the results of an extensive study of these models using 12 univariate datasets and 11 established baselines. Chapter 7 introduces ProbCast for multivariate time series through a novel training pipeline for efficient development of the multivariate model.

The 3^{rd} part of the dissertation reports on efforts to augment time series data to mitigate the impact of data scarcity on the performance of probabilistic forecasters. Chapter 8 introduces two novel assessment methods for generative models on time series data: the InceptionTime Score (ITS) and the Fréchet InceptionTime Distance (FITD). An extensive study of these methods, in conjunction with two other assessment methods, is conducted on 80 different datasets. Chapter 9 proposes a GAN-based generative model for augmenting rare extreme patterns in time series data, demonstrating its effectiveness in a real-world scenario in the water management domain. Chapter 10 presents the Structured Noise Space GAN (SNS-GAN), a novel method for effectively incorporating discrete additional information, such as labels, into the generation process. The effectiveness of this method is examined qualitatively in the image domain, with the results presented in this chapter. The chapter concludes with the results of applying this method to time series data.

The 4^{th} and final part of the dissertation is dedicated to concluding and summarizing the

findings of this research. Chapter 11 highlights the main contributions of the dissertation and outlines potential directions for further research in this domain.

Chapter 2

Foundations

This chapter is dedicated to establishing a comprehensive understanding of the fundamental concepts and methodologies that underpin the innovative advancements presented in this dissertation. It serves as a crucial groundwork, paving the way for a deeper appreciation and critical analysis of the novel approaches and models introduced in the subsequent chapters.

2.1 Generative Models Foundation

This section focuses on the critical exploration of generative models, particularly emphasizing their role as a powerful tool for learning the underlying probability distribution of a given dataset. The capacity of these models to comprehend, learn, and simulate the structure of high-dimensional data distributions underpins their significance in the realm of machine learning. With their diverse applications across various fields, from computer vision to natural language processing, and their prominence in tasks such as anomaly detection, data augmentation, and, most relevant to this thesis, probabilistic forecasting, generative models have become indispensable. Through this section, we aim to delve into the details of these models, provide comprehensive insights into their working mechanisms, and illuminate the mathematical foundations that govern their operation.

2.1.1 The Art of Fabrication

Imagine an aspiring artist keen on mastering the distinctive style of Vincent Van Gogh. The initial endeavor would entail curating a collection of Van Gogh's illustrious works for in-depth analysis. Although each piece might portray diverse subjects and narratives, an intrinsic stylistic consistency sets Van Gogh's works apart from other artists. By immersing themselves in this collection and relying upon their artistic prowess and training, the artist

might eventually craft pieces that resonate with Van Gogh's signature style.

This complicated process mirrors the essence of generative models in machine learning. Generative models are a class of machine learning models dedicated to synthesizing novel data samples similar to those in a provided training set [1]. Analogous to the ubiquitous style seen across an artist's portfolio, the samples within a training set are generated by an inherent generative process, thereby adhering to a shared distribution. Therefore, the core of generative modeling lies in discerning and replicating this intrinsic data distribution.

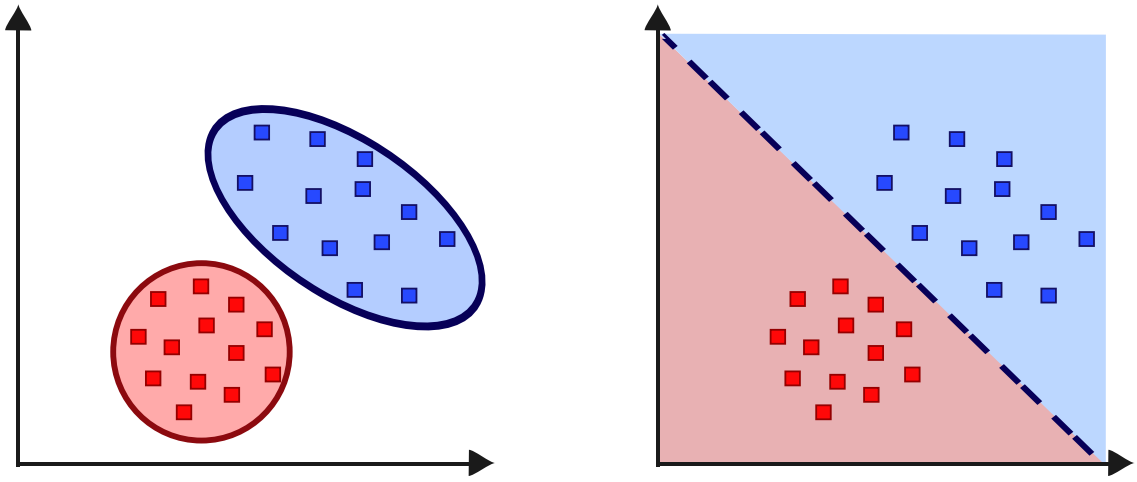
In more rigorous terms, given a dataset X comprising N independent and identically distributed samples x, drawn from a data distribution denoted as p_{data} or $p(X)$, a generative model, represented as G, tries to approximate this data distribution. Throughout this dissertation, synthesized samples from the generative model are symbolized as $\hat{x}$. In contrast, the distribution learned by the generative model is designated as p_{model} or $p(\hat{X})$.

Generative vs Discriminative modeling

To better understand the Generative models, it is helpful to compare them with the other school of machine learning models: discriminative modeling. Typically, datasets tailored for discriminative modeling encompass paired data points x and corresponding target labels y. The core objective of discriminative models is to define a function that maps each data point to its target label. One of the primary applications of these models pertains to classification tasks. Within this context, the model focused on modeling the decision boundary between classes without concerning itself with underlying data distribution (see figure 2.1). To put it more concretely, discriminative modeling endeavors to model $p(y|x)$, signifying the likelihood of a particular label y given an observation x. In contrast, generative modeling aims to model underlying data distribution $p(x)$. In essence, discriminative models offer a direct approach to decision-making tasks in machine learning by focusing on the relationship between inputs and outputs without delving into the details of how data from different classes is generated. Conversely, generative models are not concentrated on labeling observations but rather on approximating the likelihood of encountering a particular observation.

2.1.2 The Principle of Maximum Likelihood Estimation

Given a generative model parameterized by a set of parameters θ, the likelihood function, denoted as $\mathscr{L}(\theta|X)$, quantifies the capability of the model, equipped with such parameters



(a) Generative model (b) Discriminative model

Fig. 2.1 This Figure highlights the difference between generative modeling and discriminative modeling. The generative model **(a)** focuses on learning data distribution, while the discriminative model **(b)** aims to classify data points.

to explain the observations within the training set X. Mathematically, the likelihood function can be defined as:

$$\mathscr{L}(\theta \mid X) = \prod_{i=1}^{N} p_{\text{model}}\left(x^{(i)}; \theta\right). \tag{2.1}$$

The principle of maximum likelihood estimation (MLE) prescribes the selection of model parameters that maximize the likelihood of the given training data. For computational expediency and to circumvent numerical underflow issues, this optimization is often performed in the logarithmic domain, transforming the product of probabilities into a summation of log probabilities:

$$\theta^* = \arg\max_{\theta} \sum_{i=1}^{N} \log p_{\text{model}}\left(x^{(i)}; \theta\right). \tag{2.2}$$

Conceptually, the MLE procedure can be construed as minimizing the divergence between the true data distribution p_{data} and the model's estimated distribution, p_{model}. Under ideal circumstances, if the chosen model family for p_{model} aligns perfectly with the intrinsic distribution family of p_{data}, an exact recovery of p_{data} is possible. However, direct access to p_{data} is typically infeasible in real-world scenarios. Consequently, generative modeling hinges on devising an accurate modeling strategy that provides the most precise estimation of p_{data}, harnessing solely its available samples. Broadly, generative models grounded in

the MLE paradigm can be divided into two main categories: explicit generative models and implicit generative models. Subsequent sections outline a detailed explanation of these methodologies, supplemented with exemplar models that epitomize each category [2].

2.1.3 Explicit Generative Models

Explicit generative models specify a parametric form for the data distribution, denoted as $p(x;\theta)$, where θ represents the parameters of the distribution. These models are characterized by their ability to provide an explicit functional form for the likelihood, facilitating direct optimization through gradient-based methods.

However, designing effective explicit generative models poses challenges. A primary concern is achieving a balance between model expressiveness and computational efficiency. The model must be sufficiently flexible to capture the intricate data structures and variations while ensuring that likelihood evaluations and gradient computations remain tractable [2].

To address these challenges, researchers typically adopt one of two strategies:

1. **Tractable explicit models:** This involves crafting models with inherent structures that guarantee computational tractability. Such architectures ensure that evaluations of the likelihood and its gradients can be efficiently computed without compromising the model's ability to fit complex data distributions. DeepAR [3] and Wavenet [4] are two prominent autoregressive models which employ tractable explicit modeling for probabilistic forecasting. These models are discussed in detail in section 3.5.1 and 3.5.8 respectively.

2. **Tractable Approximations:** In scenarios where the exact likelihood computation is infeasible, models are designed to provide tractable approximations to both the likelihood and its gradient. This approach often relies on variational techniques or other approximation methods to ensure that optimization remains feasible. Variational auto-encoder is one the most prominent generative models in this category. An extended explanation of this model is provided in section 2.1.7, and a probabilistic forecaster based on this methodology is suggested in section 5.2.

2.1.4 Implicit Generative Models

Implicit generative models have gained significant attention due to their unique approach to modeling data distributions. Unlike explicit generative models, which define a parametric

form for the density function, implicit generative models take a more indirect route. Specifically, they forgo the explicit representation of the data distribution in favor of a mechanism that allows for sampling from the desired distribution, p_{model} [2].

The fundamental distinction lies in the accessibility of the probability density function. While explicit models provide direct access to p_{model} and its derivatives, implicit models circumvent this requirement by offering a stochastic procedure, often parameterized by neural networks or other nonlinear functions, which can generate samples from p_{model} [1]. This inherent flexibility enables implicit models to capture intricate data distributions without the need for restrictive assumptions on the form of the distribution. The Generative Adversarial Network (GAN) [5] is an eminent example of implicit generative models. A comprehensive exploration of GANs, their architectural nuances, and training dynamics is presented in the following section.

2.1.5 Generative Adversarial Networks (GANs)

Introduced by Goodfellow et al. in 2014 [5], the Generative Adversarial Network (GAN) has rapidly emerged as a groundbreaking approach in the domain of implicit generative modeling. GANs have demonstrated unparalleled proficiency in synthesizing high-fidelity artificial data, spanning various fields ranging from image generation to natural language processing.

Network Architecture

The architecture of a GAN comprises two interdependent neural networks: the Generator (G) and the Discriminator (D). These networks engage in a competitive training paradigm:

- **Generator:** Tasked with generating synthetic data, the generator, denoted as $G(z, \theta_G)$, is a differentiable function parameterized by θ_G. It aims to map samples z from a prior distribution (often termed the noise distribution) to samples that mimic the true data distribution. The choice of the prior distribution is typically a standard Gaussian:

$$\hat{x} = G(z, \theta_G), \quad z \sim \mathcal{N}(0, 1). \tag{2.3}$$

- **Discriminator:** Functioning as a binary classifier, the discriminator, $D(x, \theta_D)$, is parameterized by θ_D. Its primary objective is to discern the authenticity of samples, i.e., whether they originate from the training set or are the product of the generator G. For any given sample x, $D(x, \theta_D)$ yields a scalar with the probability that x is drawn from the genuine data distribution.

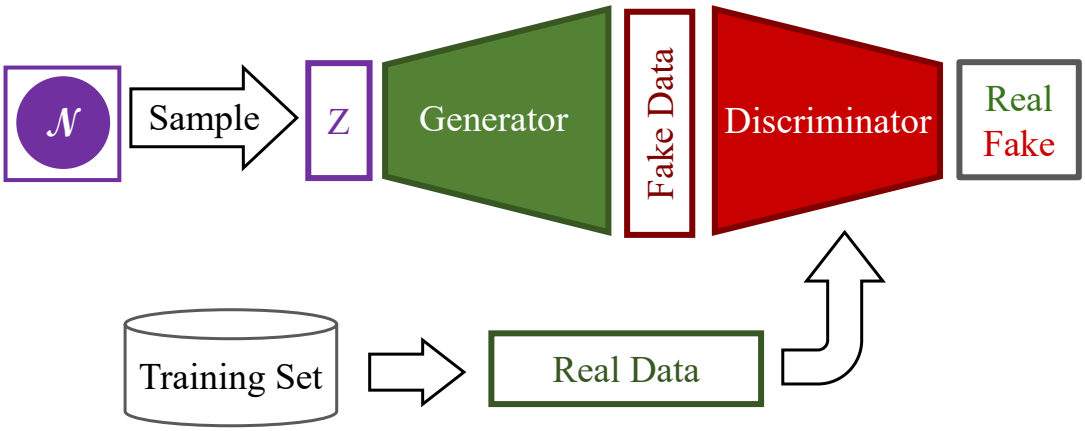


Fig. 2.2 A general illustration of GAN pipeline

A schematic representation of the GAN architecture is provided in Figure 2.2.

Training Paradigm

The training dynamics of a GAN are fundamentally adversarial, embodying a two-player minimax game between the discriminator D and the generator G. Alternating phases of training characterize this adversarial interplay:

1. The discriminator D trains to optimally categorize samples as real (from the training set) or fake (emanating from the generator G).
2. Subsequently, the generator G refines its generation mechanism to enhance the authenticity of its outputs, effectively aiming to deceive the discriminator into mistaking its outputs for genuine data samples.

Mathematically, the following objective function is optimized during training:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim P_{\text{data}}(x)}[\log(D(x))] + \mathbb{E}_{z\sim P_{\text{noise}}(z)}[\log(1-D(G(z)))]. \tag{2.4}$$

In the original proposal of GAN [5], the authors showed that the optimization of this objective function is equivalent to minimizing the Jensen Shannon divergence between p_{data} and p_{model}. During this optimization, while the discriminator D tries to maximize $V(D,G)$ for accurate categorization, the generator G aims to minimize $V(D,G)$ to generate increasingly convincing samples. Ideally, given adequate capacity and under specific conditions, this iterative process converges to a Nash equilibrium. At this equilibrium, the generator's outputs, $G(z)$, adhere

to the true data distribution, and the discriminator's classification probability, $D(x) = \frac{1}{2}$ for any input x, indicating its inability to distinguish real data from generated samples.

Non-saturated GAN

The training dynamics of GANs involve an intriguing but challenging interplay between the generator and discriminator. In the minimax formulation of GANs, while the discriminator aims to minimize a cross-entropy loss function, the generator tries to maximize it. This interplay, however, presents a substantial caveat: if the discriminator confidently rejects the generated samples, the gradient of the generator's loss with respect to its parameters approaches zero. This phenomenon, often called "gradient vanishing," hampers the generator's capacity to improve, leading to a slow or failed training process.

A non-saturated variant of the GAN's objective is proposed to overcome this gradient vanishing challenge. Instead of prompting the generator to maximize the probability of the discriminator committing an error, the non-saturated objective encourages the generator to maximize the log probability of the discriminator being deceived. This subtle yet impactful alteration results in gradients that are more conducive for training the generator, especially during the early stages when the discriminator might become too dominant. Mathematically, the generator's objective function in a non-saturated GAN is formulated as:

$$-\frac{1}{2}\mathbb{E}_{z\sim P_{\text{noise}}(z)} \log D(G(z)). \tag{2.5}$$

This alternative objective has been found to provide a smoother and more stable gradient landscape for the generator, promote a balanced evolution of both networks during the training process, and mitigate the challenges of the original minimax formulation.

Wasserstein GAN with Gradient Penalty (WGAN-GP)

The original GAN proposal, while groundbreaking in its ability to generate synthetic data, is often plagued with challenges related to training stability and mode collapse. A key contributing factor is the use of Jensen-Shannon divergence in the standard GAN framework, which can lead to vanishing gradients during training. To address this, Wasserstein GAN with Gradient Penalty (WGAN-GP) [6] proposes a paradigm shift in the divergence metric used to quantify the distance between the real and generated distributions.

In vanilla GAN, the discriminator estimates the probability of its input being real, leading to an output range between $[0, 1]$. This limited range potentially restricts the discriminator's

expressiveness regarding the authenticity of its input and can hinder convergence. The WGAN-GP approach circumvents this by employing the Earth-Mover (or Wasserstein-1) distance as its divergence metric:

$$W(p_{data}, p_{model}) = \inf_{\gamma \in \Pi(p_{data}, p_{model})} \mathbb{E}_{(x,y)\sim\gamma}[|x-y|]. \tag{2.6}$$

In this context, $\Pi(p_{data}, p_{model})$ represents the set of all coupling distributions $\gamma(x, \hat{x})$ such that x and $\hat{x}$ have the marginals p_{data} and p_{model} respectively. Intuitively, the Wasserstein distance measures the least amount of "work" required to transform one distribution into another.

Under mild conditions, the continuity and differentiability properties of the Wasserstein distance alleviate the gradient vanishing problem associated with original GANs. Leveraging the Kantorovich-Rubinstein duality [7], WGAN-GP's value function is indicated as:

$$\min_G \max_C \mathop{\mathbb{E}}_{x\sim p_{data}}[C(x)] - \mathop{\mathbb{E}}_{\hat{x}\sim p_{model}}[C(\hat{x})]. \tag{2.7}$$

An important distinction in WGAN-GP is the replacement of the discriminator with a Critic, denoted as C. The Critic is a 1-Lipschitz function that assigns an unbounded score indicating the authenticity of its input instead of outputting probabilities. WGAN-GP introduces a penalty on its gradient's norm to guarantee the Critic function's Lipschitz continuity. The final objective function for the WGAN-GP can thus be expressed as:

$$L = \mathop{\mathbb{E}}_{x\sim p_{data}}[C(x)] - \mathop{\mathbb{E}}_{\hat{x}\sim p_{model}}[C(\hat{x})] + \lambda \mathop{\mathbb{E}}_{\tilde{x}\sim p_{\tilde{x}}}[(\|\nabla_{\tilde{x}} C(\tilde{x})\|_2 - 1)^2] \tag{2.8}$$

Here, λ signifies the penalty coefficient. The distribution $p_{\tilde{x}}$ is derived by sampling uniformly along straight lines between paired samples from p_{data} and p_{model}. This gradient penalty is a regularizer, ensuring smoother training dynamics and better convergence properties.

Least Squared GAN (LSGAN)

The Least Squared GAN (LSGAN) [8] has been introduced to alleviate the vanishing gradient problem of vanilla GAN. Central to its approach is replacing the conventional binary cross-entropy loss function utilized in standard GANs with the least squares loss.The objective functions for the discriminator, denoted D, and the generator, denoted G, in the LSGAN framework are defined as:

$$\begin{aligned} \mathscr{L}_D &= 0.5 \times \mathbb{E}[(D(x)-1)^2] + 0.5 \times \mathbb{E}[D(G(z))^2], \\ \mathscr{L}_G &= 0.5 \times \mathbb{E}[(D(G(z))-1)^2] \end{aligned} \tag{2.9}$$

Instead of employing cross-entropy, which can worsen the vanishing gradient issue, the LSGAN paradigm compels its components to minimize the squared distances to target label values, specifically 0 for synthetic samples and 1 for authentic samples. One notable benefit of this quadratic form is its penalizing nature towards samples considerably distant from the discriminator's decision boundary, which, in effect, ensures mitigation of the vanishing gradient problem.

The authors of LSGAN illustrated that this model implicitly minimizes the Pearson χ^2 divergence between the true data distribution, p_{data}, and the model's data distribution, p_{model}. Considering the intrinsic smoother characteristics of Pearson χ^2 divergence compared to the Jensen-Shannon divergence, LSGANs present an optimization landscape characterized by its continuity. This is conjectured to produce more consistent gradient information during the training process, which, in turn, fosters stable convergence.

An advantageous distinction of LSGAN compared to WGAN-GP lies in its simplicity. Specifically, it obviates the necessity for auxiliary hyperparameters during the optimization phase and sidesteps the computational overhead associated with gradient penalty, streamlining the training procedure and potentially enhancing the model's scalability and efficiency.

2.1.6 Conditional GAN (cGAN)

Conditional Generative Adversarial Networks (cGANs) [9] extend the basic idea of GANs by introducing conditioning information into both the generator and discriminator, allowing the generation process to be guided by some auxiliary information. In cGANs, the generator creates outputs based on random noise and some conditioning data. The discriminator is then given both the data (real or generated) and the conditioning data to make its discrimination. The discrimination should consider both the authenticity of the input and its consistency with the condition. In this setting, the generator would model the conditional distribution of data, i.e., $p(X|c)$ instead of data distribution $p(X)$.

A distinguishing attribute of the cGAN framework is that any data type can be employed as auxiliary conditioning information. Such versatility amplifies the scope of its application in

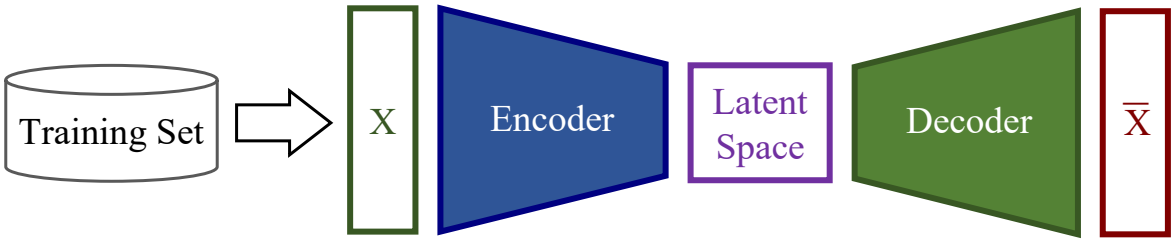


Fig. 2.3 Schematic of Autoencoder structure

conditional data modeling, catalyzing potential innovations in GAN-based solutions. However, this non-restrictiveness, while being a strength, concurrently introduces complexities. The absence of explicit guidelines concerning integrating conditioning data into the generator and discriminator necessitates rigorous empirical investigations to determine an efficacious and effective methodology for such integration, especially given the diverse typologies of conditioning data.

2.1.7 Generative Autoencoder-based Models

The landscape of generative modeling has been enriched by incorporating autoencoder structures. To grasp the nuances of generative autoencoder-based models, one must first understand the underpinning principles of autoencoders (AEs).

Autoencoders, as presented by Hinton et al. [10], are unsupervised neural network structures architected to establish compact representations of input data. This compression is often leveraged for dimensionality reduction; however, its utility extends to tasks like data reconstruction. The AE model consists of two components: the encoder and the decoder. Figure 2.3 presents a general overview of the AE pipeline. Symbolically, the encoder function $E(x)$ transforms input data into a compact latent-space representation, articulated as

$$z = E(x) \tag{2.10}$$

It encodes the input data as an internal fixed-size representation in reduced dimensionality i.e. $dim(z) << dim(x)$. The counteracting function, the decoder, $D(z)$, operates in reverse. It deciphers the latent space representation, reconstructing a resemblance of the original input.

$$\hat{x} = D(z) \tag{2.11}$$

The autoencoder training paradigm converges on minimizing the difference between the original input x and its reconstructed counterpart $\hat{x}$. This difference is often measured using a loss function, such as the mean squared error:

$$L(x,\hat{x}) = ||x-\hat{x}||^2. \tag{2.12}$$

Convergence to an acceptable loss threshold indicates the encoder's proficiency at isolating salient features, achieving robust data compression. Autoencoders, owing to their versatile design, have been requisitioned for myriad applications including, but not limited to, dimensionality reduction, anomaly detection, data denoising, and feature abstraction.

In the context of generative modeling, a pivotal modification is required to transform an AE. To reiterate, the decoder's mandate is to project latent space representations back into the data space. By enforcing a Gaussian distribution constraint upon the latent space of the AE, the decoder, in turn, acquires the ability to map samples from a Gaussian distribution onto representative data samples. Subsequently, post-training, the decoder emerges as a potent generative model. Two dominant methodologies to impose a Gaussian distribution within the AE's latent space are Variational Autoencoders (VAEs) and Adversarial Autoencoders (AAEs). Subsequent sections will explain the mechanics and elaboration of these models, offering a comprehensive overview of their operational pipelines.

Variational Auto-Encoder (VAE)

Variational Autoencoders (VAEs) [11] serve as a probabilistic extension to traditional autoencoders, facilitating unsupervised learning of probabilistic encodings. A prominent feature of VAEs is the imposition of a standard normal distribution constraint on the latent space, achieved using principles from variational calculus.

Figure 2.4 portrays the VAE structure. Contrary to traditional AEs, the VAE encompasses a probabilistic encoder. For an individual input data instance x, the encoder generates a distribution over the latent variables, expressed as $q_\phi(z|x)$, with ϕ signifying the encoder's parameters. Rather than producing a deterministic point in the latent space, the encoder manifests the parameters of a distribution, enabling the sampling of latent variables. When leveraging a Gaussian distribution for q, which is the typical choice, the encoder yields two parameters: the mean (μ) and variance (σ^2).

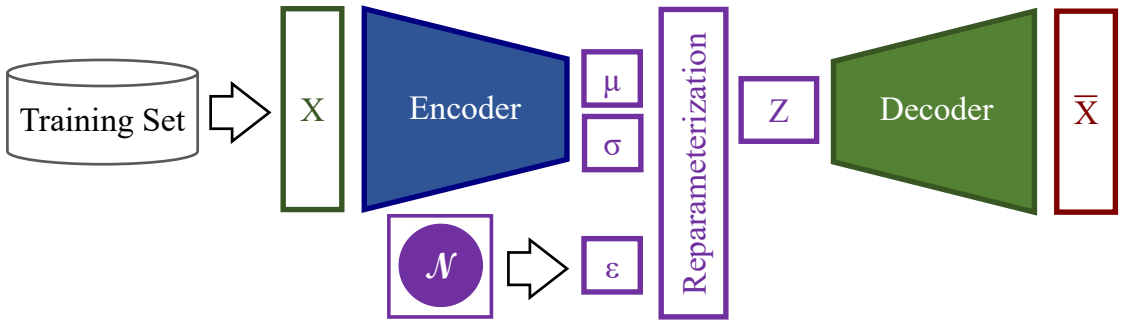


Fig. 2.4 Illustration of VAE structure

Simultaneously, the decoder introduces a likelihood function denoted as $p_\theta(x|z)$, which reflects the probability of an accurate data reconstruction given a specified latent space point, with θ representing its parameters.

Given the stochastic nature of the latent variable z, direct gradient computation becomes untenable. To counteract this, VAEs introduce the reparameterization trick. Instead of directly sampling z, a noise sample $\varepsilon \sim \mathcal{N}(0,I)$ is drawn, and z is determined using:

$$z = \mu + \sigma \odot \varepsilon \tag{2.13}$$

Such a transformation retains the differentiability of the model, thereby allowing gradient descent methodologies. The reconstruction error in this framework is the negative log-likelihood of the original data when compared with its reconstructed version:

$$\mathscr{L}_{recon}(x,\hat{x}) = -\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] \tag{2.14}$$

A cornerstone of the VAE paradigm is ensuring that the latent space closely adheres to a standard normal distribution. This is achieved by minimizing the Kullback-Leibler divergence between the encoder's output distribution and a standard normal distribution:

$$\mathscr{L}_{KL} = D_{KL}(q_\phi(z|x)||\mathcal{N}(0,I)) \tag{2.15}$$

The resultant objective, known as the Evidence Lower Bound (ELBO), is pivotal to the VAE methodology.

$$\text{ELBO} = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - D_{KL}(q_\phi(z|x)||\mathcal{N}(0,I)). \tag{2.16}$$

Training a VAE involves the maximization of the ELBO or, equivalently, minimizing its inverse. Successful training under this paradigm yields a latent space resembling a standard

normal distribution and a decoder adept at mapping from this latent representation back to the data space. Post-training, this decoder can be harnessed as a generative model to synthesize novel data samples.

Adversarial Auto-Encoder (AAE)

Adversarial Autoencoder (AAE) [12] is an elegant integration of traditional autoencoders' deterministic structure and the adversarial mechanics inherent to GANs. The inception of AAEs has enabled a sophisticated methodology to acquire latent representations that are compact and conducive to the characteristics of the original data.

The architecture of an AAE is analogous to that of an autoencoder, though integrated with an adversarial component, namely the discriminator, inspired by GANs. This adversarial module induces an additional constraint, compelling the encoder to generate latent codes that align with a standard normal distribution or any chosen prior.

The AAE training regimen embodies dual objectives. The primary objective, reminiscent of conventional autoencoders, seeks to decrease the reconstruction error between the original input and its subsequent decoded version. This endeavor ensures the encoder-decoder pair effectively represents and reconstructs intrinsic data characteristics.

The adversarial facet of the AAE is encapsulated in its secondary objective. In this context, the encoder is responsible for generating latent representations that the discriminator struggles to distinguish from genuine samples drawn from a prior distribution, often standard Gaussian. This dynamic emulates the generator-discriminator interplay observed in GANs, with the encoder approximating the role of the GAN generator, aiming to map the data distribution to a specified prior. An illustration of the AAE structure is presented in figure 2.5.
Formally, the objective function of the AAE, denoted as $\mathscr{L}$, can be decomposed into two distinct components: the reconstruction loss $\mathscr{L}_{\text{recon}}$ and the adversarial loss $\mathscr{L}_{\text{adv}}$:

$$\mathscr{L} = \mathscr{L}_{\text{recon}} + \lambda \mathscr{L}_{\text{adv}}, \tag{2.17}$$

where λ represents a hyperparameter that organizes the balance between the reconstruction and adversarial objectives. Upon achieving convergence in the training process, the resultant AAE exhibits a latent space aligned with the standard normal distribution enforced by the adversarial mechanism. The decoder, having acquired the data's nuances, can subsequently be repurposed as a generative tool to produce synthetic data instances.

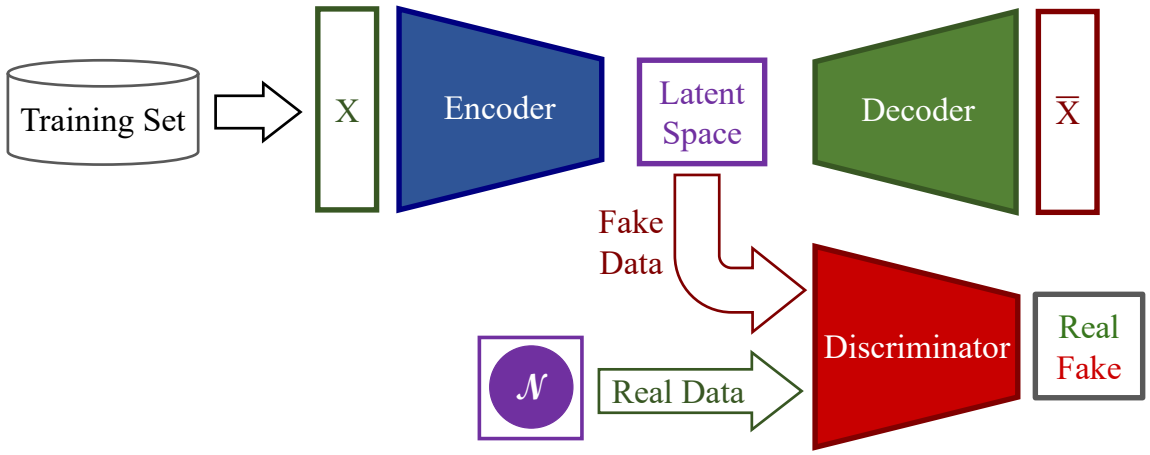


Fig. 2.5 An overview of AAE pipeline

2.2 Probabilistic Forecasting Foundation

In this section, we establish the groundwork for understanding probabilistic forecasting. We begin by defining the notation to be consistently used throughout this thesis, followed by a formal expression of the probabilistic forecasting problem. Lastly, we categorize and describe the various existing classes of methodologies devised to address this complex problem.

2.2.1 Problem Statement

A given dataset consists of a set of observations generated by the process of interest. Let $\mathscr{D}$ be the dataset containing N independent and identically distributed (i.i.d.) samples represented as:

$$\mathscr{D} = \{X_0, X_1, ..., X_N\}, \tag{2.18}$$

where, X denotes a time series data comprising T timesteps, i.e., $X = \{x_0, x_1, ..., x_T\}$. A dataset may include one or more time series ($N \geq 1$), with each time series potentially having a different length denoted by T. Each timestep x consists of a feature vector of size F, representing the observed features at that timestep. A univariate time series comprises a single feature ($F = 1$), while a multivariate time series includes more than one feature ($F > 1$). Throughout this thesis, we use X to represent a time series data, while x_t^f denotes the value of feature f at timestep t.

In forecasting, our ultimate objective is to obtain a predictive probability distribution for

future quantities based on historical information. To be more specific, we aim to model the conditional probability distribution as follows:

$$P(x_{t+1:t+h}|x_{0:t}). \tag{2.19}$$

Here, h represents the forecast horizon. If $h = 1$, we are dealing with a one-step-ahead forecasting task. For $h > 1$, it is referred to as a multistep ahead forecasting task. When the dataset contains multiple time series, i.e., $N > 1$, there are two primary approaches for constructing a forecasting model: local modeling and global modeling. In local modeling, we independently model each time series, resulting in N forecasting models for the dataset. On the other hand, global modeling involves employing a single model for all time series data in the dataset.

2.2.2 Approaches

There are various approaches to constructing forecasting models. These approaches fall roughly into three categories: deterministic modeling, explicit probabilistic modeling, and implicit probabilistic modeling.

Deterministic Modeling aims to model only one specific property of predictive distribution. To achieve this goal, the forecast model defines a deterministic function $\mathscr{F} : \mathbb{R}^t \to \mathbb{R}^h$ that maps the historical window to the property of interest. This approach is mainly employed in literature to model the predictive distribution's mean, median, or specific quantile. For instance, a deterministic model that models the mean of the predictive distribution for the one-step-ahead forecasting task approximates the following function:

$$\mathbb{E}(x_{t+1}) = \mathscr{F}(x_{0:t}). \tag{2.20}$$

Deterministic modeling simplifies the forecasting problem by modeling only a specific property of the predictive distribution. Therefore, these models are generally easier and faster to train and can provide an accurate approximation for the target property of the predictive distribution. However, a limited perspective of future events may not be sufficient in many real-world scenarios. To make sensitive and complex decisions, we require a comprehensive understanding of what the future holds.

Explicit Probabilistic Modeling defines a parametric probability distribution explicitly Q_Φ to approximate predictive distribution P. Then, the parameters of the assumed distribution (Φ) are estimated using the historical data. For instance, assume that we consider Q a

t-distribution for explicit probabilistic modeling. Since the t-distribution can be fully defined with only one parameter ($\Phi = t$), the explicit model needs to approximate the following deterministic function:

$$\Phi = \mathscr{F}(x_{0:t}). \tag{2.21}$$

By assuming a specific distribution type, this approach simplifies maximizing the model likelihood. To train the model for the distribution parameters Φ, the density function definition is incorporated into the likelihood expression [2]. Once the parameters are specified, Q can explicitly approximate the predictive distribution or future forecasts can be obtained by sampling from the Q distribution.

Accurately defining a parametric probability distribution that properly represents the predictive distribution poses a challenge for explicit models. When domain knowledge is available regarding the type of predictive distribution, explicit models are easy to train and can yield accurate results. However, if baseless or simplistic assumptions are made regarding the predictive distribution type, the model's performance becomes unreliable and may provide misleading outcomes. Unfortunately, real-world scenarios often involve complex and high-dimensional predictive distributions, making it difficult to make precise and well-informed assumptions.

Implicit Probabilistic Modeling interacts with predictive distribution indirectly, primarily through sampling. Since these models do not require an explicit definition density function for predictive distribution, they are more flexible and can accurately represent complex and high-dimensional probability distribution. However, our access to model distribution is limited to samples of predictive distribution with Monte Carlo sampling.

Part II

Probabilistic Forecasting

Chapter 3

Review of Probabilistic Forecasting Literature

The realm of probabilistic forecasting represents a long-standing challenge that has captivated scientists for decades. The quest to devise an optimally accurate probabilistic forecaster has led to a wealth of research and a diverse array of methodologies. Consequently, the literature in this field is extensive and multifaceted. This chapter aims to conduct a comprehensive review of the significant probabilistic forecasting methodologies, ranging from classical, model-driven approaches to modern, data-driven techniques. The objective is to provide a broad overview of the probabilistic forecasting landscape, highlighting the strengths and weaknesses of various approaches. By doing so, this chapter endeavors to offer a coherent and insightful perspective on the evolution and current state of probabilistic forecasting techniques.

3.1 Classical Approaches

Forecasting is a well-established research field with a wide range of methods developed to tackle this task. Deterministic statistical methods were the pioneers of forecasting models. Some notable examples include the autoregressive (AR) model [13], moving average (MA) model [14], autoregressive moving average (ARMA) models [15], Exponential Smoothing (ES) methods [16, 17], and structural time series models [18, 19, 20]. These methods offer simple yet effective models that are still widely used today. Unfortunately, the lack of uncertainty quantification limited the application of these models in sensitive domains.

It did not take long for the first probabilistic forecast model to join the forecasting tool-

box. The Autoregressive Conditional Heteroscedastic (ARCH) [21] was proposed in the finance domain to model forecasting volatility based on past observations over a short period. Generalized ARCH (GARCH) [22] extended the ARCH model, which employed an ARMA model over past error terms to capture volatility forecasting. Since then, numerous variations of GARCH models have suggested [23, 24, 25, 26]. Gneiting et al. [27] proposed ensemble model output statistics (EMOS) for probabilistic forecasting. EMOS consists of linear regression models trained to minimize the Continuous Ranked Probability Score (CRPS) to obtain unbiased and calibrated probabilistic forecasts. Bayesian modeling has also been widely used for probabilistic forecasting. For example, Raftery et al.[28] employed Bayesian model averaging (BMA) to combine forecasts from multiple models using Bayesian techniques. Frazier et al.[29] applied Approximate Bayesian Computation (ABC) to generate probabilistic forecasts by combining Bayesian inference with the autoregressive fractionally integrated moving average (ARFIMA) model. Loaiza-Maya et al. [30] incorporated a scoring rule in a Bayesian forecaster to construct a likelihood-free model.

3.2 Machine Learning Approaches

Researchers have explored machine learning methods for probabilistic forecasting. Clements et al. [31] investigated three bootstrapping methods for providing prediction intervals as probabilistic forecasts. The application of random forests for probabilistic forecasting has been studied in [32]. Corani et al. [33] automated kernel selection and hyperparameter estimation of a Gaussian process for time series forecasting, creating an automatic forecasting pipeline. Tajmouri et al. [34] utilized the nearest neighbor algorithm to generate a conformal prediction for time series. Conformal prediction is a method that produces predictive regions given a confidence level. Gradient Boosted Decision Trees were examined for probabilistic time series forecasting in [35].

3.3 Neural Network based Approaches

With the rise of neural networks, a new set of approaches for probabilistic forecasting has emerged, offering end-to-end solutions that effectively learn from large datasets and leverage parallel computation on GPUs for fast computations. As a result, researchers quickly adopted neural networks in the forecasting pipeline, leading to significant advancements in the field.

Khosravi et al. [36] combined neural networks with the GARCH model to provide prediction interval for time series forecasting. WaveNet [4] employed dilated convolutional

neural networks (CNNs) to process long input time series data effectively and has shown outstanding performance in audio generation and time series forecasting. Long short-term memory(LSTM) networks have been used for parameterizing a Linear State Space Model (SSM) to generate probabilistic forecast [37]. Rasp et al. [38] employed a neural network to post-process an ensemble weather prediction model. Prophet [39] is a modular regression model built upon the Generalised Additive Model [20], which can integrate domain knowledge into the modeling process.

Wen [40] suggested a multi-horizon quantile forecaster (MQ-R(C)NN) based on Sequence-to-Sequence RNN (Seq2Seq) [41] model. The author proposed an approach called forking sequences to improve the stability and performance of encoder-decoder models. Gesthaus et al. [42] introduced another method for quantile forecasting based on the spline functions for representing output distributions. Gouttes [43] suggested an autoregressive RNN model based on Implicit Quantile Networks [44] for quantile forecasting. Lim et al. [45] proposed the Temporal Fusion Transformer(TFT). This attention-based model utilized a sequence-to-sequence model for the local process of the input window and a temporal self-attention decoder to model long-term dependencies. Kan et al. [46] proposed multivariate quantile functions, which are parameterized using gradients of "input convex neural networks" [47]. The model employs an RNN-based feature extractor to build a probabilistic forecaster. Xu et al.[48] expanded quantile RNN forecaster models to handle time series with mixed sample frequency.

Salinas et al. [49] proposed an autoregressive forecaster for high-dimensional multivariate time series. To do so, the authors utilized a low-rank covariance matrix to model the output distribution. They took a copula-based approach in conjunction with an RNN model to construct the forecaster. DeepAR [3] is suggested as a simple yet effective autoregressive explicit model based on RNNs. Wang et al. [50] presented a global-local forecasting model. The suggested model uses a global model based on deep neural networks for capturing global non-linear patterns, while a probabilistic graphical model extracts the individual random effects locally. DeepTCN [51] is proposed as an encoder-decoder forecasting model based on dilated CNN models. This model can be utilized either as an explicit model or a quantile estimator.

Normalizing Kalman Filter [52] has been proposed for probabilistic forecasting. This model uses normalizing flows [53] to augment a linear Gaussian state space model. Rasul et al. [54] employed conditional normalizing flows to compose an autoregressive multivariate probabilistic forecasting model. Hasson et al. [55] introduced Level Set Forecaster

(LSF). LSF organizes training data into groups, considering those that are sufficiently similar. Subsequently, the bins containing the actual values are employed to generate the predicted distributions. Denoising diffusion models have been used to develop TimeGrad [56], an autoregressive multivariate forecaster. Türkmen et al. [57] combined classical discrete-time conditional renewal processes [58] with model neural networks architecture to build a forecasting model suitable for intermittent demand forecasting.

3.4 GAN-based Approaches

This thesis investigates the application of GANs on the time series forecasting task. Although GANs are widely applied in various tasks in the image domain, their application on time series data is relatively new. Nevertheless, a few studies utilize the GAN for probabilistic forecasting, in addition to the contributions presented in this thesis. Zhou et al. [59] proposed a GAN-based point forecaster for the stock market data. The generator receives basic indicator information to forecast the next closing price. The discriminator employs past closing prices to distinguish the generated forecast from the actual closing price. Notably, the generator does not use a noise vector as input. The authors utilized prediction loss (e.g., mean squared loss) and direction prediction loss in conjunction with adversarial loss to train the model. Zhang et al. [60] developed a point forecaster for the stock market, utilizing prediction and adversarial losses to train their model. Their approach is similar to the previous work mentioned, focusing on forecasting the next closing price. Lin et al. [61] introduced a probabilistic forecasting model for traffic flow prediction. The generator output the reconstructed condition in conjunction with the forecast. The authors employed Wasserstein GAN with gradient penalty to train the GAN and utilized prediction loss and reconstruction loss to train the generator. Kabir et al. [62] used adversarial training to quantify the electricity price's uncertainty with a prediction interval. Yin et al. [63] combined a variational autoencoder (VAE) with a conditional GAN to construct a forecaster for multivariate time series. The VAE encoder maps the exogenous data sequence to a Gaussian distribution. Then the generator, which also acts as a decoder of VAE, maps from the aforementioned Gaussian distribution to predictive distribution.

3.5 Established Baselines

This section is dedicated to a detailed examination of 11 advanced probabilistic forecasting models. These models represent the latest advancements in the field and have demonstrated their effectiveness and robustness across various research studies. The primary focus will

be on elucidating the distinctive features, methodologies, and underlying principles of each model, highlighting their contributions to advancing probabilistic forecasting. In subsequent chapters, this thesis will benchmark these models as baselines, employing them as comparative standards to evaluate the performance of newly proposed methodologies in diverse experimental settings. This comparative analysis aims to provide a comprehensive understanding of where the newly proposed models stand in the context of current state-of-the-art probabilistic forecasting techniques.

3.5.1 DeepAR

DeepAR, as presented by Salinas et al. [3], serves as a prominent explicit probabilistic forecasting model in the realm of time series prediction. At its core, DeepAR constructs a predictive distribution utilizing a Gaussian distribution, wherein the distribution's parameters, specifically the mean and standard deviation, are derived from the historical data.

To facilitate this parameter extraction, DeepAR integrates a neural network structure. This architecture predominantly features a multi-layer Recurrent Neural Network (RNN) to map the time series history onto the Gaussian distribution parameters. The training regimen for the network adopts a maximum likelihood estimation paradigm. Within this paradigm, the probability density function of the Gaussian distribution acts as the likelihood function, ensuring optimal parameter approximation.

3.5.2 DeepState

Rangapuram et al. [37] proposed the DeepState model as a novel approach that combines state space models (SSMs) with deep learning for probabilistic time series forecasting. The paper proposes bridging the gap between SSMs and deep learning by parametrizing a linear SSM with a jointly-learned recurrent neural network (RNN). The deep recurrent neural network parametrizes the mapping from covariates to state space model parameters. The model is interpretable, as it allows inspection and modification of SSM parameters for each time series. It can automatically extract features and learn complex temporal patterns from raw time series and associated covariates. The model promises to contain the interpretability and data efficiency of SSMs and large data handling of deep neural networks.

3.5.3 DeepFactor

The DeepFactor [50] employs both classical and deep neural time series forecasting models to construct probabilistic forecaster. DeepFactor is a global-local method where each time series or its latent function is represented as a combination of a global time series and a corresponding local model. The global factors are modeled by Recurrent Neural Networks (RNNs). These latent global deep factors can be thought of as dynamic principal components driving the underlying dynamics of all time series. Local models can include various choices like white noise processes, linear dynamical systems (LDS), or Gaussian processes (GPs). This stochastic component captures individual random effects for each time series. DeepFactor systematically combines of deep neural networks and probabilistic models in a global-local framework and develops an efficient and scalable inference algorithm for non-Gaussian likelihoods.

3.5.4 Deep Renewal Processes

Deep Renewal Processes [57] is a novel framework for probabilistic forecasting of intermittent and sparse time series. The method focuses on forecasting demand data that appears sporadically, posing unique challenges due to long periods of zero demand followed by non-zero demand. The framework is based on discrete-time renewal processes, which are adept at handling patterns like aging, clustering, and quasi-periodicity in demand arrivals. The model innovatively incorporates neural networks, specifically recurrent neural networks (RNNs), replacing exponential smoothing with a more flexible neural approach. Moreover, the framework also extends to model continuous-time demand arrivals, enhancing flexibility and applicability.

3.5.5 GPForecaster

For the Gaussian Process (GP), we employed a model implemented in GluonTS [64]. This model defines a Gaussian Process with Radial Basis Function (RBF) as the kernel for each time step. GP with an RBF kernel can capture both the trends and nuances of time series data, while its probabilistic nature provides a measure of uncertainty or confidence in its predictions. This combination is especially valuable in scenarios where understanding the uncertainty of future events is as crucial as the predictions themselves.

3.5.6 Multi-Horizon Quantile Recurrent Forecaster

The Multi-Horizon Quantile Recurrent Forecaster, or in short MQ-RNN [40], is a direct quantile regression method that aims to forecast the quantile of predictive distribution for multiple horizons directly. The model employs Seq2Seq architecture for quantile regression. Besides MQ-RNN, this thesis also utilized MQ-CNN, which is the same model with CNN components as the main components of Seq2Seq architecture instead of RNN modules.

3.5.7 Prophet

The prophet [39] is a robust and scalable solution for forecasting time series data, particularly in business contexts where handling specific scenarios such as multiple strong seasonalities, trend changes, outliers, and holiday effects becomes paramount. The prophet decomposes time series into three components. The first component, trend, Captures non-periodic changes in the data. The second component, seasonality, represents periodic changes, like weekly yearly cycles. finally, the last component, holidays, accounts for holidays and events that occur on irregular schedules. Each component is modeled separately and fits using Bayesian infeProbabilistic Stream Flow Forecasting for Reservoir Operatiorence to estimate parameters. Furthermore, the model is designed to be intuitive, allowing analysts to make informed adjustments based on domain knowledge.

3.5.8 WaveNet

The Wavenet [4] is a powerful generative model originally proposed for voice synthesis. Wavenet forecasts sequentially in an autoregressive fashion using previously generated samples as input to predict the next.WaveNet employs a deep neural network consisting of stacks of dilated causal convolutions, enabling it to capture a wide range of audio frequencies and temporal dependencies. Wavenet defines predictive distribution for each time step in the horizon by quantifying the time series and using a softmax at the output layer of the model. In other words, the wavenet quantifies the time series to make the problem similar to a classification problem and provides the output of the softmax layer as the predictive distribution.

3.5.9 Transformer

The transformer [65] is a novel sequence-to-sequence model that represents a departure from previous methods by relying entirely on attention mechanisms, dispensing with recurrent and convolutional neural networks. The transformer follows the typical encoder-decoder

structure, but both the encoder and the decoder are composed of a stack of identical layers, each with a novel self-attention mechanism. The self-attention mechanism allows the model to weigh the importance of different time steps in the input sequence, enabling it to capture context more effectively. At the core of the model is the scaled dot-product attention, a mechanism that calculates the attention scores based on the scaled dot products of queries and keys. Unlike recurrent models, the transformer allows for much more parallelization, making training faster. Moreover, the self-attention mechanism enables the model to learn long-range dependencies more effectively.

3.5.10 Temporal Fusion Transformers

Temporal Fusion Transformers (TFT) [45] is an encoder-decoder-based quantile regression model for multi-horizon forecasting. It has many novelties under the hood that make its performance excel in many scenarios, including:

- **Static Covariate Encoders**: These encode static context vectors, which are crucial for conditioning the temporal dynamics within the network.
- **Gating Mechanisms**: They provide adaptive depth and complexity to the network, allowing it to handle a wide range of datasets and scenarios.
- **Variable Selection Networks**: These networks select relevant input variables at each time step, enhancing interpretability and model performance by focusing on the most salient features.
- **Temporal Processing**: The model employs a sequence-to-sequence layer for local processing and interpretable multi-head attention mechanisms to capture long-term dependencies.

The TFT demonstrates superior performance across various real-world datasets, outperforming existing benchmarks in multi-horizon forecasting. The model facilitates interpretability, allowing users to identify globally important variables, persistent temporal patterns, and significant events.

Chapter 4

Assessment Methods for Probabilistic Forecasting

This chapter is dedicated to a thorough exploration of how the performance of probabilistic forecasters is assessed, focusing on three conventional methods: the Continuous Ranked Probability Score (CRPS), CRPS-Sum, and the Energy Score. These methods are pivotal in understanding and evaluating the efficacy of probabilistic forecasting models, and this chapter delves into their intricacies and applications. Moreover, this chapter critically examines their discrimination abilities. It investigates how effectively each method distinguishes between varying levels of forecasting accuracy and reliability, providing insight into their strengths and limitations. This analysis is crucial, as it reveals nuanced aspects of these methods that are often overlooked but are essential for accurately interpreting the performance of probabilistic forecasters.

4.1 Scoring Rule

A scoring rule provides a general framework for evaluating the alignment of P_{data} with P_{model}. A scoring rule is any real-valued function that provides a numerical score based on a predictive distribution (i.e., P_{model}) and a set of observations X.

$$S(P_{\text{model}}, X) \tag{4.1}$$

The scoring can be defined as positively or negatively orientated. In this paper, we consider the negative orientation since it can be interpreted as a model error, and as a result, it is more prevalent in the scientific community. Hence, a lower score indicates a better probabilistic model.

A scoring rule is proper if the following inequality holds:

$$S(P_{\text{model}}, X) \geq S(P_{\text{data}}, X) \tag{4.2}$$

A scoring rule is strictly proper if the equality in Equation (4.2) holds if and only if $P_{\text{model}} = P_{\text{data}}$ [66]. Therefore, only a model perfectly aligned with the generative process can acquire the lowest strictly proper score. Various realizations of scoring rules have been proposed to evaluate the performance of a probabilistic forecaster. Bellow, we introduced three scoring rules commonly used to assess probabilistic forecasting models.

4.2 Continuous Ranked Probability Score (CRPS)

CRPS is a univariate strictly proper scoring rule which measures the compatibility of a cumulative distribution function F with an observation $x \in \mathbb{R}$ as

$$\text{CRPS}(F, x) = \int_{\mathbb{R}} (F(y) - \mathbb{1}\{x \leq y\})^2 \, \mathrm{d}y, \tag{4.3}$$

where $\mathbb{1}\{x \leq y\}$ is the indicator function, which is one if $x \leq y$ and zero otherwise.

The predictive distributions are often expressed in terms of samples, possibly through Monte-Carlo sampling [66]. Fortunately, several methods exist to estimate CRPS given only samples from a predictive distribution. The precision of these approximation methods depends on the number of samples we use for estimation. Below you can find a list of the most used techniques.

Empirical CDF:

In this technique, we approximate the CDF of a predictive model using its samples.

$$\hat{F}(y) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{x_i \leq y\} \, . \tag{4.4}$$

Then, we can use $\hat{F}(y)$ in conjunction with Equation(4.3) to approximate CRPS.

Quantile-based:

The pinball loss or quantile loss at a quantile level $\alpha \in [0,1]$ and with a predicted α^{th} quantile q is defined as

$$\Lambda_\alpha(q,x) = (\alpha - \mathbb{1}\{x < q\})(x-q). \tag{4.5}$$

The CRPS has an intuitive definition as the pinball loss integrates over all quantile levels $\alpha \in [0,1]$,

$$\text{CRPS}\left(F^{-1},x\right) = \int_0^1 2\Lambda_\alpha\left(F^{-1}(\alpha),x\right) d\alpha\,, \tag{4.6}$$

where F^{-1} represents the quantile function. In practice, we approximate quantiles based on the samples we have. Therefore, Equation (4.6) can be approximated as a summation over N quantiles. The precision of our approximation depends on the number of quantiles as well as the number of samples we have.

Sample Estimation:

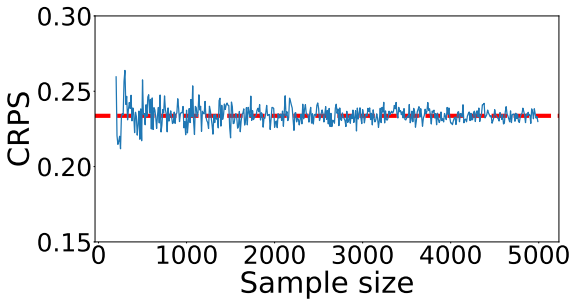
Using lemma 2.2 of [67] or identity 17 of [68], we can approximate CRPS by

$$\text{CRPS}(F,x) = \text{E}_F\left|X-x\right| - \frac{1}{2}\text{E}_F\left|X-X'\right|\,, \tag{4.7}$$

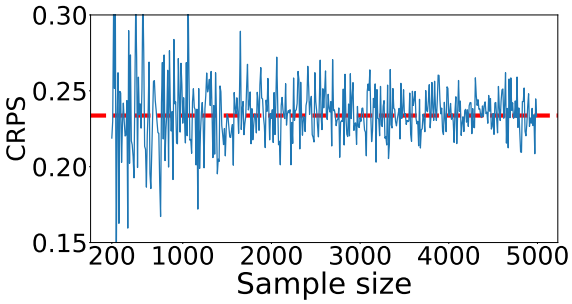
where X and X' are independent copies of a random variable with distribution function F and a finite first moment [66].

We ran a simple experiment to investigate the significance of sample size on the accuracy of different approximation methods. In this experiment, we assumed that the probabilistic model follows a Gaussian distribution with $\mu = 0$ and $\sigma = 1$. Then, we approximated CRPS (F,x) where $x = 0$ with various sample sizes in the range $[200, 5000]$. Since we know the probabilistic model distribution, we can calculate the value of CRPS analytically, i.e., $\text{CRPS}(F,x) \approx 0.2337$.

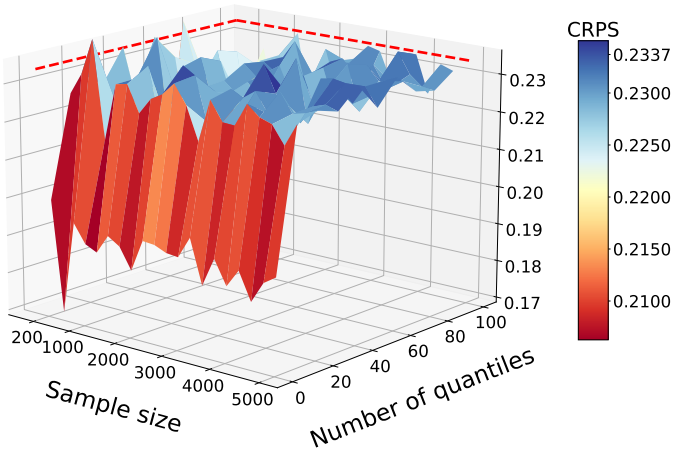
From Figures 4.1a and 4.1b, we can perceive that the empirical CDF method and sample estimation method can converge to the close vicinity of the true value efficiently. However, the empirical CDF method has less variance than sample estimation. The method based on pinball loss depends on sample size and the number of quantiles. Figure 4.1c portrays how these two factors affect the approximation. With the number of quantiles greater than 20, the pinball loss method can produce an excellent approximation using only a few samples (circa 500 samples).



(a) Empirical CDF



(b) Sample estimation



(c) Quantile based

Fig. 4.1 The effect of sample size on precision of CRPS approximation using different methods:(**a**) Empirical CDF, (**b**) Sample estimation, (**c**) Quantile based. We can see that all approximation methods can provide close estimation; however, the sample estimation method has more variance in estimation.

4.3 Energy Score (ES)

Energy Score (ES) is a strictly proper scoring rule for multivariate time series. For an m-dimensional observation x in $\mathbb{R}^m$ and a predictive cumulative distribution function F, the energy score (ES) [66] is defined as

$$\mathrm{ES}(F,\mathbf{x}) = E_F\|\mathbf{X}-\mathbf{x}\|^{\beta} - \frac{1}{2}E_F\left\|\mathbf{X}-\mathbf{X}'\right\|^{\beta}, \tag{4.8}$$

where $\|\ .\ \|$ denotes Euclidean distance and $\beta \in (0,2)$. Here, CRPS is a special case of ES, where $\beta = 1$ and $m = 1$. While ES is a strictly proper scoring rule for all choices of β, the

standard choice in the application is normally $\beta = 1$ [66].

ES provides a probabilistic forecast model assessment method that works well on multivariate time series. Unfortunately, ES suffers from the curse of dimensionality [69], and its discrimination power decreases with increasing data dimensions. Still, the performance of ES in lower dimensionalities complies with the expected behavior of an honest and careful assessor. Hence, we can use its behavior in lower dimensionalities as the reference for comparison with newly suggested assessment methods.

4.4 CRPS-Sum

To address the limitation of ES in multidimensional data, Salinas et al. [49] introduced CRPS-Sum for evaluating a multivariate probabilistic forecasting model. CRPS-Sum is a proper scoring rule, and it is not strictly proper. CRPS-Sum extends CRPS to multivariate time series with a simple modification. It is defined as

$$\text{CRPS-Sum} = \mathrm{E}_t \left[\text{CRPS} \left(F_{sum}^{-1}, \sum_i x_t^i \right) \right], \tag{4.9}$$

where F_{sum}^{-1} is calculated by summing samples across dimensions and then sorting to obtain quantiles. Equation (4.9) calculates CRPS based on the quantile-based method (Equation (4.6)). More generally, one can calculate the CRPS-Sum by summing samples and observations across the dimensions. This way, we would acquire a univariate vector of samples and observation. Then, we can apply any aforementioned approximating methods to calculate CRPS-Sum.

CRPS-Sum has been widely welcomed by the scientific community. Many researchers have used it to report the performance of their models [49, 54, 56, 52]. However, the capabilities of CRPS-Sum have not been investigated thoroughly, unlike the vast studies dedicated to the properties of ES and CRPS [66, 69, 70]. To evaluate the discrimination ability of CRPS-Sum, we conducted several experiments on a toy dataset and outlined the results in the following sections. Furthermore, these findings have been published in [71]

4.5 CRPS-Sum Sensitivity Study

In this study, we inspected the sensitivity of CRPS-Sum concerning the changes in the covariance matrix. This study extends the sensitivity study previously conducted by [69, 70] for various scoring rules, including CRPS and ES. For a more straightforward interpretation of the scoring-rule response to changes in a model or data, we defined relative changes in the scoring rule Δ_{rel}.

We ran our experiment N times, where CS_i denotes the obtained CRPS-Sum from the i-th experiment. We define

$$\overline{\mathrm{CS}} = \frac{1}{N}\sum_{i=1}^{N} \mathrm{CS}_i\,, \tag{4.10}$$

as the mean value of CRPS-Sum for the N experiments. Furthermore, let CS^* signify the CRPS-Sum for a model identical to the true data distribution. Now, the relative changes [69] in CRPS-Sum is defined as

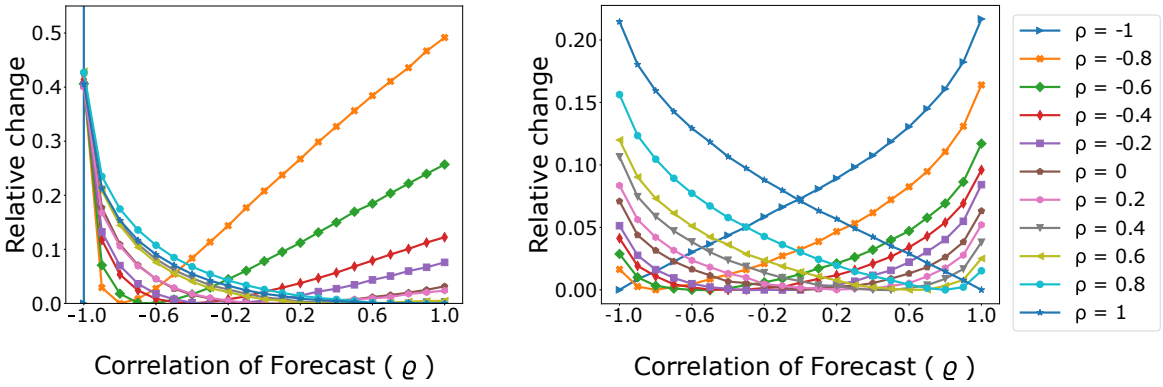
$$\Delta_{\mathrm{rel}}(\mathrm{CS}) = \frac{\overline{\mathrm{CS}} - \overline{\mathrm{CS}}^*}{\overline{\mathrm{CS}}^*}\,. \tag{4.11}$$

This metric frames the relative changes in the CRPS-Sum of a forecasting modeling across our experiments as the differences between the predicted and actual density of the stochastic process. The main idea is to determine the sensitivity of the scores with respect to some biased non-optimal forecast in a relative manner.

In this study, we have a true data distribution that follows a bivariate normal distribution with $\mu = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ and $\Sigma = \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}$ where $\rho \in [-1, -0.8, ..., 0.8, 1]$. Furthermore, we specified a forecasting model f that follows the same distribution; however, this time, the off-diagonal element of the covariance matrix is $\varrho \in [-1, -0.9, -0.8, ..., 0.8, 0.9, 1]$. In our study, we sampled $n = 2^{14}$ windows of size $w = 2^9$, as suggested in [70].

Figure 4.2 illustrates the relative change in CRPS-Sum and ES with respect to changes in correlation ρ of the data-generating process as a function of the correlation coefficient ϱ of the family of models we studied. We can observe that ES behavior is unbiased with regard to ρ, and its figure is symmetric. This is the expected behavior from a scoring rule in this scenario. In contrast, the response of CRPS-Sum to change in ρ is not symmetric. It is more sensitive to the changes when the covariance ρ of the data is negative and almost indifferent to the changes when the covariance ρ of the data is positive.

Hence, the sensitivity of CRPS-Sum to changes in covariance depends on the dependency structure of true data. In real-world scenarios, where we cannot access the covariance matrix of the data-generative process, we cannot reliably interpret CRPS-Sum and compare various models based on CRPS-Sum.



(a) CRPS-Sum (b) ES

Fig. 4.2 The relative change in CRPS-Sum (**a**) and ES (**b**) with respect to ρ and ϱ. The correlation of forecast (ϱ) is presented on the x axis, and the correlation of data (ρ) is depicted with different lines. Unlike ES, the CRPS-Sum figure is not symmetric, which indicates that it is biased with regard to the ρ value.

4.6 The Effect of Summation on CRPS-Sum

To calculate CRPS-Sum, first, we summed the time series over the dimensions [49]. Although this aggregation let us turn a multivariate time series into a univariate one, we lost important information concerning the model's performance in each dimension. Furthermore, the values of negatively correlated dimensions negate each other, and consequently, those dimensions will not be presented in aggregated time series.

For instance, assume we have a multivariate time series x with two dimensions. Our data follow a bivariate Gaussian distribution with $\mu = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ and $\Sigma = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$. Hence, the following relation holds between dimensions:

$$x^0 = -x^1. \tag{4.12}$$

By summing over dimensions, we have:

$$\sum_i x^i = 0\,. \tag{4.13}$$

After summation, we acquire a signal with constant zero, and all the information regarding the variability of the original time series is lost.

To acquire information regarding the model's performance on each dimension, we can calculate CRPS first. Once the CRPS was validated, we could calculate the CRPS-Sum to check how well the model learned the relationship between the dimensions, and even, at this point, we should not forget the flaws of CRPS-Sum that we witnessed, e.g., sensitivity toward data covariance and loss of information during summation. Unfortunately, the importance of CRPS is ignored in most of the recent papers in the probabilistic forecast domain. In these papers, the CRPS is either not reported at all [52, 56], or the argument about the performance of the model is made solely based on CRPS-Sum [54, 49]. Considering the flaws of CRPS-Sum, this trend can put the assessment results of these recent models in jeopardy.

4.7 Closer Look into CRPS-Sum in Practice

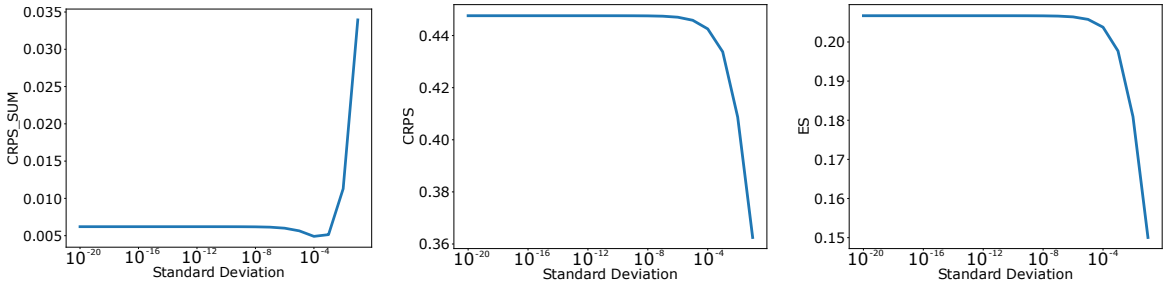
In the previous section, we discussed the properties of CRPS-Sum and indicated its shortcomings in hypothetical scenarios using toy data settings. In this section, we aim to investigate CRPS-Sum capabilities with real datasets. To do so, we conducted experiments by constructing simple models based on random noise and studied their performance using CRPS-Sum. In our first experiment, we employed the exchange-rate dataset [72]. The exchange-rate dataset is a multivariate time series dataset containing the daily exchange rate of eight countries: Australia, British, Canada, Switzerland, China, Japan, New Zealand, and Singapore, collected between 1990 and 2016. This dataset has a few dimensions, which lets us use ES alongside CRPS and CRPS-Sum. Additionally, it is easier to perform qualitative assessments on lower dimensionalities. We used the dataset in the same setting proposed in [49].

We also utilized the low-rank Gaussian copula processes method (GP-copula) from [49]. GP-copula combines an RNN-based time series model with a Gaussian copula process output model for probabilistic forecasting. Furthermore, the model employs a low-rank covariance structure to reduce the computational complexity and handle non-Gaussian marginal distributions. We selected this model since the model performance has been reported in CRPS-Sum.

Our first model is a dummy univariate model which follows a Gaussian distribution. The mean of the Gaussian distribution is $\mu = \mu_{last}$ where μ_{last} is the mean of the last values in the input vector over the dimensions, i.e.,

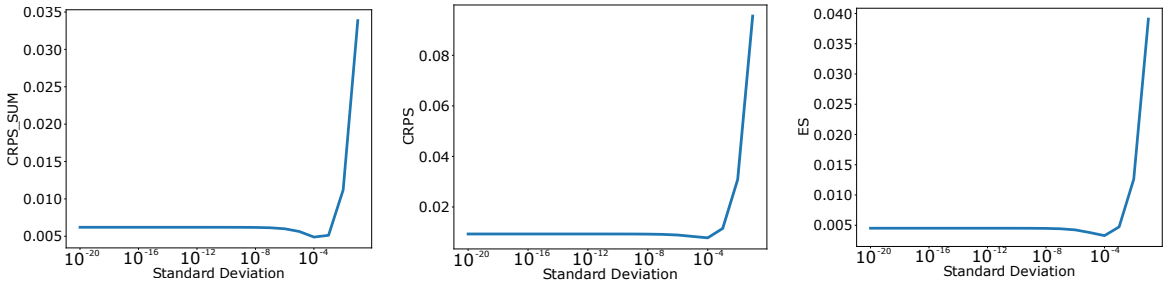
$$\mu_{last} = \frac{1}{D}\sum_{i=1}^{D} x_T^i. \tag{4.14}$$

We used $\sigma = 10^{-4}$ as the standard deviation of the dummy univariate model in our experiments. Nevertheless, as shown in figure 4.3 and 4.4, the results are not dependent on the σ value when $\sigma \leq 10^{-4}$. We used this model to generate the forecast for every dimension.



(a) CRPS-Sum (b) CRPS (c) ES

Fig. 4.3 The assessment of univariate dummy model with $\sigma \in \{10^{-1}, 10^{-2}, ..., 10^{-20}\}$ using CRPS-Sum, CRPS, and ES. The plot is depicted on a logarithmic scale.



(a) CRPS-Sum (b) CRPS (c) ES

Fig. 4.4 The assessment of multivariate dummy model with $\sigma \in \{10^{-1}, 10^{-2}, ..., 10^{-20}\}$ using CRPS-Sum, CRPS, and ES. The plot is depicted on a logarithmic scale.

We employed a multivariate Gaussian distribution for the second model to build a dummy multivariate forecaster. The mean of the i-th dimension of the multivariate Gaussian distribution is the value of the last time step in the input window, i.e., $\mu_i = x_T^i$. The covariance

matrix is zero everywhere except on its main diagonal, filled with 10^{-4}. In other words, we extended the last observation of the input window as the prediction and applied a small perturbation from a Gaussian distribution.

Table 4.1 presents the CRPS-Sum, CRPS, and ES of the two dummy models and the result of the GP-copula model from [49] on the exchange-rate dataset. Note that all values in this paper were calculated using the sampling method. We also calculated these metrics using the quantile method, which yielded similar results. While the CRPS-Sum suggests that the dummy univariate model is much better than the GP-copula, the CRPS and ES indicate that the performance of the dummy univariate model is worse than the GP-copula. The results reported by CRPS and ES are aligned with our expectations; however, the CRPS-Sum reports a misleading assessment.

Table 4.1 This table illustrates the results from dummy models on the exchange-rate dataset and compares their performance with GP-copula based on CRPS-Sum, CRPS, and ES. It shows that CRPS-Sum dummy models have better performance in comparison to GP-copula.

	GP-copula	**Dummy Univariate**	**Dummy Multivariate**
CRPS-Sum	0.0070	0.0049	0.0048
CRPS	0.0092	0.4425	0.0077
ES	0.0043	0.2037	0.0032

On the other hand, the quantitative results for the dummy multivariate model are quite surprising. All three assessment methods denote that the dummy multivariate has a superior performance compared to the GP-copula. To further explain this unexpected result, we analyzed the performance of these models qualitatively.

Figure 4.5 depicts the forecasts from GP-copula and the dummy multivariate model for the exchange-rate dataset.

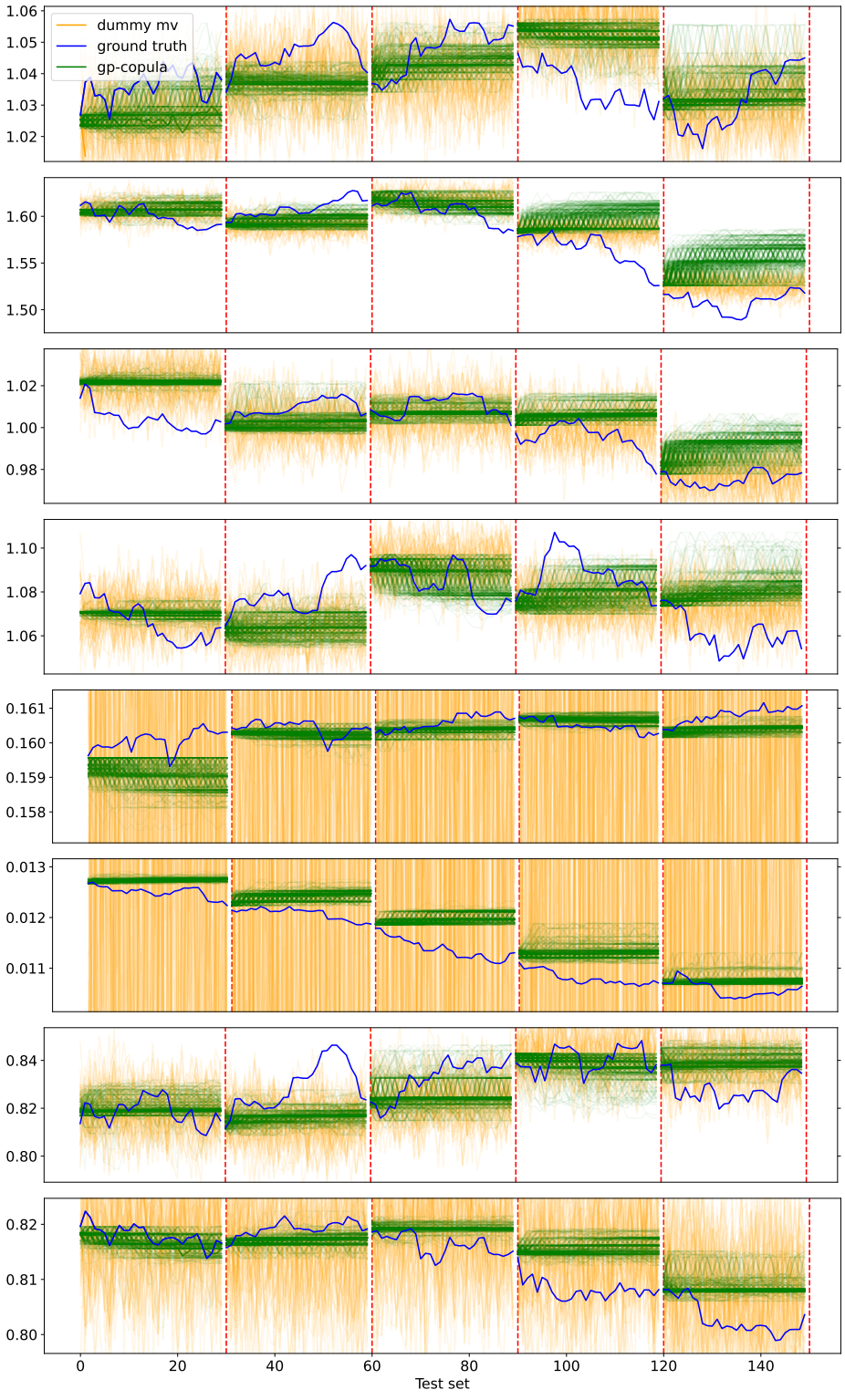


Fig. 4.5 This figure presents the sample forecasts from GP-copula for the exchange-rate dataset test set. The dataset has eight dimensions, and the test set consists of five batches with 30 timesteps. Each subfigure corresponds to one of the data dimensions, presented in original order from top to bottom. We used 400 samples for the visualization of each forecast batch.

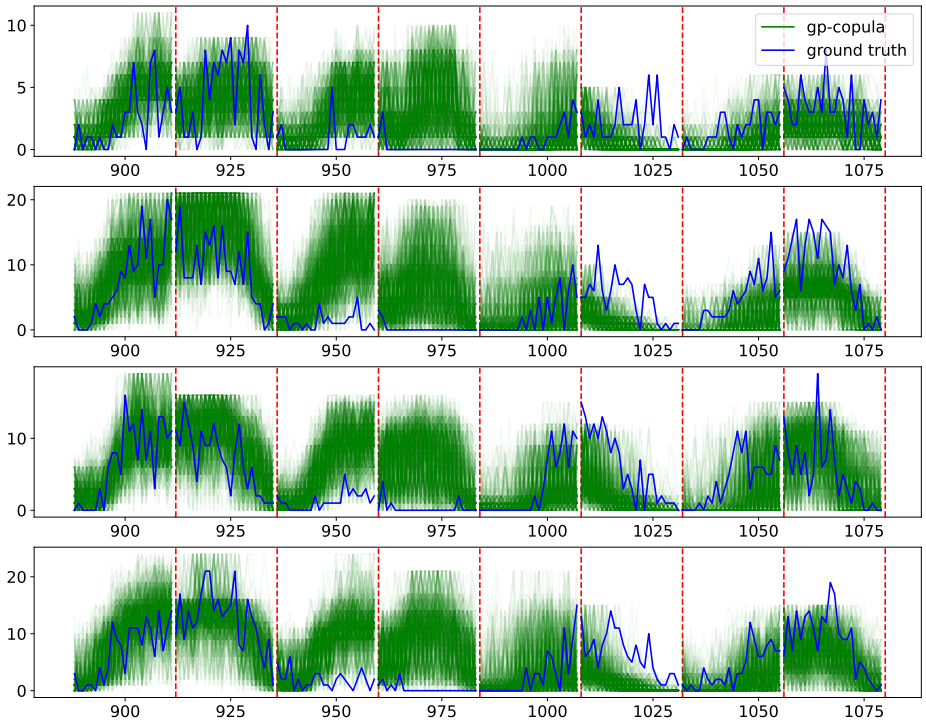
This experiment shows us that the border between a dummy and a genuine model can become blurry if we rely solely on CRPS-Sum. Furthermore, we learned that CRPS and visualization can help us better understand model performance.

In the second experiment, we performed a similar experiment on the taxi dataset [73]. The taxi dataset contains the spatio-temporal traffic time series of New York taxi rides taken at 1214 locations every 30 min in January 2015 (training set) and January 2016 (test set). This dataset consists of 1214 dimensions. Table 4.2 presents results for the experiment on the taxi dataset. In contrast to the previous experiment on the exchange-rate dataset, we cannot examine the discrimination ability of CRPS-Sum by comparing it to other metrics in higher dimensionalities. As mentioned in Section 4.3, the ES is not a reliable indicator of model performance in higher dimensionalities. CRPS cannot reflect the dependency structure learned by the model.

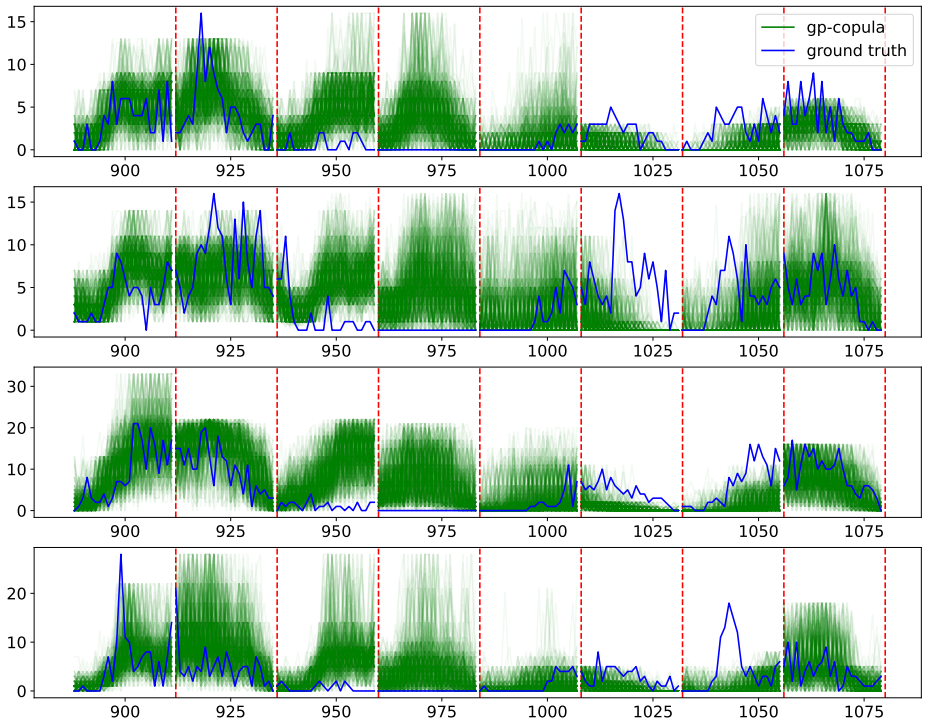
Furthermore, we cannot cross-check the CRPS-Sum discrimination ability with the qualitative performance of the model since it is impossible to investigate the model's performance intuitively due to the size of the data and the unintuitive nature of the time series data. For instance, Figure 4.6a,b illustrates the performance of GP-copula on the dimensions where the model has the best and the worst performance based on CRPS. By comparing these two figures, we can perceive clearly that the qualitative analysis of model performance is not feasible and straightforward. This experiment emphasizes again the importance of a strictly proper scoring rule for probabilistic multivariate time series forecasting, which is sound in its definition and analyzed carefully with real-world datasets with low dimensionalities.

Table 4.2 This table illustrates the results from dummy models on the taxi dataset and compares their performance with GP-copula based on CRPS-Sum, CRPS, and ES. In contrast to the exchange-rate dataset, it is not feasible to cross-check these quantities in higher dimensionalities.

	GP-copula	**Dummy Univariate**	**Dummy Multivariate**
CRPS-Sum	0.1703	0.4685	0.4705
CRPS	0.3336	0.6778	0.7543
ES	0.0138	0.0284	0.0318



(a) Best performance based on CRPS



(b) Worst performance based on CRPS

Fig. 4.6 These figures illustrate the 400 forecast samples from the GP-copula model on the taxi dataset. (**a**) Visualization of the GP-copula model on four dimensions of the taxi dataset with the best performance based on CRPS. (**b**) Visualization of the GP-copula model on four dimensions of the taxi dataset with the worst performance based on CRPS.

4.8 Final remarks on assessment of probabilistic forecasting models

This section reviewed various existing methods for assessing probabilistic forecast models and discussed their advantages and disadvantages. While CRPS only applies to univariate models and ES suffers from the curse of dimensionality, CRPS-Sum was introduced to help us assess multivariate probabilistic forecast models. Unlike CRPS and ES, the properties of CRPS-Sum have not been studied. Our sensitivity study illustrates that the CRPS-Sum behavior is not symmetric concerning the covariance of data distribution. CRPS-Sum is more sensitive to changes in the covariance of the model when the covariance of the data is negative. This is an undesirable property and makes result interpretation difficult.

Furthermore, CRPS-Sum cannot reflect the performance of a model on each dimension due to the loss of information caused by summation during its calculation. W demonstrated this problem with simple examples and experiments on the exchange-rate dataset, where a dummy model based on random noise achieved better CRPS-Sum than the state-of-the-art model. Additionally, with the experiment on the taxi dataset, we portrayed that the study of the CRPS-Sum discrimination ability in higher dimensionalities is not feasible.

To conclude, CRPS-Sum cannot provide an unbiased and accurate assessment for multivariate probabilistic forecasters. Thus, we suggest avoiding CRPS-Sum if possible. For data with low dimensionality, we can use ES. Assessing the probabilistic forecast model in higher dimensions is still an open problem. In the current state, relying solely on any existing metric is complicated, and manual qualitative analysis should also be used to evaluate the performance.

Chapter 5

Univariate One-step-ahead Forecasting

This chapter introduces the fundamental principles of utilizing implicit generative models, more specifically Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs), for probabilistic forecasting, which serves as the main contribution of this thesis. Following these principles, we present several models that apply these principles to perform one-step-ahead probabilistic forecasting. Furthermore, we empirically evaluated the potential of the proposed models by conducting several experiments on different datasets. Multiple investigations are conducted to explore the capabilities of these models thoroughly. The results obtained from these experiments are analyzed and discussed in detail. Through these experiments and discussions, we aim to demonstrate the effectiveness and performance of the proposed model in generating probabilistic forecasts. This empirical exploration provides insights into the strengths and limitations of the proposed models, offering valuable knowledge for further research and practical applications in the field of probabilistic forecasting.

5.1 Probabilistic Forecasting with GANs - ForGAN

This thesis proposes to utilize conditional GANs to achieve this goal. The core idea is to train a GAN to generate samples from predictive distribution while conditioning the generative model on historical information. To address this objective, two key challenges need to be overcome. First, we need to integrate the history window effectively in GAN architecture. Second, design an architecture that transforms noise distribution into predictive distribution accurately. Figure 5.1 illustrates the data flow of the *ForGAN* pipeline, the proposed solution

The content of this chapter has been adopted from A. Koochali et al. 'Probabilistic Forecasting of Sensory Data with Generative Adversarial Networks - ForGAN' IEEE Access, vol. 7, pp. 63868–63880, 2019, doi: 10.1109/ACCESS.2019.2915544.

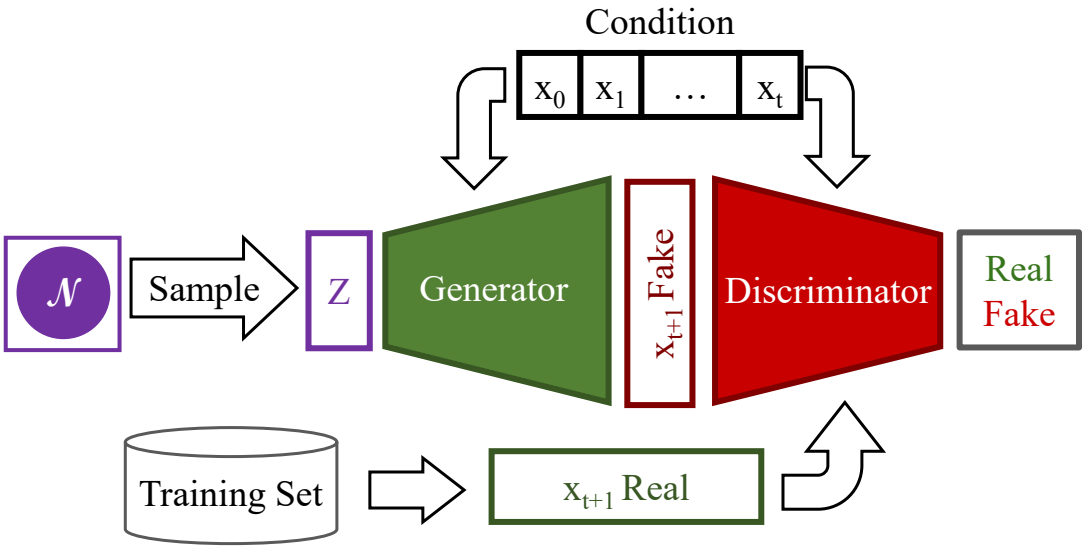


Fig. 5.1 Overview of ForGAN pipeline

to address these challenges. Figure 5.2 depicts the architecture of ForGAN in detail.

The generator distribution is characterized by $P_G(x_{t+1:t+h}|x_{0:t}, \Phi)$, where Φ represents the parameters of the generator. The generator consists of two functions. The first function h_G maps the historical window into a latent space to provide its representation:

$$\mathbf{h}_{0:t} = h_G(x_{0:t}). \tag{5.1}$$

The second function m_G transforms a sample from a standard Gaussian distribution space ($z \sim \mathcal{N}(0,1)$) to a sample scenario in predictive distribution, taking into account the historical window information via its representation:

$$\hat{x}_{t+i} = m_G(\mathbf{h}_{0:t}, z_i, k_i). \tag{5.2}$$

Here, i indicates a time step in the future ($0 < i \leq h$), and k is an optional argument that contains extra information related to the forecast at time step i, such as the previous model forecast for time steps before i ($0 < j < i$). In this thesis, functions h_G and m_G are approximated using neural networks. Various architectures are explored and discussed in this chapter and the subsequent chapters, considering the specific requirements and characteristics of the forecasting task.

The discriminator's primary goal is to determine the validity of a given forecast for the

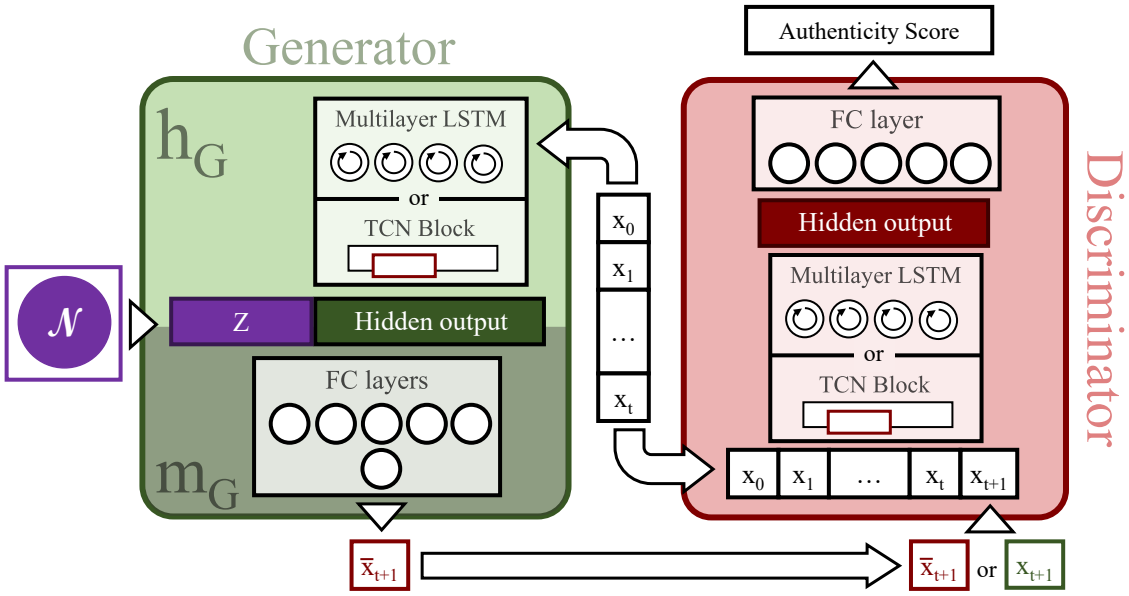


Fig. 5.2 The structure of ForGAN in detail

historical window. To achieve this, the discriminator takes both the historical window and the forecast as input and produces a score that indicates the forecast's validity:

$$score = D(\hat{x}_{t+1:t+h}, x_{0:t}). \tag{5.3}$$

ForGAN uses a consistent pipeline for the discriminator across different forecasting tasks and generator architectures. This pipeline involves concatenating the historical window and the forecast to create a complete input window $x_{0:t+h}$. The entire input window is then mapped to its representation using a neural network, followed by passing the representation through a fully connected layer to generate the score. In ForGAN, different architectures, such as RNN-based and CNN-based models, are experimented with to obtain the input representation in the discriminator.

The ForGAN model is trained using LSGAN [8] objective function. In an iteration of the training of ForGAN, we trained the generator on one batch and the discriminator on two batches.

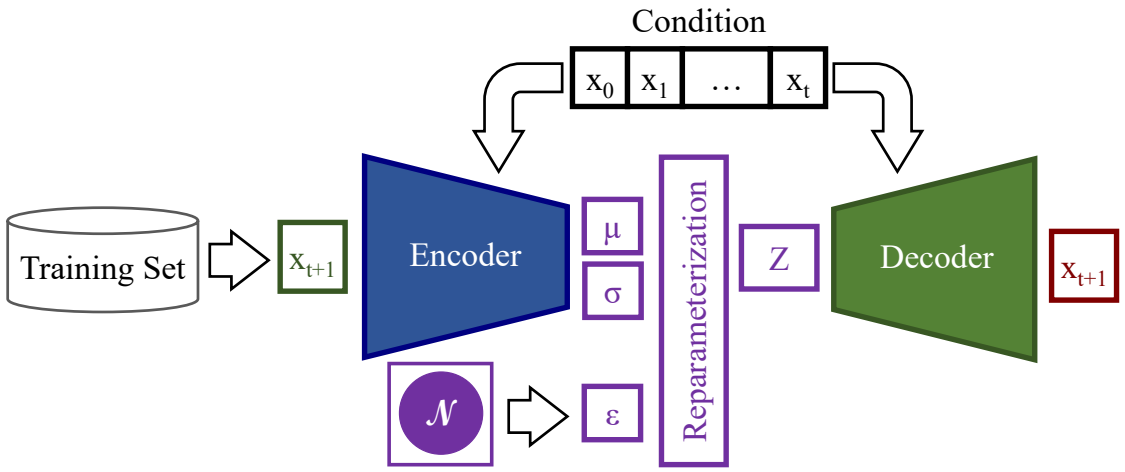


Fig. 5.3 Schematic representation of the VAEneu architecture.

5.2 Probabilistic Forecasting with Variational Autoencoders - VAEneu

Drawing inspiration from ForGAN, we explore the viability of leveraging Conditional VAEs (CVAEs) as potent probabilistic forecasting tools. However, a direct extrapolation of conventional CVAEs for this purpose is untenable, primarily due to the inherent incompatibility of VAEs' reconstruction loss for probabilistic forecasting. Typically, this loss, when applied to unbounded continuous variables, is an error metric (like mean squared error (MSE) or mean absolute error (MAE)) that measures the distance between the actual and reconstructed data points. By merely minimizing this distance, the model converges to a single point in the distribution, where it minimizes the selected error function. For instance, in the case of MSE, the model converges to the mean of the distribution, and in the case of MAE, the model converges to the target distribution's median. Therefore, the model is unable to learn the predictive distribution genuinely. Thus, to facilitate VAE's role as a probabilistic forecaster, a new loss function catering to predictive distribution learning is necessary.

5.2.1 CRPS Estimation as Reconstruction Loss

CRPS, a strictly proper scoring rule, is an apt scoring rule for assessing univariate probabilistic forecasting models. As articulated in Equation(4.7), given a target value x and two independent sample sets from the predictive distribution, $\hat{X}$ and $\hat{X}'$, CRPS can be computed as:

$$\mathrm{CRPS}(F,x) = \mathrm{E}_F\left|\hat{X} - x\right| - \frac{1}{2}\mathrm{E}_F\left|\hat{X} - \hat{X}'\right|. \tag{5.4}$$

The accuracy of CRPS hinges on the sample sizes of $\hat{X}$ and $\hat{X}'$. The inherent probabilistic nature of VAE's encoder offers a novel avenue to employ CRPS estimation as a reconstruction loss, empowering the model to discern the predictive distribution. During VAE training, we can sample multiple forecasting instances from the probabilistic latent space for every input instance, thereby optimizing Equation(5.4) as the reconstruction loss.

Figure 5.3 depicts the data flow of *VAEneu*, while figure 5.4 elucidates the intricacies of VAEneu's architecture in depth. The probabilistic encoder in a CVAE seeks to approximate the latent variable z posterior distribution, given a target forecast $x_{t+1:t+h}$ and the condition $x_{0:t}$. The encoder distribution is represented as $P_E(z|x_{0:t+h};\theta)$, where θ encompasses the encoder's parameters. Typically, P_E is a Gaussian distribution parameterized by mean $\mu(x_{0:t+h};\theta)$ and variance $\sigma^2(x_{0:t+h};\theta)$.

The reparameterization trick allows the derivation of the latent variable z using μ and σ^2. However, to harness the CRPS loss, each input condition window ($x_{0:t}$) requires multiple forecast samples ($\hat{X}_{t+1:t+h}$). This mandates sampling multiple latent variables Z from the probabilistic latent space for each input condition window. To do so, During reparameterization, we sample multiple ε samples from a standard normal distribution for each μ and σ^2. Consequently, every condition window is accompanied by an array of latent variables Z, which the decoder then translates into forecasts($\hat{X}_{t+1:t+h}$).

The decoder's distribution is represented by $P_D(x_{t+1:t+h}|x_{0:t},\Psi)$, where Ψ signifies the decoder's parameters. The generator comprises two functional components. The first, h_D, maps the historical window to the latent space, denoting its representation:

$$\mathbf{h}_{0:t} = h_D(x_{0:t}). \tag{5.5}$$

The second function, m_D, transforms the encoder's latent variable Z into sample scenarios in the predictive distribution, incorporating historical window information via its representation:

$$\hat{S}_{t+i} = m_D(\mathbf{h}_{0:t}, z_i, k_i). \tag{5.6}$$

In the above equation, i indexes a future timestep ($0 < i \leq h$), and k is an optional variable containing auxiliary information related to the forecast at timestep i. Functions h_D and m_D are approximated using neural networks. In subsequent sections, we further delve into

various h_D architectures and their comparative efficacies.

A unique advantage for VAEneu stems from its probabilistic encoder, which facilitates multiple forecast samples for every input condition window. This differentiates it from the Adversarial Autoencoder (AAE), as the latter's pipeline is entirely deterministic, despite both models inheriting from the autoencoder-based implicit generative model paradigm.

5.2.2 Structural Parallels between ForGAN and VAEneu

During inference, ForGAN's generator functions as the probabilistic forecaster. Similarly, post-training, VAEneu's decoder assumes this role. A structural juxtaposition of ForGAN's generator and VAEneu's decoder (as evident in figures 5.2 and 5.4 respectively) reveals striking similarities. This posits that ForGAN and VAEneu essentially offer divergent training methodologies for an analogous probabilistic forecasting neural architecture. Further, the similarities aren't confined to the probabilistic forecaster. The VAEneu encoder remarkably resembles ForGAN's discriminator, except for the final linear layer. Therefore, the distinction between ForGAN and VAEneu resides predominantly in their distinct data flows and objective functions. When appraising model capacity, they stand identical. This parallelism provides a conducive platform to evaluate and compare these models' prowess empirically. By employing identical hyperparameters for VAEneu and ForGAN, we can precisely assess the relative merits of each training paradigm.

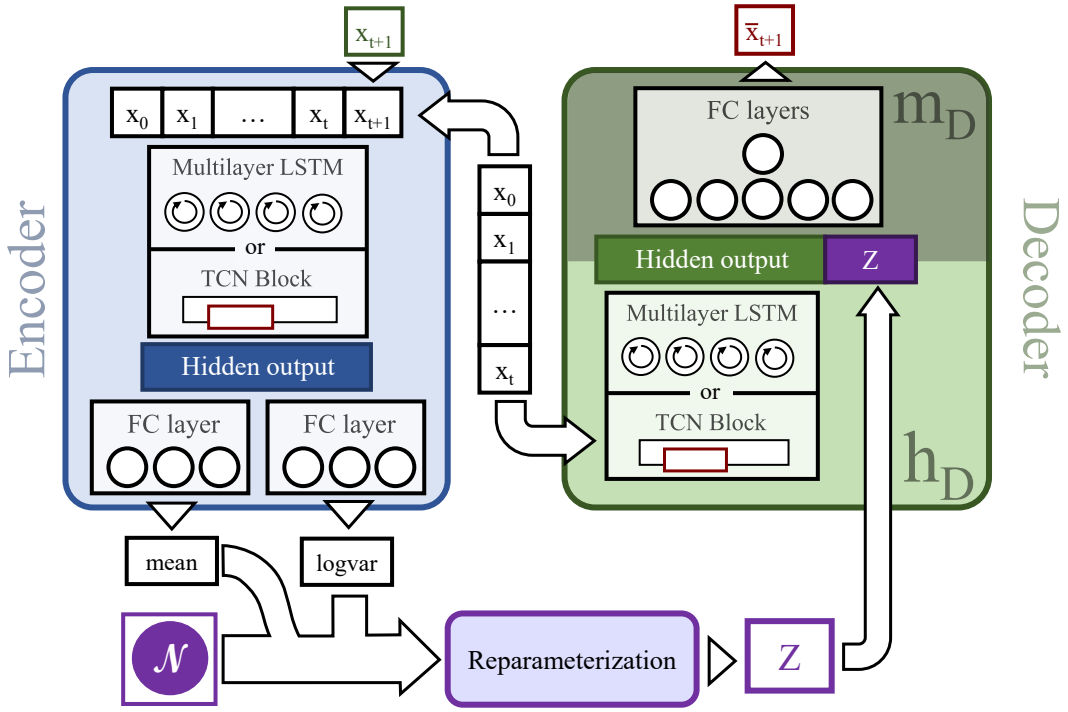


Fig. 5.4 The structure of VAEneu in detail

5.3 CRPS-ForGAN: Improved ForGAN training with CRPS loss

Inspired by the CRPS loss suggested by VAEneu, we introduce CRPS-ForGAN. This ForGAN-based model enjoys using the CRPS loss function as the secondary loss for the generator to improve training stability and model accuracy.

The architecture of the CRPS-ForGAN remains consistent with that of the original ForGAN. The generator's input comprises a condition window of historical data and a random noise vector. However, instead of generating a singular forecast scenario, the generator in CRPS-ForGAN, aided by multiple noise vectors, generates multiple forecast scenarios for each input condition window. This capacity for multiple outputs is pivotal for utilizing the CRPS loss function.

CRPS-ForGAN's training regimen is a fusion of the adversarial training mechanism inherent to GANs and the CRPS-guided loss employed by VAEneu.

For every training iteration:

1. *Adversarial Training:* The discriminator and generator are trained in tandem, adhering to the standard adversarial training dynamics. The generator tries to produce realistic forecasts, while the discriminator distinguishes between the true and generated forecasts.

2. *CRPS-guided Generator Training:* Post the adversarial training, the generator undergoes another training iteration, this time guided by the CRPS loss (Equation(5.4)). The generator, leveraging the multiple noise vectors, produces an array of forecast scenarios for every condition window. The CRPS loss is then calculated between these generated forecasts and the true future values, pushing the generator towards producing forecasts that better approximate the predictive distribution.

5.3.1 Benefits of the CRPS-guided Training

Integrating the CRPS loss into the ForGAN training pipeline offers several advantages:

- **Enhanced Training Stability:** The adversarial game intrinsic to GANs can often lead to unstable training dynamics. By introducing a secondary CRPS-guided training step, the generator receives more consistent and structured feedback on its forecasts, fostering more stable and efficient training.

- **Holistic Distribution Learning:** While the adversarial training process inclines the generator towards producing realistic forecasts, it doesn't necessarily guide it to capture the entirety of the predictive distribution. The CRPS loss fills this gap by ensuring the generator does not just focus on specific modes of distribution but encompasses its full extent.

- **Mitigation of Mode Collapse:** A notorious issue with GAN training is mode collapse, where the generator produces a limited variety of outputs. The CRPS loss, by emphasizing the entire predictive distribution, reduces the likelihood of this phenomenon.

5.4 Probabilistic Forecasting for Univariate Time Series Using One-step-ahead Predictions

This section delves into developing and assessing probabilistic forecasters designed specifically for univariate time series data, focusing on one-step-ahead predictions. Leveraging the

foundational principles behind the ForGAN and VAEneu architectures, we aspire to design a predictive model capable of approximating the target distribution, denoted as $p(x_{t+1}|x_{0:t})$.

Within the scope of our investigation, Recurrent Neural Networks (RNNs) and Temporal Convolutional Networks (TCNs) are explored as core components to devise these probabilistic forecasters. We derive six distinct models by coupling these networks with the architectures mentioned above. These are: ForGAN-RNN, ForGAN-TCN, CRPS-ForGAN-RNN, CRPS-ForGAN-TCN, VAEneu-RNN, and VAEneu-TCN.

Most hyperparameters are either fixed throughout all experiments or are a function of Input Window Size (IWS). IWS is the time series length fed to the TCN or RNN block. For the Decoder and Generator, the IWS equals the history window size. For the Encoder and Discriminator, the IWL is equal to the summation of the history window size and horizon.

5.4.1 Hyperparameter Configuration

We have enumerated the hyperparameters associated with each model and their respective values for comprehensive transparency and replicability. These details are succinctly captured in tables 5.1. The hyperparameters adopted in our experiments remain invariant across different datasets or depend on the Input Window Size (IWS). The IWS defines the extent of the time series provided to the TCN or RNN block. The IWS is set to be identical to the history window size for both the decoder and the Generator components. In the case of the Encoder and Discriminator components, IWS is computed as the cumulative sum of the history window size and the forecasting horizon.

5.4.2 Datasets

This section introduces an array of datasets utilized to evaluate ForGAN and VAEneu performance nuances under different data conditions. Out of the 12 datasets deployed, one is a custom toy dataset—explicitly devised as a part of this study, making it a noteworthy contribution to this work. The remaining 11 datasets are sourced from public repositories, each offering distinct challenges and insights due to their inherent data characteristics. Such a diverse selection enables a comprehensive analysis of the ForGAN and VAEneu models, ensuring their evaluation under varied conditions.

Table 5.2 encapsulates the salient features of these datasets for quick reference. A standard data partitioning scheme has been adopted for experiments involving public datasets:

Table 5.1 List of hyperparameters for proposed models and their corresponding employed values in experiments.

Hyperparameter	Value
RNN Block	
Cell type	LSTM
Number of layers	1
Hidden size	$\frac{IWS}{2}$
Noise size	$\frac{IWS}{2}$
TCN Block	
Kernel size	5
Number of layers	Log($\frac{IWS}{8}$)
Hidden size	2×Log(IWS)
Noise size	2×Log(IWS)

80% of the data is earmarked for training, leaving the residual 20% for model validation and final performance measurement. A visual representation of each dataset can be found in Figure 5.6, showcasing a segment from each to provide an overview of the diverse dynamics of these datasets.

Lorenz Dataset

The Lorenz dataset is crafted to assess the proficiency of probabilistic forecasters. It encapsulates complex time window clusters generated from the renowned Lorenz system. These equations describing atmospheric convection are pivotal in understanding chaotic behavior in physical systems.

Mathematical Formulation The Lorenz system embodies a set of coupled differential equations, representing atmospheric convection (x), horizontal temperature variation (y), and vertical temperature (z) as functions of time (t). Defined as:

$$\begin{aligned} \dot{x} &= \sigma(y-x), \\ \dot{y} &= x(\rho-z), \\ \dot{z} &= yx-\beta z, \end{aligned} \tag{5.7}$$

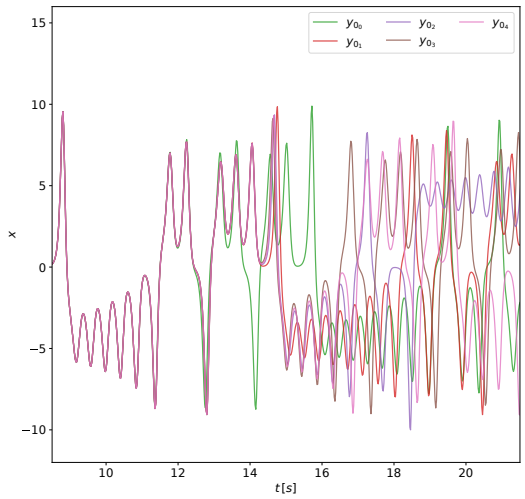
Table 5.2 The properties of univariate datasets utilized for one-step-ahead forecasting experiments

Name	Domain	Frequency	Length	History Window Size
Lorenz dataset	Artificial	-	-	24
Gold Price Dataset	Economic	Daily	2487	60
HPEC Dataset	Energy	Hourly	34569	48
Internet Traffic A1H Dataset	Web	Hourly	1231	48
Internet Traffic A5M Dataset	Web	5 minutes	14772	24
Internet Traffic B1H Dataset	Web	Hourly	1657	48
Internet Traffic B5M Dataset	Web	5 minutes	19888	24
Mackey Glass Dataset	Physics	Seconds	20000	120
Saugeen River Flow Dataset	Nature	Daily	23741	60
Sunspot Dataset	Nature	Daily	73924	60
US Births Dataset	Nature	Daily	7305	60
Solar Dataset	Energy	Hourly	8219	48
Wind Dataset	Energy	Hourly	8219	48

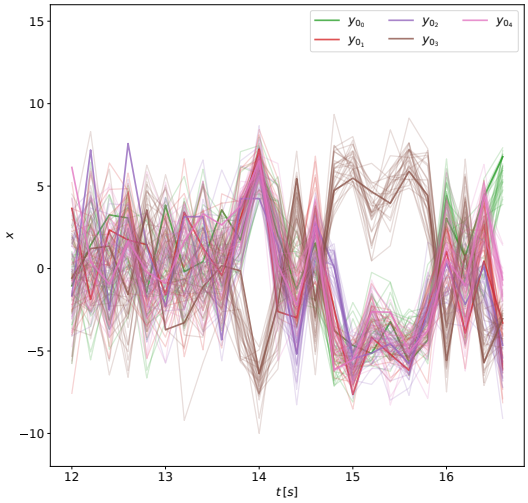
where σ is proportional to the Prandtl number [74], ρ is proportional to the Rayleigh number [75] and β is connected to physical dimensions of the atmospheric layer of interest [76]. One of the most interesting features of the Lorenz equations is the emergence of chaotic behavior [77, 76] for certain choices of the parameters σ, ρ, and β. In the following we fix $\sigma = 16$, $\rho = 45.92$, and $\beta = 4$.

Dataset Creation The dataset's generation involved initiating with $x_0 = 1$, $z_0 = 1$, and five y_0 values, serving as cluster seeds (Table 5.3). Utilizing these seeds, 100,000 data samples spanning 26 seconds (with a 0.02s resolution) were generated (Figure 5.5a). A Gaussian noise ($\mu = 0, \sigma = 7.2$) was introduced to induce distinct time windows, yet ensuring intra-cluster resemblance. The bifurcation region (12-17 seconds) was designated as the condition time window for training (Figure 5.5b).

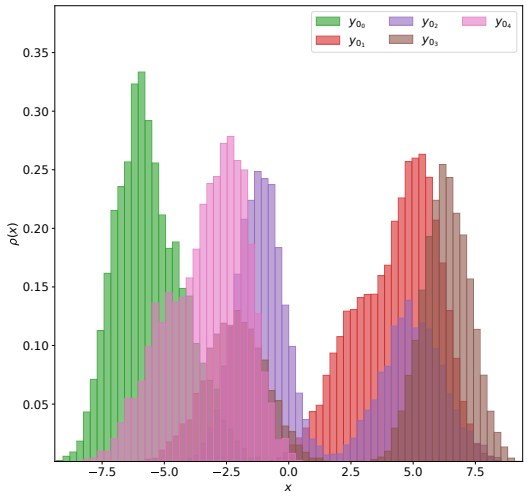
Target Formulation Target values, x_{t+1}, were sampled from intervals $t \in (20, 22, 25)$, shaping the probability distributions evident in Figure 5.5c. Figure 5.5d elucidates the cumulative probability distribution for the entire dataset.



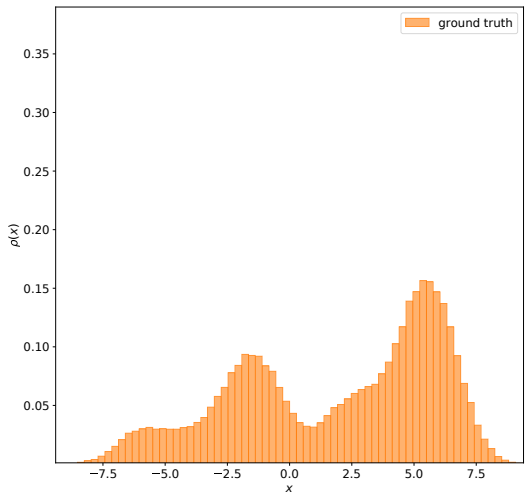
(a) Solution to the Lorenz system for different initial values y_0.



(b) The bifurcation region after the data augmentation steps described in the text



(c) Possible values x_{t+1} distinguished by initial value y_0



(d) Full probability distribution of x_{t+1}

Fig. 5.5 Visualizing the pipeline of Lorenz dataset creation

Table 5.3 Initial values y_0 and relative composition of the Lorenz dataset.

Index	y_0	**Relative Occurrence**
0	1.0001	5.5 %
1	1.000001	22 %
2	1.00000001	42 %
3	1.0000000001	24 %
4	1.000000000001	6.5 %

Gold Price Dataset

Gold is a vital commodity with a significant history of global trading. The Gold Price dataset provides daily pricing details, encompassing 2,487 data points from September 1st, 2010, to March 13th, 2020. This dataset facilitates analytical and predictive modeling of the gold market due to its comprehensive coverage during the specified period.

Household Electric Power Consumption Dataset

The Household Electric Power Consumption (HEPC) dataset [78] comprises electric power consumption readings from a singular household with a sampling rate of one minute. From December 16th, 2006, to November 26th, 2010, it encapsulates nearly four years of data. Notably, there are timestamps with missing values. An imputation approach, where missing values are replaced with the average of their adjacent observations, is adopted to ensure data integrity. Moreover, the dataset is resampled to aggregate the readings into hourly intervals, ensuring manageability while preserving inherent patterns.

Internet Traffic Datasets

Internet traffic datasets [79] offer data from two distinct ISPs, named A and B. The A dataset pertains to a private ISP with nodes across 11 European cities, capturing data on a transatlantic link from June 7the to July 29the, 2005. The B dataset is sourced from UKERNA1 and aggregates traffic across the UK's academic network from November 19th, 2004, to January 27th, 2005. The A dataset logs every 30 seconds, whereas B records every 5 minutes. Aggregated variants, A5M, A1H, B5M, and B1H, offer data at 5-minute and 1-hour resolutions respectively.

Accessible in www.kaggle.com/datasets/arashnic/learn-timeseries-forecasting-from-gold-price

Macky-Glass Dataset

The time-delay differential equation proposed by Mackey and Glass [80] stands as a benchmark for generating chaotic time series. The equation is given by:

$$\dot{x} = \frac{ax(t-\tau)}{(1+10\cdot(t-\tau))-bx(t)}\,. \tag{5.8}$$

Adhering to the parameters in [81], where $a = 0.1$, $b = 0.2$, and $\tau = 17$, a dataset of length 20,000 is generated using Equation (5.8.

Saugeen River Flow Dataset

This dataset [82] includes a univariate time series showcasing the daily mean flow of the Saugeen River at Walkerton measured in cubic meters per second ($\frac{m^3}{s}$). It captures data from January 1st, 1915, to December 31st, 1979. The dataset is archived in the Monash Time Series Forecasting Archive [83].

Solar-4-seconds and Wind-4-seconds datasets

These datasets capture solar and wind power production every 4 seconds from August 1st, 2019, for approximately one year. For the purpose of this research, the datasets are aggregated to represent hourly resolutions. Sourced from the Australian Energy Market Operator (AEMO) online platform, they are also part of the Monash Time Series Forecasting Archive [83].

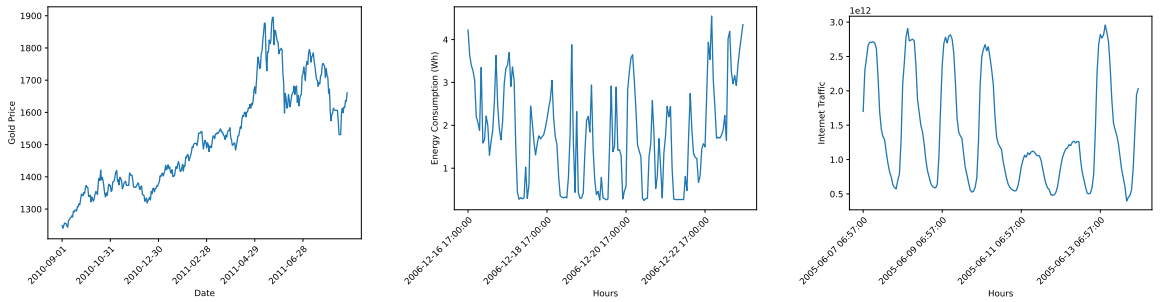
Sunspot Dataset

The dataset contains a time series depicting daily sunspot counts from January 8th, 1818, to May 31st, 2020. Any missing values within the dataset are addressed using the Last Observation Carried Forward (LOCF) method. The Solar Influences Data Analysis Center is the primary data source, with the dataset also being part of the Monash Time Series Forecasting Archive [83].

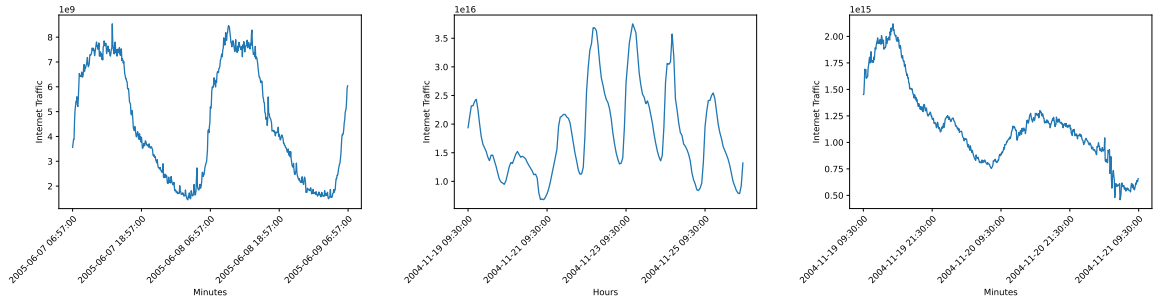
US Birth Dataset

This dataset illustrates the number of births in the US from January 1st, 1969, to December 31st, 1988. Originally sourced from the R mosaicData package [84], it's accessed for this research via the Monash Time Series Forecasting Archive [83].

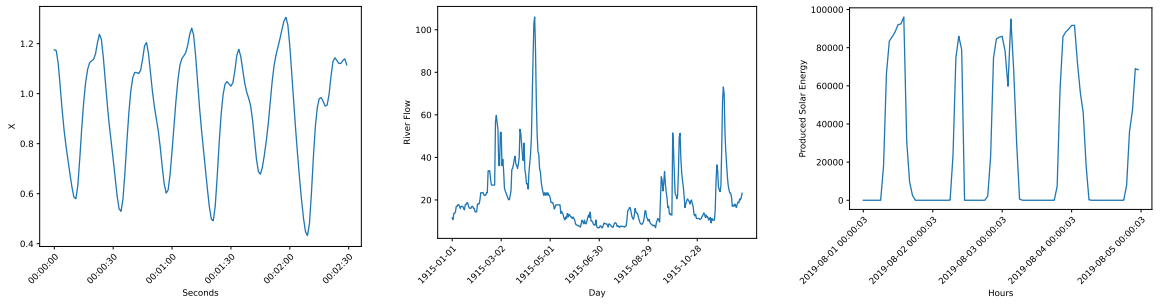
The dataset can be accessed at https://git.opendfki.de/koochali/forgan



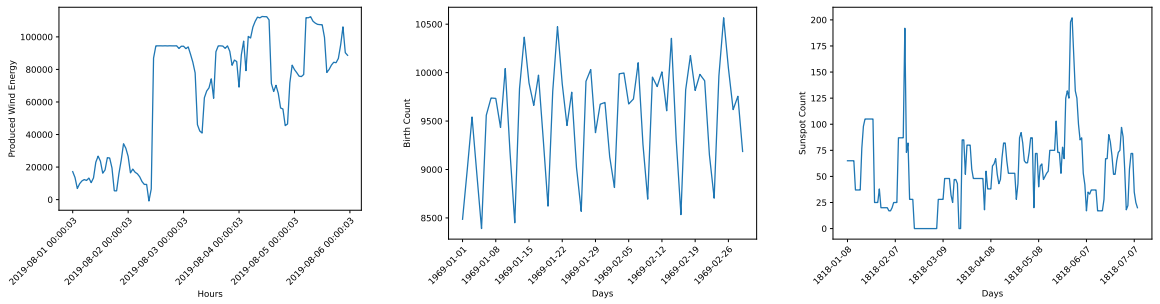
(a) Gold Price Dataset (b) HEPC Dataset (c) Internet Traffic A1H Dataset



(d) Internet Traffic A1M Dataset (e) Internet Traffic B1H Dataset (f) Internet Traffic B5M Dataset



(g) Mackey Glass Dataset (h) Saugeenday River Dataset (i) Solar-4-seconds Dataset



(j) Wind-4-seconds Dataset (k) US Births Dataset (l) Sunspot Dataset

Fig. 5.6 Illustration of a slice of each univariate time series dataset utilized in this thesis for one-step-ahead forecasting

5.4.3 Experimental Configuration

Implementation Details: The models detailed in this study have been implemented leveraging the Pytorch framework [85]. The training process was conducted on a dedicated workstation equipped with an Intel(R) Xeon(R) CPU E5-1650 v4 processor, a memory capacity of 64 GB RAM, and a Geforce RTX 4080 GPU.

Optimizer and Learning Rate Schedule: The RMSProp optimizer [86] was employed for the optimization of all models. An adaptive learning rate strategy was adopted, commencing with a rate of 0.01. The learning rate was systematically halved in cases where no discernible improvement in the validation set error was observed over a span of 500 consecutive batches.

Batch Configuration and Early Stopping: A consistent batch size of 256 samples was used across all models to ensure uniformity in the training regimen. To promote computational efficiency, an early stopping mechanism was integrated. Specifically, while the models were set to train for a maximum of 100,000 batches, the training would be preemptively halted if the validation error remained stagnant for an extended sequence of 10,000 steps.

5.5 Experiments and Results

Within this section, we meticulously analyze the empirical outcomes derived from our experimental evaluation. We begin our exposition with the results of the Lorenz dataset. A unique facet of this dataset is our perception of its underlying generative mechanism. This empowers us with the capability to engage in a detailed and comprehensive qualitative analysis of the results. In view of its distinct nature and the depth of discussion it warrants, we have dedicated an isolated section to dissect the outcomes from this dataset. Subsequent to the Lorenz dataset, we cast our focus on the remaining datasets. For each dataset, we provide a holistic portrayal of the models' performance, followed by elaborated analytical insights. The CRPS serves as our primary metric of choice for assessing the predictive accuracy of the models. To circumvent any anomalies and ensure robustness, each model is subjected to three independent training iterations on every dataset. The reported performance metric is an arithmetic mean of these three runs, which is indicated as $\overline{CRPS}$, appended with the standard deviation to shed light on the stability and reproducibility of each technique.

Table 5.4 This Table presents the quantitative results of our proposed models alongside DeepAR as the baseline model. The bold CRPS number indicates the best-performing result considering $\overline{CRPS}$.

Models	CRPS
ForGAN-RNN	1.518 ± 0.011
ForGAN-TCN	1.518 ± 0.027
CRPS-ForGAN-RNN	1.488 ± 0.004
CRPS-ForGAN-TCN	1.485 ± 0.005
VAEneu-RNN	1.484 ± 0.001
VAEneu-TCN	$\mathbf{1.482 \pm 0.001}$
DeepAR	2.604 ± 0.200

5.5.1 Lorenz Dataset

Table 5.4 showcases the empirical outcomes of our investigations on this dataset. As evident, VAEneu models exhibit superior performance. The ForGAN models, while trailing the VAEneu marginally, display an improved performance upon the incorporation of CRPS loss, aligning their accuracy closely with the VAEneu models. Notably, the standard deviation for the ForGAN models underscores their relative instability as opposed to their VAEneu counterparts. Yet, with the inclusion of the CRPS loss in the ForGAN training regime, we witness a significant uptick in their stability. This provides a strong testament to the potency of incorporating the CRPS loss in enhancing both the accuracy and stability of the ForGAN model. On the other end of the spectrum, the DeepAR model manifested suboptimal performance, lagging behind all other models. This is due to the fact that the assumption of the DeepAR model on the normality of predictive distribution does not hold on some of the condition clusters.

Given our knowledge of the true generative process underpinning the dataset, we can approximate the empirical PDF of the predictive distribution of the probabilistic forecasting model through model samples, comparing them with the authentic distribution. Figures 5.7 and 5.8 represent the qualitative insights for condition clusters y_{0_3} and y_{0_4}, respectively. The distribution of condition cluster y_{0_4} (Figure 5.8) resonates closely with a Gaussian distribution. Consequently, DeepAR, predicated on the Gaussian assumption, renders accurate forecasts for this cluster. Conversely, the distribution for cluster condition y_{0_3} (Figure 5.7) evokes a mixture of two Gaussian distributions. This divergence from DeepAR's foundational assumption results in its inability to accurately model the y_{0_3} cluster accurately. This serves as a compelling demonstration of the pivotal role of preliminary knowledge on genuine

data distribution when leveraging explicit models. Absent this, they risk underperformance and potentially misleading outcomes. Examining the qualitative attributes of the proposed models, we discern their precision in modeling these distributions, echoing the quantitative metrics.

5.5.2 Public Dataset

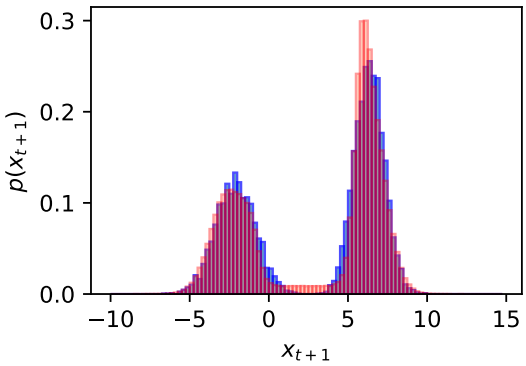
In Table 5.5, we provide a comprehensive evaluation of our experiments conducted on 12 univariate datasets. VAEneu models are found to be particularly competitive, leading in performance on 10 out of the 12 datasets. In the remaining two datasets, the VAEneu models are only marginally outperformed by their closest competitors.

The results indicate that the TCN adaptations of the ForGAN models generally lead to performance improvements across most datasets. An interesting observation is the neck-and-neck performance of VAEneu-RNN and VAEneu-TCN, indicating that the choice between recurrent and temporal convolutional architectures might be data-dependent.

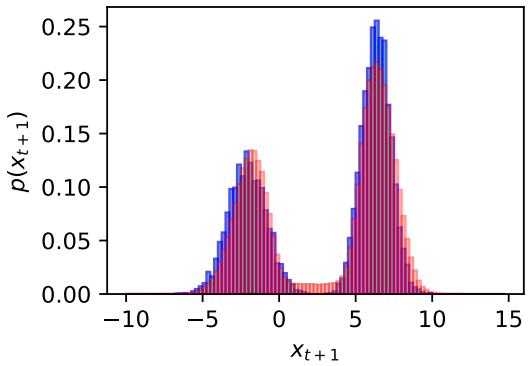
In line with our observations from the Lorenz dataset experiments, the CRPS-ForGAN models demonstrate enhanced stability during the training phase and improved precision in forecasts. Specifically, the CRPS-ForGAN-TCN model emerges as the most effective among the ForGAN-based architectures.

The DeepAR model displays varied results across different datasets. For instance, while it exhibits exemplary performance on the US births dataset, its performance is subpar on several other datasets. This inconsistency underscores the significance of selecting an appropriate predictive distribution family in models that make explicit distributional assumptions. Utilizing a fixed predictive distribution, such as the Gaussian in these experiments, can yield excellent results when it aligns with the underlying data distribution but can falter when there's a mismatch.

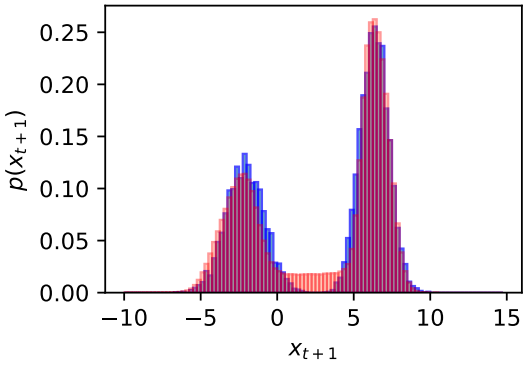
Upon a meticulous examination of Table 5.5, it becomes evident that the performances of the evaluated models exhibit minimal deviation, particularly when accounting for variations over three independent runs. Such close proximities in performance render it challenging to assertively claim the superiority of one model over the others on a given dataset. The expansive array of datasets and models incorporated in our study intensifies this challenge, complicating the task of drawing overarching conclusions on the models' efficacy across all datasets.



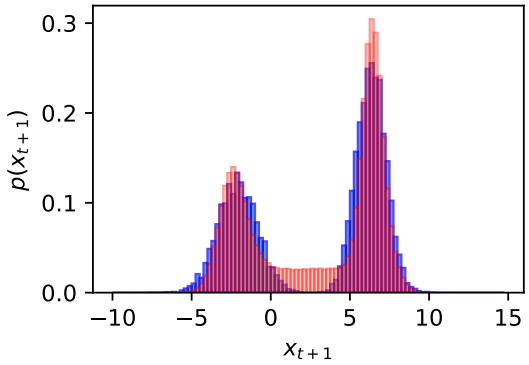
(a) ForGAN RNN



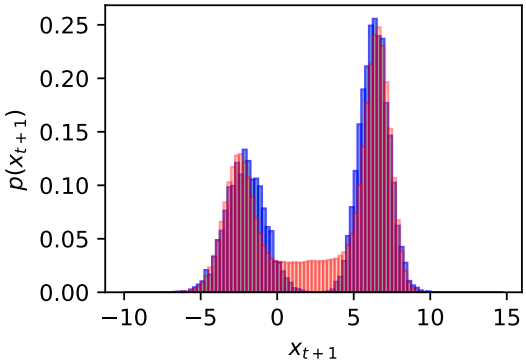
(b) ForGAN TCN



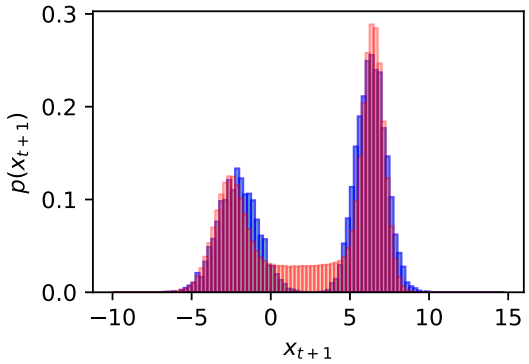
(c) CRPS-ForGAN RNN



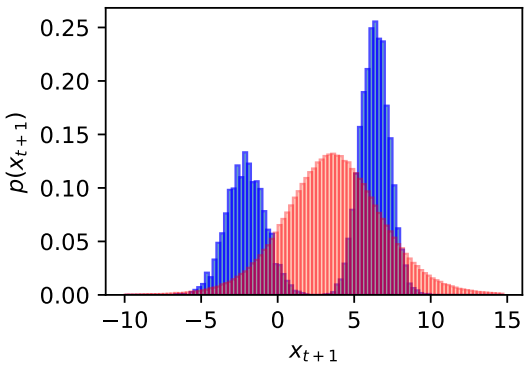
(d) CRPS-ForGAN TCN



(e) VAEneu RNN

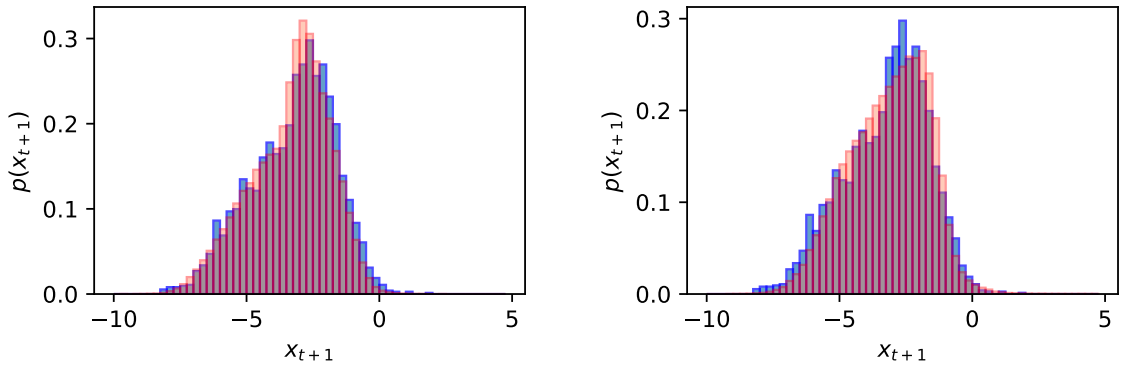


(f) VAEneu TCN



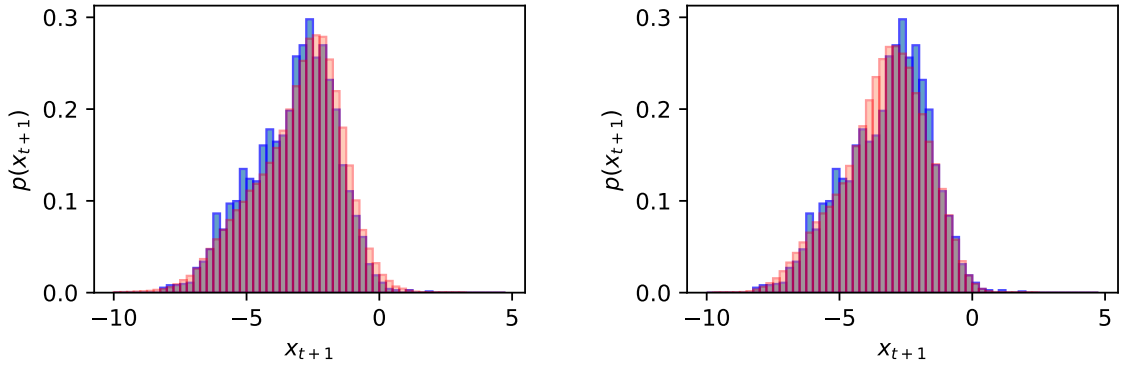
(g) DeepAR

Fig. 5.7 Qualitative results of the experiment on the Lorenz Dataset for the condition cluster y_{0_3}. The blue histogram indicates the true distribution, and the red histogram represents the predictive distribution.



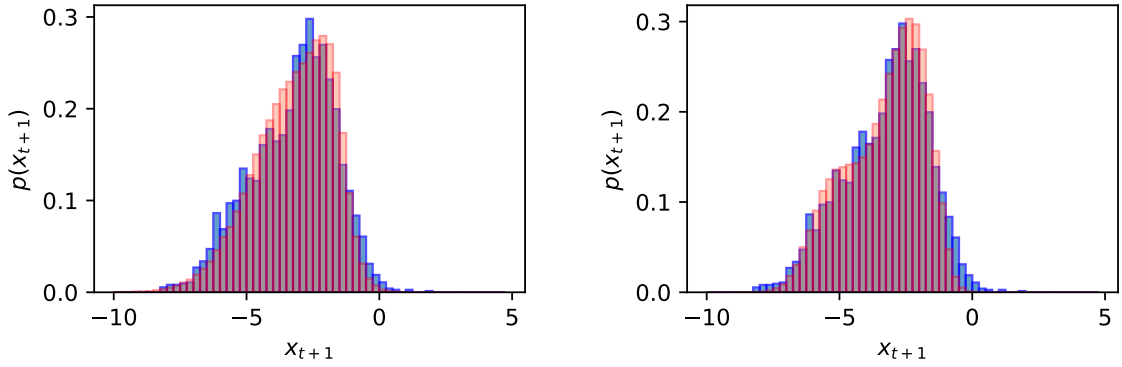
(a) ForGAN RNN

(b) ForGAN TCN



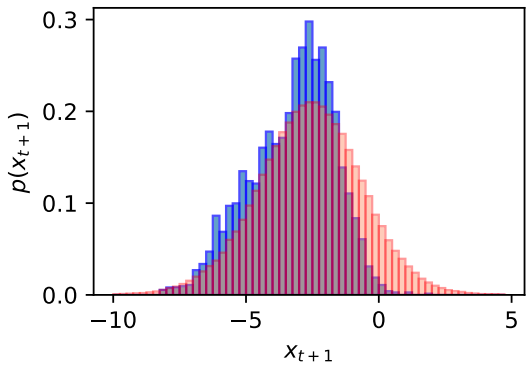
(c) CRPS-ForGAN RNN

(d) CRPS-ForGAN TCN



(e) VAEneu RNN

(f) VAEneu TCN



(g) DeepAR

Fig. 5.8 Qualitative results of the experiment on the Lorenz Dataset for the condition cluster y_{0_4}. The blue histogram indicates the true distribution, and the red histogram represents the predictive distribution.

To streamline the presentation and facilitate a clearer understanding of the results, we leverage the Critical Difference (CD) diagram. This diagram serves as an efficacious visual instrument designed explicitly to compare the performance of multiple algorithms across an array of datasets. Its primary utility lies in determining statistically significant distinctions in the rank orders of different models.

The horizontal axis of the diagram enumerates the models or their respective identifiers, arranged based on their mean rankings. A model with superior performance, indicated by a lower average rank, occupies a position further to the right. A bar denoting the critical difference is drawn above the ranked algorithms. Should the gap between the average rankings of two distinct algorithms fall beneath the critical difference, their performance difference is deemed statistically insignificant, adhering to a predetermined confidence level.

For the computation of the critical difference depicted in our diagram, we rely on the outcomes from three statistical tests. This study restricts statistical significance at the 0.05 threshold level ($\alpha = 0.05$).

Table 5.5 Quantitative results of our proposed models benchmarked against DeepAR on 12 public datasets. The cell colors indicate the divergence of the model performance from the best-performing model on that dataset. The legend on the bottom right corner provides information on cell color.

	ForGAN-RNN	ForGAN-TCN	CRPS-ForGAN-RNN	CRPS-ForGAN-TCN
Gold price	5.288E+00 ± 9.527E-02	5.387E+00 ± 1.223E-01	5.514E+00 ± 2.499E-01	5.257E+00 ± 3.891E-02
HEPC	2.695E-01 ± 8.000E-03	2.411E-01 ± 1.104E-03	2.433E-01 ± 2.173E-03	**2.337E-01 ± 1.344E-03**
Internet traffic A1H	3.601E+10 ± 1.418E+09	3.629E+10 ± 7.822E+08	3.217E+10 ± 1.066E+09	3.361E+10 ± 5.605E+08
Internet traffic A5M	7.019E+07 ± 1.774E+06	6.567E+07 ± 1.829E+06	6.531E+07 ± 7.108E+05	6.496E+07 ± 1.549E+06
Internet traffic B1H	3.649E+14 ± 4.107E+13	3.624E+14 ± 4.780E+13	2.726E+14 ± 9.490E+12	2.640E+14 ± 1.984E+13
Internet traffic B5M	1.314E+13 ± 3.743E+11	1.225E+13 ± 1.152E+11	1.212E+13 ± 2.161E+11	1.204E+13 ± 1.550E+11
Mackey glass	6.992E-04 ± 6.518E-04	6.165E-04 ± 2.900E-04	7.502E-04 ± 1.605E-04	3.167E-04 ± 6.638E-05
Saugeen river	3.057E+00 ± 4.696E-02	3.434E+00 ± 6.072E-01	2.947E+00 ± 4.619E-02	2.863E+00 ± 4.618E-02
Solar 4 seconds	4.013E+03 ± 9.245E+02	3.884E+03 ± 5.195E+02	3.073E+03 ± 8.419E+01	2.841E+03 ± 4.156E+01
Sunspot	8.162E+00 ± 1.495E-01	7.970E+00 ± 2.475E-02	8.029E+00 ± 1.122E-01	7.681E+00 ± 3.304E-02
US births	2.125E+02 ± 3.639E+01	1.813E+02 ± 3.175E+00	2.055E+02 ± 9.947E+00	1.839E+02 ± 3.198E+00
Wind 4 seconds	4.520E+03 ± 5.910E+01	4.356E+03 ± 3.044E+01	4.435E+03 ± 4.476E+01	4.237E+03 ± 3.146E+01

	VAEneu-RNN	VAEneu-TCN	DeepAR	Color Legend
Gold price	**5.173E+00 ± 1.305E-02**	5.369E+00 ± 4.771E-02	7.450E+00 ± 1.235E+00	**Best Model**
HEPC	2.351E-01 ± 1.457E-03	2.346E-01 ± 7.066E-04	2.348E-01 ± 3.272E-03	$\leq 10\%$
Internet traffic A1H	**2.774E+10 ± 9.519E+08**	2.946E+10 ± 5.720E+08	5.219E+10 ± 2.420E+09	$\leq 50\%$
Internet traffic A5M	**6.316E+07 ± 1.920E+05**	6.327E+07 ± 3.119E+05	6.936E+07 ± 4.681E+05	$\leq 100\%$
Internet traffic B1H	2.377E+14 ± 6.827E+12	**2.317E+14 ± 5.933E+12**	4.312E+14 ± 7.133E+13	$> 100\%$
Internet traffic B5M	1.211E+13 ± 7.052E+10	**1.188E+13 ± 2.019E+10**	1.326E+13 ± 1.127E+12	
Mackey glass	**1.385E-04 ± 3.092E-05**	2.115E-04 ± 2.479E-05	4.820E-03 ± 3.013E-04	
Saugeen river	**2.805E+00 ± 1.769E-02**	2.866E+00 ± 2.816E-03	3.698E+00 ± 4.304E-02	
Solar 4 seconds	2.738E+03 ± 4.673E+01	**2.680E+03 ± 4.295E+01**	2.962E+03 ± 5.721E+01	
Sunspot	7.723E+00 ± 3.712E-03	**7.655E+00 ± 9.443E-03**	7.715E+00 ± 4.368E-02	
US births	1.827E+02 ± 1.266E+01	1.658E+02 ± 3.560E+00	**1.554E+02 ± 3.463E+00**	
Wind 4 seconds	**4.190E+03 ± 4.044E+01**	4.301E+03 ± 3.825E+01	6.375E+03 ± 4.631E+02	

Friedman test: The initial analytical step entails the application of the Friedman test, aimed at discerning any significant differences among the algorithms spanning multiple datasets. The null hypothesis, denoted as $\boldsymbol{H_0}$, posits the equivalence in performance across all tested models. Contrarily, the alternative hypothesis, $\boldsymbol{H_1}$, suggests a differential performance exhibited by at least one algorithm relative to the others. The rejection of $\boldsymbol{H_0}$ in favor of $\boldsymbol{H_1}$ signifies the potential utility of the CD diagram, thereby certifying the progression to subsequent statistical tests.

Wilcoxon signed-rank test: For pairwise comparisons of model performances across all datasets, the study employs the Wilcoxon signed-rank test. This non-parametric test facilitates the comparison of two paired samples without assuming that the differences between said pairs adhere to a normal distribution. Given two models, A and B, with their performances represented as sets of $\overline{CRPS}$ values spanning the datasets, the null hypothesis, $\boldsymbol{H_0}$, asserts a median difference of zero between paired $\overline{CRPS}$ values, thereby suggesting similar performance between Models A and B across all dataset. The outcome, be it acceptance or rejection of $\boldsymbol{H_0}$, is utilized for defining the critical difference band within the CD diagram.

Paired t-test: Finally, the paired t-test is applied to rank model performances within specific datasets. A paired t-test is used to compare the means of two related groups when the samples are dependent; that is, each observation in one sample can be paired with an observation in the other sample. Within the scope of model performance evaluated over multiple runs, these dependent observations represent the performances of two distinct models assessed on an identical dataset across separate runs. Let $\overline{CRPS}_A$ symbolize the mean CRPS of model A, computed over three runs on a specific dataset. If the inequality $\overline{CRPS}_A < \overline{CRPS}_B$ holds for a dataset, then the null hypothesis $\boldsymbol{H_0}$ of the one-sided paired t-test posits a superior or equivalent performance by Model B relative to Model A across all runs. The alternative hypothesis, $\boldsymbol{H_1}$, counters this by indicating a statistically superior performance by Model A. We sequentially applied this test to every possible model pair within a dataset to determine their ranks. In scenarios where the null hypothesis faced rejection, Model A was accorded a superior rank. Conversely, an acceptance led to the assignment of identical ranks to both Model A and Model B. Subsequent to the ranking procedure, the models' average ranks across all datasets were computed to establish their respective positions on the CD diagram.

Figure 5.9 outlines the CD diagram, encapsulating the main findings from our experimental results and offering a condensed perspective of Table 5.5. It is evident that the models

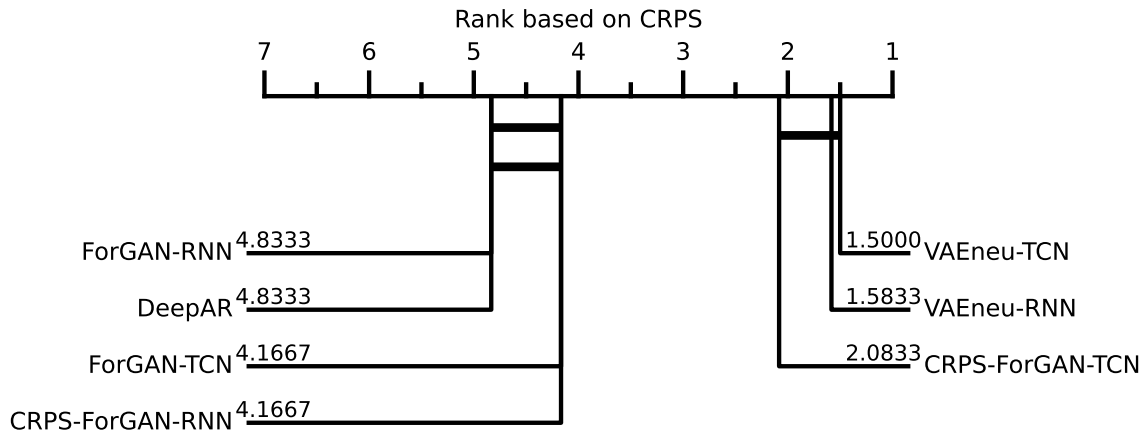


Fig. 5.9 The critical difference diagram provides a holistic overview of the examined models' comparative performances across the entire dataset.

separate into two discernible clusters. The first cluster encompasses the VAEneu-based models and the CRPS-ForGAN-TCN, establishing themselves as the top-performing models within this investigation. Despite VAEneu-TCN securing a marginally superior rank on an average scale, the overall performance of these three models remains largely identical across all examined datasets. This observation reinforces our preceding inference, where the VAEneu-based models consistently showcased superior efficacy in contrast to their counterparts. Additionally, within the ForGAN-based model variants, CRPS-ForGAN-TCN emerged as the most proficient.

While the CD diagram provides an aggregate portrayal of model performance across datasets, it omits certain granular details pivotal to determining the elaborateness of their efficiencies. Specifically, this diagram illuminates the ranking of models and denotes whether performance differences are statistically significant. However, it falls short of conveying the quantitative extent of performance divergence in terms of CRPS when one model markedly outperforms another. To bridge this informational gap, we employed the relative score concept. The relative CRPS, denoted as Δ_{CRPS} is mathematically formulated as follows:

$$\Delta_{CRPS} = \frac{\overline{CRPS} - \overline{CRPS}^*}{\overline{CRPS}^*}, \tag{5.9}$$

wherein $\overline{CRPS}^*$ represents the $\overline{CRPS}$ of the top-performing model for a given dataset. In essence, Δ_{CRPS} quantifies the deviation of each model's performance from that dataset's best-performing model.

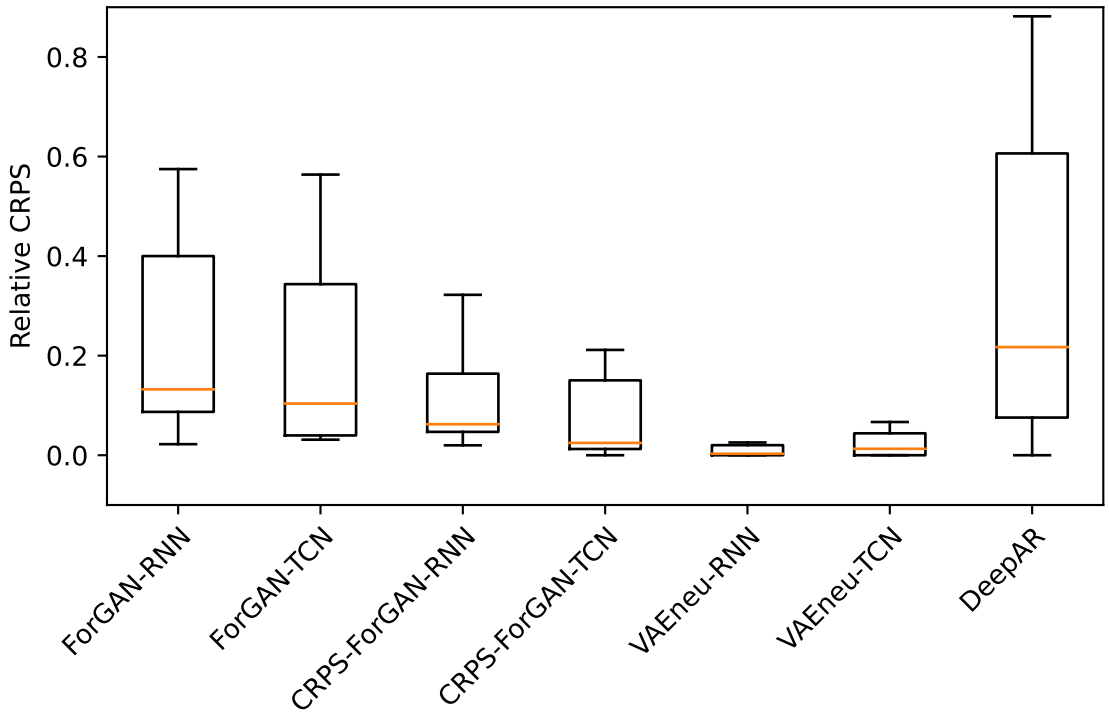


Fig. 5.10 Distribution of the relative CRPS scores of each model over all datasets, encapsulated in a box-plot representation.

Figure 5.10 shows the distribution of Δ_{CRPS} across datasets for each model. This visualization yields intriguing insights: notably, DeepAR manifests the most noticeable variability in its performance. Subsequently, the ForGAN model variants display similar Δ_{CRPS} distributions to DeepAR with slightly smaller variance. An observable enhancement in performance and robustness is evident upon the integration of the CRPS loss into the ForGAN models. In stark contrast, the VAEneu models consistently deliver prime performances across all datasets. Based on these empirical findings, it is secure to conclude that the VAEneu models clinch the title of the best-performing models for one-step-ahead forecasting in univariate datasets.

5.5.3 Analysis of Model Convergence

The speed at which a machine learning model converges to optimal parameters during training is a pivotal factor that impacts its efficiency and applicability, particularly for extensive datasets. Convergence can be measured in terms of computational time (wall clock) and the number of required training steps. Given that the models under consideration possess

comparable parameter counts and were trained under analogous hardware configurations and experiment conditions, a fair basis for performance comparison is established.

Training Time Analysis

Figure 5.11 presents the distribution of the duration each model necessitates to complete a single iteration of training i.e. training step, expressed in milliseconds across all datasets. A noticeable observation from the figure indicates that the VAEneu models demonstrate superior speed compared to other model variants. Furthermore, TCN-based models, on average, exhibit a reduced speed relative to their RNN counterparts. This speed reduction in TCN variants can be attributed to our choice of a fixed kernel size of 5. This decision was motivated to facilitate the processing of input in finer granularities, enhancing the model's capability to discern intricate patterns. Nevertheless, this entails incrementing the network's depth to ensure the TCN model's receptive field adequately encompasses the complete input window, leading to an increased parameter count. Alternative configurations with larger kernels may permit shallower networks, though they could potentially compromise model efficacy. A configuration with even finer kernels might achieve enhanced precision, but this might come at the expense of extended processing durations per training step. The augmented computational overhead observed in ForGAN models with CRPS loss is anticipated due to the auxiliary execution of the generator for CRPS loss computation. Intriguingly, ForGAN models tend to be more time-intensive than their VAEneu counterparts despite comparable parameter numbers. This disparity arises from the ForGAN training approach, where the discriminator is trained on two batches in each step, whereas all VAEneu components are trained on a single batch during a training step.

Convergence Steps Analysis

Besides the computational time, the number of steps a model necessitates to converge toward the optimal parameters is an essential metric. Figure 5.12 portrays the distribution of requisite convergence steps across datasets for each model. A key insight drawn from this figure underscores the relatively swift convergence of ForGAN models, despite often to sub-optimal parameter sets. Conversely, the integration of CRPS loss causes ForGAN models to exhibit variable convergence steps depending upon the dataset. Overall, the addition of CRPS loss enables models to stagnate at GAN's objective function saddle points, facilitating prolonged training and culminating in a more refined parameter set. A comparative study between the VAEneu models reveals that the VAEneu-TCN converges more rapidly than its VAEneu-RNN counterpart, counterbalancing its slower training step processing time.

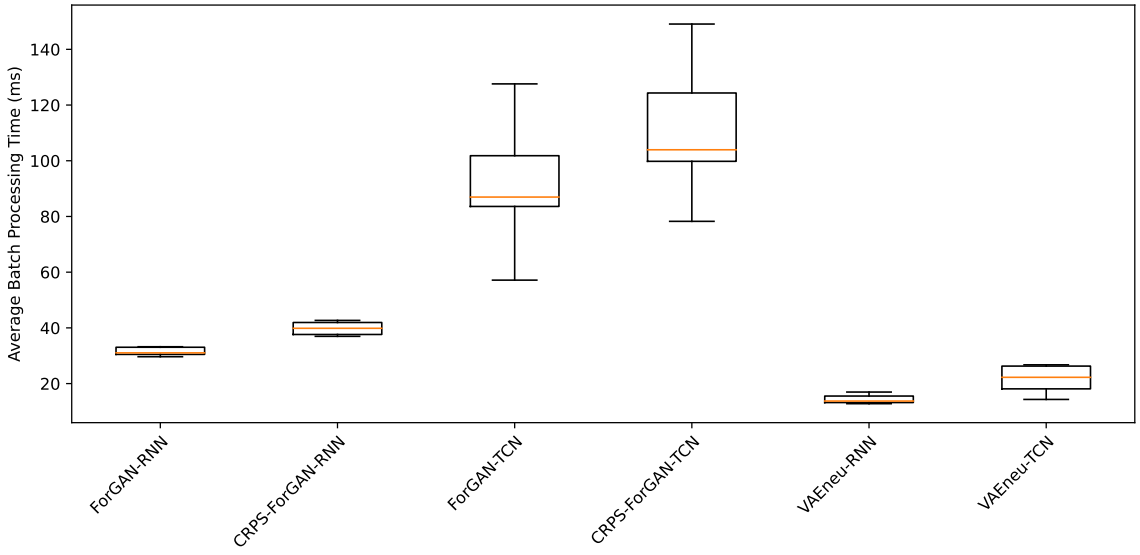


Fig. 5.11 Average batch processing time of each model during training, illustrated across all datasets.

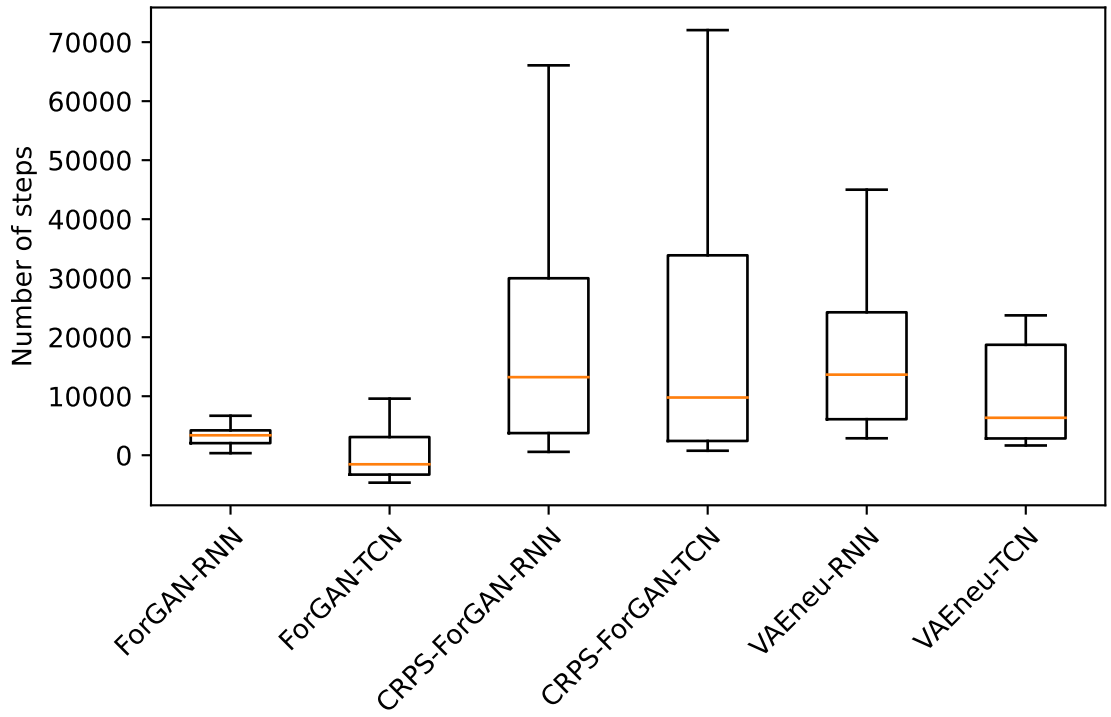


Fig. 5.12 Box-plot representation of models' convergence steps across multiple executions on various datasets.

5.5.4 Analyzing the Impact of the Repeat Factor on Training

Incorporated alongside the CRPS loss, the *repeat factor* designates the number of samples for each input used to approximate the CRPS during training. A larger repeat factor is hypothesized to enhance the accuracy of CRPS estimation. This, in turn, should refine the quality of gradients obtained from the CRPS loss, facilitating the weight update process within the network. Nevertheless, the influence of the repeat factor on the model's training trajectory remains an area yet to be fully explored. In this section, we aim to clarify the consequence of varying repeat factor values on both the CRPS of the resultant optimal model and the average time taken to complete a training step, focusing specifically on models employing the CRPS loss.

To assess the interplay of different repeat factors, we trained models based on CRPS-ForGAN and VAEneu architectures across three distinct datasets. Repeat factors were chosen following the relation:

$$\text{Repeat factor} = 2^i, \text{where } i \in \{1, 2, \ldots, 7\}. \tag{5.10}$$

Figure 5.13 encapsulates our findings. An evident trend emerges: as the repeat factor escalates, models tend to converge to a configuration with a more favorable CRPS. This improvement, however, comes at the cost of an increased batch processing duration. The VAEneu-based model showcases this pattern more pronouncedly, given that its training exclusively relies on optimizing the CRPS loss. In juxtaposition, the CRPS-ForGAN model concurrently optimizes the CRPS loss (viewed as an auxiliary loss) and the primary GAN objective function. An interesting observation to note is that by choosing a repeat factor within the interval $[8, 64]$, one can achieve commendable CRPS values without significantly extending the training step processing time.

5.5.5 Analyzing the Accuracy of the CRPS Loss Approximation

In the initial exposition of the CRPS loss, it was posited that this loss provides an approximated calculation of CRPS based on a limited subset of samples. This section endeavors to probe the fidelity of the CRPS loss as an approximation to the true CRPS value. Specifically, for the sake of this analysis, we determined the CRPS over 100 samples from a holdout dataset at every 10th step during training, denoting this as *CRPS test*. This was compared with the *CRPS train* obtained from the CRPS loss.

Figure 5.14 sketches the evolution of both *CRPS train* and *CRPS test* for VAEneu-based and CRPS-ForGAN-based architectures across three different datasets. The graphic illu-

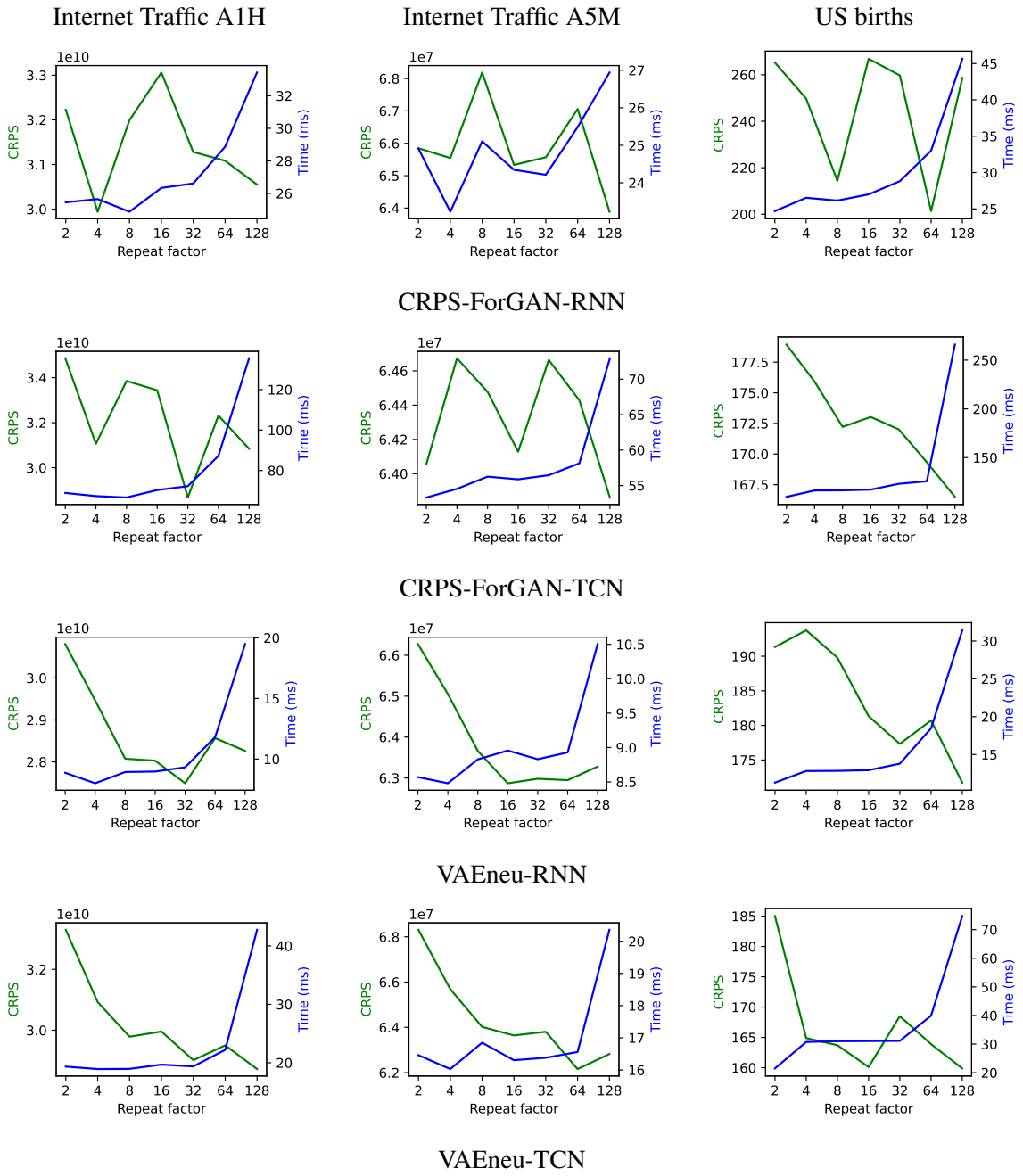


Fig. 5.13 The effect of repeat factor on CRPS and training step processing time on three datasets.

minates that *CRPS train* consistently yields values proximate to the *CRPS test* with high variation. Given that the CRPS loss is an estimate derived from merely 32 samples of a single batch, the closeness of the approximation is notably admirable. This observation provides a rationale for the efficacy of the CRPS loss in yielding top-tier models in our empirical studies.

In some instances, such as with the Sunspot dataset depicted in Figure 5.14, the *CRPS train* slightly overestimates the *CRPS test*. Conversely, for the Saugeen River dataset, the *CRPS test* appears to be almost central to the *CRPS train* approximations. However, a divergence is noted with the US birth dataset exclusively where the *CRPS train* undervalues the *CRPS test*. Intriguingly, it's this particular dataset where our proposed architectures were unable to surpass the forecasting accuracy of the DeepAR model, lagging by a noticeable margin.

These observations raise the prospect that an underestimation by the CRPS loss might signal sub-optimal model performance. Yet, given the occurrence of this phenomenon in just a singular dataset, it would be overhasty to establish this as a definitive correlation. Such an inference remains suspended as a potential avenue for future research explorations.

5.6 ForGAN Application in the Automotive Sector

The ForGAN architecture demonstrates significant utility when adapted to specific scenarios within the automotive industry, specifically in the context of internal combustion engines. One of the pivotal forecasting targets within this domain is the accurate prediction of the filling quantity within individual engine cylinders.

To facilitate this prediction, historical records of cylinder filling quantities are fed into the ForGAN model. Additionally, 12 auxiliary predictors are incorporated, each bearing a substantial correlation to the target variable. These predictors include but are not limited to, engine speed, intake pressure, camshaft positioning, throttle valve configurations, lambda values, and coolant temperature metrics.

For benchmarking purposes, the efficacy of ForGAN was juxtaposed against a contemporary CNN-based time series forecasting model as proposed in [87]. The comparative results, presented in Table 5.6, unequivocally illustrate the superior performance of the

This research's findings are protected under patent A. Koochali, et al., "Method and Device for the Probabilistic Prediction of Sensor Data." US20220187772A1, June 16, 2022. Access here

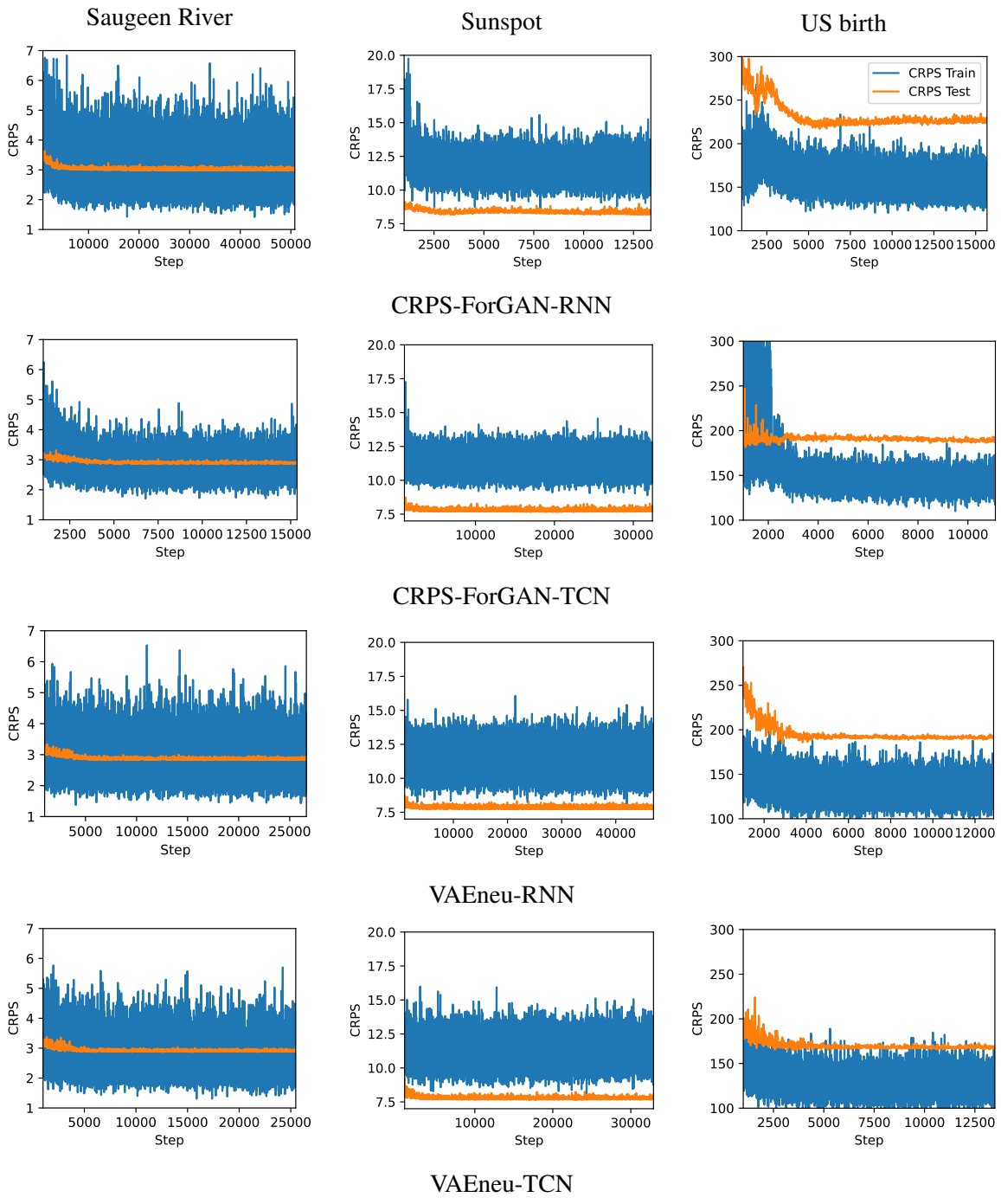


Fig. 5.14 Illustration of CRPS train and CRPS test on three datasets for models that use CRPS loss.

Table 5.6 Comparison of model performance (CRPS) on the cylinder filling prediction dataset.

Model	ForGAN	CNN-based Forecaster [87]
Predictive Accuracy (CRPS)	0.0081	0.0110

ForGAN model over its CNN-based counterpart.

The technical significance of the ForGAN method for predicting the filling quantity in a combustion engine cylinder is profound, touching upon various aspects of automotive engineering. Below, we reviewed some of the important impacts of this research.

Enhanced Precision in Predictive Control : The utilization of ForGAN is a departure from traditional deterministic methods, providing a more nuanced understanding of the system's behavior. It allows for the consideration of temporal dependencies and uncertainties in the sensor data, leading to predictions that are not only precise but also carry a measure of confidence.

Optimization of Engine Performance : By accurately predicting the filling quantity in the cylinders, the method enables engine performance optimization. This is crucial for achieving optimal power output, fuel efficiency, and emission control. The engine control unit (ECU) can utilize the predicted values to adjust parameters in real-time, ensuring that the engine operates within the desired performance envelope.

Reduction in Wear and Tear : Predictive control facilitated by accurate predictions can lead to a reduction in wear and tear of engine components. By preemptively adjusting engine parameters, the system can avoid operating conditions that are known to cause excessive stress on components, thereby extending the lifespan of the engine and reducing maintenance costs.

Improved Fuel Efficiency and Emission Control ForGAN's ability to provide accurate and probabilistic predictions directly contributes to improved fuel efficiency and emission control. By optimizing the filling quantity in the cylinders, the combustion process can be made more efficient, leading to better utilization of fuel and a reduction in harmful emissions. This is particularly significant in the context of rigorous environmental regulations and the automotive industry's ongoing efforts to reduce the carbon footprint of vehicles.

Chapter 6

Univariate Multi-step-ahead Forecasting

In the realm of probabilistic forecasting, the complexity of multi-step-ahead predictions magnifies the importance of probabilistic approaches. Unlike one-step-ahead forecasting, where the immediate future is predicted with relatively higher confidence, multi-step-ahead forecasting must account for compounded uncertainties that grow with each subsequent prediction horizon. As the forecast extends further into the future, the range of possible outcomes expands, making the task increasingly uncertain and complex. Probabilistic forecasting, in this context, becomes crucial as it provides a distribution of possible future values rather than a single-point estimate, thus offering a more comprehensive view of the future by encapsulating the inherent uncertainty.

In this chapter, we employed the Auto-Regression technique to extend 6 models proposed in the previous chapter to the multi-step-ahead forecasting models. Furthermore, we introduce 6 novel methods using sequence-to-sequence modeling designed for multi-step-ahead probabilistic forecasting. We conducted an extensive study of 12 proposed models against 11 established probabilistic forecasting models on 12 datasets. Finally, we studied various aspects of the proposed models to paint a full picture of the proposed models' competencies and shortcomings.

6.1 Extension to Multi-Step-Ahead Forecasting

6.1.1 Auto-Regressive strategy

The extension of our forecasting capabilities from a one-step-ahead prediction to a multi-step-ahead horizon necessitates a methodological shift. To this end, we integrate an Auto-Regressive (AR) strategy during the inference phase. This approach hinges on utilizing

the model's predictions as input for subsequent forecasts, thereby iteratively projecting further into the future. Such AR methods, while conceptually straightforward, often face with performance degradation as the forecasting horizon extends—primarily due to the propagation and accumulation of prediction errors.

To mitigate the inherent challenge of error accumulation in AR models, precision in the initial forecasts becomes crucial. The less erroneous the initial predictions, the lesser the impact of error propagation on long-range forecasts. In our investigation, we rigorously examine the Auto-Regressive variant of our models—denoted with the "AR" prefix—and analyze the influence of error compounding over extended horizons. This analysis not only provides insights into the robustness of our forecasting models but also contributes to the broader understanding of the reliability of AR techniques in probabilistic time series forecasting. The outcomes of this investigation are presented in a subsequent section of this chapter, offering a comprehensive evaluation of the models' performances across varying forecast lengths.

6.1.2 Attention-based Strategy

The advent of Sequence-to-Sequence (Seq2Seq) modeling [88], initially conceived for machine translation tasks, has significantly impacted the field of time series forecasting. This method marks a departure from traditional auto-regressive (AR) strategies, which typically generate future values through a recursive process, often leading to the compounding of errors. In contrast, Seq2Seq models adopt a holistic approach, encapsulating the entire input sequence to forecast the output sequence in a non-recursive manner. This structure inherently captures the full spectrum of temporal dependencies within the sequence, enhancing robustness in forecasting, especially over extended horizons.

Central to Seq2Seq modeling is the encoder-decoder architecture, as depicted in Figure 6.1. The encoder processes the input time series, condensing its salient features into a context vector. The decoder then uses this context vector to generate future values. This method's ability to handle variable-length input and output sequences has revolutionized time series forecasting. The Seq2Seq framework's adaptability has been proven in numerous forecasting scenarios, particularly where encoding intricate temporal dynamics is essential.

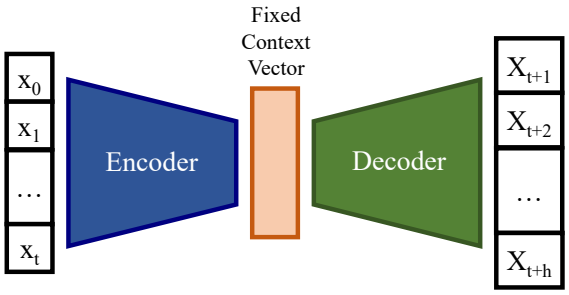


Fig. 6.1 The original architecture of a Seq2Seq model.

The integration of the attention mechanism into the Seq2Seq architecture, as illustrated in Figure 6.2, addresses a critical limitation of the original design: the constraint of a fixed-length context vector. The attention mechanism allows for dynamic weighting of different input sequence segments during prediction, creating a context vector that is continuously updated. This addition enriches the forecasting process with greater nuance and context sensitivity. In the realm of time series forecasting, attention-augmented Seq2Seq models excel in identifying subtle patterns and dependencies, key to accurate future predictions.

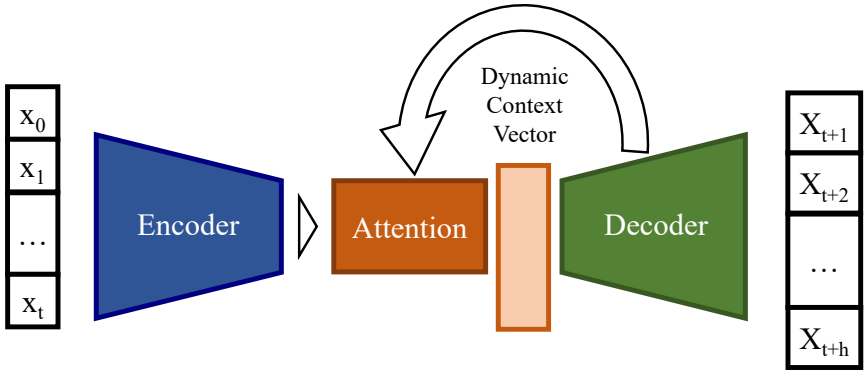


Fig. 6.2 Integration of the attention mechanism into the Seq2Seq model architecture.

To facilitate multi-step-ahead probabilistic forecasting within the ForGAN and VAEneu frameworks, we introduced a novel generator/decoder model integrating an Attention-based Seq2Seq structure. Figure 6.3 displays the ForGAN generator, incorporating this architecture. The noise vector, combined with the dynamic context vector, is fed into the decoder, which maps this composite vector to the predictive distribution, taking into account the evolving context. A similar approach is employed for the VAEneu's decoder model, with the noise vector derived from the reparameterization unit. We denote these Attention-based models with the prefix 'Ab-' and have developed six variants, namely AbForGAN-RNN, AbForGAN-TCN, CRPS-AbForGAN-RNN, CRPS-AbForGAN-TCN, AbVAEneu-RNN, and AbVAEneu-

TCN, with the suffix indicating the type of discriminator/encoder employed. Table 6.1 details the specific hyperparameters for these Attention-based models.

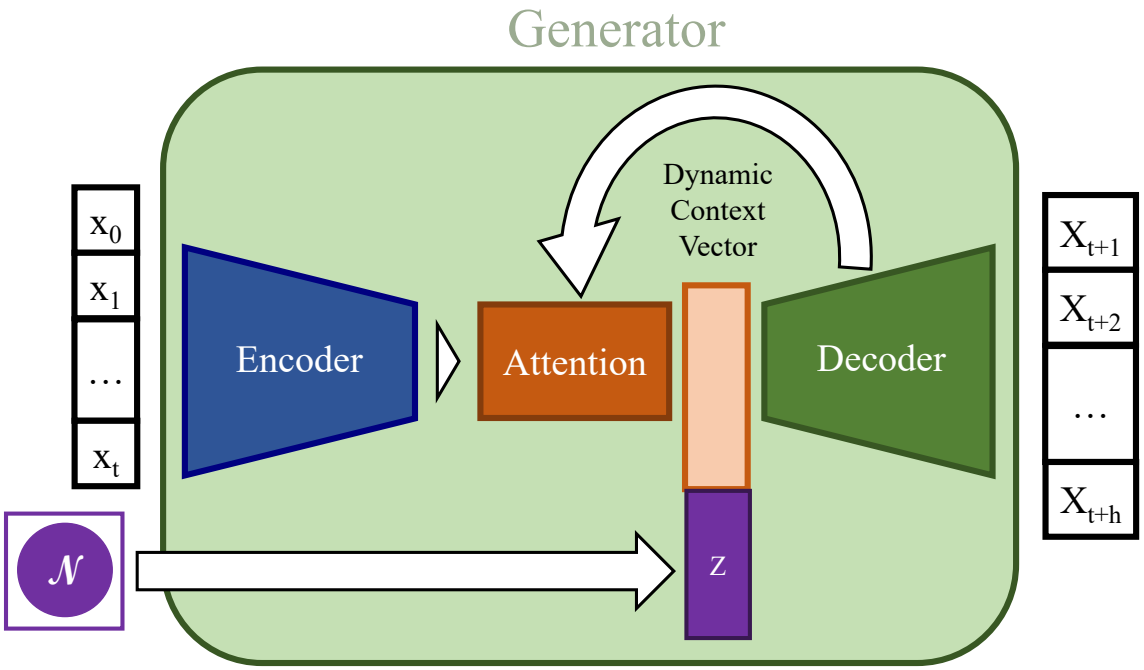


Fig. 6.3 Data pipeline of ForGAN's generator with Attention-based Seq2Seq model.

Table 6.1 Hyperparameters for Attention-based Models

Hyperparameter	Value
Size of Context Vector	$4 \times \log(\text{Input History Window})$
Size of Noise Vector	$4 \times \log(\text{Input History Window})$

6.2 Experiments on Probabilistic Forecasting

In this section, we delve into the empirical evaluation of probabilistic multi-step-ahead forecasting. The experiments encompass a comparative analysis against 11 baseline models, followed by an in-depth discussion of the results. This section also includes studies focusing on various aspects of the proposed models, offering comprehensive insights into their performance characteristics.

We begin this section by introducing 11 baseline models, each representing distinct approaches in probabilistic forecasting. These models serve as a comparative framework to

Table 6.2 The properties of univariate datasets utilized for multi-step-ahead forecasting experiments

Name	Frequency	History Window Size	Horizon
Gold Price Dataset	Daily	60	30
HPEC Dataset	Hourly	48	24
Internet Traffic A1H Dataset	Hourly	48	24
Internet Traffic A5M Dataset	5 minutes	24	12
Internet Traffic B1H Dataset	Hourly	48	24
Internet Traffic B5M Dataset	5 minutes	24	12
Mackey Glass Dataset	Seconds	120	60
Saugeen River Flow Dataset	Daily	60	30
Sunspot Dataset	Daily	60	30
US Births Dataset	Daily	60	30
Solar Dataset	Hourly	48	24
Wind Dataset	Hourly	48	24

assess the efficacy of our proposed methods in multi-step-ahead forecasting scenarios. A brief description of each model is provided, highlighting their unique forecasting strategies and underlying methodologies.

Our experimental setup involves partitioning each dataset, reserving a portion equivalent to five forecast horizons for testing while the remainder is utilized for model training. The proposed models undergo training for a maximum of 100,000 steps. However, training is ceased prematurely if no improvement in model performance is observed over 5,000 consecutive training steps.

For the estimation of the Continuous Ranked Probability Score (CRPS), as delineated in Equation(4.7), we use 1,000 samples per time step within the forecast horizon. The computational environment for the experiments comprises a machine equipped with an Nvidia RTX A6000 GPU, providing the necessary computational power for efficient model training and evaluation. The remaining details of the experimental setup align with those specified for the one-step-ahead forecasting experiment, as outlined in Section 5.4.3.

The datasets employed for these experiments are the same as those used in the one-step-ahead forecasting experiment, with their properties detailed in Section 5.4.2. Additionally, we define the forecast horizon for each dataset in Table 6.2.

6.2.1 Results Analysis

The comprehensive evaluation of 12 proposed models and 11 baselines models across 12 diverse datasets is encapsulated in Tables 6.3, 6.4, and 6.5. The extensive data presented in these tables can initially seem overwhelming. However, a color-coded scheme aids in interpreting the relative performance of the models, with each color denoting the model's performance in comparison to the best-performing model on each dataset. A clear pattern emerges from this visual representation: the proposed models consistently outperform the baseline models across the majority of datasets. Notably, models based on TCN demonstrate marginally superior performance relative to their RNN counterparts.

To synthesize these extensive results into a more digestible format, we utilize the critical difference diagram. Figure 6.4 showcases this diagram for the multi-step-ahead forecasting experiment. An intriguing insight from this diagram is the superior performance of AR models, particularly those utilizing TCN architecture. The top critical difference band includes all the proposed models alongside TFT, Wavenet, and DeepAR, suggesting their comparable performance levels considering their rank across all datasets. These models, hereafter referred to as "top tier models," exhibit performances that are statistically indistinguishable from one another based on their ranking across the datasets.

In Figure 6.5, the distribution of relative scores among the top-tier models is illustrated. The Auto-Regressive ForGAN model with TCN architecture and enhanced by CRPS loss emerges as the leading performer. Amongst the proposed models, TCN-based variants display uniformly strong performances. For RNN-based models, Attention-based VAEneu and Attention-based ForGAN with CRPS loss are on par with their TCN counterparts. This observation highlights the positive impact of the CRPS loss function, affirming its utility in multi-step-ahead forecasting.

The baseline models, while trailing behind the proposed models, exhibit their own set of strengths. DeepAR, in particular, distinguishes itself as the most competent among the baselines. This outcome is somewhat unexpected, considering DeepAR's explicit assumptions about the nature of uncertainty distribution. The impressive performance of DeepAR suggests that Gaussian distributions can aptly approximate uncertainty in many real-world scenarios.

Table 6.3 The results of multi-step-ahead forecasting experiment. The cell colors represent the divergence of a model result from the best model on the dataset. The legend of cell color is placed on the last row.

	Gold price	HEPC	Internet traffic A1H	Internet traffic A5M
Prophet	1.068E+02 ± 3.325E-01	3.505E-01 ± 7.260E-03	4.909E+11 ± 4.765E+09	1.349E+09 ± 2.006E+07
MQ-RNN	3.158E+02 ± 2.966E+01	5.282E-01 ± 8.993E-02	1.384E+12 ± 0.000E+00	5.113E+09 ± 8.783E+03
MQ-CNN	4.559E+01 ± 7.067E+00	7.314E-01 ± 2.027E-01	4.680E+11 ± 1.769E+10	9.308E+08 ± 1.482E+07
TFT	4.727E+01 ± 8.706E+00	2.909E-01 ± 7.120E-02	2.458E+11 ± 1.047E+11	1.707E+08 ± 8.986E+06
Transformer	7.763E+01 ± 4.331E+01	2.801E-01 ± 2.191E-02	9.958E+10 ± 3.720E+10	1.787E+08 ± 5.814E+07
DeepFactor	1.148E+02 ± 8.890E+01	1.142E+00 ± 1.134E-01	1.517E+12 ± 8.429E+11	3.351E+09 ± 7.390E+07
DeepAR	6.510E+01 ± 3.140E+01	**2.584E-01 ± 1.079E-02**	9.868E+10 ± 1.127E+10	1.035E+08 ± 1.385E+07
Deep Renewal Processes	1.311E+03 ± 2.219E+00	5.580E-01 ± 1.316E-02	1.398E+12 ± 0.000E+00	5.165E+09 ± 0.000E+00
Wavenet	3.318E+01 ± 7.699E+00	2.936E-01 ± 1.978E-02	1.821E+11 ± 4.357E+10	1.308E+08 ± 1.219E+07
GPForecaster	3.888E+01 ± 8.487E-01	6.021E-01 ± 9.254E-03	1.398E+12 ± 1.992E+06	5.109E+09 ± 8.269E+05
DeepState	4.503E+01 ± 4.653E+00	7.153E-01 ± 8.870E-02	7.917E+11 ± 4.126E+10	1.227E+09 ± 3.086E+08
AbVAEneu-TCN	2.224E+01 ± 1.579E-01	3.021E-01 ± 2.385E-02	7.427E+10 ± 4.220E+09	9.797E+07 ± 9.119E+06
CRPS-AbForGAN-TCN	2.049E+01 ± 1.417E+00	3.171E-01 ± 1.535E-02	6.232E+10 ± 7.001E+09	1.131E+08 ± 7.362E+06
AbForGAN-TCN	2.213E+01 ± 9.108E-01	4.176E-01 ± 2.152E-02	6.566E+10 ± 1.739E+10	1.017E+08 ± 1.272E+07
ARVAEneu-TCN	2.107E+01 ± 5.676E-01	2.873E-01 ± 1.050E-02	5.944E+10 ± 2.376E+09	1.094E+08 ± 1.386E+07
CRPS-ARForGAN-TCN	1.893E+01 ± 5.264E-01	3.021E-01 ± 8.493E-03	7.413E+10 ± 1.505E+10	1.143E+08 ± 9.292E+06
ARForGAN-TCN	2.068E+01 ± 6.792E-01	3.019E-01 ± 2.020E-02	7.713E+10 ± 7.337E+09	1.004E+08 ± 2.143E+07
AbVAEneu-RNN	2.036E+01 ± 2.125E+00	2.830E-01 ± 7.968E-03	7.778E+10 ± 2.679E+09	9.968E+07 ± 6.968E+06
CRPS-AbForGAN-RNN	2.204E+01 ± 6.048E-01	3.251E-01 ± 8.432E-03	6.833E+10 ± 4.690E+09	1.215E+08 ± 1.763E+07
AbForGAN-RNN	3.149E+01 ± 7.654E+00	4.344E-01 ± 1.211E-02	1.409E+11 ± 7.438E+10	**7.899E+07 ± 1.167E+07**
ARVAEneu-RNN	**1.890E+01 ± 2.409E-01**	3.115E-01 ± 2.620E-02	**5.266E+10 ± 3.054E+09**	1.090E+08 ± 6.614E+06
CRPS-ARForGAN-RNN	2.101E+01 ± 6.515E-01	2.987E-01 ± 3.822E-03	6.321E+10 ± 1.357E+10	1.327E+08 ± 2.639E+07
ARForGAN-RNN	1.940E+01 ± 8.139E-01	2.951E-01 ± 1.680E-02	8.866E+10 ± 4.083E+09	9.573E+07 ± 4.398E+06
Color legend	**Best Model**	≤ 10% ≤ 50%	≤ 100% > 100%	

Table 6.4 The results of multi-step-ahead forecasting experiment. The cell colors represent the divergence of a model result from the best model on the dataset. The legend of cell color is placed on the last row.

	Internet traffic B1H	Internet traffic B5M	Mackey Glass	Saugeen river
Prophet	1.925E+15 ± 3.225E+13	1.415E+14 ± 1.646E+12	1.327E-01 ± 9.006E-04	1.194E+01 ± 2.700E-01
MQ-RNN	1.581E+16 ± 0.000E+00	1.171E+15 ± 0.000E+00	1.446E-01 ± 7.526E-03	2.091E+01 ± 7.186E-01
MQ-CNN	2.375E+15 ± 1.920E+14	2.444E+14 ± 7.673E+12	1.247E-01 ± 9.235E-03	1.412E+01 ± 1.665E-01
TFT	8.051E+14 ± 2.390E+13	**3.043E+13 ± 3.083E+12**	5.918E-03 ± 1.056E-03	8.872E+00 ± 6.238E-01
Transformer	8.580E+14 ± 1.645E+14	5.081E+13 ± 1.611E+13	1.187E-01 ± 2.262E-02	1.161E+01 ± 1.725E+00
DeepFactor	5.780E+15 ± 1.402E+15	4.906E+14 ± 1.993E+13	2.513E-01 ± 4.804E-03	2.232E+01 ± 2.121E-01
DeepAR	6.546E+14 ± 1.921E+14	3.463E+13 ± 1.904E+12	7.630E-02 ± 6.160E-03	1.293E+01 ± 3.054E-01
Deep Renewal Processes	1.597E+16 ± 0.000E+00	1.183E+15 ± 0.000E+00	1.915E-01 ± 3.415E-05	1.010E+01 ± 2.412E-01
Wavenet	1.487E+15 ± 2.118E+14	3.291E+13 ± 7.423E+12	7.115E-03 ± 1.281E-04	9.413E+00 ± 4.448E-01
GPForecaster	1.597E+16 ± 1.156E+08	1.183E+15 ± 9.302E+07	3.252E-01 ± 1.440E-01	1.140E+01 ± 2.043E-01
DeepState	2.761E+15 ± 2.387E+14	2.694E+14 ± 5.305E+13	2.841E-01 ± 3.193E-03	1.484E+01 ± 1.995E+00
AbVAEneu-TCN	4.485E+14 ± 1.856E+13	3.821E+13 ± 1.582E+12	3.274E-03 ± 5.482E-04	8.631E+00 ± 1.383E-01
CRPS-AbForGAN-TCN	**4.460E+14 ± 7.685E+13**	4.129E+13 ± 6.742E+12	8.257E-03 ± 5.326E-03	8.526E+00 ± 2.259E-02
AbForGAN-TCN	9.359E+14 ± 3.503E+14	5.311E+13 ± 1.325E+13	4.793E-02 ± 2.900E-02	8.545E+00 ± 1.243E+00
ARVAEneu-TCN	5.273E+14 ± 6.020E+13	3.456E+13 ± 2.420E+12	**4.471E-04 ± 1.112E-04**	8.433E+00 ± 9.476E-02
CRPS-ARForGAN-TCN	4.647E+14 ± 3.201E+13	4.387E+13 ± 5.952E+12	6.740E-04 ± 1.950E-04	8.280E+00 ± 1.340E-01
ARForGAN-TCN	5.982E+14 ± 1.836E+14	3.244E+13 ± 6.031E+11	3.576E-03 ± 1.584E-03	8.433E+00 ± 2.263E-01
AbVAEneu-RNN	4.712E+14 ± 2.212E+13	3.996E+13 ± 6.789E+11	2.746E-03 ± 3.471E-04	8.666E+00 ± 7.088E-02
CRPS-AbForGAN-RNN	5.278E+14 ± 6.229E+13	5.374E+13 ± 1.066E+13	4.056E-03 ± 1.154E-03	8.332E+00 ± 5.433E-01
AbForGAN-RNN	1.241E+15 ± 2.182E+14	3.905E+13 ± 7.390E+12	1.527E-01 ± 2.047E-02	**8.035E+00 ± 2.357E-01**
ARVAEneu-RNN	5.751E+14 ± 1.191E+14	4.196E+13 ± 5.294E+12	8.570E-04 ± 9.274E-05	8.441E+00 ± 1.981E-02
CRPS-ARForGAN-RNN	8.170E+14 ± 1.516E+14	4.599E+13 ± 2.049E+12	4.862E-03 ± 2.930E-03	8.302E+00 ± 1.939E-01
ARForGAN-RNN	1.080E+15 ± 3.793E+14	3.957E+13 ± 2.358E+12	4.877E-03 ± 1.523E-03	8.303E+00 ± 3.844E-02
Color legend	**Best Model**	≤ 10% ≤ 50%	≤ 100% > 100%	

Table 6.5 The results of multi-step-ahead forecasting experiment. The cell colors represent the divergence of a model result from the best model on the dataset. The legend of cell color is placed on the last row.

	Solar 4 seconds	Sunspot	US births	Wind 4 seconds
Prophet	7.505E+03 ± 1.015E+02	2.227E+01 ± 3.251E-01	3.601E+02 ± 3.117E+00	1.173E+04 ± 2.783E+02
MQ-RNN	5.440E+03 ± 4.428E+02	3.408E+00 ± 1.624E-01	9.468E+02 ± 2.302E+01	1.409E+04 ± 1.797E+02
MQ-CNN	4.755E+03 ± 5.661E+02	3.469E+00 ± 5.075E-01	5.010E+02 ± 7.133E+01	1.286E+04 ± 1.878E+02
TFT	3.287E+03 ± 5.986E+02	3.048E+00 ± 9.106E-02	**1.324E+02 ± 2.443E+00**	9.313E+03 ± 1.301E+03
Transformer	1.887E+03 ± 1.971E+02	2.627E+00 ± 1.789E-02	3.900E+02 ± 2.299E+02	8.912E+03 ± 2.063E+03
DeepFactor	3.043E+04 ± 7.307E+02	2.795E+00 ± 3.432E-01	3.062E+03 ± 1.629E+03	1.223E+04 ± 4.748E+02
DeepAR	1.599E+03 ± 2.118E+02	2.622E+00 ± 6.136E-02	1.849E+02 ± 1.446E+01	9.664E+03 ± 1.157E+03
Deep Renewal Processes	2.953E+04 ± 8.191E+00	3.397E+01 ± 7.952E+00	1.057E+04 ± 7.027E+00	1.724E+04 ± 5.735E-02
Wavenet	4.222E+03 ± 1.339E+03	2.600E+00 ± 3.132E-01	3.155E+02 ± 3.350E+01	1.338E+04 ± 2.940E+03
GPForecaster	2.126E+04 ± 3.377E+02	4.008E+00 ± 6.619E-01	4.248E+02 ± 7.614E-01	1.153E+04 ± 3.518E+02
DeepState	7.449E+03 ± 1.362E+03	9.865E+00 ± 1.352E+00	5.376E+02 ± 2.587E+01	2.820E+04 ± 7.989E+03
AbVAEneu-TCN	2.118E+03 ± 7.299E+01	**2.483E+00 ± 2.146E-02**	2.928E+02 ± 8.977E+00	8.072E+03 ± 5.277E+01
CRPS-AbForGAN-TCN	2.198E+03 ± 3.357E+01	3.134E+00 ± 4.622E-01	2.911E+02 ± 2.316E+01	7.715E+03 ± 1.405E+02
AbForGAN-TCN	1.588E+03 ± 2.659E+02	3.072E+00 ± 2.129E-01	8.340E+02 ± 1.127E+02	7.535E+03 ± 2.880E+02
ARVAEneu-TCN	1.800E+03 ± 1.344E+02	2.672E+00 ± 3.969E-02	2.875E+02 ± 2.054E+01	7.360E+03 ± 3.616E+01
CRPS-ARForGAN-TCN	1.773E+03 ± 2.306E+02	3.177E+00 ± 3.800E-01	2.912E+02 ± 1.509E+01	7.163E+03 ± 3.742E+02
ARForGAN-TCN	**1.449E+03 ± 7.190E+01**	2.726E+00 ± 3.155E-02	3.125E+02 ± 2.791E+01	**6.888E+03 ± 2.701E+02**
AbVAEneu-RNN	2.175E+03 ± 2.104E+01	2.494E+00 ± 1.028E-01	2.991E+02 ± 7.796E+00	8.009E+03 ± 2.646E+02
CRPS-AbForGAN-RNN	2.208E+03 ± 5.464E+01	3.443E+00 ± 2.848E-01	2.964E+02 ± 2.758E+01	8.362E+03 ± 1.727E+02
AbForGAN-RNN	1.007E+04 ± 1.445E+04	3.064E+00 ± 2.353E-01	7.414E+02 ± 3.879E+02	8.061E+03 ± 4.018E+02
ARVAEneu-RNN	2.776E+03 ± 1.035E+02	3.378E+00 ± 8.062E-02	2.868E+02 ± 9.752E+00	6.892E+03 ± 2.090E+01
CRPS-ARForGAN-RNN	3.009E+03 ± 2.661E+02	4.134E+00 ± 4.991E-01	3.010E+02 ± 1.095E+01	7.041E+03 ± 1.600E+02
ARForGAN-RNN	3.069E+03 ± 1.977E+02	3.634E+00 ± 1.247E-01	2.847E+02 ± 1.931E+01	6.921E+03 ± 2.521E+02
Color legend	**Best Model**	$\leq$ 10% $\leq$ 50%	$\leq$ 100% $>$ 100%	

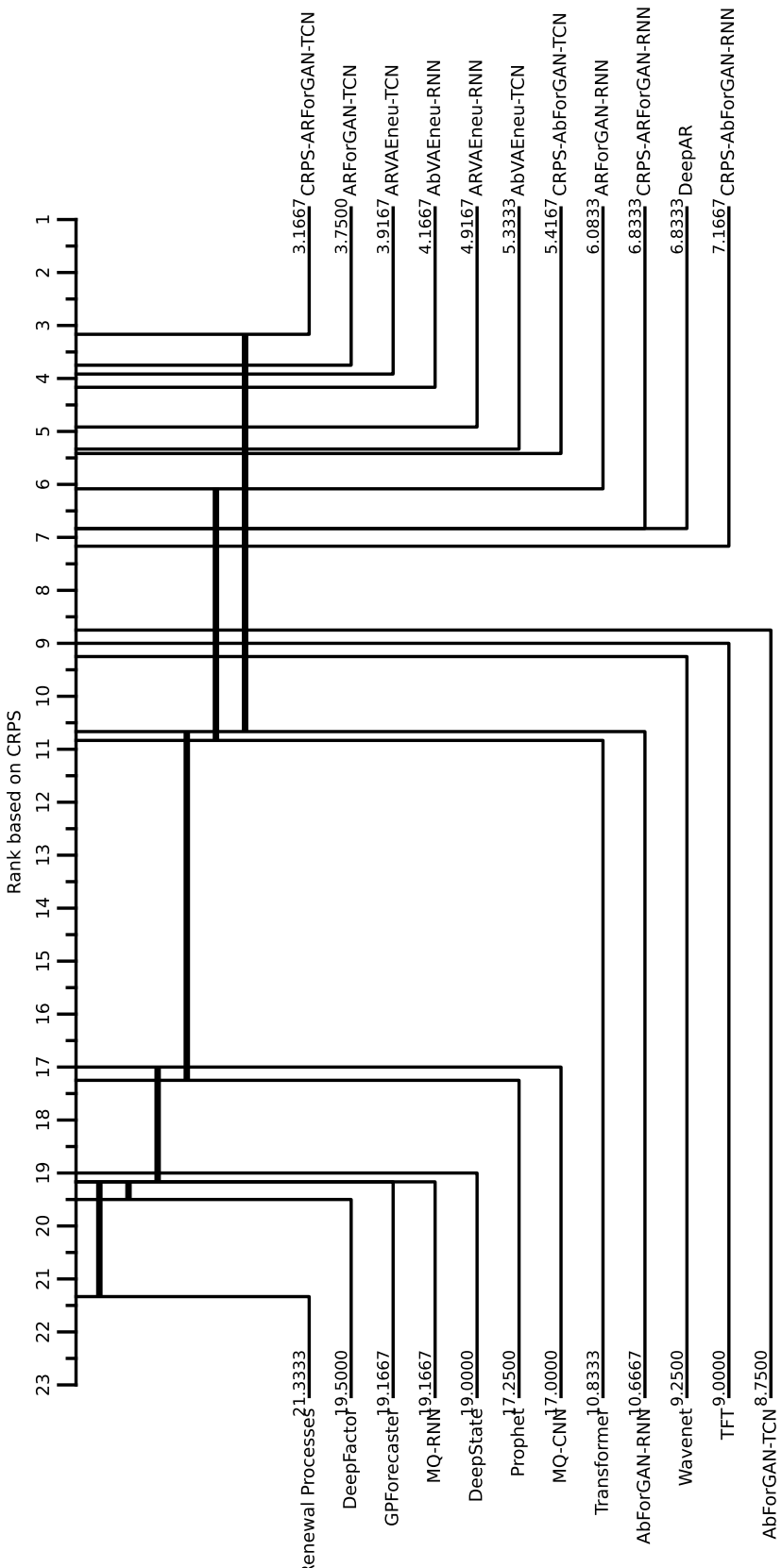


Fig. 6.4 The critical difference diagram for multi-step-ahead forecasting experiment.

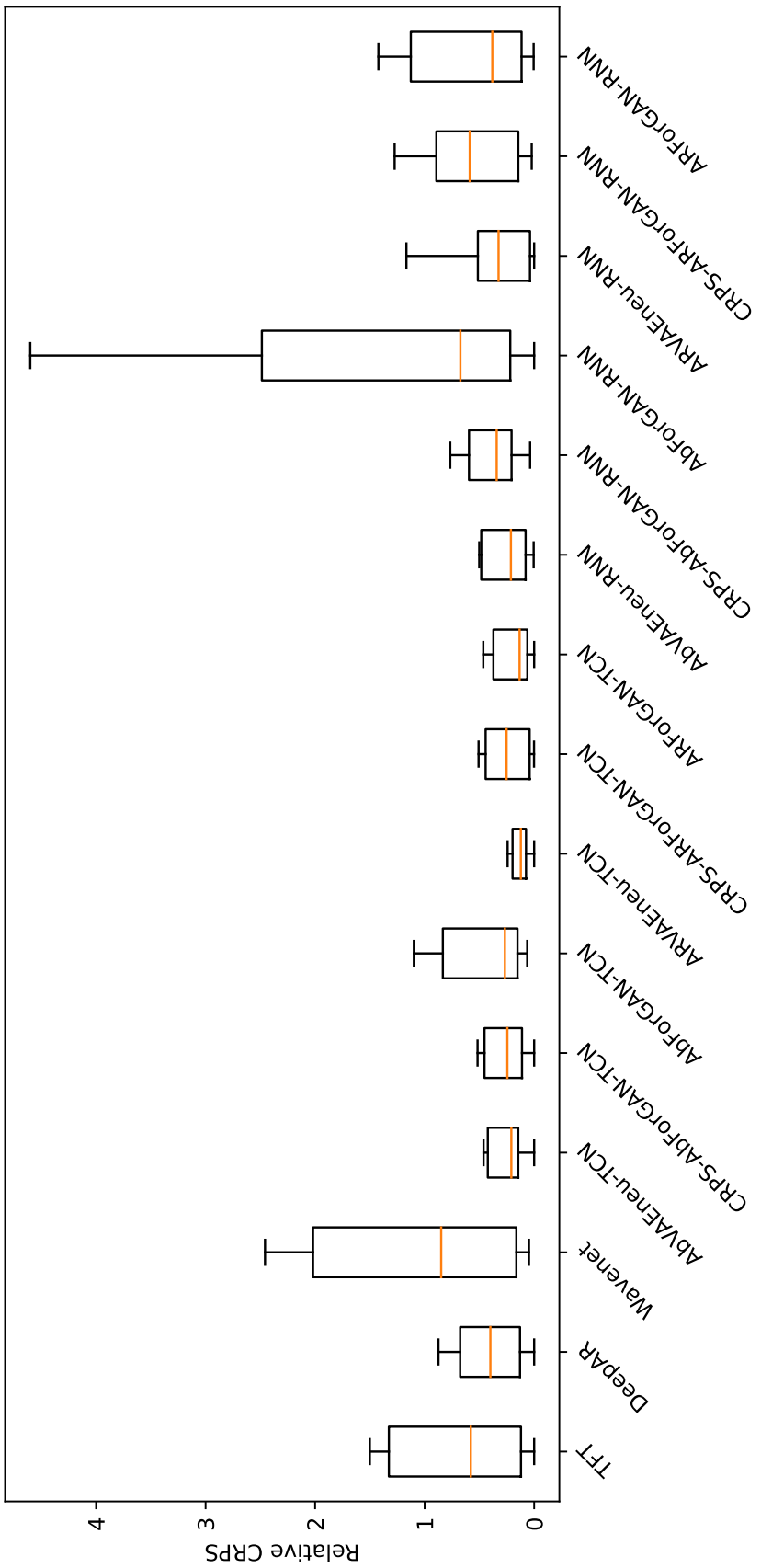


Fig. 6.5 The relative CRPS for top tier model for multi-step-ahead forecasting experiment.

Contrary to expectations, the integration of complex sequence-to-sequence models into the generator and decoder architecture did not yield a significant performance advantage. A plausible explanation for this could be the necessity of a correspondingly sophisticated discriminator/encoder to leverage the capabilities of these complex models comprehensively. For the sake of a uniform comparison, we paired these models with a discriminator/encoder akin to those used in Auto-Regressive models. Thus, the study of Attention-based models hyperparameters and their influence on models' performance opens an interesting research front for future investigation.

Finally, Figures 6.6 and 6.7 present qualitative insights into models' presents. In these figures, the samples from best-performing models from each category of models, namely, VAEneu, ForGAN, and baseline for 7 datasets, are presented. Figure 6.6 showcases sample forecasts from 4 datasets where the proposed models—VAEneu and ForGAN—outperformed the baseline models. This visualization serves to validate and complement the quantitative findings, offering a tangible representation of the superiority of the proposed models. It elucidates how these models capture the underlying patterns and dynamics in the datasets more effectively, resulting in more accurate and reliable forecasts.

Conversely, Figure 6.7 presents samples from the three datasets where baseline models demonstrated better performance than the proposed ones. While the baselines exhibit superior results in these cases, a close examination of the samples from the proposed models reveals that their forecasts are still competitively close to the expected outcomes. This observation indicates that, even in scenarios where they are not leading, the proposed models maintain a high level of forecasting quality, closely rivaling the baseline models.

6.2.2 Analysis of Model Convergence

In our exploration of multi-step-ahead forecasting, a key focus has been on understanding how swiftly our proposed models reach their optimal parameters during the training phase. This analysis encompasses both the number of training steps and the wall clock time, which are critical factors in evaluating the efficiency of the models.

Figure 6.8 illustrates the convergence patterns of the models in terms of training steps. A notable observation is that the Attention-based models exhibit a significantly faster convergence rate compared to the Auto-Regressive models. This rapid convergence can be considered as further evidence of our previously mentioned theory that these models may not have reached their full potential due to the relatively simple architecture of the en-

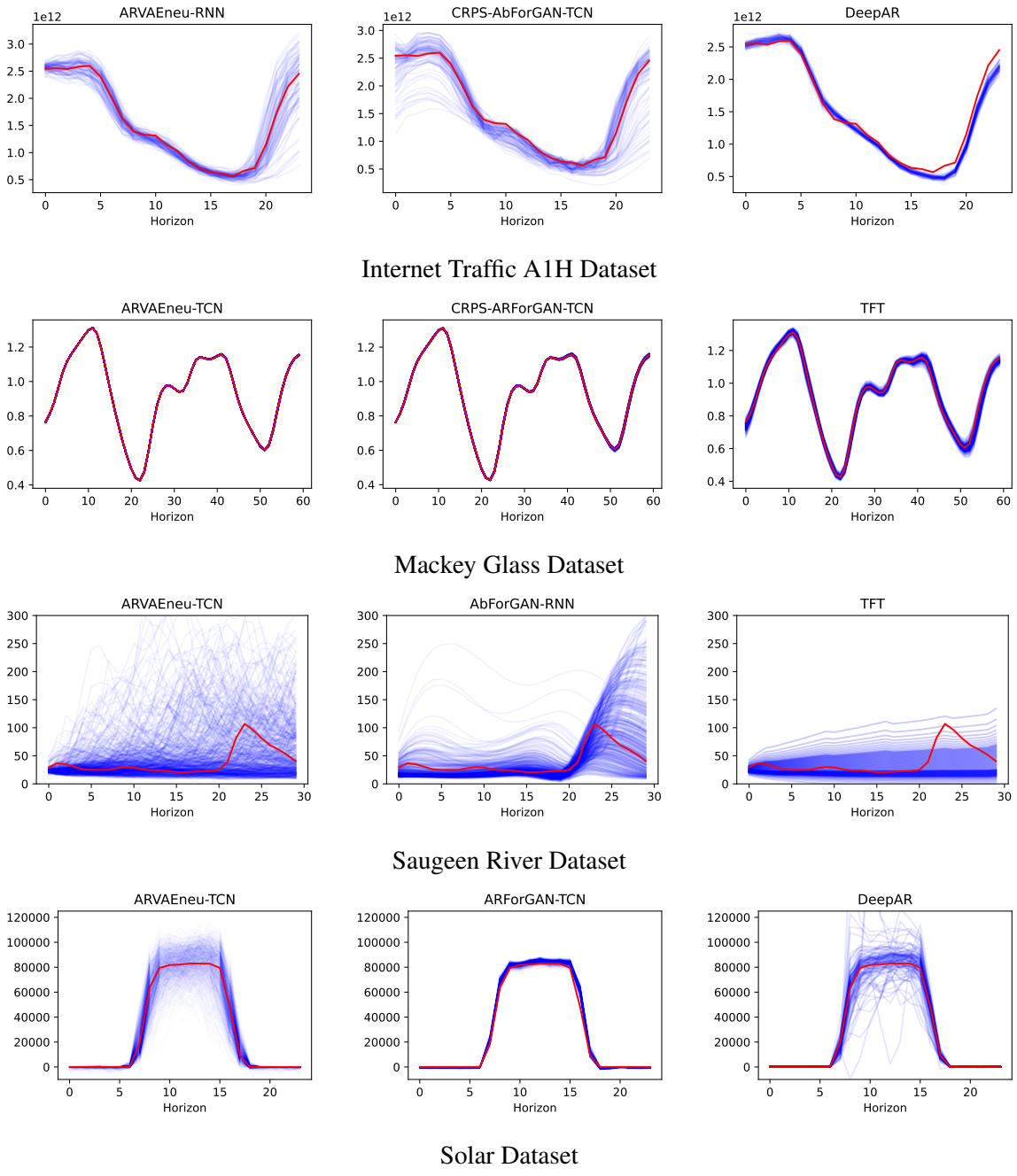


Fig. 6.6 Samples of forecasts from 4 datasets from our experiment where proposed models acquire best results. The figures on each row present samples for a dataset. The first column is the best-performing VAEneu model. The second column is the best-performing ForGAN model. The third column is the best-performing baseline model. The red line color is ground truth, and the blue lines are forecasts.

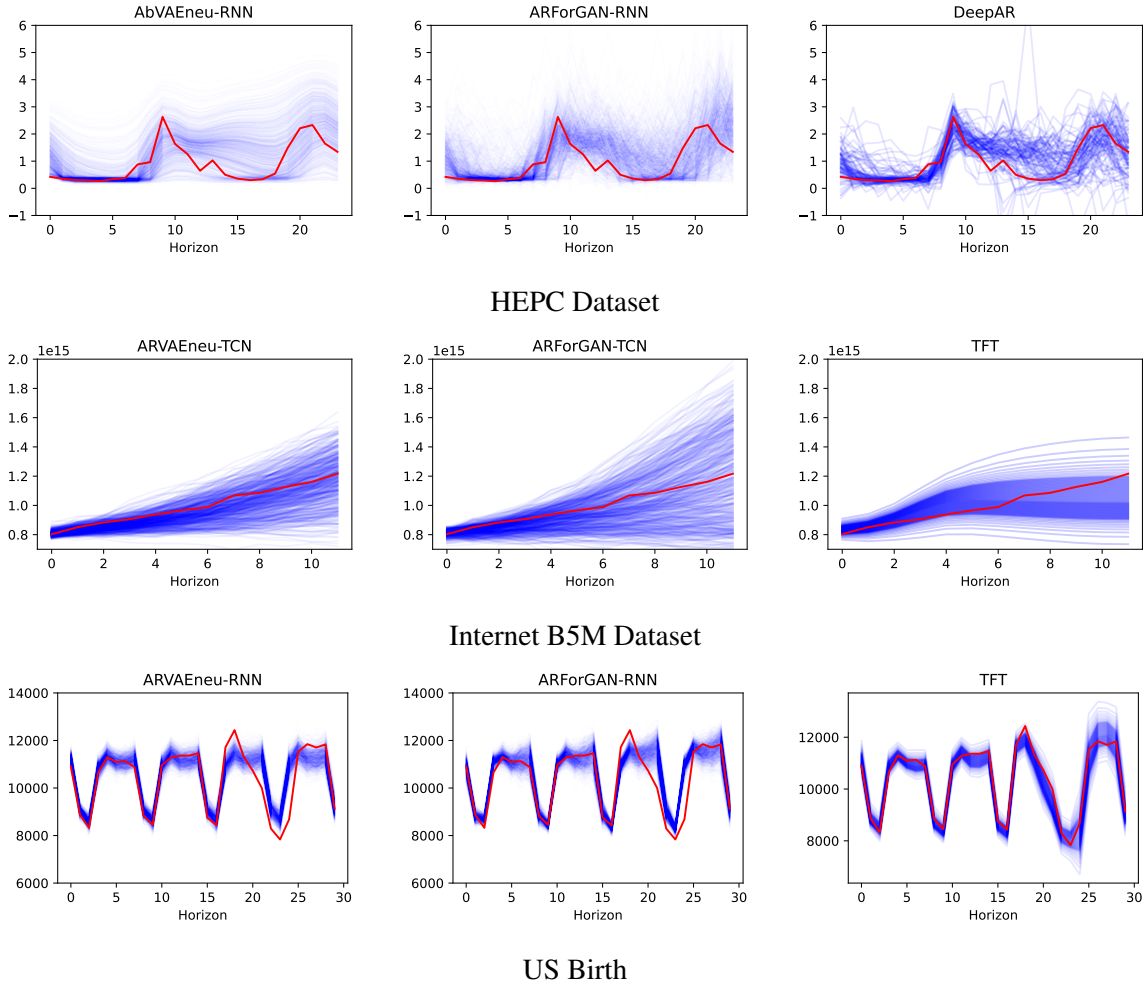


Fig. 6.7 Samples of forecasts of all datasets where baseline models acquire best results. The figures on each row present samples for a dataset. The first column is the best-performing VAEneu model. The second column is the best-performing ForGAN model. The third column is the best-performing baseline model. The red line color is ground truth, and the blue lines are forecasts.

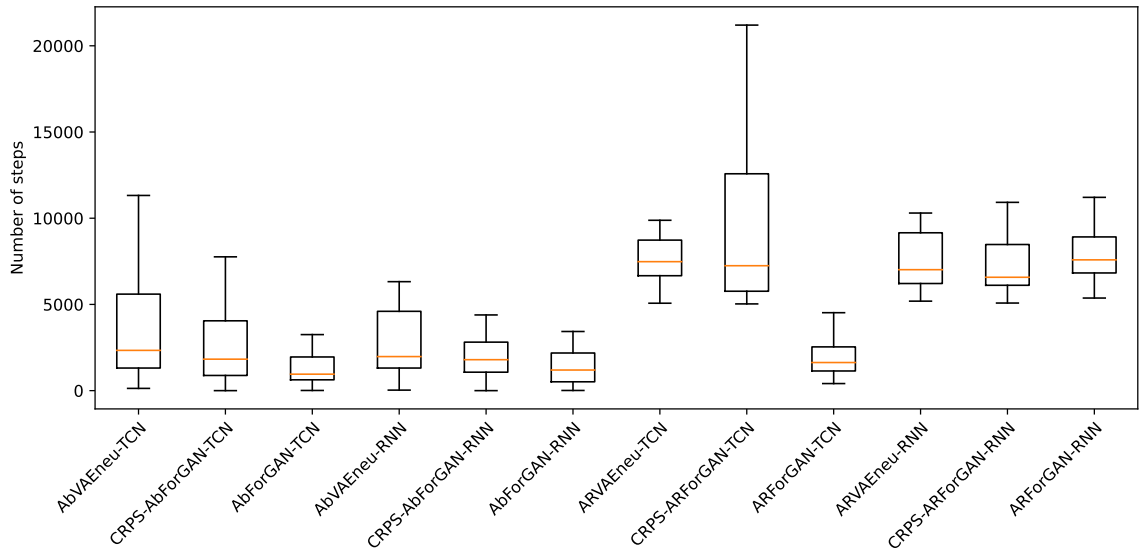


Fig. 6.8 Distribution of convergence steps required by different models to reach optimal parameters during training, spanning all datasets.

coder/discriminator with which they were paired. Consequently, they tend to settle into sub-optimal parameters more quickly. However, it is crucial to note that the performance of these Attention-based models remains competitive across most datasets, as delineated in the preceding sections.

Turning our attention to the average time required to process a single batch during training, Figure 6.9 provides valuable insights. The Attention-based models demonstrate a relatively slower batch processing speed compared to the Auto-Regressive models. The distinct training objectives of these models can explain this difference in processing speed. While Attention-based models are tasked with predicting the entire forecast horizon during training, the Auto-Regressive models focus on one-step-ahead forecasting during training and extend their capabilities to multi-step-ahead forecasting only during the inference phase.

6.2.3 Analysis of Inference Time

The efficiency of a forecasting model in a real-time application is critically dependent on its inference time. Particularly in scenarios demanding rapid probabilistic forecasts, the ability of a model to generate predictions within strict time constraints becomes a pivotal factor. In this context, our study aims to evaluate the inference time of the proposed models rigorously. We quantified the time required to generate 1000 samples for a single time step in the forecast horizon, and these measurements were aggregated across all datasets, as depicted

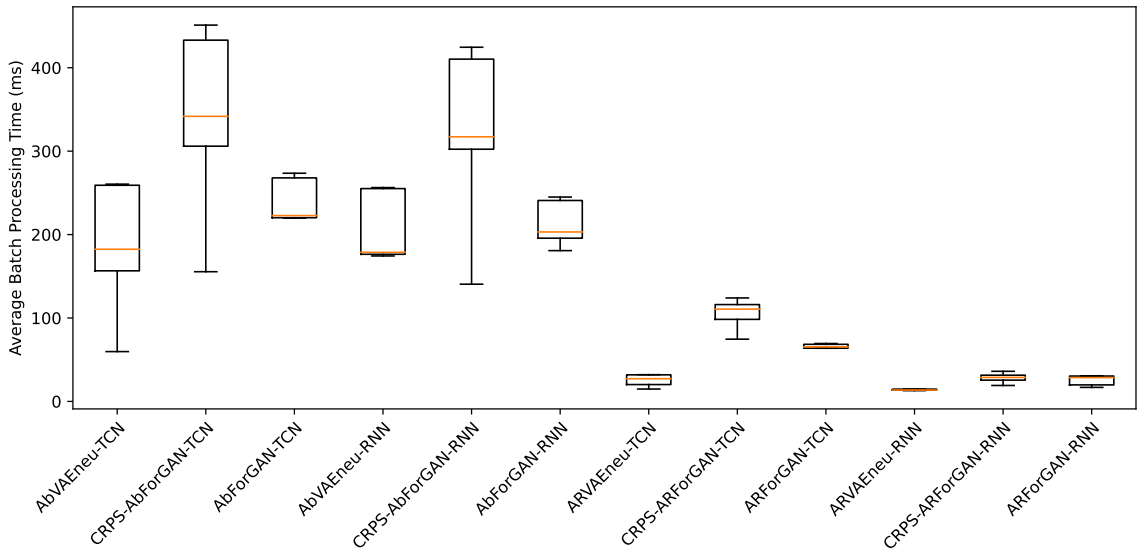


Fig. 6.9 Average batch processing time of each model during training, illustrated across all datasets.

in Figure 6.10.

The Attention-based models demonstrate consistent performance in terms of inference time across the board, attributable to their uniform structure in generating probabilistic forecasts. RNN-based Auto-Regressive models exhibit the fastest inference times among all the proposed models. This efficiency is largely due to the memory aspect inherent in RNN units. For the first forecast step, these models require only a single feed of historical data. Subsequently, for each consecutive forecast, the model leverages the RNN cell's state, effectively serving as a memory mechanism, and necessitates only the forecasted output from the previous step for further predictions. This memory-centric approach significantly streamlines the inference process.

Conversely, the TCN-based Auto-Regressive models face challenges regarding inference speed. Without a memory mechanism akin to RNN units, these models require the entire history window, augmented with all preceding forecasts, to be fed into the network for each time step within the forecast horizon. This repetitive process considerably slows down the inference time, positioning the TCN Auto-Regressive models as the most time-intensive among the proposed models. Despite their exemplary forecasting accuracy, this time constraint might limit their applicability in time-sensitive environments.

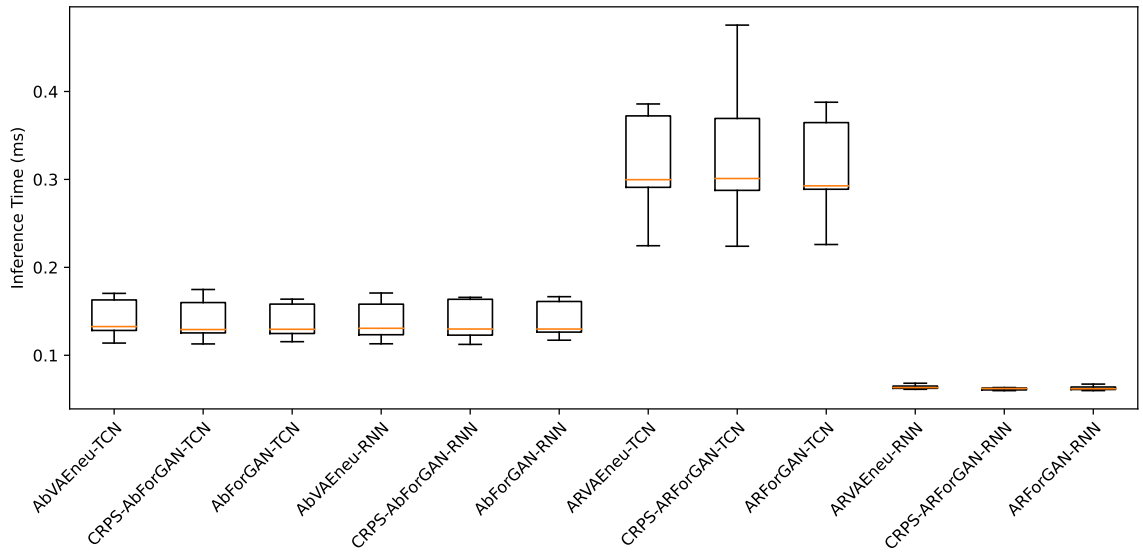


Fig. 6.10 Inference time distribution for the proposed models, measured for the generation of 1000 samples per time step in the forecast horizon.

The selection of an appropriate model for a given task hinges on balancing accuracy with time efficiency. If the application imposes strict constraints on inference time, the RNN variant of the Auto-Regressive VAEneu model emerges as a viable choice. It offers a compromise between speed and accuracy, catering to scenarios with severe time limitations. In cases where moderately relaxed time constraints are present, Attention-based models stand as a suitable option, offering near-paralleled accuracy and stability to TCN-based models. However, for applications where high accuracy is paramount and time constraints are more flexible, the TCN Auto-Regressive models, despite their longer inference times, provide the best performance.

This comprehensive analysis of inference times across various model architectures offers crucial insights for choosing the right model in accordance with the specific requirements of real-time forecasting tasks. Understanding these trade-offs between inference speed and forecasting accuracy is essential for effective model deployment in real-world applications.

6.2.4 Investigating Error Accumulation in Forecasting Horizons

A critical aspect of probabilistic forecasting, particularly in models employing Auto-Regressive structures, is the potential accumulation of errors across the forecast horizon. This phenomenon, where errors in early predictions compound and adversely affect subsequent predictions, was a primary driver for exploring the Attention-based model as an alternative.

To rigorously evaluate the extent of error accumulation in AR models and compare it with their Attention-based counterparts, we conducted a detailed analysis focusing on the CRPS at each time step within the forecast horizon.

Our analysis entailed measuring CRPS at individual time steps across the forecast horizon for both AR and Attention-based models. This approach allowed us to examine the forecast accuracy and error propagation over the horizon. The results of our analysis for three datasets are illustrated in Figure 6.11. Contrary to our initial hypothesis, we found that the AR models did not exhibit a significant accumulation of errors over the horizon. The performance of these models remained consistent and on par with the Attention-based models. This finding suggests that the AR models, despite their sequential prediction approach, maintain robustness against error propagation, a characteristic pivotal for reliable long-term forecasting.

An intriguing observation emerged regarding the Attention-based ForGAN models. These models exhibited a noticeable deterioration in performance compared to other Attention-based models on some datasets. This distinction raises questions about the model's adaptability or specific architectural nuances that might contribute to its relative underperformance in certain contexts.

In conclusion, our study challenges the notion of inherent error accumulation in AR models for probabilistic forecasting. The findings underscore the competence of AR models in maintaining forecast accuracy over extended horizons. Additionally, the varying performances of different Attention-based models, particularly the ForGAN variant, open avenues for further exploration into model-specific characteristics and their impact on long-term forecasting accuracy.

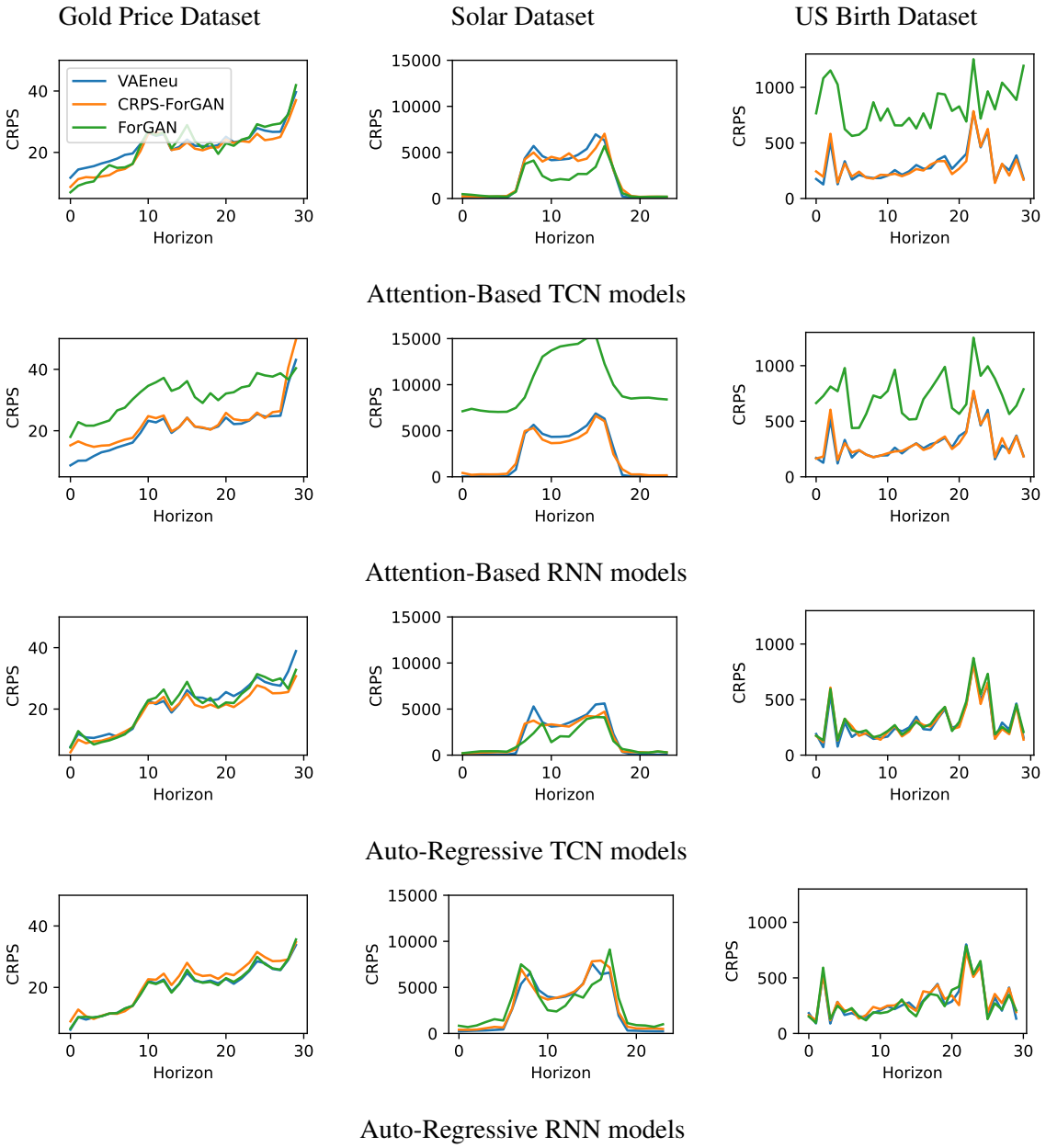


Fig. 6.11 The CRPS of the proposed models over forecast horizon on 3 datasets.

Chapter 7

Multivariate One-step-ahead Forecasting

Time series forecasting remains a cornerstone of many crucial applications, from financial modeling to energy consumption prediction. The field traditionally leans towards deterministic point prediction techniques, which, while intuitive, tend to offer an average perspective on possible outcomes. Such models do not encapsulate the inherent uncertainty prevalent in real-world scenarios. As articulated by Gneiting et al. [89], probabilistic forecasting models are designed to mitigate this lacuna. These models attempt to quantify predictive uncertainties by generating a probability distribution over potential outcomes.

Within this context, Generative Adversarial Networks (GANs), despite their proven efficacy in modeling intricate probability distributions, are fraught with challenges in training stability, requiring meticulous model architecture and hyperparameter selection [90]. Drawing inspiration from the ForGAN architecture, this chapter introduces *ProbCast*: a novel probabilistic forecaster crafted around the principles of conditional GANs for multivariate time series data. This article also paves the way for a transformative framework specifically designed to metamorphose established deterministic forecasters into their probabilistic counterparts. This framework not only streamlines the transition but also restricts the expansive search space typically associated with GAN architecture identification. In essence, it empowers deterministic models to embrace the probabilistic paradigm without diminishing accuracy, harnessing the latent capabilities of GANs.

The core contributions of this study can be distilled into the following facets:

The content of this chapter has been adopted from A. Koochali et al., "If you like it, GAN it—probabilistic multivariate times series forecast with GAN." Engineering proceedings 5, no. 1, 2020, doi:10.3390/engproc2021005040.

- Introduction of *ProbCast*, a groundbreaking probabilistic model tailored for multivariate time series forecasting, leveraging the mechanics of a conditional GAN for training.
- Development and presentation of a robust framework devised to transform deterministic point forecasting architectures into enhanced probabilistic variants.
- Comprehensive experimental evaluation on two open-source datasets, underscoring the preeminence of *ProbCast*. The results further validate the potential of our proposed framework in transmuting deterministic predictors into probabilistic models that resonate with enhanced accuracy benchmarks.

7.1 Methodology

7.1.1 ProbCast Framework: From Deterministic Forecasters to Probabilistic Forecasters using GANs

Multivariate time series forecasting contains complicated challenges. While the need to identify inter-feature dependencies necessitates complex model architectures, achieving high-accuracy forecasting further magnifies these complexities. An additional layer of difficulty is introduced when using Generative Adversarial Networks (GANs), as they demand thorough hyperparameter tuning to stabilize the training process. Given the pronounced model complexity required for multivariate time series, deriving concurrent optimal architectures for both the generator and discriminator that exhibit robust performance becomes a daunting endeavor.

To circumvent these challenges, we introduce ProbCast, a novel framework that builds upon the premise of GANs, tailored to harness the power of an existing deterministic forecaster for probabilistic forecasting.

Central to the ProbCast framework is the idea of constructing the generator by repurposing the architecture and hyperparameters of a given deterministic forecaster. Subsequently, with the generator architecture in place, we embark on the search for a suitable discriminator. This sequential approach not only isolates the complexities of determining the architecture of the generator and discriminator but also substantially streamlines the GAN architecture search process.

Our framework's paradigm shift brings forth two significant benefits:

1. It paves the way for the seamless transformation of a high-performing deterministic model into its probabilistic counterpart.

2. It ensures the resultant probabilistic model, thus derived, offers enhanced precision and closer congruence with real-world distributions.

Through the juxtaposition of existing forecasting techniques with the adversarial framework, we offer an innovative avenue for advancing the state of probabilistic forecasting in multivariate time series.

7.1.2 Training Pipeline of the ProbCast

Figure 7.1 illustrates our innovative framework, encompassing the conditional GAN setup required for the training process. This framework simplifies and outlines the pathway from deterministic to probabilistic forecasting.

Deterministic Model Construction

The initial phase necessitates the creation of an optimal deterministic forecasting model. Given an extant and proficient point forecast model, this stage can be omitted, directly leveraging the pre-existing model. Otherwise, our approach facilitates the architecture search to pinpoint the optimal deterministic forecaster.

Generator Design and Noise Vector Integration

After the establishment of the deterministic model, the next step is to integrate the noise vector, denoted as z, into its architecture. Empirical findings from our experiments indicate pronounced efficacy when this noise vector is inserted within the latter layers of the neural network. This strategic placement ensures that the first layers predominantly focus on learning the essence of the input window representation.

Discriminator Search and Adversarial Training

With a definitive generator architecture, the subsequent task is singularly streamlined to discover an apt time series classifier to play the role of the discriminator in the adversarial training phase. This specialized focus on only the discriminator for the architecture search not only elevates efficiency but, as our results testify, leads to a discriminator competent

in training the generator. The latter demonstrates a marked enhancement in performance relative to its deterministic counterpart.

The summarized workflow of ProbCast is as follows:

1. Obtain or construct an optimal deterministic forecaster.
2. Design the generator: repurpose the deterministic architecture and embed the noise vector optimally within the posterior layers.
3. Initiate the discriminator hyperparameter search and, subsequently, train the generator within this adversarial setting.

Training Objective

Our optimization criterion, based on a value function, seeks to balance the objectives of both the generator and discriminator and is defined as:

$$\begin{aligned}\min_G \max_D V(D,G) = \mathbb{E}_{X_{t+1}\sim P_{\text{data}}(X_{t+1})}[\,\log D(X_{t+1}|X_t,..,X_0)] + \\ \mathbb{E}_{z\sim P_z(z)}[\,\log(1-D(G(z|X_t,..,X_0)))].\end{aligned} \tag{7.1}$$

In this expression, the first term corresponds to the expectation over the real data distribution, while the second term embodies the generator's attempt to deceive the discriminator using its generated samples. Together, this forms the adversarial game that characterizes the GAN framework.

7.2 Experimental Evaluation

7.2.1 Dataset Description

To assess the efficacy of our proposed methodology, we conducted experiments on two distinct and publicly accessible datasets: the Electricity Consumption and the Exchange-Rate datasets .

The datasets were sourced from the repository https://github.com/laiguokun/multivariate-time-series-data, curated and prepared by the authors of [72].

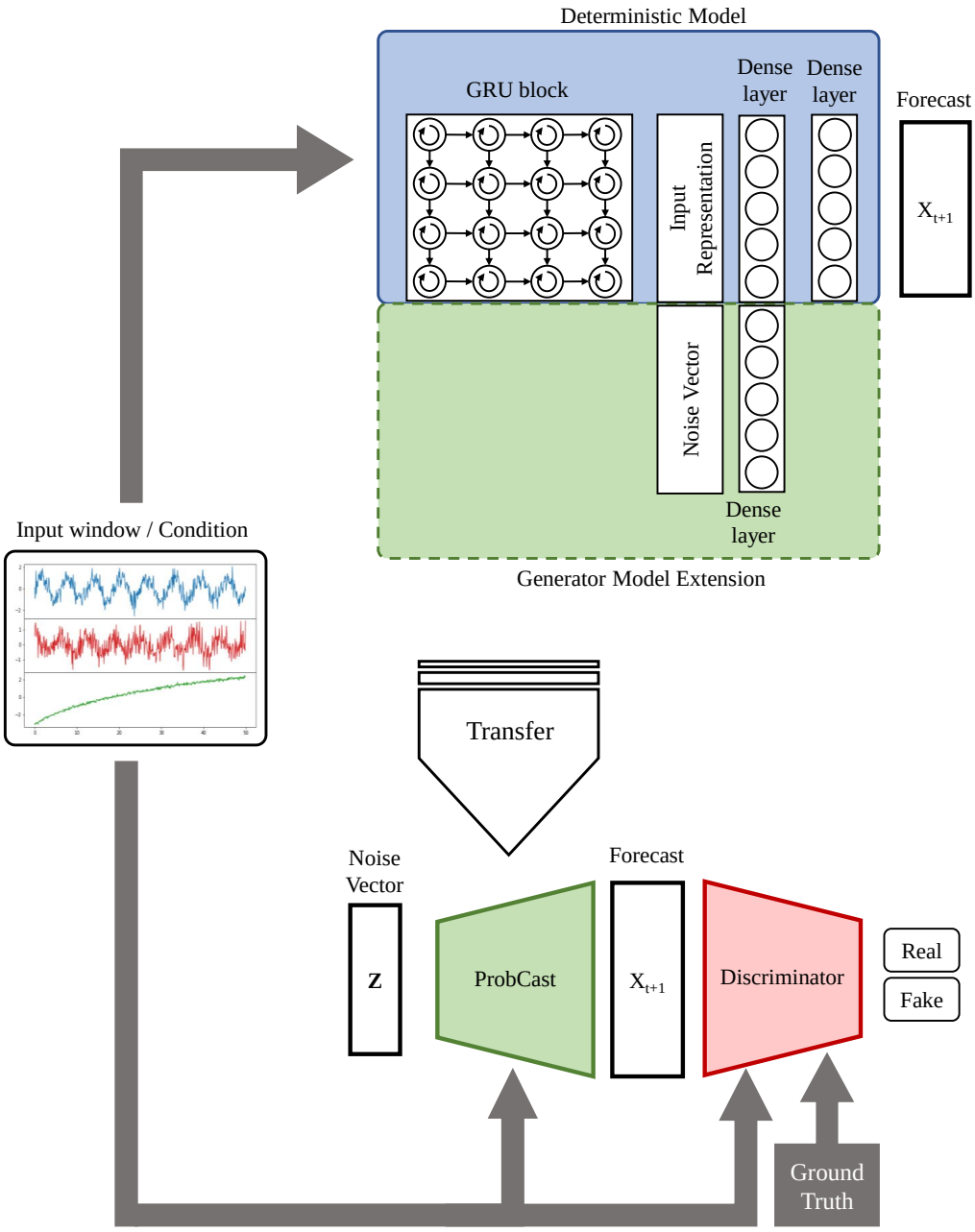


Fig. 7.1 Illustration of the ProbCast framework alongside the adversarial training schema. The procedure cascades from top to bottom, beginning with the deterministic model optimization. This model employs a GRU block to encapsulate input window representations, further transformed via two dense layers to yield the forecast. Subsequent to this, noise vector z integration occurs to create the generator, which, under the support of an apt discriminator, undergoes training within the conditional GAN structure to give rise to the forecaster.

Electricity Consumption Dataset

This dataset encapsulates the electricity consumption, quantified in KWh, of 321 clients. Data was captured at 15-minute intervals spanning from 2012 to 2014. For the purposes of our experiments, the temporal granularity of the dataset was adjusted to represent hourly consumption metrics.

Exchange-Rate Dataset

The Exchange-Rate dataset comprises daily exchange rates of eight nations from 1990 to 2016. The countries under consideration are Australia, Great Britain, Canada, Switzerland, China, Japan, New Zealand, and Singapore.

Table 7.1 delineates the inherent characteristics of both datasets. In line with conventional experimental partitioning, 75% of the data was earmarked for training, 5% was reserved for validation purposes, and the remaining 20% was utilized for testing.

Table 7.1 Characteristics of the evaluated datasets.

	Electricity Dataset	**Exchange-Rate Dataset**
Duration of Data Collection	2012-2014	1990-2016
Dataset length	62,304	7588
Number of features	321	8
Sampling frequency	Hourly	Daily

7.2.2 Experimental Configuration

To ensure robust evaluation and model fidelity, we systematically performed an architecture search prior to each experiment, aiming to identify a proficient deterministic model.

Deterministic Model Training

The Mean Absolute Error (MAE) was selected as the loss function for training the deterministic model. As illustrated in Figure 7.1, the model's architecture leverages a Gated Recurrent Unit (GRU) block [91] for adept representation of the input temporal window. After the GRU block, the model incorporates two dense layers responsible for mapping the derived representations to the forecasting outputs.

ProbCast Construction

After establishing the architecture of the most precise deterministic model, this structure was repurposed for ProbCast development. Specifically, the noise vector was appended to the output of the GRU block, enhancing the model's capability to capture inherent data variability. Moreover, the width of fully connected layers is expanded to accommodate the representation, which is now extended with noise vector as detailed in Figure 7.1.

Discriminator Architecture and Training

An integral component of our methodology is the discriminator, whose role is pivotal in a Generative Adversarial Network (GAN) paradigm. As depicted in Figure 7.2, our discriminator first concatenates the time instance X_{t+1} to the preceding input window, resulting in a chronologically arranged sequence $\{X_{t+1}, X_t, ..., X_0\}$. A GRU block, followed by several dense layers, inspects the temporal consistency and authenticity of this sequence. To pinpoint the optimal architecture, we employed a genetic algorithm approach. Our experimental pipeline was implemented using the PyTorch framework [92].

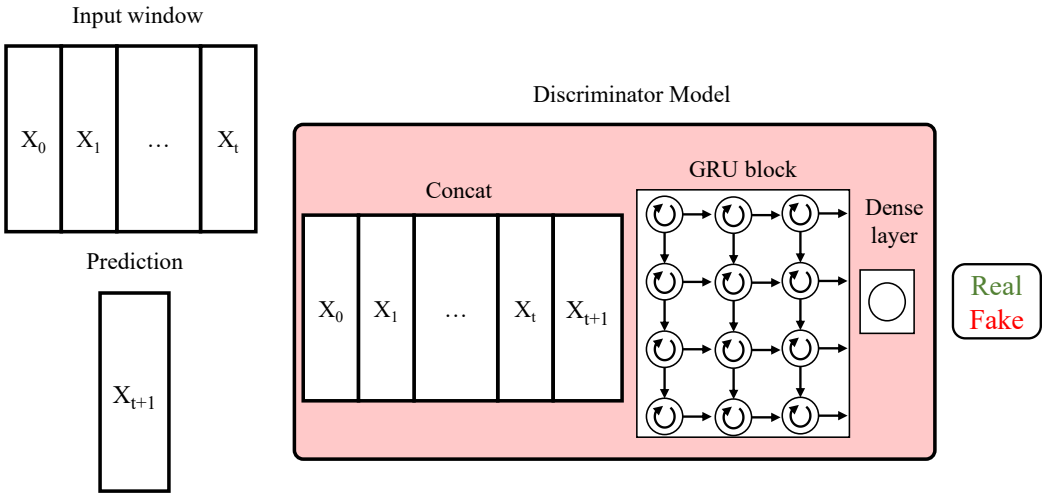


Fig. 7.2 Depiction of the discriminator's architecture within our conditional GAN setup. The GRU block's configuration, in terms of layers and cells, is determined through hyperparameter optimization.

7.2.3 Performance Evaluation

To objectively evaluate and compare the forecasting capabilities of both deterministic and probabilistic models, the Continuous Ranked Probability Score (CRPS) is adopted as the primary performance metric, as defined in Equation(4.7).

Properties of CRPS

The CRPS offers a dual advantage: not only does it evaluate probabilistic forecasts, but it can also assess deterministic ones. When applied to a deterministic forecast, the CRPS simplifies to the Mean Absolute Error (MAE), a widely accepted point-wise error measure. This inherent property of the CRPS facilitates seamless comparisons between the deterministic and probabilistic paradigms, thereby ensuring an unbiased evaluation of both model types. In essence, for deterministic forecasts, the relationship can be concisely stated as:

$$\text{CRPS} = \text{MAE}$$

ProbCast Evaluation

Upon the culmination of the GAN training process, we quantified the performance of both the ProbCast and the deterministic model using CRPS. To compute CRPS for the ProbCast model, we use 200 samples for each input.

7.3 Results and Analysis

The optimal hyperparameter configurations identified for each dataset through our framework are presented in Table 7.2. Meanwhile, Table 7.3 outlines the outcomes of our empirical analyses, comparing the Continuous Ranked Probability Score (CRPS) of the deterministic forecaster against that of the ProbCast across the datasets.

Table 7.2 Optimal hyperparameter configurations for each dataset.

Generator Hyperparameters		
	Electricity	**Exchange-Rate**
Input windows size	174	170
Noise size	303	183
Number of GRU layers	1	1
GRU cells per layer	119	119
Discriminator Hyperparameters		
Number of GRU layers	3	1
GRU cells per layer	146	149

Within the scope of the electricity dataset experimentation, despite structural similarities, the ProbCast demonstrated superior accuracy compared to the deterministic model. This outcome underscores the model's adeptness at furnishing accurate forecasts for multivariate time series, particularly when encountering a large number of features in data. In the context of the exchange-rate dataset, the ProbCast, mirroring its earlier success, surpassed the deterministic model in performance, an achievement made more significant given the dataset's relatively small size.

Such empirical findings confirm the efficacy of our framework in adeptly transforming a deterministic model into its more accurate probabilistic counterpart. An intriguing observation surfaces: Why, despite the well-documented sensitivity of GANs to component architectures, does the deterministic model's architecture, when transposed onto the ProbCast, yield superior results? We speculated that the deterministic model, fine-tuned to determine the mean of prospective outcomes, excels in extracting important features from the input time window, subsequently facilitating the precise transformation of the noise vector z into the future value's probability distribution.

Table 7.3 Performance comparison between the deterministic model and the ProbCast, gauged using CRPS.

Dataset	**Deterministic Model**	**ProbCast**
Electricity	235.96	232.00
Exchange-rate	1.04×10^{-2}	8.66×10^{-3}

7.4 Conclusion

Within the scope of this study, we introduced ProbCast, an advanced probabilistic model tailored for one-step-ahead forecasts in multivariate time series. Leveraging the robust capabilities of conditional Generative Adversarial Networks (GANs), we harnessed them to forecast the conditional probability distribution of future values given their historical counterparts, denoted as $P(X_{t+1}|X_t,..,X_0)$.

In conjunction, we proposed an innovative framework designed to streamline the discovery of optimal architectures for the components of GANs. A salient feature of this framework is its ability to construct the probabilistic model from a deterministic forecaster, thereby enhancing performance. This scheme not only simplifies the independent architectural optimization for

both the generator and discriminator but also furnishes a mechanism to adeptly transform a pre-existing deterministic model into its probabilistic counterpart, enriched in precision and authenticity.

To validate our methodology, we conduct empirical analyses utilizing two public datasets. While its modest size and feature-set characterize the exchange-rate dataset, the electricity dataset stands in distinct contrast due to its extensive volume and multiplicity of features. Benchmarks were established by juxtaposing the efficacy of ProbCast against its deterministic analog. Across both datasets, our ProbCast model surpassed its deterministic peer. These empirical outcomes underscore ProbCast's dual capability: Its prowess in determining complex patterns from sparse data and its proficiency in distinguishing inter-feature dependencies and forecasting with precision, even when confronted with comprehensive datasets. Such revelations certify the success of our framework, endorsing a structured and intuitive paradigm for transitioning from accurate deterministic models to their enhanced probabilistic derivatives.

Part III

Time Series Generative Models and Assessment

Chapter 8

Class Conditional Time Series Generation Assessment

The rapid rise of implicit generative models, especially with the advent of Generative Adversarial Networks (GANs) [5], has catalyzed a paradigm shift in the realm of data generation and modeling across diverse domains, including images, video, music, and speech. Despite their significant success, a critical limitation in their wide-scale adoption, particularly in the time series domain, is the lack of a universally accepted metric to evaluate and benchmark their performance rigorously. While certain domains, notably images, benefit from established metrics like the Inception Score (IS) and Fréchet Inception Distance (FID) [93, 94], their counterparts for time series data remain unavailable. This cavity not only hinders the comparative analysis of existent models but also hampers potential advancements in the field.

The essence of a generative model is embodied in its ability to perfectly replicate the data distribution, denoted as P_{data}, through its learned distribution P_{model}. A reliable assessment metric, δ, should be proficient in determining the alignment between these distributions. In light of these requirements and inspired by the success of IS and FID in the image domain, this chapter introduces two novel metrics for the time series domain: the InceptionTime Score (ITS) and the Fréchet Inception Time Distance (FITD). The objective of our investigation is twofold: to establish the feasibility of transforming image domain evaluation paradigms to the time series domain and to contextualize our newly coined metrics with regard to existing ones; notably the Train on Synthetic Test on Real (TSTR) and Train on Real Test on Synthetic (TRTS) [95].

The content of this chapter has been adopted from A. Koochali et al. 'Quantifying Quality of Class-Conditional Generative Models in Time Series Domain' Applied Intelligence, July 26, 2023. doi: 10.1007/s10489-023-04644-y.

To verify the integrity and utility of our proposed metrics, we performed a comprehensive experimental campaign spanning 80 datasets from the UCR archive. This empirical rigor seeks to illuminate the quality of the samples generated by P_{model} and its prowess in replicating the mode space. By comparing ITS and FITD with TSTR and TRTS, our intent is to provide researchers with a rational perspective into the mechanics and implications of our assessment tools, thereby paving the way for the confident deployment of implicit generative models in time series analysis.

8.1 Related Work

Assessing the performance of implicit generative models quantitatively remains a formidable challenge [93]. We do not have access to distribution learned by the model explicitly to validate them. Furthermore, manual inspection of artificial data authenticity is not feasible, particularly for unintuitive data such as time series data. Hence, the application of generative models for fabricating artificial time series is limited. In this section, we explore four primary approaches for evaluating these models on the time series domain and review the relevant literature.

The first approach is to assess the resemblance between the statistical characteristics of real and generated data as the proxy for the closeness between real data and artificial data probability distribution. Wiese et al. [96] utilized several metrics and scores to quantify the similarity of various statistical properties, such as empirical probability density function (epdf), different moments of the probability distribution, auto-correlation factor (ACF), and correlation. [97] suggested cross-correlation as the difference between the auto-correlation estimator by real data and synthetic data. Later, [98] used cross-correlation as a difference between the correlation between generated and real multivariate time series channels. Furthermore, [98] used the difference between the histogram of real and artificial samples as well as Kullback–Leibler divergence [99] between the distribution of real and generated samples to measure the authenticity of generated time series. These measures can offer insight into the extent to which the probability distributions of the real and artificial data align for unconditional time series generation. However, they are not effective for class conditional time series generation as they do not capture the ability of the generative model to respect the given class during the generation process.

https://www.cs.ucr.edu/~eamonn/time_series_data_2018/

The second approach investigates the similarities between real and fabricated samples through a binary classification task. In this method, the classifier is trained to distinguish between real and artificial data and then tests against a holdout set from the generated sample. If the generated samples were similar to real ones, the classifier could not classify them accurately; hence, a high classification error would indicate realistic artificial data. [100] utilized this method with an LSTM-based neural network as the classifier. [98] utilized classifiers based on K-nearest neighbor and random forest to evaluate generative models with this method. Similar to the first approach, this method does not assess the incorporation of classes.

The third approach for evaluating time series generative models involves using a surrogate task to assess their performance indirectly. The underlying assumption is that if a generative model accurately captures the underlying data distribution, it will perform well in downstream tasks. In line with this intuition, Esteban et al. [95] proposed two metrics for evaluating class conditional GANs for the time series domain using classification as a downstream task. The first metric, "Train on Real, Test on Synthetic" (TRTS), involves training a classifier using real data and testing it with synthetic data. This metric is similar to GAN-test [101], commonly used in the image domain. The second metric, "Train on Synthetic, Test on Real" (TSTR), involves training a classifier using synthetic data and testing it with real data. In the image domain, this metric is referred to as GAN-train [101] and Classification Accuracy Score (CAS) [102]. Although these metrics have been extensively studied in the image domain [101, 102], there is a lack of research regarding their application in the time series domain. Additionally, the absence of consensus on classifier structure in the time series domain makes it challenging to compare generative models using these metrics across different studies.

The fourth evaluation approach involves assessing the quality of generated samples in the representation space. For instance, in [103], authors manually extracted seven features from time series data and applied similarity scores based on Jensen-Shannon divergence [104] and cosine similarity to assess the quality of generated samples. In another work, inspired by Fréchet Inception Distance (FID) [105], Smith et al. [106] proposed 1D FID as the Fréchet Distance between real and synthetic samples representation. The authors used a CNN-based classifier to obtain representations; however, they did not disclose the structure and network weights of the classifier. As a result, generative models cannot be compared using 1D FID across studies, similar to TSTR and TRTS. Additionally, the authors noted that 1D FID did not always agree with the visual inspection, highlighting the need for further investigation into its strengths and limitations.

This study introduces two metrics with a defined assessment pipeline for class conditional time series generation. We also propose a framework for TSTR and TRTS that enables comparison across studies. Finally, we explore the strengths and weaknesses of each metric through various experiments to provide a more interpretable evaluation.

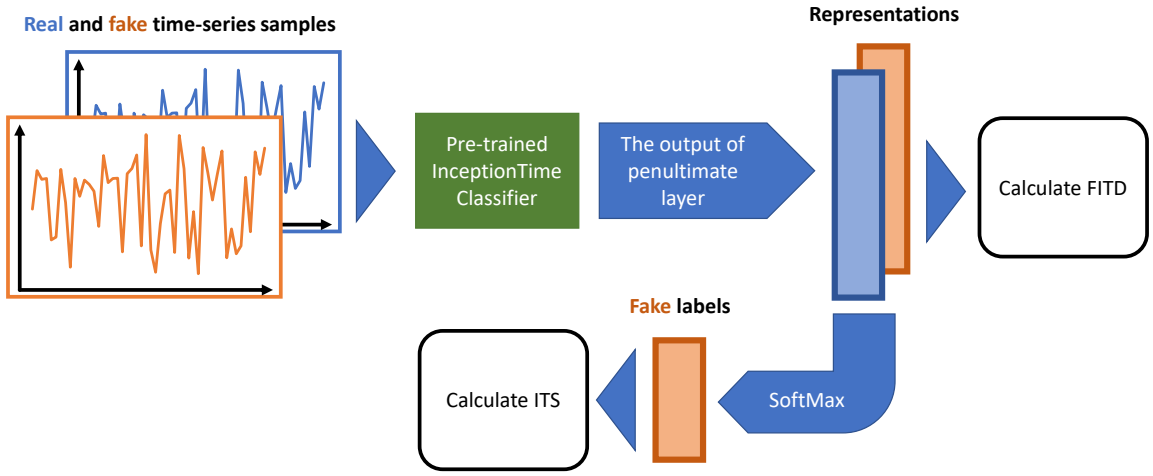


Fig. 8.1 The proposed evaluation pipeline for FITD and ITS.

8.2 Quantitative Assessment for Deep Generative Models on the Time Series Domain

In this section, we present our evaluation pipeline for class conditional time series generative models and introduce two new evaluation metrics based on this pipeline. An overview of this pipeline is represented graphically in Fig. 8.1. We aim to establish a standardized framework that can be easily adopted by other studies, enabling comparison of assessments across different studies. The existing and proposed metrics assess the quality of generated samples based on highly classifiable and diverse features. To have comparable results across various studies, it is crucial to define a standard framework for obtaining these features. Therefore, in this study, we propose to adopt InceptionTime [107] for determining our evaluation metrics (the green block in Fig. 8.1).InceptionTime is a CNN-based time series classifier that acquired impressive accuracy on the time series classification task. We utilize a similar network structure across all datasets. Still, given the high variability in dynamics across different time series datasets, it is not feasible to use a single pre-trained network for all datasets. Hence, the InceptionTime network is trained separately for each dataset, adopting

the same network structure and training pipeline as the authors of InceptionTime provided in the project git repository.

8.2.1 InceptionTime Score (ITS)

Inspired by IS for assessing generative models in the image domain, we proposed the InceptionTime Score (ITS) as the evaluation metric for the quality synthetic data in the time series domain. Given x as the set of synthetic time series samples and y as their corresponding labels, we expect high-quality generated data to have low entropy conditional label distribution $p(y \mid x)$. This is to be compared with the data's marginal distribution $p(y)$, which is expected to be high for diverse samples. Thus, in the ideal case, the shapes of $p(y \mid x)$ and $p(y)$ are opposite: namely, narrow vs. uniform. The score should reflect this property and be higher the more the conditional label, and the marginal distributions differ. This is achieved by taking the exponentiation of their respective KL divergence:

$$\begin{aligned} \text{ITS} &= \exp\left(H(y) - \mathbb{E}_{\mathbf{x}}[H(y \mid \mathbf{x})]\right) \\ &= \exp\left(\mathbb{E}_{\mathbf{x}}[\mathbb{KL}(p(\mathrm{y} \mid \mathbf{x}) \| p(\mathrm{y}))]\right). \end{aligned} \tag{8.1}$$

By definition, ITS is a positively oriented metric. Its lowest value is 1.0, and its upper bound is the number of classes in the dataset. To acquire the label of synthetic time series data, we employed a pre-trained InceptionTime network.

8.2.2 Fréchet InceptionTime Distance (FITD)

ITS relies solely on the statistics of the generated samples and ignores real samples. Hence, it assigns a high score to a model with sharply distributed marginal and diverse training samples, regardless of whether the generated samples follow the target distribution. To address this problem in the image domain, Heusel et al. [105] proposed Fréchet Inception Distance (FID). To exploit FID on time series data, we defined Fréchet InceptionTime Distance (FITD). We extract the feature vectors for real and generated samples from the penultimate layer of a pre-trained InceptionTime Classifier. We assume each of these feature vectors follows a continuous multivariate Gaussian. Subsequently, we calculate the Fréchet Distance (also known as Wasserstein-2 distance) between these two Gaussian, i.e.

$$\text{FITD(r, g)} = \left\|\mu_r - \mu_g\right\|_2^2 + \text{Tr}\left(\Sigma_r + \Sigma_g - 2\left(\Sigma_r \Sigma_g\right)^{\frac{1}{2}}\right), \tag{8.2}$$

https://github.com/hfawaz/InceptionTime

where (μ_r, Σ_r) and (μ_g, Σ_g) are the mean and covariance matrices of the real data and generated data, respectively, lower FITD indicates a smaller distance between data distribution and real distribution, and the minimum value is zero. FITD is a robust and efficient metric; however, its assumption on multivariate Gaussian distribution in feature space is not always true.

8.2.3 Assessment Based on Classification Accuracy

We can use a classifier to explicitly benefit from labeled data to assess the class-conditional generative models. The core idea is that if a generative model can generate realistic data samples, it should perform well in the downstream tasks. In this case, a classifier can be trained on real data and tested on synthetic data regarding classification accuracy. This thesis refers to this metric as TRTS (Train on Real, Test on Synthetic). TRTS implies that if the distribution learned by the generative model P_{model} matches the data distribution P_{data}, then a discriminative model trained on samples from P_{data} can accurately classify generated samples from P_{model}. TRTS outputs low accuracy if generated samples fall out of P_{data}. However, if $P_{model} \subset P_{data}$, then TRTS might assign a high accuracy, in neglection of the fact that the mode space is only partially covered by P_{model}.

Another classifier-based assessment metric is to train a model on synthetic data and test it on real data. We refer to this metric as TSTR (Train on Synthetic, Test on Real). Similar to TRTS, the TSTR argues that if $P_{model} \approx P_{data}$, then a classifier trained on generated samples can score high accuracy while classifying real samples. Unlike TRTS, the TSTR can detect the situation where P_{model} partially covers P_{data}; however, it cannot reflect the existence of synthetic samples that do not follow P_{data}. In other words, TSTR provides high accuracy even if $P_{data} \subset P_{model}$. This latter case is more intuitively known as an over-parameterized model.

In this study, we employed the InceptionTime model as the classifier for calculating TRTS and TSTR.

8.3 Experiment

In contrast to evaluating models, which is accomplished via statistical analysis of the models on a large and diverse enough variety of datasets, assessing an evaluation metric presents a more challenging problem. The purpose of an evaluation metric is to determine whether a

metric is reliable. Reliability in the context of generative models means that the evaluation metric can effectively quantify the quality of generated samples. Furthermore, it is essential that the metric can identify common problems faced by generative models. Thus, we investigate the reliability of our proposed metric through three primary directions:

- Decline in Quality
- Mode Drop, and
- Mode Collapse.

In the following, we designed three experiments to replicate these situations and investigate the discriminative ability of ITS, FITD, TRTS, and TSTR in the time series domain.

8.3.1 Empirical Evaluation Score

In our experiments, we train InceptionTime on the train set of a dataset and calculate our target scores (ITS, FITD, TRTS, and TSTR) on the respective test set to obtain the base score ($\text{score}_{\text{base}}$) for each dataset. Since the test set is obtained from data distribution, we consider $\text{score}_{\text{base}}$ as the best score we can acquire on each dataset empirically. Also, we indicate the score of generated samples as $\text{score}_{\text{gen}}$. Finally, we define

$$\text{rel(score)} = \text{score}_{\text{base}} - \text{score}_{\text{gen}} \tag{8.3}$$

as the score of generated samples relative to the base score. We expect $\text{rel(ITS)} \geq 0$, $\text{rel(TRTS)} \geq 0$, $\text{rel(TSTR)} \geq 0$ and $\text{rel(FITD)} \leq 0$ in all cases. In other words, we do not expect a better score than the base score.

8.3.2 Datasets

The UCR archive [108] is a collection of 128 univariate time series datasets designed for the classification task. It thus enables us to perform our experiments on a broad spread of datasets with various properties across different domains. Furthermore, the InceptionTime model has demonstrated impressive performance in the classification task on the UCR archive. As discussed above, we need highly classifiable and diverse features to calculate FITD and ITS precisely. Therefore, for our experimental setting, we select a subset of datasets from the UCR archive on which the InceptionTime model acquires at least 80% accuracy, resulting in 80 datasets. Table 8.1 lists the names of these datasets and their properties, and Figure 8.2 outlines the accuracy scored by the InceptionTime model.

Table 8.1 This table presents the list of selected datasets from the UCR Archive alongside their properties.

No	Name	Type	Class	Length	No	Name	Type	Class	Length
1	Adiac	Image	37	176	41	ProximalPhalanxTW	Image	6	80
2	ArrowHead	Image	3	251	42	ShapeletSim	Simulated	2	500
3	Beef	Spectro	5	470	43	SmallKitchenAppliances	Device	3	720
4	BeetleFly	Image	2	512	44	SonyAIBORobotSurface1	Sensor	2	70
5	BirdChicken	Image	2	512	45	SonyAIBORobotSurface2	Sensor	2	65
6	Car	Sensor	4	577	46	Strawberry	Spectro	2	235
7	CBF	Simulated	3	128	47	SwedishLeaf	Image	15	128
8	ChlorineConcentration	Sensor	3	166	48	Symbols	Image	6	398
9	Coffee	Spectro	2	286	49	SyntheticControl	Simulated	6	60
10	Computers	Device	2	720	50	ToeSegmentation1	Motion	2	277
11	CricketX	Motion	12	300	51	ToeSegmentation2	Motion	2	343
12	CricketY	Motion	12	300	52	Trace	Sensor	4	275
13	CricketZ	Motion	12	300	53	TwoLeadECG	ECG	2	82
14	DiatomSizeReduction	Image	4	345	54	TwoPatterns	Simulated	4	128
15	DistalPhalanxOutlineCorrect	Image	2	80	55	UWaveGestureLibraryX	Motion	8	315
16	ECG5000	ECG	5	140	56	Wafer	Sensor	2	152
17	ECGFiveDays	ECG	2	136	57	Wine	Spectro	2	234
18	FaceAll	Image	14	131	58	Worms	Motion	5	900
19	FaceFour	Image	4	350	59	WormsTwoClass	Motion	2	900
20	FacesUCR	Image	14	131	60	Yoga	Image	2	426
21	FordA	Sensor	2	500	61	ACSF1	Device	10	1460
22	FordB	Sensor	2	500	62	BME	Simulated	3	128
23	GunPoint	Motion	2	150	63	Chinatown	Traffic	2	24
24	Ham	Spectro	2	431	64	EthanolLevel	Spectro	4	1751
25	HandOutlines	Image	2	2709	65	FreezerRegularTrain	Sensor	2	301
26	ItalyPowerDemand	Sensor	2	24	66	FreezerSmallTrain	Sensor	2	301
27	LargeKitchenAppliances	Device	3	720	67	Fungi	HRM	18	201
28	Lightning2	Sensor	2	637	68	GunPointAgeSpan	Motion	2	150
29	Lightning7	Sensor	7	319	69	GunPointMaleVersusFemale	Motion	2	150
30	MedicalImages	Image	10	99	70	GunPointOldVersusYoung	Motion	2	150
31	MiddlePhalanxOutlineCorrect	Image	2	80	71	HouseTwenty	Device	2	2000
32	MoteStrain	Sensor	2	84	72	InsectEPGRegularTrain	EPG	3	601
33	NonInvasiveFetalECGThorax1	ECG	42	750	73	InsectEPGSmallTrain	EPG	3	601
34	NonInvasiveFetalECGThorax2	ECG	42	750	74	PigArtPressure	Hemodynamics	52	2000
35	OliveOil	Spectro	4	570	75	PigCVP	Hemodynamics	52	2000
36	OSULeaf	Image	6	427	76	PowerCons	Power	2	144
37	PhalangesOutlinesCorrect	Image	2	80	77	Rock	Spectrum	4	2844
38	Plane	Sensor	7	144	78	SemgHandGenderCh2	Spectrum	2	1500
39	ProximalPhalanxOutlineAgeGroup	Image	3	80	79	SmoothSubspace	Simulated	3	15
40	ProximalPhalanxOutlineCorrect	Image	2	80	80	UMD	Simulated	3	150

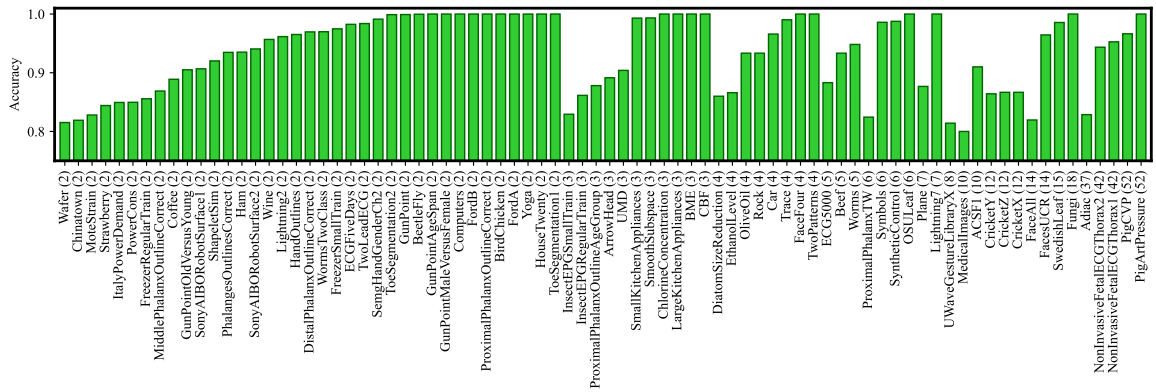


Fig. 8.2 The list of 80 datasets from the UCR archive alongside the accuracy of the InceptionTime classifier on these datasets. The numbers in the parentheses indicate the number of classes in the dataset.

8.4 Results and Discussion

We perform three sets of experiments to analyze the robustness of our evaluation metrics against Decline in Quality, Mode Drop, and Mode Collapse. We will start our evaluation with Decline in Quality in the following section.

8.4.1 Experiment 1 - Decline in Quality

One requirement for a reliable assessment metric is that it should quantitatively express the quality of the generated samples. To investigate this experimentally, we added a noise signal to the samples in the test set to simulate the decrease in quality. The noise is sampled from a Gaussian distribution with $\mu = 0$, and σ is selected from an equally spaced grid of values in $[0,5]$. The standard deviation value indicates the noise strength and the amount of corruption in the original data. We expect the assessment scores to worsen with the increase in standard deviation. Figures 8.3 to 8.9 present our experiment's results on all datasets.

FITD: The FITD response behaves differently than others. Since FITD does not have an upper bound, it increases with the increase of corruption into data while other scores converge to their lower bound at some noise strength.

TRTS and ITS: The behavior of TRTS and ITS are very similar. Both ITS and TRTS use the InceptionTime model trained on the train set as the backbone of their computation. Once the extent of the decline in quality passes from a certain threshold ($\sigma > \Delta$ where The value of Δ depends on the scale of the data), the classifier fails to classify the samples, and its prediction is not better than a random guess. The TRTS converges to random guess accuracy, which depends on the number of classes on the dataset, and ITS converges to 1.0.

TSTR: The TSTR response has more variance than TRTS. The reason is that TRTS is trained on a train-set of real data, which does not change during experiments, while TSTR is trained on synthetic data, and as a result, we trained a new model for each value of σ.

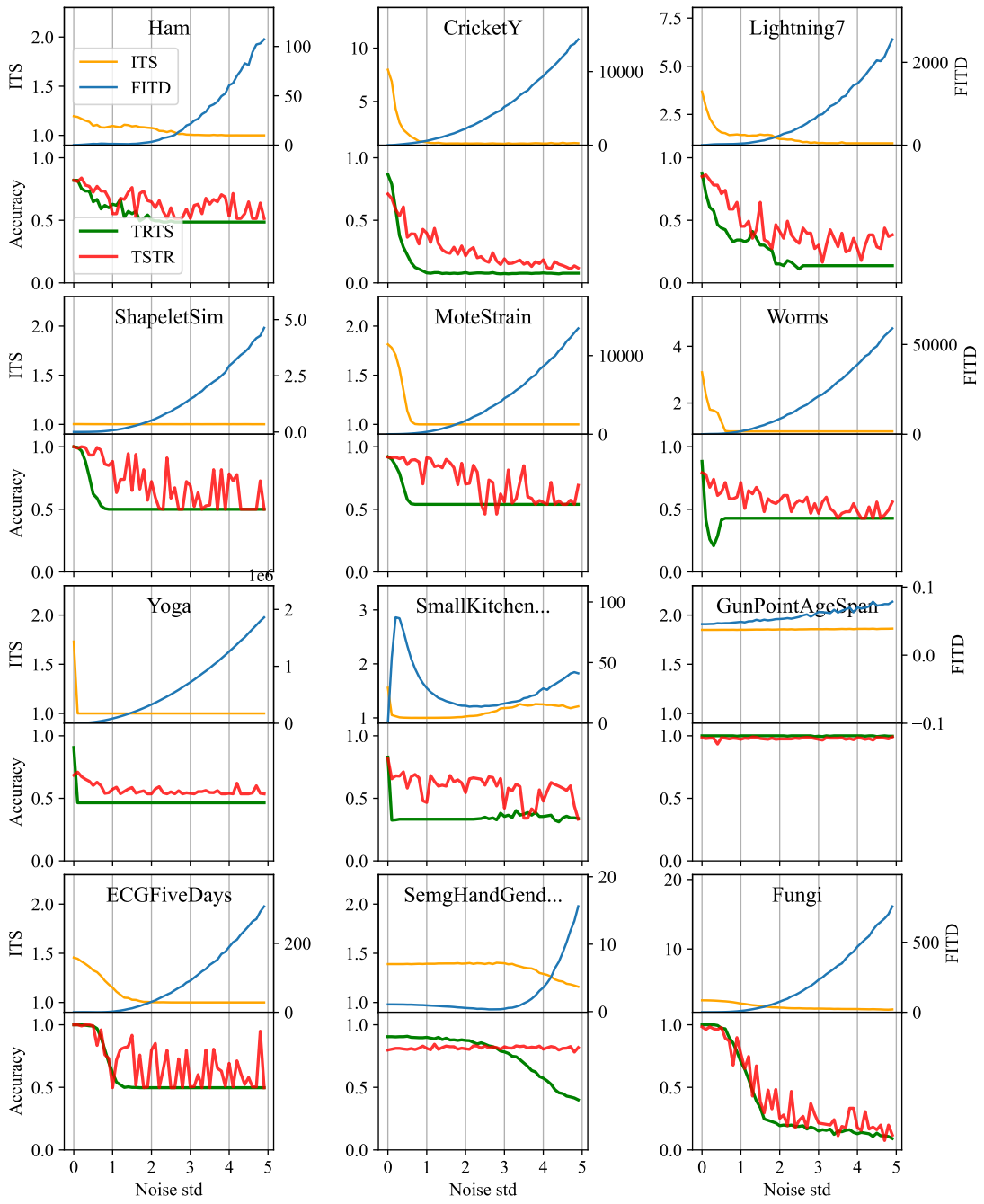


Fig. 8.3 Changes in studied metrics when data quality is declined by introducing noise into data progressively.

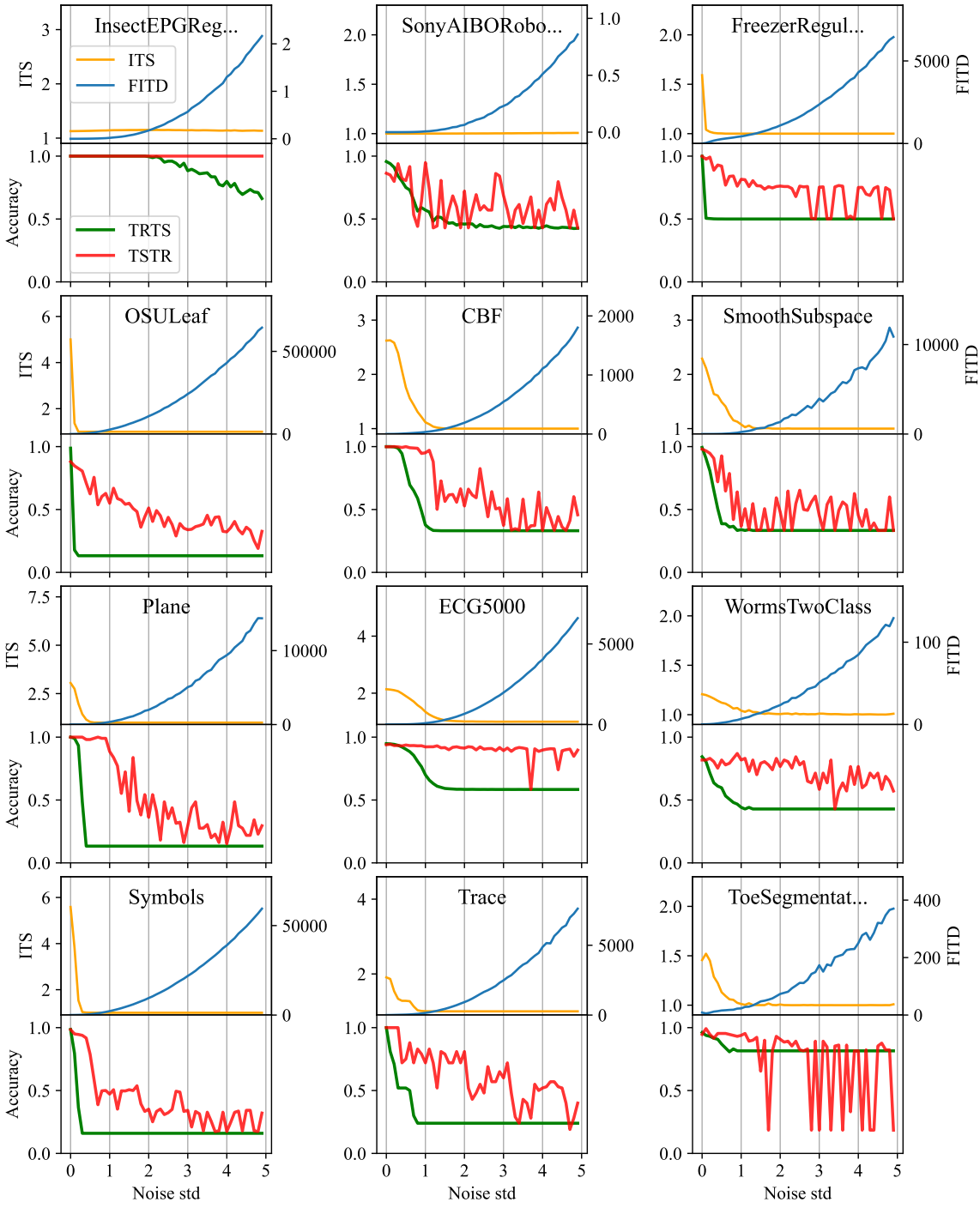


Fig. 8.4 Changes in studied metrics when data quality is declined by introducing noise into data progressively.

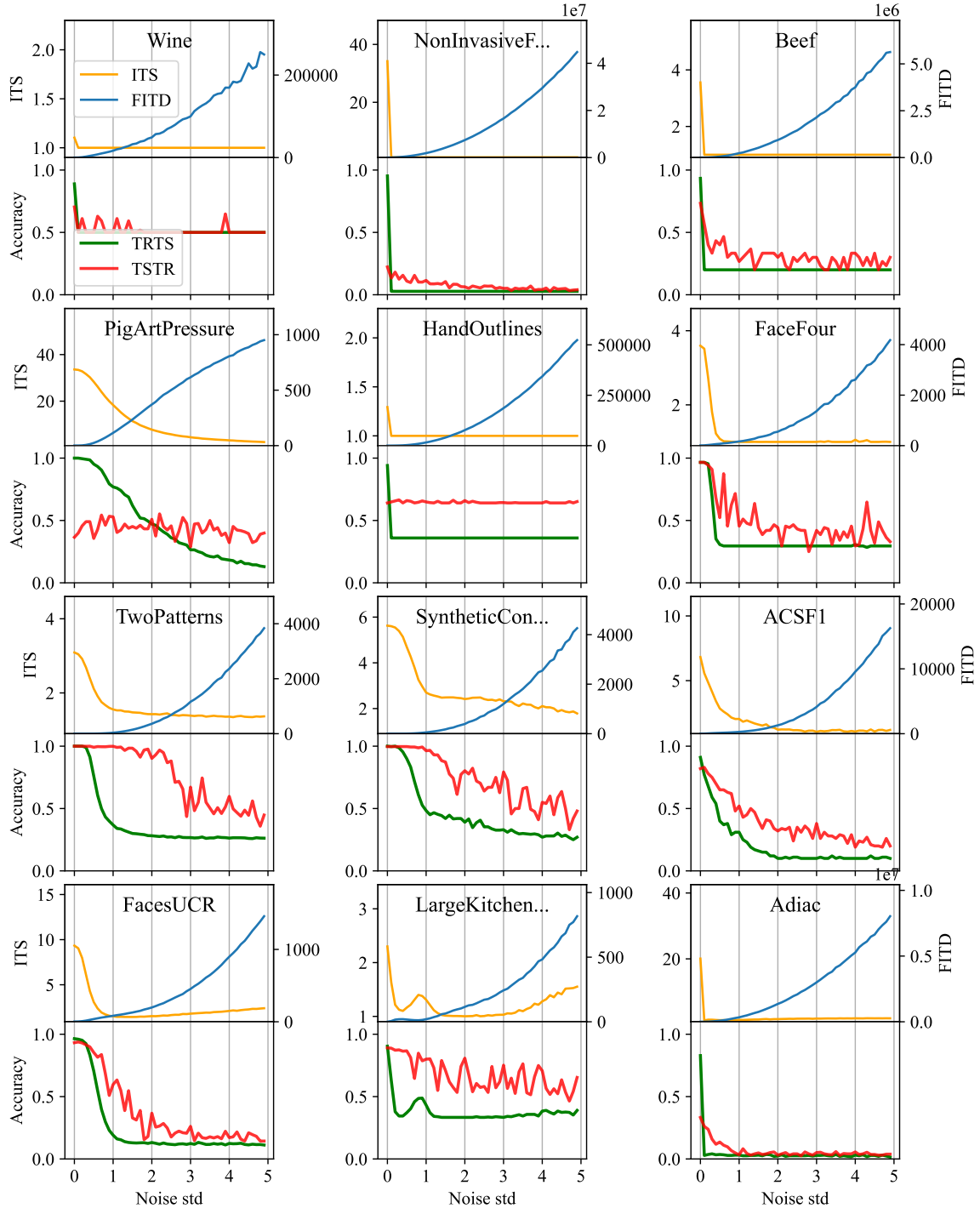


Fig. 8.5 Changes in studied metrics when data quality is declined by introducing noise into data progressively.

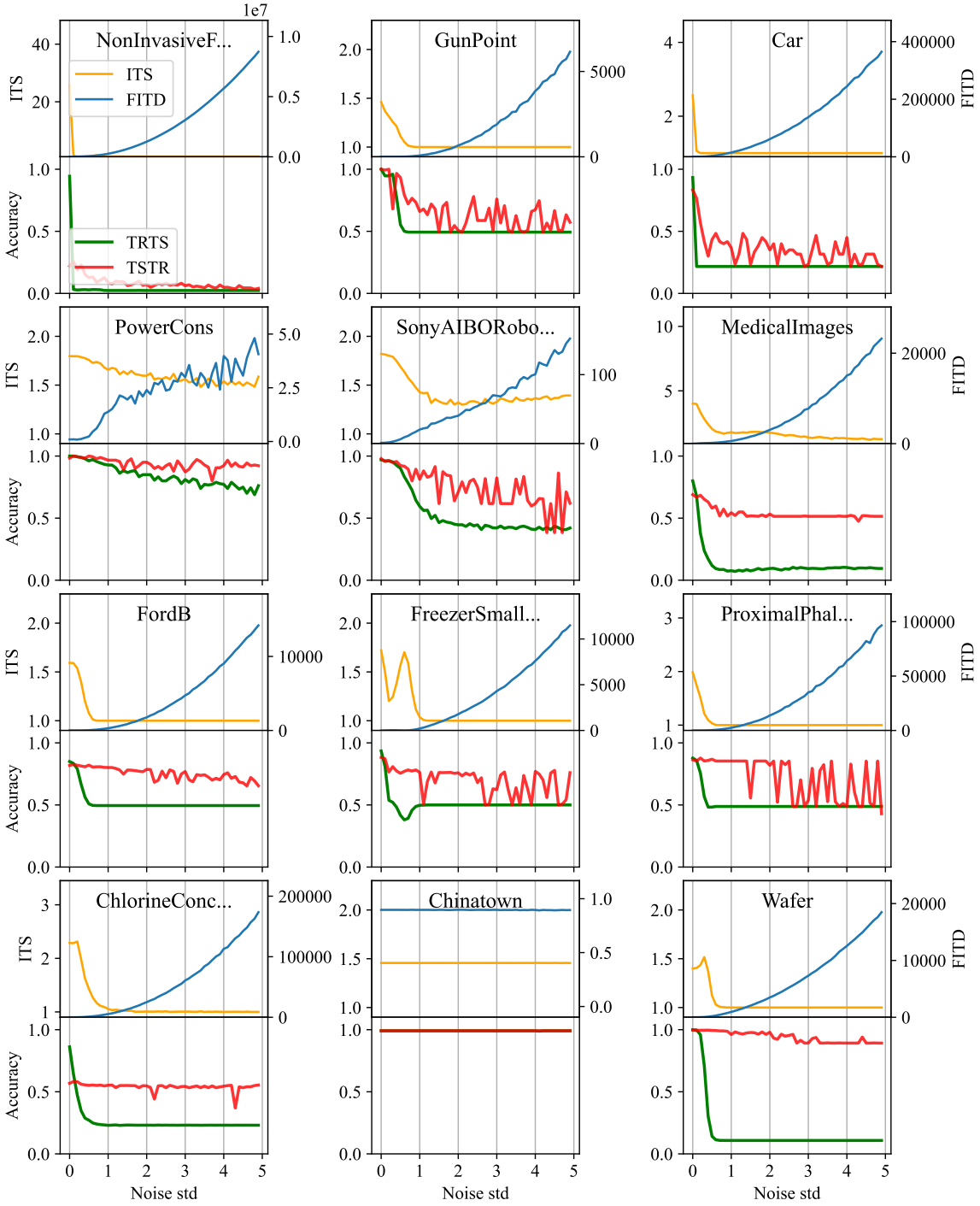


Fig. 8.6 Changes in studied metrics when data quality is declined by introducing noise into data progressively.

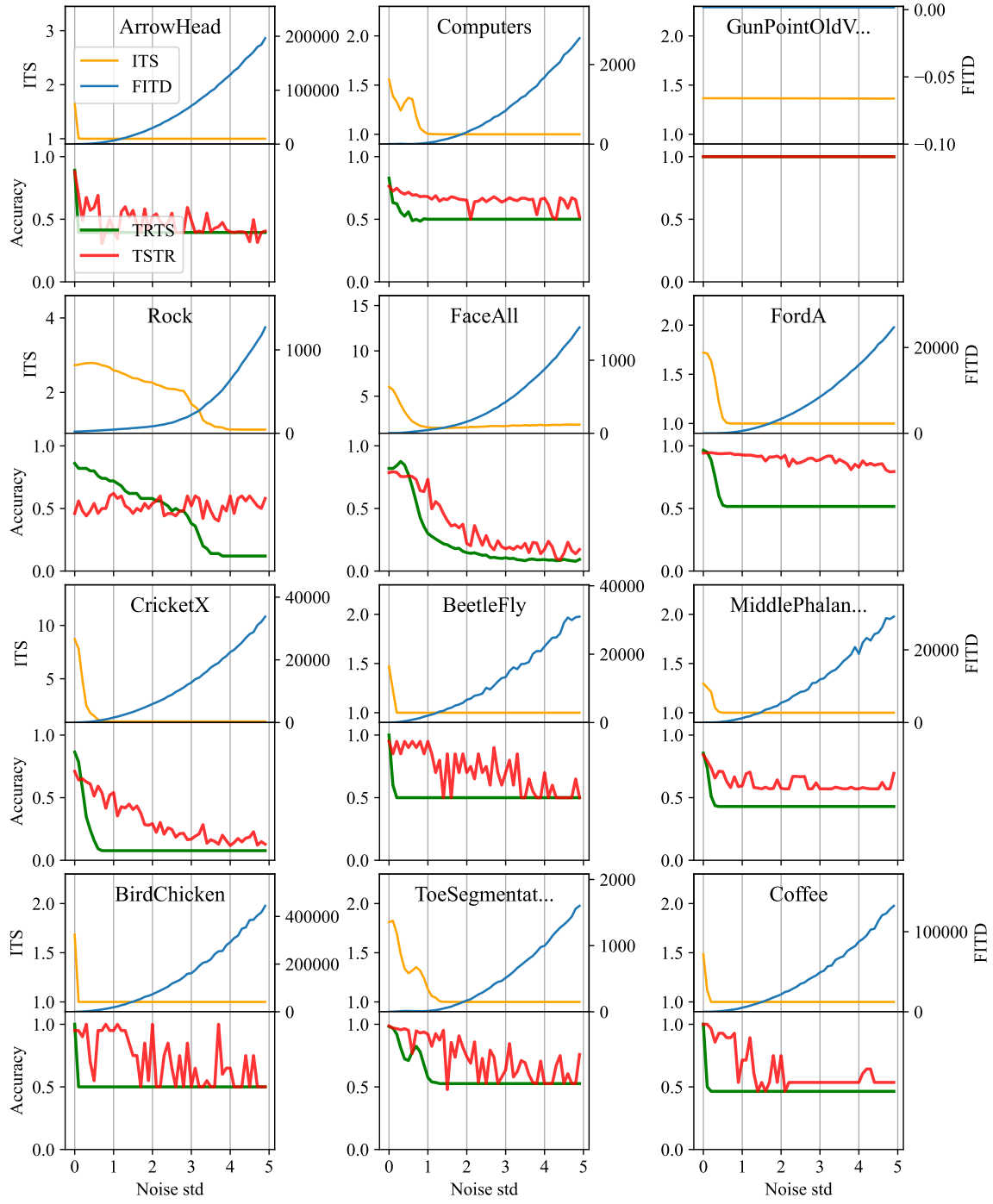


Fig. 8.7 Changes in studied metrics when data quality is declined by introducing noise into data progressively.

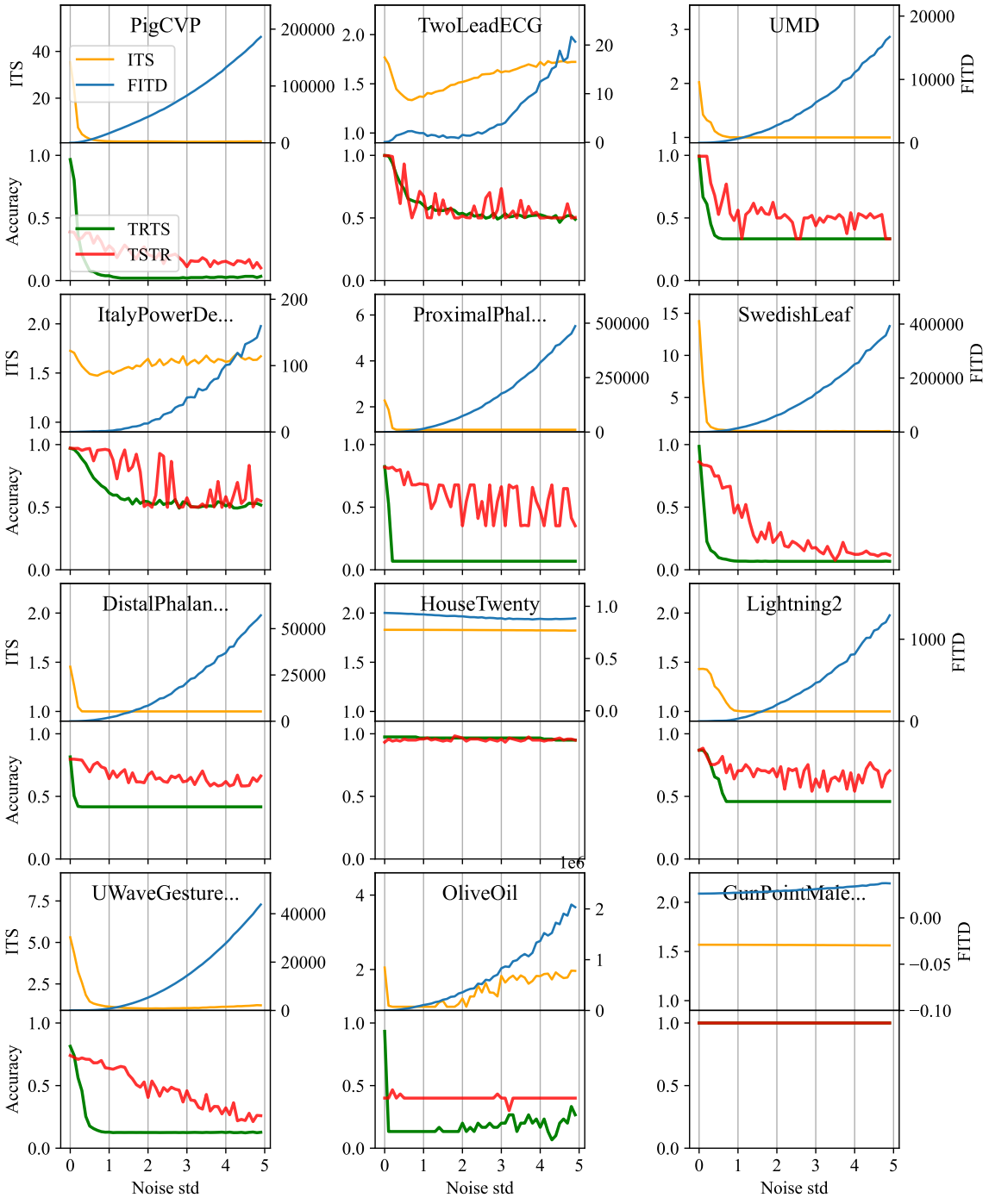


Fig. 8.8 Changes in studied metrics when data quality is declined by introducing noise into data progressively.

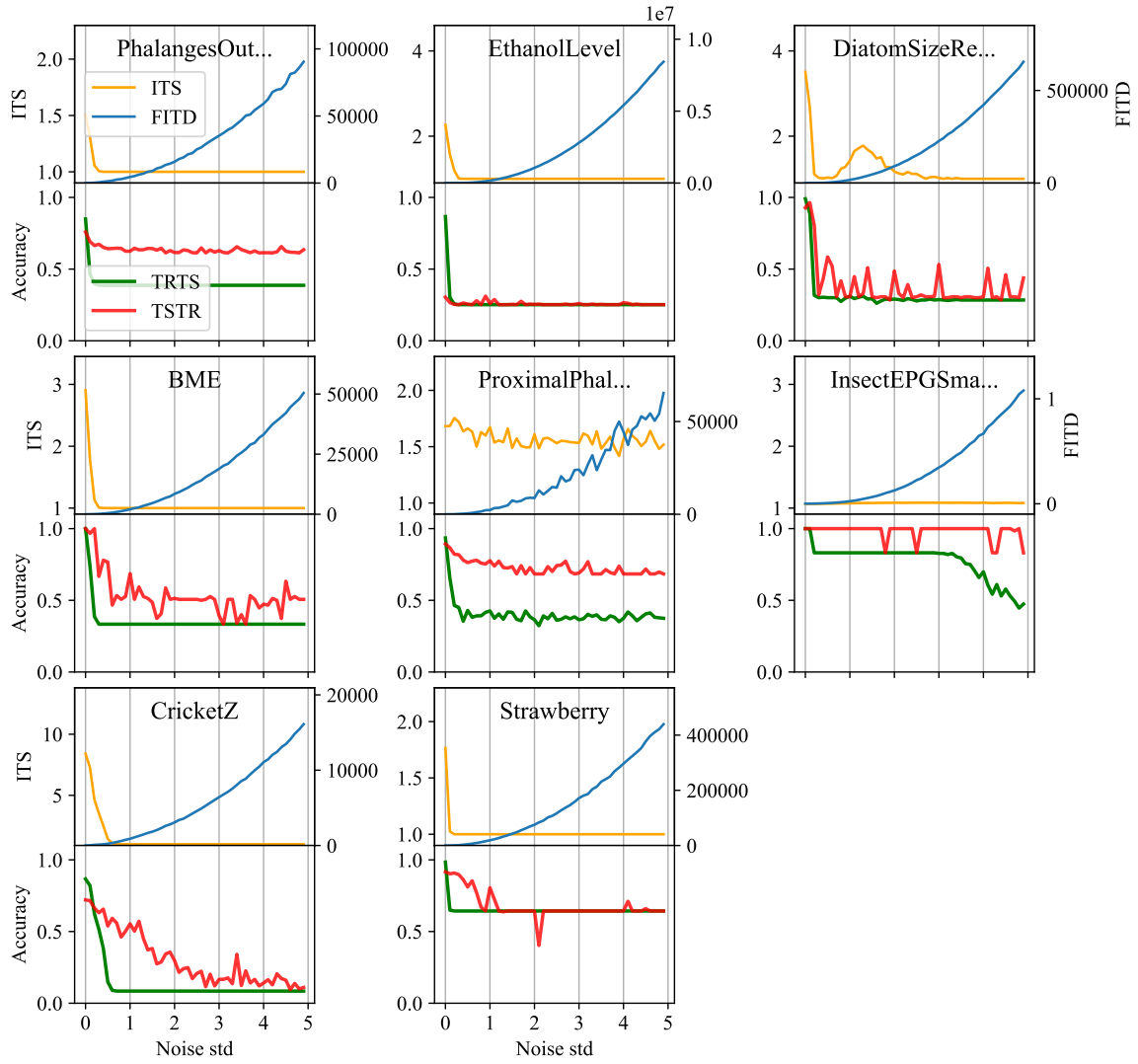
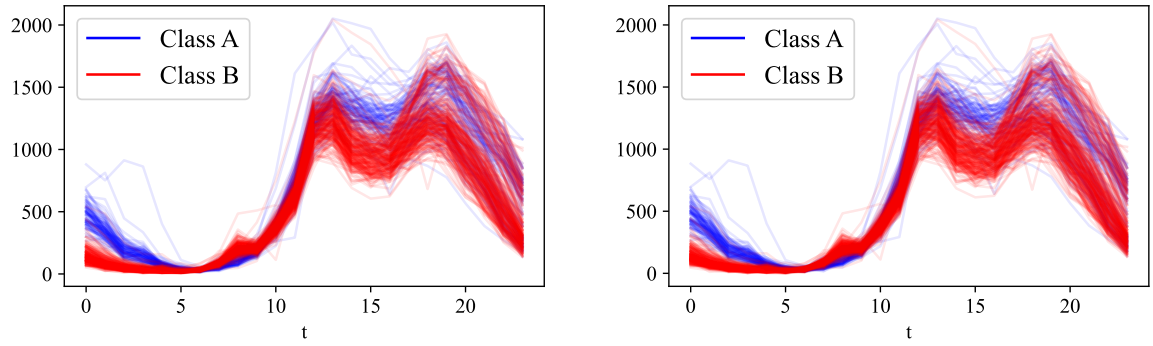


Fig. 8.9 Changes in studied metrics when data quality is declined by introducing noise into data progressively.

We need more substantial noise to corrupt the data with a larger data scale. For instance, it seems that our scores cannot detect the presence of noise on data in the Chinatown data set in Figure 8.6. However, Figure 8.10 reveals that this data set has a great scale, ranging approximately between [0,2000]. Therefore, we need a much larger σ to corrupt the data meaningfully.

8.4.2 Experiment 2 - Mode Drop

Mode Drop happens when the generative model ignores some modes of real data while generating artificial samples. This could be due to a lack of model capacity or inadequate



(a) The original data

(b) Data after adding noise with $\sigma = 5$

Fig. 8.10 Visualization of China town dataset before and after inducing noise into data

optimization [109]. We designed three experimental scenarios, namely Single Mode Drop, Extreme Mode Drop, and Successive Mode Drop, to evaluate the capabilities of ITS, FITD, TSTR, and TRTS in recognizing mode drop in the time series domain.

Single Mode Drop

In the first experiment, we remove all the samples belonging to one class from the test set to simulate the mode drop scenario. We calculate all scores for the mode drop caused by removing each class. Hence, for the dataset with N classes, we would have N values for each score. Figures 8.11 and 8.12 illustrate $rel(score)$ of our scores' responses on all datasets.

FITD: The changes in FITD depend on the degree to which the removed class affects the properties of assumed Gaussian distribution in latent space. In most datasets, the drop of a single class did not change the Gaussian distribution properties in latent space significantly. Thus, the FITD reflects the single-mode drop poorly. On the other hand, on a few datasets, the FITD response with high variance indicates that at least one of the class samples significantly impacts the mean and covariance matrix of points in latent space. Since the feature vectors are generated with a non-linear transformation to a high dimensional space, it is impossible to interpret the FITD response given the samples in the data space.
ITS: The ITS response is mostly positive but has a great variance. When we remove a class, we change the diversity of labels. Therefore, we expect that $H(P(y \mid x))$ remains unaffected while $H(P(y))$ decreases due to the reduction in diversity. The drop of each class affects $H(P(y))$ differently, which results in a high variance between responses. If the distribution of labels is closer to a uniform distribution, the drop of each class will decrease the $H(P(y))$ similarly. In contrast, if the label distribution is heavily unbalanced, then the drop of a major

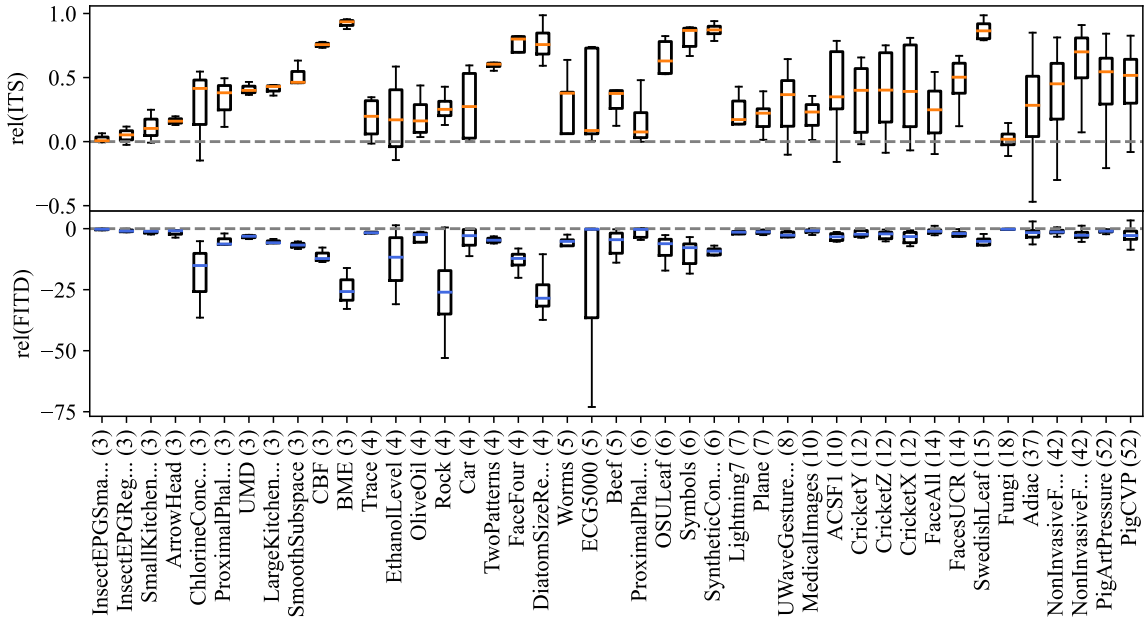


Fig. 8.11 Relative ITS and FITD score when one mode is dropped from a dataset.

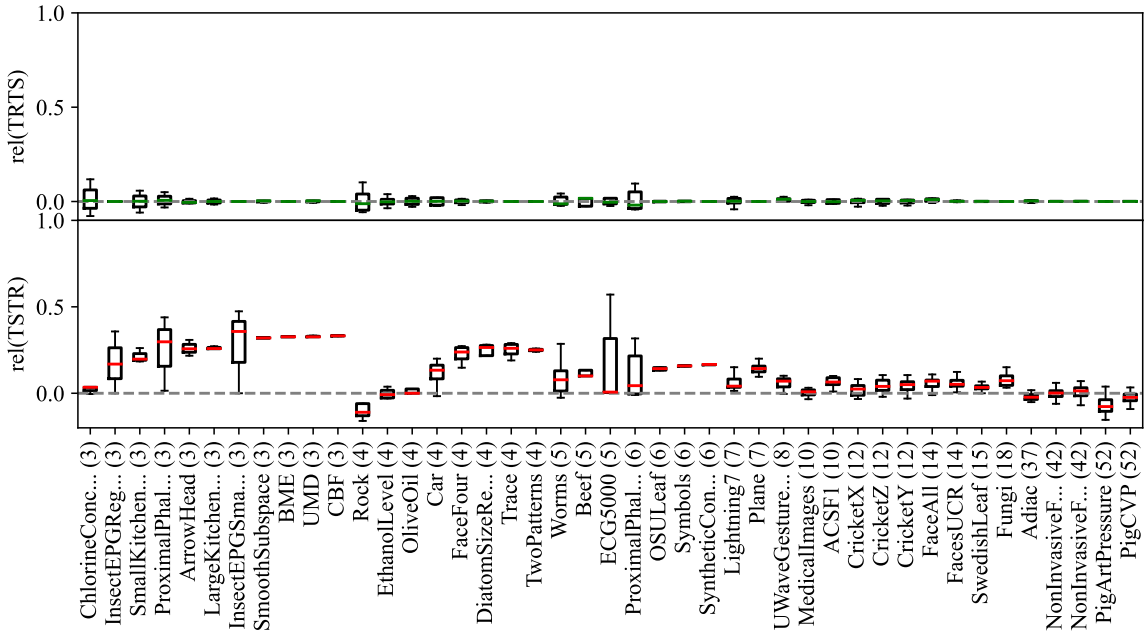


Fig. 8.12 Relative TRTS and TSTR score when one mode is dropped from a dataset.

class would increase $H(P(y))$. That is why we can observe the improvement in ITS after mode drop in some rare cases.

TRTS: With the drop of a class, we have $P_{model} \subset P_{data}$. As we mentioned previously, we expect TSTR to identify this situation, while TRTS is not capable of detecting it. Our results presented in Figure 8.12 are aligned with these metrics' expected behavior. The TRTS did not change on most datasets.

TSTR: The positive $rel(TSTR)$ indicates that TSTR decreases in most datasets. The impact of a single-mode drop is more prominent when the dataset has fewer classes. In a few datasets, the drop of single mode has improved TSTR. The class drop has made the classification task easier for the classifier. Therefore, in a few datasets, although we have an increase in classification error due to the missing class, the classification error of other classes has been improved, which results in a marginal improvement of overall accuracy.

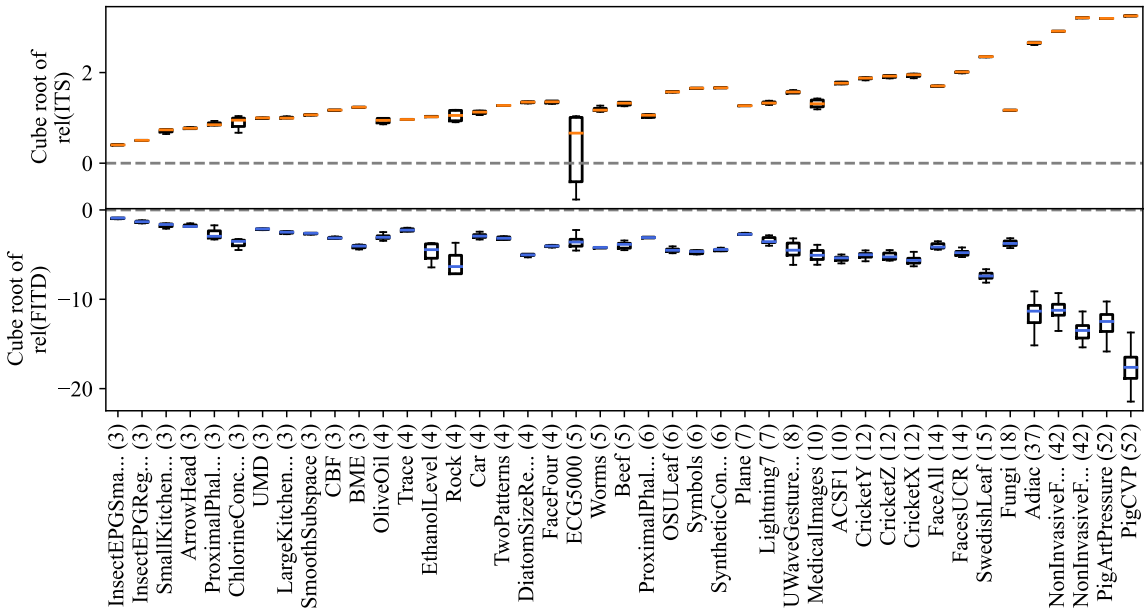


Fig. 8.13 Relative ITS and FITD score for extreme mode drop scenario.

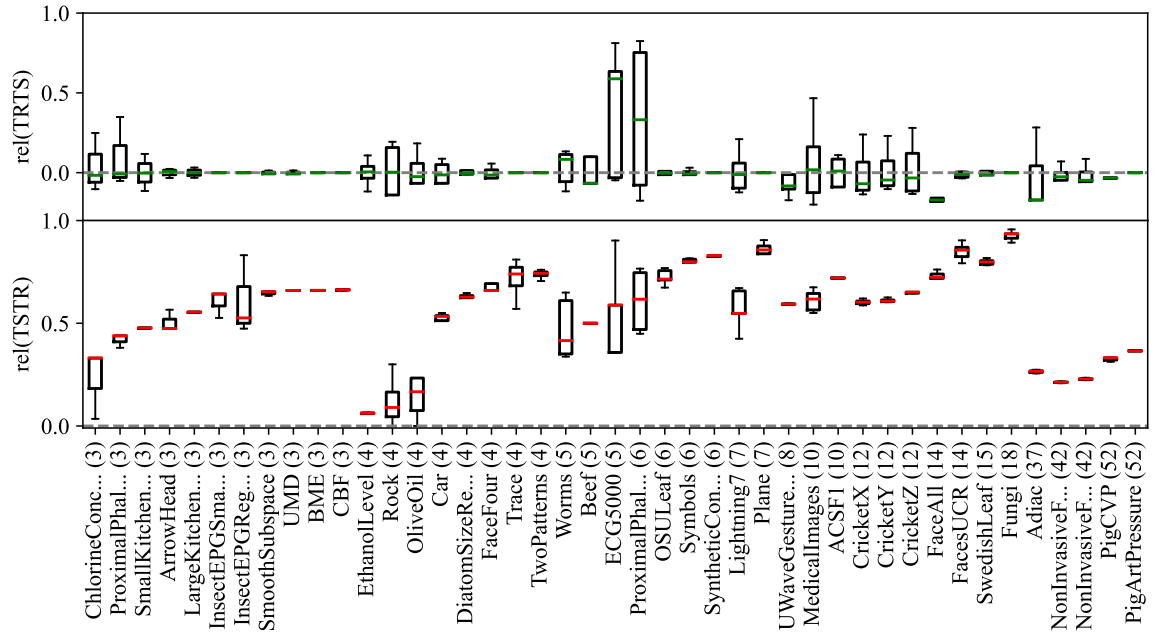


Fig. 8.14 Relative TRTS and TSTR score for extreme mode drop scenario.

Extreme Mode Drop

In the second case, we simulate the extreme case of mode drop, where we keep only one of the classes in the test set. We follow the same approach as the previous experiment but retain only one class. Therefore, for the dataset with N classes, we would have N values for each score. Figures 8.13 and 8.14 portray the results. To make the comparison easier across all datasets, the cube root of $rel(Score)$ for FITD and ITS has been presented. All scores respond to the extreme mode drop scenario correctly except TRTS.

FITD: In the case of FITD, the extreme mode drop drastically changes the properties of the assumed Gaussian in latent space, and we can see this shift in the FITD response. Additionally, this change is more prominent with a large number of classes.

ITS: If we assume error-free classification, with the drop of all modes except one, the $ITS = 1$ since $H(P(y)) = 0$ and $H(P(y \mid x)) = 0$. Hence, $rel(ITS) = ITS_{base} - 1 = N - 1$ where N is the number of classes. In practice and considering classification error, we still observe that the ITS response is close to theoretical expectation.

TRTS: Similar to the previous experiment, TRTS response cannot highlight extreme mode drop since $P_{model} \subset P_{data}$.

TSTR: The TSTR denotes the extreme mode drop in all datasets. Furthermore, with the increase in the number of classes, we have a more significant divergence from $TSTR_{base}$. Please note that we have low accuracy for $TSTR_{base}$ for datasets with $N > 20$ since we trained the InceptionTime classifier for all datasets similarly regardless of the number of classes.

Successive Mode Dropping

In our final experiment, we fill the gap between the first and second experiments, drop the modes one by one, and inspect the response of our assessment metric. Figures 8.15 and 8.16 demonstrate the scores on all datasets. The results are consistent with previous experiments.

FITD: FITD is less sensitive when a few classes are dropped. However, when the number of dropped classes crosses a certain threshold, FITD increases sharply. Seemingly, the properties of the assumed Gaussian distribution are quite robust against removing a few samples from the test set. However, once we remove samples belonging to most classes, the distribution begins to change dramatically with every additional class we drop from the test set.

ITS and TSTR: ITS and TSTR decrease linearly with the number of dropped classes.

TRTS: TRTS does not change with successive drop modes.

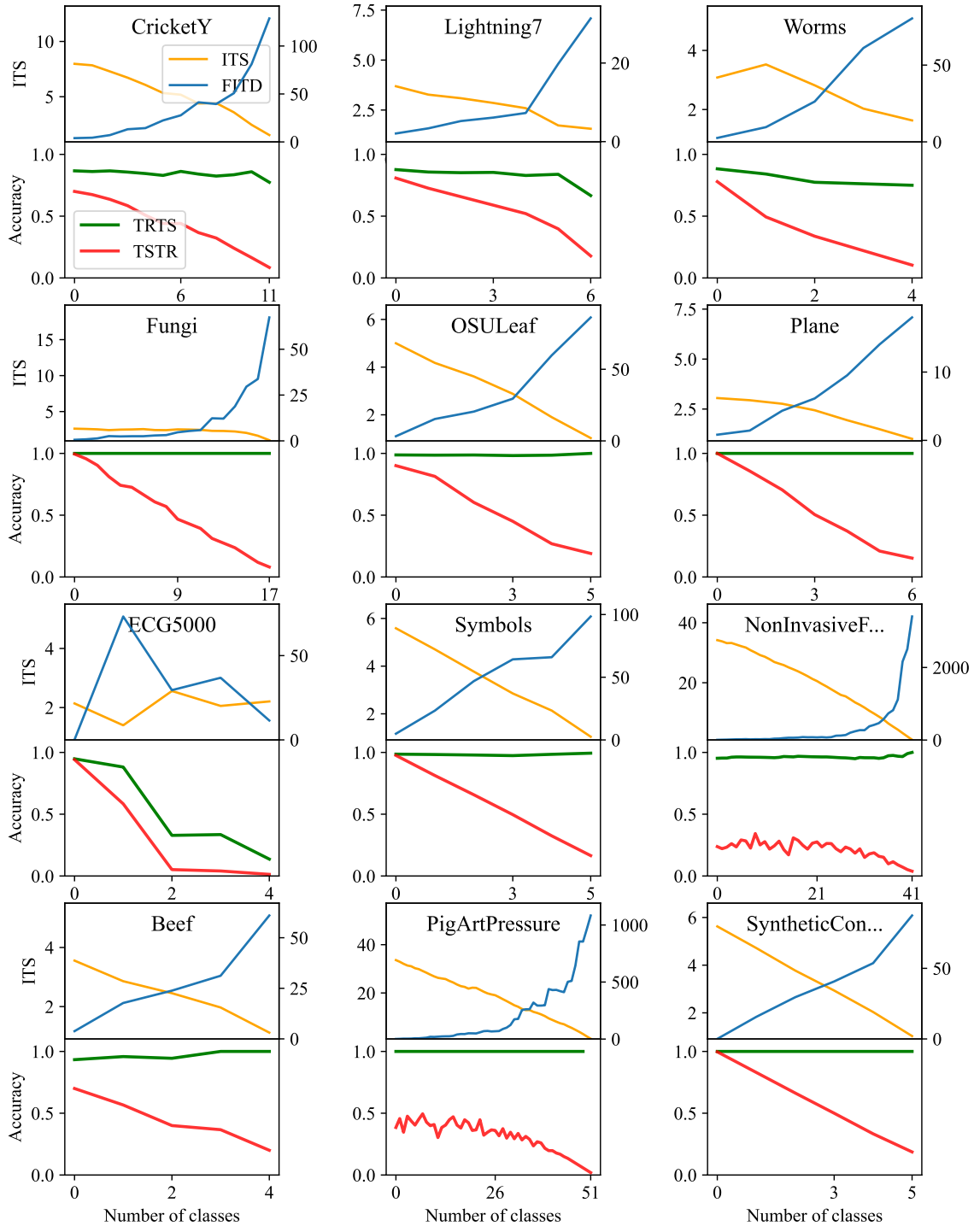


Fig. 8.15 Changes in studied metrics when the modes are removed one by one from a dataset.

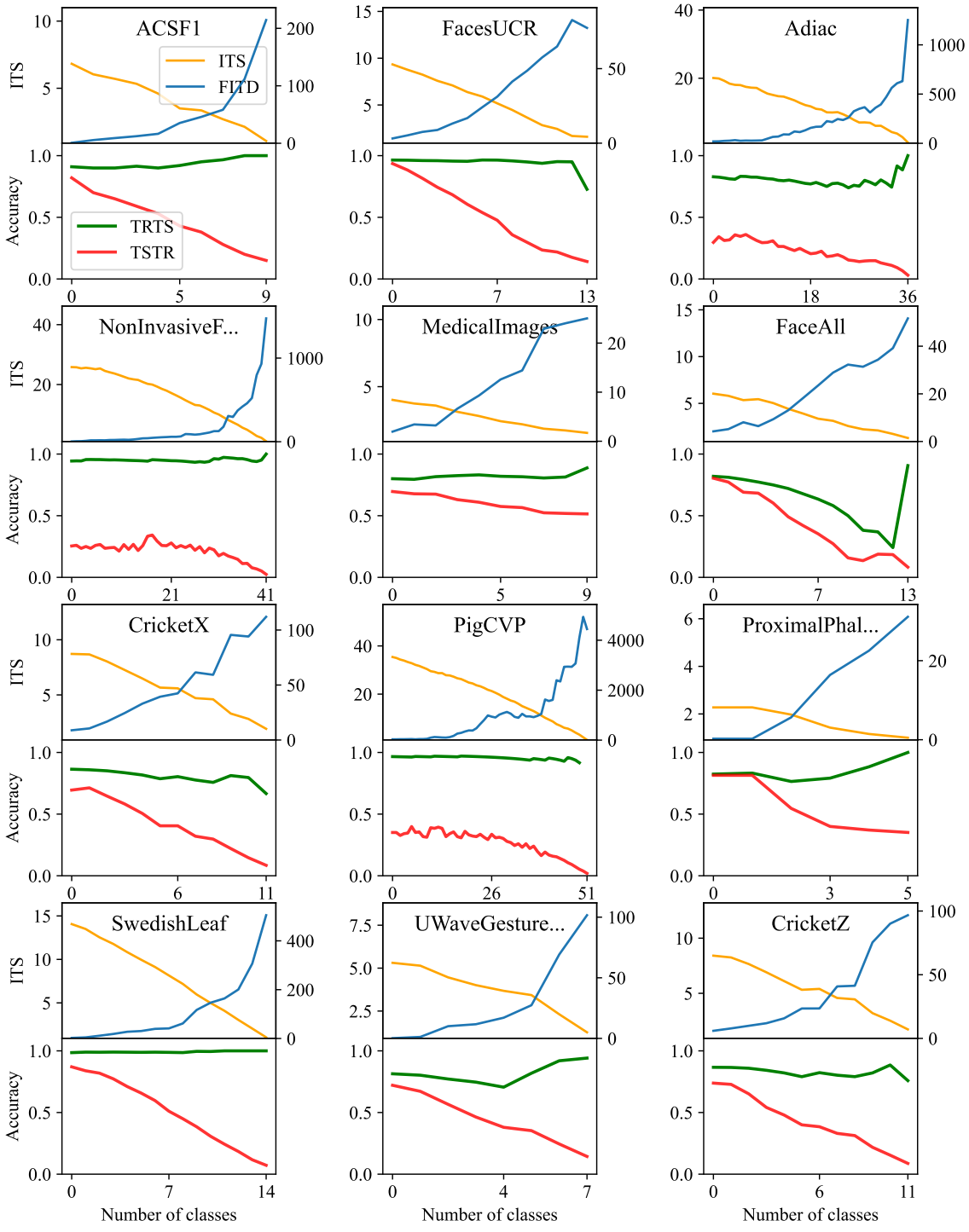


Fig. 8.16 Changes in studied metrics when the modes are removed one by one from a dataset.

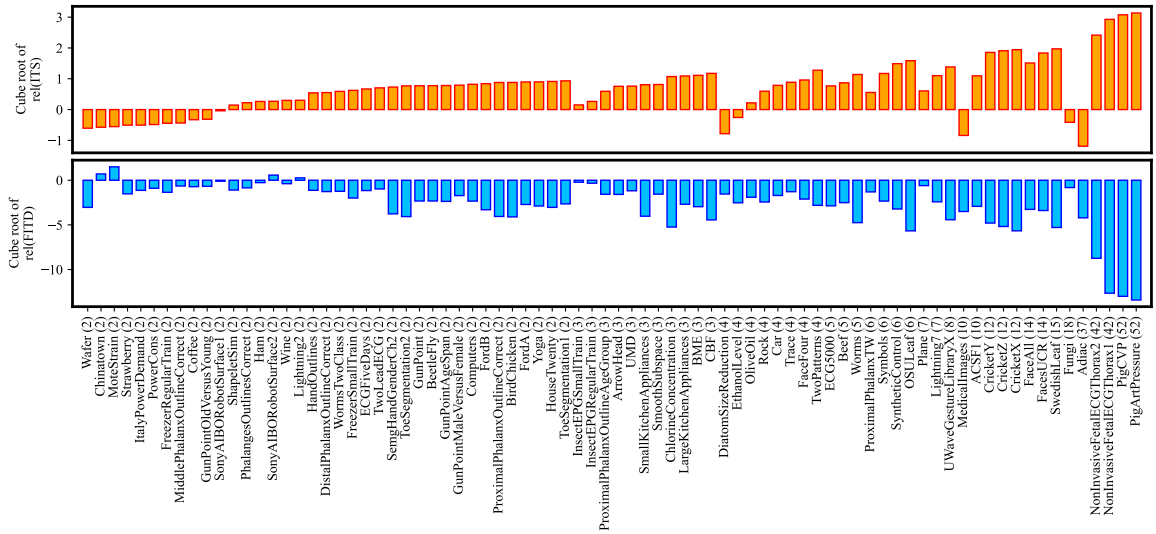


Fig. 8.17 Cube root of relative ITS and FITD score when mode collapse happens in a dataset.

8.4.3 Experiment 3 - Mode Collapse

The mode collapse problem happens when multiple modes of real data are averaged in generated data and presented as a single mode [93]. To simulate mode collapse, we replaced samples of a class with the averaged sample. This was calculated by averaging samples in each time step as follows: Given a set of samples $\{X^0, X^1, ..., X^N\}$ from a class where each sample consists of T time steps ($X^i = \{X_0^i, X_1^i, ...X_T^i\}$), we define the averaged sample $\overline{X}$ at time step $t \in T$ as

$$\overline{X_t} = \frac{1}{N} \sum_{i=0}^{N} X_t^i. \tag{8.4}$$

Figures 8.17 and 8.18 summarize the performance of our scores relative to their base score in detecting this simulated mode collapse.

ITS: In the presence of a perfect classifier, ITS should reach its maximum since then $H(P(y \mid x)) = 0$ in (8.1) , and we have maximum diversity among labels, hence $H(P(y)) = N$, where N indicates the number of classes. However, the average sample might not accurately represent a class's samples. Therefore, there is a high chance of misclassification. Since our generated samples are small and are limited to a single average sample per class, any misclassification would significantly change ITS from its expected value. Therefore, we can observe in Figure 8.17 that the ITS has been improved in some datasets.

FITD: The FITD responds correctly to mode collapse on most datasets, but its responses' strength is inconsistent across datasets. Again, interpreting the FITD response depends on

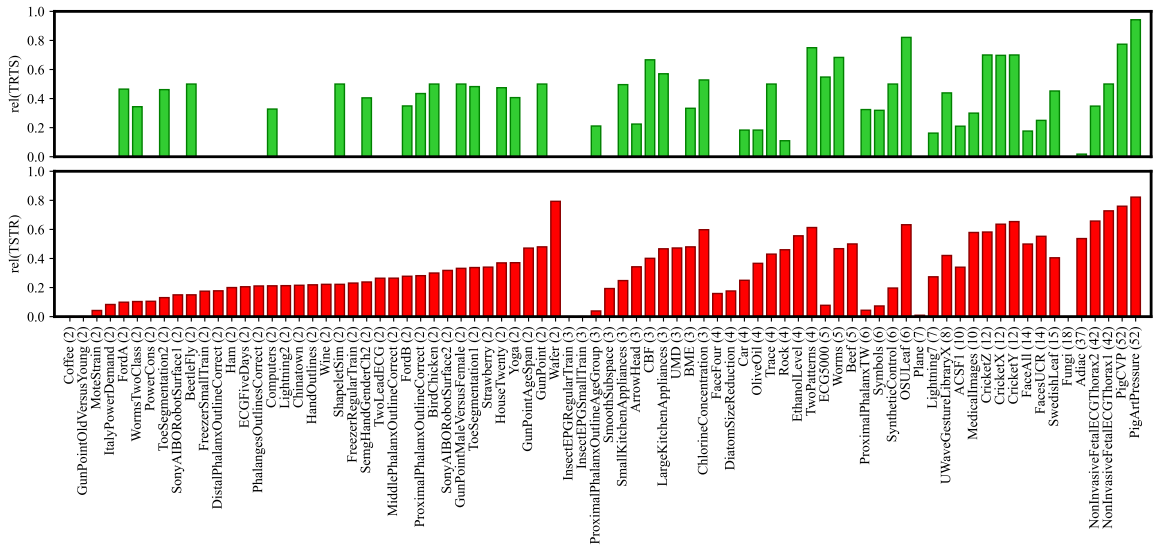


Fig. 8.18 Relative TRTS and TSTR score when mode collapse happens in a dataset.

how samples are mapped in latent space. If the averaged samples can replicate the test set Gaussian distribution properties, we would obtain FITD close to $FITD_{base}$. Otherwise, FITD would diverge from its base score.

TRTS: The TRTS displays a hit-and-miss behavior. If the averaged samples can represent the original samples of the dataset, then they would classify correctly, and TRTS cannot detect mode collapse. Otherwise, the misclassification of averaged samples would reflect the mode collapse problem.

TSTR: The TSTR can detect mode collapse in most datasets. When the mode collapse happens, the diversity of generated samples decreases. Therefore, it is difficult for a classifier to learn the probability distribution of a class accurately, given only samples from the mode of the distribution. Thus, we expect a high classification error once the classifier evaluates the real data due to the limited generalization capacity of the model. The TSTR behavior illustrated in Figure 8.18 is aligned with our expectations.

8.5 Conclusion and Final Remarks

With new advancements in the deep neural network front, generative models are on the rise; however, their application has been hindered in the time series domain due to the lack of a

standard assessment metric. Hand in hand with the prospering variety of new model designs comes the need for reliable assessment metrics to compare them.

8.5.1 Main Contributions

In this chapter, we proposed a framework that converts two popular image domain evaluation metrics, IS and FID, to time series, resulting in two new evaluation metrics, ITS and FITD. To ensure the comparability of the proposed metrics across studies, we utilized the InceptionTime classifier as the framework's backbone. We then conducted experiments on 80 datasets to assess the discriminative abilities of ITS and FITD in detecting common generative model problems. We compared them with two commonly used assessment metrics, TRTS and TSTR. Our findings, presented in Table 8.2, provide quantitative evidence of the metrics' capabilities. Our main contribution is the introduction of two novel assessment metrics and a framework that allows for cross-study comparisons of class conditional generative models in the time series domain. Our main findings on each metric are summarized as follows:

- **ITS** can respond correctly to all the studied problems in most datasets; however, its behavior is most consistent in detecting the Mode Drop problem. Furthermore, $H(P(y))$ seems to be the most defining component of ITS response in detecting the studied problems.
- **FITD** behavior heavily depends on how the samples are mapped into latent space. Since the transformation to latent space is complex and non-linear, interpreting the FITD response is not straightforward. Additionally, since FITD does not have an upper bound, it can quantify the quality of generated samples better than the other metrics.
- **TRTS** performance is disappointing compared to others. In the presence of other metrics, it is unnecessary to compute TRTS for investigating studied problems.
- **TSTR** shines when the generative model has learned a subset of the real distribution. Therefore, it is the most reliable to detect Mode Drop and Mode Collapse compared to others.

This work can be extended by adopting the recent advancement of generative model assessment on image domain [94] to the time series domain. Another potential direction is to extend the list of studied problems or investigate other aspects of evaluation metrics, such as computation time or sample efficiency.

	Decline in Quality	Mode Drop	Mode Collapse
ITS	+	++	+
FITD	++	+	+
TRTS	+	-	-
TSTR	-	++	++

Table 8.2 The summary of the scores' capabilities in detecting common problems of generative models.

Chapter 9

Time series Dataset Augmentation : A Case Study on Sewer Prediction

Generative Adversarial Networks (GANs) have showcased their proficiency in various domains, proving adept at generating synthetic data that mirrors the characteristics of real-world datasets. Motivated by this potential, we put forth a pioneering approach harnessing GANs to generate time series data tailored specifically for urban water management applications. This chapter outlines the complexities of the urban water management domain, illuminates the specific data challenges being addressed, and subsequently unveils our proposed GAN model and the accompanying data preprocessing pipeline. Following the methodological exposition, we analyze the empirical results, discussing their implications for urban water management. Conclusively, we glance forward, pinpointing prospective avenues for further integrating GANs into the vast landscape of water management challenges.

Urban water management is a critical component of sustainable urban development. With the ever-increasing integration of machine learning techniques into this domain, we find ourselves on the precipice of a smarter, more efficient approach to managing our urban water resources [110, 111, 82]. Time series data plays a pivotal role across a spectrum of industries, from healthcare and finance to aerospace and, of course, water resources management [112, 113, 114, 115, 116, 117]. In the face of a rapidly changing environment, harnessing these large datasets becomes even more essential, allowing us to transition from conventional infrastructure development to intelligent, interconnected systems that anticipate the needs of

A.E. Bakhshipour, A. Koochali et al., 'A Bayesian Generative Adversarial Network (GAN) to Generate Synthetic Time Series Data, Application in Combined Sewer Flow Prediction' 2nd International Joint Conference on Water Distribution Systems Analysis and Computing and Control in the Water Industry, Valencia, Spain 2022.

the future [110, 118, 119, 120].

However, harnessing this potential is not without challenges. Data scarcity, whether due to privacy, cost, or the inherent rarity of certain events, stands as a formidable barrier [119]. Additionally, the absence of standardized benchmarks, robust anomaly detection techniques, and probabilistic forecasting methodologies further compounds these challenges, inhibiting the realization of smarter urban water infrastructure.

Against this backdrop, Generative Adversarial Networks (GANs) emerge as a promising solution [5]. Originally developed for domains with intuitively assessable outputs like images, GANs have demonstrated their versatility in generating time series data across sectors such as healthcare [121], finance [122], and engineering [123, 124, 125]. Their prowess extends to tasks such as anomaly detection and probabilistic time series prediction. Yet, their application in Urban Water Management (UWM) remains relatively uncharted territory.

Our research endeavors to bridge this gap. We aim to leverage GANs, specifically for data augmentation, in the context of urban drainage systems. Recognizing the challenges posed by the scarcity of extreme event data, we employ GANs to generate synthetic time series, balancing our dataset and enhancing the accuracy of predictive models for combined sewer flows. By comparing models trained on both real and synthetic datasets, we illuminate the efficacy of our approach using a real-world test case, advocating for more widespread adoption of GANs in UWM.

9.1 Problem Formulation

Combined Sewer Overflows (CSOs) are systems designed to simultaneously collect rainwater runoff, domestic sewage, and industrial wastewater in a single-pipe system. However, during periods of heavy rainfall or snowmelt, the wastewater volume in a CSO can exceed the capacity of the sewer system or treatment plant. As a result, untreated stormwater and wastewater discharge directly into nearby bodies of water, leading to significant environmental and public health challenges [126]. Implementing control strategies, such as releasing retained water into treatment facilities, modulating throttle flow within the sewer system, or channeling water to holding reservoirs, is imperative to mitigate these adverse impacts. To optimize these control measures, accurate and timely predictions of combined sewer flow become paramount.

Recent advancements in deep learning have paved the way for the successful application of deep neural network (DNN) models in predicting water flow dynamics in CSOs [127, 126, 128, 129, 130]. The efficacy of these models, especially in capturing peak flow events, is heavily reliant on the presence of comprehensive, high-resolution datasets encompassing diverse rainfall and extreme events. Regrettably, acquiring such exhaustive datasets often proves challenging, leading to models that may lack robustness and reliability in real-world scenarios.

Recognizing this data deficiency, our research introduces an innovative solution leveraging Generative Adversarial Networks (GANs) to synthesize time series data. By doing so, we aim to enrich datasets, correct imbalances, and ultimately strengthen the precision of DNN-based CSO water depth prediction models. To validate our approach, we examined a real-world case study situated in Germany. In this investigation, we precisely examined the effect of data augmentation using synthetic instances and assessed its implications for the performance of data-driven models.

9.1.1 Case Study: Kaiserslautern's Combined Sewer Catchment

To ascertain the efficacy of our proposed model, we selected a combined sewer catchment located in Kaiserslautern, Germany, as our empirical setting (Figure. 9.1). The selected catchment spans an area of 67.22 ha, with an effective impervious region of 32.79 ha. Notably, the catchment incorporates a storage retention tank (SRT) endowed with a volumetric capacity of 14,000 m^3.

The dataset under consideration encompasses variables such as precipitation, water depth, inflow, flow velocity, and ambient temperature, where precipitation has a positive correlation with other variables of interest. Data collection frequencies varied: 1-minute intervals during meteorologically active periods and less frequent measurements during drier intervals. The data acquisition spanned from August 2020 to December 2021. Figure. 9.2 provides an insightful summary of the gathered data.

Emphasizing the reliability metric, water depth data were observed to exhibit a higher fidelity in our case study. Consequently, our primary objective transitioned to constructing a data-centric model capable of predicting water depths up to a temporal span of one hour in advance. The prediction is facilitated by leveraging previously measured water depths and corresponding rainfall data. Such models, in operational scenarios, could be instrumental in achieving optimal system operations, such as minimizing the volumetric extent and persistence of CSO events through real-time control (RTC) measures.

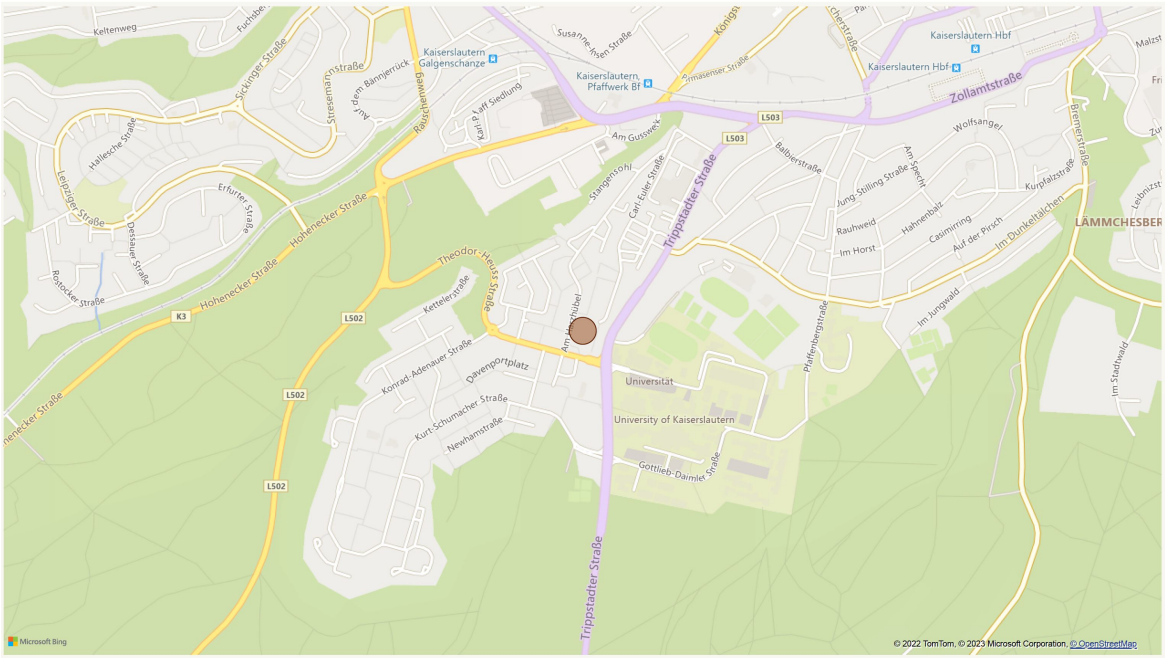


Fig. 9.1 Geographical disposition of the SRT in Kaiserslautern, Germany.

For synthetic data generation and model training purposes, only data recorded during wet weather conditions were considered. Dry weather periods were outlined using two criteria: (1) The cumulative rainfall during the preceding two hours (denoted as the catchment's maximum concentration time) must not exceed 1 millimeter, and (2) peak water depth during such periods should remain below the 0.15 millimeter threshold. A systematic analysis was performed on the rainfall and water depth data to isolate negligible rainfall instances during dry weather that had an inconsequential influence on water depths. Subsequent to this, a rudimentary Artificial Neural Network (ANN) was orchestrated to predict dry weather patterns, premised on temperature readings and the time of day, as visualized in Figure. 9.3. Leveraging the predictive prowess of the dry weather model, we reconstructed the dry weather time series for the entirety of the data recording duration. The ensuing step involved deducing the wet weather water depth time series by subtracting the synthetically constructed dry weather depth from the empirically measured water depth. To enhance granularity, this time series was subsequently re-sampled at 15-minute intervals, serving as the predecessor to our endeavors in synthetic data generation and water depth forecasting, as elucidated in the following segments.

Year	Month	Percipitation [mm]	Average of H(m)	Max of H(m)	Average of Q(l/s)	Max of Q(l/s)	Average of T(c)	Average of V(m/s)	Max of V(m/s)
2020	August	29.43	0.05	0.38	9.54	394.72	18.38	0.54	1.66
2020	September	33.65	0.05	0.34	8.96	337.51	17.84	0.53	1.64
2020	October	70.85	0.06	0.31	11.67	286.82	15.74	0.57	1.57
2020	November	23.32	0.05	0.40	8.49	430.25	13.88	0.52	1.75
2020	December	95.45	0.07	0.32	17.45	306.57	11.31	0.63	1.60
2021	January	62.09	0.07	0.30	15.45	271.79	9.72	0.62	1.55
2021	February	59.21	0.06	0.35	13.78	344.01	9.32	0.58	1.65
2021	March	50.39	0.06	0.20	8.48	118.93	10.18	0.55	1.23
2021	April	20.66	0.14	1.46	9.94	922.20	11.28	0.48	2.00
2021	May	93.03	0.06	0.38	12.88	408.54	12.58	0.57	1.73
2021	June	67.78	0.06	0.40	10.89	438.18	14.77	0.55	1.76
2021	July	97.72	0.06	0.48	15.01	661.89	16.65	0.57	2.17
2021	August	55.56	0.05	0.40	9.61	438.68	17.16	0.53	1.76
2021	September	28.34	0.05	0.37	8.64	382.47	16.88	0.52	1.70
2021	October	54.87	0.05	0.43	10.10	484.99	15.21	0.53	1.75
2021	November	46.00	0.06	0.27	10.92	214.56	12.84	0.56	1.45
2021	December	65.72	0.06	0.33	12.77	309.72	10.73	0.58	1.61
Total		**954.07**	**0.06**	**1.46**	**11.49**	**922.20**	**13.72**	**0.56**	**2.17**

Fig. 9.2 Consolidated visualization of the procured data metrics.

9.2 Data Augmentation with GAN

In this research, our goal is to map from prior distribution to data distribution and generate artificial time series $\hat{X}$ with size T i.e., $\hat{X} = \{\hat{x}_1, ..., \hat{x}_T\}$. To generate an artificial sample at the time-step t, the generator $\mathscr{G}$ parameterized by Θ aims to model the following distribution:

$$\mathscr{G}_\Theta(\hat{x}_t|\hat{x}_{1:t-1}) = \mathscr{G}_\Theta(\hat{x}_t|h_t), \tag{9.1}$$

where h_t is the output of the auto-regressive neural network:

$$h_t = H(h_{t-1}, x_{t-1}, \Theta). \tag{9.2}$$

To implement H, we employ a multi-layered Gated Recurrent Unit (GRU) and a fully connected layer to map h_t to $\hat{x}_t$. To initiate time series generation, we used $h_0 = z$ and $x_0 = \mathscr{S}$ where z is sampled from the prior distribution and $\mathscr{S}$ is the start token which is set as the mean value of given time series dataset $\mathscr{D}$. Furthermore, we employed a standard Gaussian distribution as the prior distribution. Figure 9.4 illustrates the generator's structure $\mathscr{G}$.

The aim of the Critic $\mathscr{C}$ is to approximate a function that maps a given time series sample X to a score that reflects the authenticity of its input. In more concrete terms, we try to find a parameter set Φ which:

$$\text{Score} = \mathscr{C}_\Phi(X_{1:T}) = \mathscr{C}_\Phi(k_T), \tag{9.3}$$

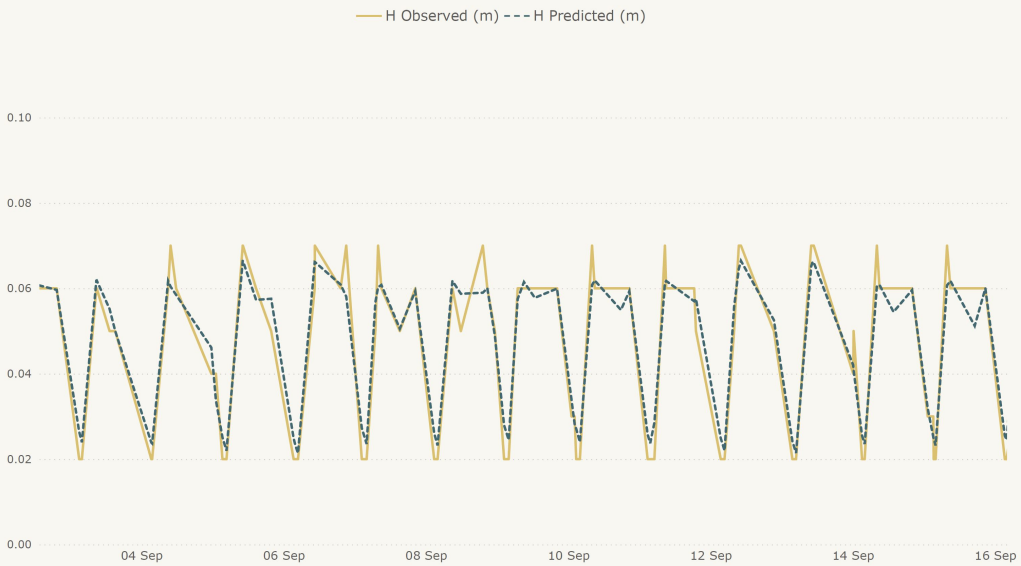


Fig. 9.3 Predictive outcomes for dry weather conditions, facilitated by a foundational ANN model.

where k_t is the last output of a Recurrent Neural Network:

$$k_T = K(X_{1:T}, \Phi). \tag{9.4}$$

Figure 9.5 present the Critic $\mathscr{C}$ structure. A GRU implements the K, and the k_T is mapped to the score with a fully connected layer.

We employed Wasserstein GAN with gradient penalty (WGANGP) to train our model. WGAN-GP employs Earth-Mover (also called Wasserstein-1) to approximate divergence between p_{data} and p_{model} i.e. $W(p_{data}, p_{model})$. Wasserstein distance $W(q,p)$ is informally defined as the minimum cost of transporting mass to transform the distribution q into the distribution p (where the cost is mass times transport distance). Under mild assumptions, it is continuous everywhere and differentiable almost everywhere. Using Kantorovich-Rubinstein duality[7], the value function of WGAN-GP is defined as:

$$\min_{\mathscr{G}} \max_{\mathscr{C}} \mathop{\mathbb{E}}_{x \sim p_{data}} [\mathscr{C}(x)] - \mathop{\mathbb{E}}_{\hat{x} \sim p_{model}} [\mathscr{C}(\hat{x})], \tag{9.5}$$

where Critic $\mathscr{C}$ approximates a 1-Lipschitz function. To enforce the Lipschitz constraint on the Critic $\mathscr{C}$, the WGAN-GP imposes a penalty on the gradient norm of the Critic. Thus, the final objective function for WGAN-GP is:

$$L = \mathop{\mathbb{E}}_{x \sim p_{data}} [\mathscr{C}(x)] - \mathop{\mathbb{E}}_{\hat{x} \sim p_{model}} [\mathscr{C}(\hat{x})] + \lambda \mathop{\mathbb{E}}_{\tilde{x} \sim p_{\tilde{x}}} [(\|\nabla_{\tilde{x}} \mathscr{C}(\tilde{x})\|_2 - 1)^2], \tag{9.6}$$

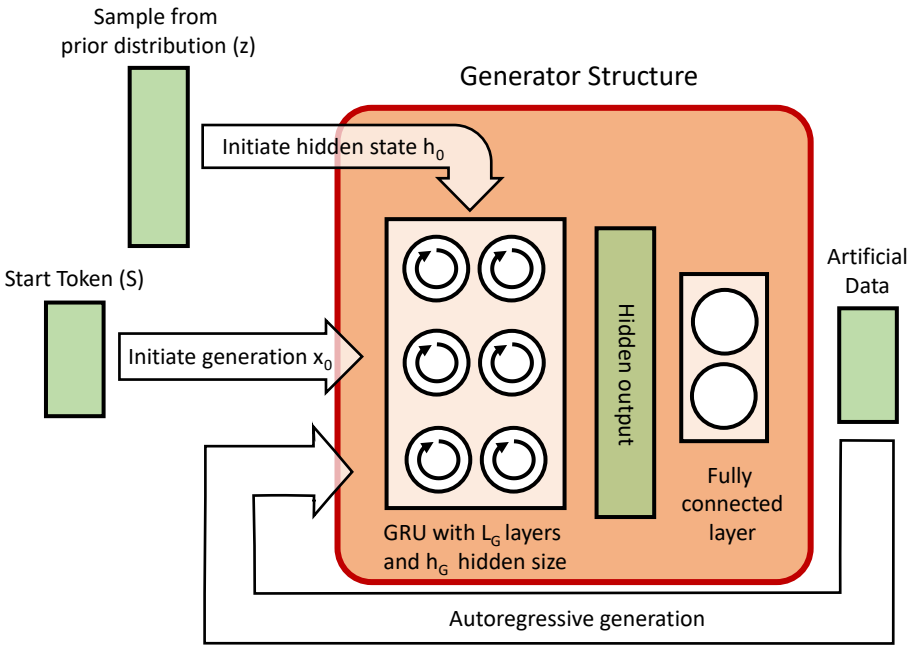


Fig. 9.4 The architecture of proposed WGAN-GP Generator

where λ is the penalty coefficient, and $p_{\tilde{x}}$ is implicitly defined by sampling uniformly on lines between pairs of points sampled from p_{data} and p_{model}.

9.2.1 Tail-Oriented Generative Learning

Data augmentation via artificial generation is primarily executed with the intent of enriching the dataset, thereby fostering enhanced performance in subsequent tasks. Critical to the success of this strategy is the fidelity of the generated data in mirroring the nuances of the original dataset, particularly in scenarios involving underrepresented patterns. A threat inherent to generative models lies in their tendency to predominantly model and replicate prominent data patterns, potentially neglecting less frequent patterns. Such an oversight could inadvertently intensify data pattern imbalances, thereby undermining the efficacy of tasks that leverage the augmented dataset.

As delineated in Figure. 9.6, our dataset embodies peaks that manifest sporadically and with variances in regularity. These episodic peaks can be discerned as outliers within the long-tail distribution of data points, as depicted in Figure. 9.7. An efficacious generative model, to produce credible artificially-induced peaks, must render a perfect approximation of this distribution's tail. However, given the scarcity of data at the extremities of distributions, the generative model often confronts challenges in capturing these nuances, either overesti-

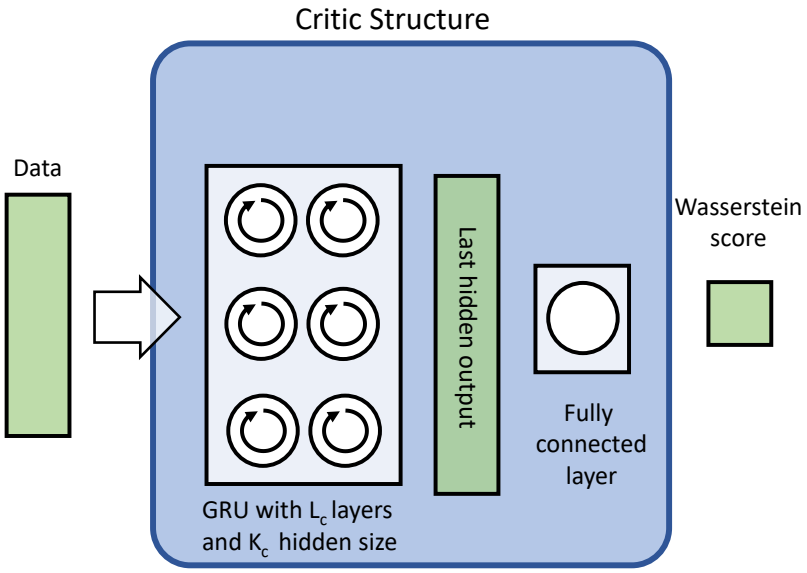


Fig. 9.5 The architecture of proposed WGAN-GP Critic

mating or underestimating the distribution's tail. Such imprecisions cause unauthentic peaks within the generated data, which, when used to train a forecasting model, can compromise its adeptness at peak predictions.

To counteract this predicament, it becomes imperative to penalize the generative model for its deviations in modeling the distribution tail. Our proposed strategy involves the incorporation of a quantile accuracy-focused penalization mechanism within the loss function. The quantile loss function, which gauges differences between generated and real data at a specified quantile, is formulated as:

$$\text{QLoss}(q,\hat{x}_q,x_q) = \max\left[q(\hat{x}_q - x_q),(q-1)(\hat{x}_q - x_q)\right], \tag{9.7}$$

where q represents the designated quantile, while x_q and $\hat{x}_q$ denote the actual and generated data values at this quantile, respectively. Integrating the quantile loss function within the WGAN-GP's objective function facilitates its optimization during the generator's training phase. Thus, the modified GAN objective function is articulated as:

$$L = \text{WGAN-GP Loss} + \gamma \times \text{QLoss}, \tag{9.8}$$

with γ designating the penalty coefficient assigned to the quantile loss. Empirical investigations have underscored that calibrating the quantile loss for $q = 0.99$ in conjunction

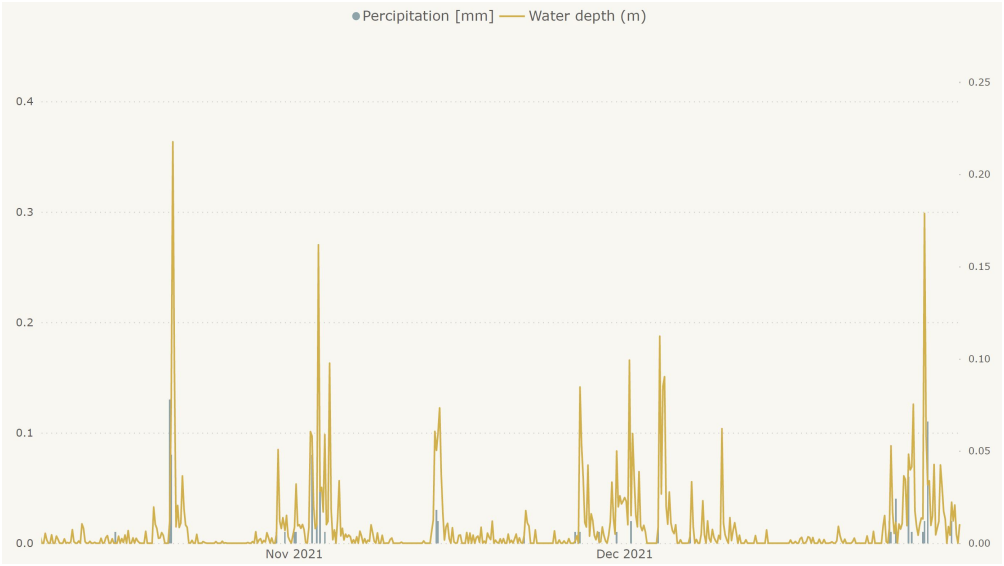


Fig. 9.6 Exemplary illustration from the dataset.

with setting $\gamma = 1$ offers the most sensible trade-off, leading to an accurate modeling of the distribution's tail.

9.3 Experiment set-up

9.3.1 Optimization of Hyperparameters

To optimize the hyperparameters intrinsic to our proposed architecture, we leveraged the Bayesian optimization in conjunction with Hyperband, termed as BOHB [131], facilitated by the Ray Tune library [132]. Central to the BOHB algorithm is its adeptness at combining the strengths of Bayesian optimization for hyperparameter sampling with the resource allocation efficiency furnished by the Hyperband strategy [133].

At its inception, a subset of hyperparameters is randomly sampled, and the resultant model's performance score is noted. The Bayesian optimizer subsequently fits a probabilistic model to these outcomes. Harnessing this model, it proposes a new set of hyperparameters that are probabilistically fit towards achieving superior performance.

Hyperband, grounded in multi-armed bandit theory, efficiently allocates resources to varying configurations. Implementing the principle of successive halving [134], configurations demonstrating suboptimal performance are progressively pruned. Within this framework, 'resources' are conceptualized as the number of training iterations. Therefore, top-performing

models are allocated a higher number of iterations, while sub-par models are terminated prematurely.

However, a challenge emerges in the context of generative models. Traditional value functions for Generative Adversarial Networks (GAN) are not conducive to gauging in-training performance. Moreover, authenticating the veracity of generated data poses an additional layer of complexity. In light of these challenges and the necessity for a computationally light metric (owing to its frequent computation), we elected to deploy the Jensen-Shannon Distance (JSD) to discern the divergence between the data and model distributions. The mathematical formulation of JSD for two discrete probability distributions, P and Q, is:

$$\mathrm{JSD}(P\|Q) = \frac{1}{2}D(P\|M) + \frac{1}{2}D(Q\|M), \tag{9.9}$$

with M being the averaged distribution, $M = \frac{1}{2}(P+Q)$. The divergence metric, D, represents the Kullback-Leibler Divergence (KLD):

$$D_{\mathrm{KL}}(P\|Q) = \sum_{x\in\mathscr{X}} P(x)\log\left(\frac{P(x)}{Q(x)}\right). \tag{9.10}$$

A key assumption posits that each time step's values are statistically independent. By constructing histograms (with a granularity of 100 bins) from both the dataset and generated samples, we obtain discrete probability distributions. Admittedly, this approach results in a somewhat coarse representation of the true distributions, given that the temporal dependency between successive data points is disregarded. Nonetheless, a significant distinction between these approximate distributions can be indicative of suboptimal model performance. Given BOHB's reliance on performance scores for pruning during the hyperparameter optimization phase, the JSD metric, despite its approximate nature, suffices for our needs.

Table 9.1 enumerates the hyperparameters under consideration, outlining both the explored domain and the optimal value as identified by BOHB.

9.3.2 Data Preprocessing

Effective training of Generative Adversarial Networks (GANs) requires that the input data be suitably processed to mirror the desired distribution characteristics as closely as possible. We initiated our preprocessing with a detailed analysis of the data distribution, identifying and addressing any prominent issues.

Hyperparameter	Search Space	Optimal Value
Prior sample size (z)	[10 - 100]	30
Generator GRU layers (L_G)	[2 - 5]	3
Generator GRU hidden size (h_G)	[16 - 256]	160
Critic GRU layers (L_C)	[2 - 7]	2
Critic GRU hidden size (k_C)	[16 - 256]	23
Quantile loss coefficient (γ)	[0.5 - 2]	1
WGAN-GP coefficient (λ)	[0.1 - 10]	5

Table 9.1 Hyperparameters assessed via BOHB, including the search range and the resultant optimal value.

Figure 9.7 presents the initial distribution of our variables of interest, namely precipitation and sewage water flow. An examination reveals that the sewage flow data exhibits a discrete distribution, beginning at zero and incrementing in steps of 0.1. Given that GANs necessitate a differentiable generator function incapable of producing discrete outcomes directly, we applied a smoothing transformation to shift from a discrete to a continuous distribution, thus enabling the later recovery of the original data characteristics. This transformation is represented as follows:

$$X_{continuous} = X_{discrete} + u, \quad \text{where } u \sim \mathscr{U}_{[0,0.1]}, \tag{9.11}$$

where $\mathscr{U}_{[0,0.1]}$ is a uniform distribution with minimum value 0 and maximum value 0.1. The inverse operation for recovering discrete values post-generation is formulated as follows:

$$\hat{X} = \hat{X} - \text{mod}(\hat{X}, 0.1), \tag{9.12}$$

where mod is the modulo operation, effectively quantizing the continuous value back into the appropriate discrete interval.

Another challenge presented by the data is its pronounced skewness toward zero, coupled with a long-tailed distribution of positive values. Such distributions can complicate the learning process for generative models, as they might converge rapidly by capturing the dense regions of the distribution, neglecting the tail which, although sparse, contains critical information—in this instance, representing the instances of heavy rainfall. To mitigate the risks of mode collapse—a condition where multiple data modes are collapsed into a singular mode by the generative model—we applied a logarithmic transformation to non-zero values and mapped zero values to a negative domain via a beta distribution, as depicted by the

transformation function $F(X)$ below:

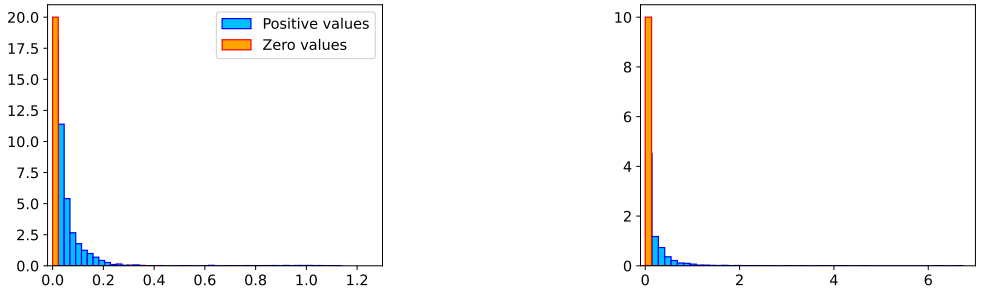
$$F(X) = \begin{cases} \log(X) & \text{if } X > 0, \\ -\mathscr{B}(\alpha, \beta) & \text{if } X = 0, \end{cases} \tag{9.13}$$

where $\mathscr{B}(\alpha, \beta)$ denotes a sample drawn from the beta distribution parameterized by shape parameters α and β.

The inverse transformation, applied to the generated samples to recover their original scale, is given by:

$$F^{-1}(\hat{X}) = \begin{cases} \exp(\hat{X}) & \text{if } \hat{X} > 0, \\ 0 & \text{if } \hat{X} \leq 0. \end{cases} \tag{9.14}$$

Figure 9.8 graphically details the complete preprocessing and postprocessing pipeline. Subsequent to the preprocessing steps, Figure 9.9 illustrates the transformed distribution of the dataset.



(a) The distribution of water depth values.

(b) The distribution of precipitation values.

Fig. 9.7 The original distribution of the data points in the studied dataset.

9.4 Results and Discussion

Quantitative assessment of implicit generative models such as GANs is a challenging task [93]. The loss function of the generator does not indicate the quality of generated samples, and the lack of a consensual and reliable standard for evaluating generative models in the time series domain poses a significant obstacle [135]. To overcome this challenge, we propose a two-fold approach. First, we conduct statistical analysis of the generated samples and the real data to gauge the accuracy of the generative model's estimation of the underlying

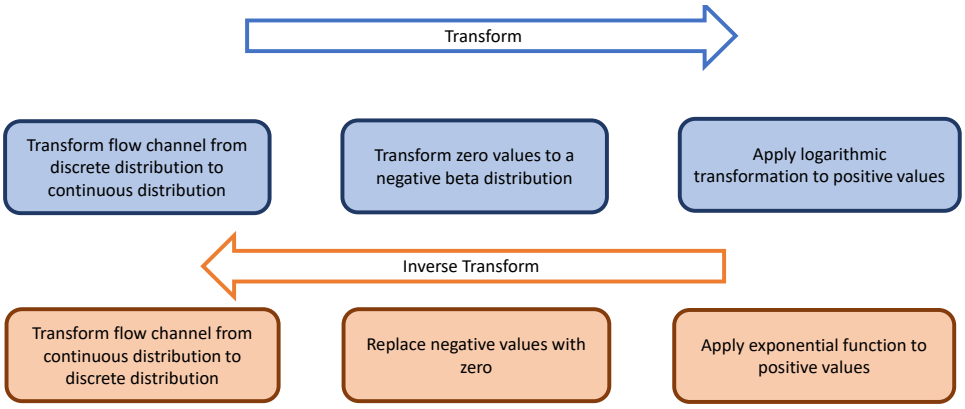
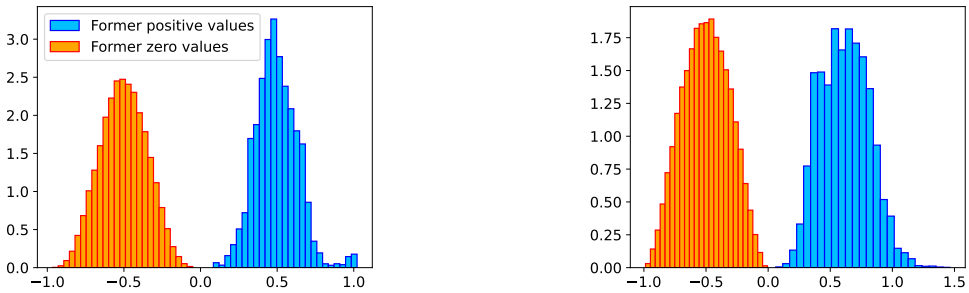


Fig. 9.8 Illustration of the preprocessing and postprocessing pipeline, showing the sequence of transformations and their inverses.



(a) The distribution of transformed water depth values.

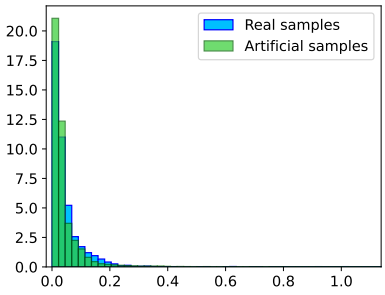
(b) The distribution of transformed precipitation values.

Fig. 9.9 The distribution of data points following preprocessing in the studied dataset.

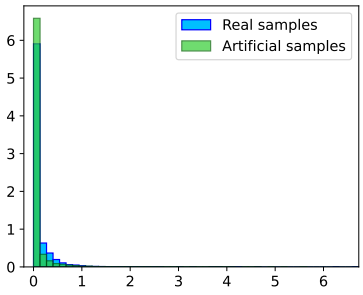
generative process. Second, we augment our dataset with samples generated by the model and investigate the impact on the performance of a downstream forecasting task. Through this, we aim to infer the generated samples' quality indirectly.

9.4.1 Statistical Result Analysis

The ultimate goal of a generative model is to approximate the underlying generative process accurately. This is reflected in the alignment of the distribution of artificial data with real data. Figure 9.10 visualizes the distribution of artificial data compared to the real data distribution, demonstrating that the proposed generative model has effectively learned the distribution of the real data. Additionally, Table 9.2 provides a quantitative analysis of the first four moments of both the real and artificial distributions. The close match between the moments further supports the accuracy of our generative model.



(a) The distribution of water depth values



(b) The distribution of precipitation values

Fig. 9.10 The original distribution of the data points alongside the distribution of artificial data.

Distribution moments	Real water depth	Artificial water depth	Real precipitation	Artificial precipitation
Mean	0.051	0.041	0.102	0.070
Standard Deviation	0.093	0.102	0.253	0.239
Skewness	7.059	8.427	6.669	8.691
Kurtosis	63.589	104.753	81.617	151.838

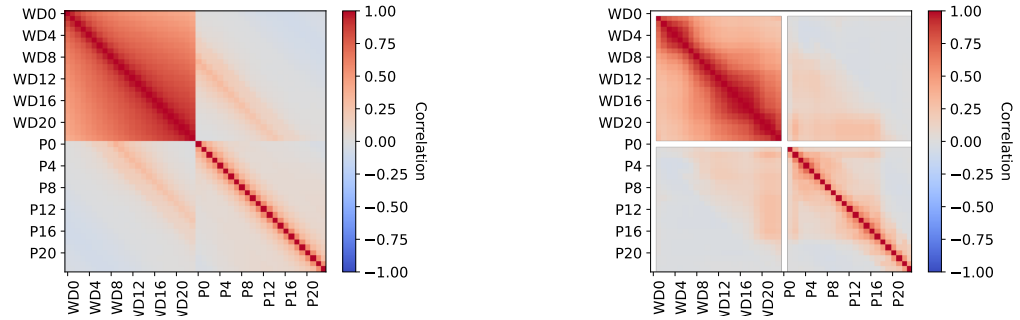
Table 9.2 The list of hyperparameters tuned by BOHB alongside the hyperparameters space that is searched and final selected values by BOHB

Furthermore, we aim to evaluate the ability of our generative model to capture the dependency structure of data over time and between channels. To accomplish this, we analyze the correlation matrix of values in a 6-hour time window (i.e., 24 time-steps) for both real and generated samples, as illustrated in Figure 9.11. The high similarity between the real and generated samples correlation matrix, as shown in Figure 9.11, indicates that our generative model has effectively estimated the dependency structure of real data.

To conclude, the statistical analysis demonstrates the effectiveness of our proposed GAN-based model in approximating the underlying generative process. Our model exhibits the capability of accurately modeling the unconditional distribution of real data points and capturing the real data dependency structure.

9.4.2 Downstream Task Analysis

In this study, we aim to improve the performance of a forecasting model by utilizing artificial data. The presence of a downstream task for the generative model enables us to assess the



(a) The correlation matrix of real data

(b) The correlation matrix of artificial data

Fig. 9.11 The correlation matrix of values in a 6-hour window (.i.e., 24 time-steps). WD stands for water depth, and P stands for precipitation. The numbers that follow the abbreviations express the time step.

quality of the generated sample indirectly through the inspection of artificial data impact on the performance forecast model.

The downstream task is the multi-step-ahead forecast of water depth values. Our forecaster model receives a 150-minute window of past data (10 time-steps) and estimates the water depth values for the next 60 minutes (4 time-steps):

$$[x^{wd}_{t+4}, x^{wd}_{t+3}, x^{wd}_{t+2}, x^{wd}_{t+1}] = f(X_{t-9}, ..., X_{t-1}, X_t), \tag{9.15}$$

where x^{wd}_i is the value of water depth at i-th time-step and X_i is the data-point at i-th time-step. We implemented function f with a feed-forward neural network with 3 layers and 256 neurons at each layer (Figure 9.12) and trained it with the mean squared error (MSE) loss function. We used 50% of our data for training, 20% for validation, and 30% for testing. Furthermore, we generated a set of artificial data from our GAN model with the same size as the real training set for augmentation. We trained our forecast model in three scenarios, namely, training only with real data (real scenario), training only with artificial data (artificial scenario), and training with real and artificial data (augmentation scenario). The performance of forecasting models is reported using a similar test set from real data in all scenarios. Table 9.3 summarizes the result of our forecasting experiments. The performance of the forecasting model in the artificial scenario is very close to the real scenario, which signifies that the generated samples look realistic and authentic. Furthermore, the accuracy of the forecasting model has been increased in the augmentation scenario in comparison to the real scenario. The improvement shows that doubling the size of the training set results in a better estimation of future values.

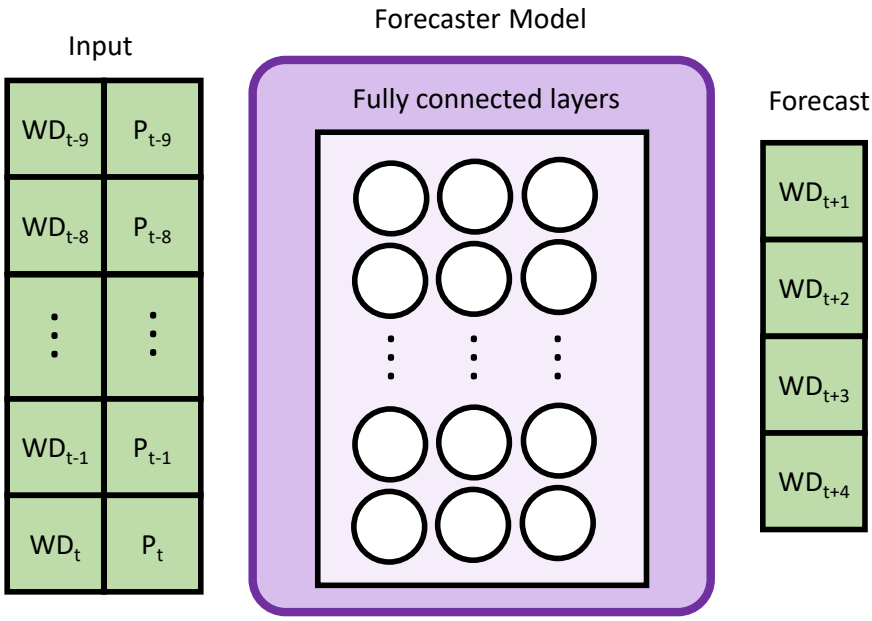


Fig. 9.12 The architecture of forecaster model

	MSE
Real Scenario	$1.671 \times 10^{-3} \pm 0.126 \times 10^{-3}$
Artificial Scenario	$1.679 \times 10^{-3} \pm 0.231 \times 10^{-3}$
Augmentation Scenario	$1.527 \times 10^{-3} \pm 0.149 \times 10^{-3}$

Table 9.3 The performance of forecasting model in different scenarios reported in mean squared error (MSE)

Figure 9.13 provides detailed information on the performance of the forecasting model on each time step of the future horizon for different scenarios. We can observe that the model trained on artificial data only shows superior performance in comparison to the real scenario at the beginning of the forecast window; however, the model accuracy deteriorates quickly once we forecast further in the future. In the augmentation scenario, the model exhibits the strengths of both real and artificial scenarios. Similar to the artificial scenario, the model forecast accurately in the first step forecasting horizon, while its performance remain relatively constant throughout the forecasting horizon.

In addition to quantitative results, Figure 9.14 portrays the future estimation by the forecasting model in different scenarios and provides qualitative insight into the model performance. This figure displays multiple rain events that are combined by removing the dry-weather intervals in between. As a result, the time on the x-axis is not continuous. In the artificial

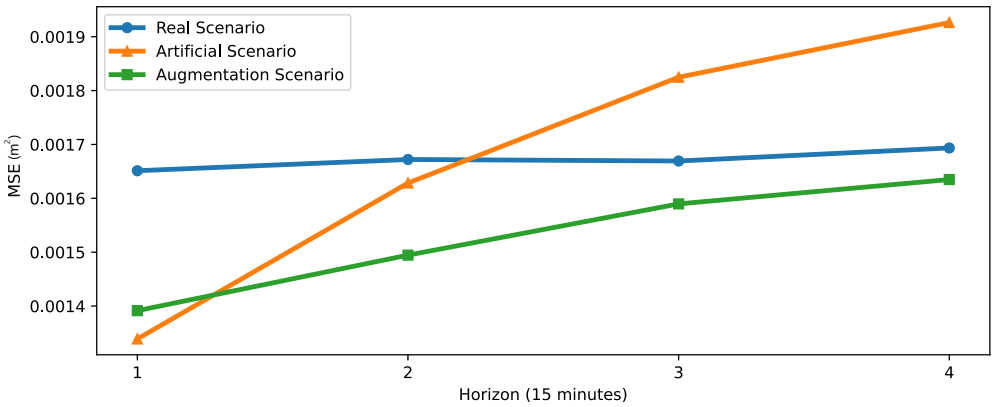


Fig. 9.13 The forecast error over 1-hour horizon

scenario, the model tends to overestimate the pick, while the artificial data improves the model in an augmented scenario.

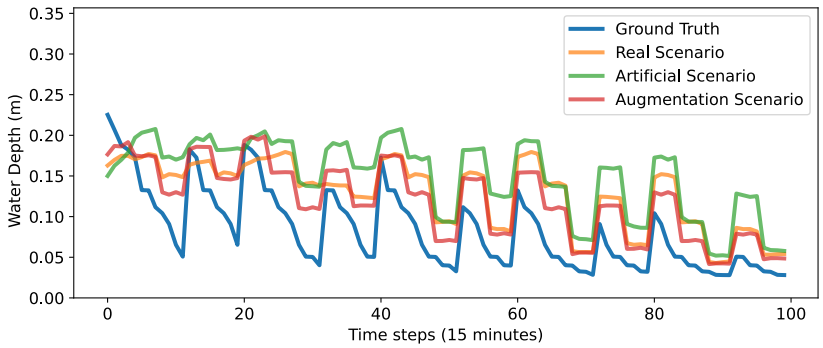


Fig. 9.14 Sample of forecast from our model in different scenarios

9.5 Conclusion and Potential Directions

In this study, we investigated the potential applications of Generative Adversarial Networks (GANs) in different fields of urban water management. We divided these potentials into three main categories and provided some recommendations for further studies as follows:

1. **Synthetic data generation:** Regarding urban drainage systems, GANs can assist by synthesizing rainfall data where actual data is insufficient. Additionally, GANs can generate rare events for robust optimization of the planning of these systems. As for urban flood modeling, GANs can use high-resolution urban catchment data from

low-resolution satellite data. Considering water distribution networks, using GANs to generate water consumption data from real data can address data privacy issues. GANs can generate a significant amount of synthetic data, leveraging both collected and generated data from hydraulic models. This can help address the issue of overfitting in deep learning models for leak detection by providing data with unknown sources of uncertainty.

2. **Anomaly detection:** GANs can learn the overall distribution of a time series dataset and analytically generate the missing values or detect anomalies. They can help by providing an alternative to autoencoders for identifying normal and abnormal events for tasks such as leak and contamination detection in water distribution networks.

3. **Probabilistic prediction:** GANs can transform deterministic models into probabilistic ones with improved performance for optimal and robust decision-making under uncertainty. Some examples will be probabilistic rainfall runoff, water demand forecasting, and pluvial flood prediction models.

In Summary, using synthetic data generated by GAN models can accelerate the development of deep learning models in the water sector, increasing data sets and reducing privacy risks. GANs can also produce similar structured and unstructured data, making them an important research area in data-limited situations [111]. It is important to note that the potential applications of GANs in urban water management discussed in this paper can also be applied to other problems and challenges involving time series data, data privacy, anomaly detection, and probabilistic prediction.

In this study, we evaluated the primary application of GAN, data augmentation, by using it to generate synthetic time series for balancing our dataset. The results showed that the proposed method slightly improved the accuracy of combined sewer water depth predictions. Furthermore, a model trained solely with synthetic data was found to be comparable to one trained with real data. It remains to be demonstrated if this approach is more effective than deterministic model predictive control in forecasting flow and water level. Due to the highly stochastic nature of pollutant transport in urban drainage systems, deterministic models fail to predict runoff quality with reasonable accuracy. In this sense, we believe GAN can be a valuable tool for improving quality-based runoff prediction models. Our future studies will explore this potential.

Chapter 10

Beyond Unconditional Synthesis: Structured Noise Space GANs for Class-Specific Data Generation

Class-conditional generative models represent significant progress in machine learning, empowering tasks that necessitate controlled data synthesis. Contrary to their unconditional counterparts, these models synthesize samples conditioned on specific class labels, facilitating a more focused generative process. This conditional generation enables the production of class-representative data, which has a multitude of practical applications. For example, such models can augment the diversity of a class in a training dataset, thereby enhancing the robustness of classifiers in subsequent tasks.

Moreover, Generative Adversarial Networks (GANs) stand to gain from integrating class labels during the training phase. The inclusion of such labels steers the generation process towards enhanced quality and diversity. These labels enrich the model's understanding of the data distribution, allowing the generator to create samples that are not merely plausible but also class-specific. This specificity potentially leads to outputs of higher fidelity and greater variety. Concurrently, the discriminator benefits from class labels by providing a more nuanced task of discriminating between genuine and generated data, evaluating not only the authenticity but also the class alignment, thereby furnishing a more robust learning signal for the generator.

Nevertheless, incorporating class labels as a condition into the GAN architecture is crucial for the effective functioning of Conditional GANs (CGANs). The field lacks a unified approach for the integration of conditional information into GAN architectures, leaving

researchers to adopt a trial-and-error methodology to identify optimal strategies for condition integration. Scientific efforts have been bifurcated into two primary streams based on discriminator conditioning: classifier-based GANs [136, 137, 138, 139, 140, 141] and projection-based GANs [142, 143, 144, 145]. These categorizations reveal the scarcity of options for conditioning generators. The original CGAN proposal [9] suggests concatenating one-hot encoded class labels with the noise vector. However, this method may lead to the generator disregarding the condition vector when the number of classes is small or negatively impacting generator performance when the number of classes is large. Another prevalent approach, particularly in the image domain, is the injection of class label information into the generator's batch normalization layer [146]. Although successful in the image domain, the batch normalization is not utilized for time series data due to the adverse effects of it on the time-dependent correlation between consecutive steps. Thus, this method is not applicable to time series data.

To mitigate these challenges, we introduce Structured Noise Space GAN (SNS-GAN), a novel methodology for projecting class-conditional information into the generator without requiring alterations to the network structure. Our approach is agnostic to network architecture, rendering it compatible with both image and time series data.

The subsequent section delineates the proposed SNS-GAN model in detail. Following that, we apply the method to the image domain to provide a qualitative validation of our approach. Lastly, we demonstrate the model's superior efficacy in the time series domain compared to baseline models.

10.1 Structured Noise Space GAN (SNS-GAN)

In the realm of generative modeling, the ability to generate data specific to a given class out of multiple classes in a dataset is a significant milestone. Typically, data belonging to N classes exhibit N modes, with each mode corresponding to a particular class. The challenge lies in synthesizing samples that not only look realistic but also adhere to the specified class label. This requires the generative model to correctly map the input noise to the appropriate data mode, leveraging the class label information.

Current methods in Generative Adversarial Networks (GANs) typically involve sampling noise from a standard Gaussian distribution and then conditioning the generator with explicit class labels to produce class-specific outputs. However, this study introduces an innovative

approach that deviates from the norm by proposing a structured noise space for GANs. This structured noise space is multimodal, with each mode directly linked to a data class. By sampling noise from this structured distribution, the noise vector inherently carries the class information, eliminating the need for explicit class labels during generation.

To actualize the SNS-GAN, we utilize an N-dimensional Gaussian distribution to represent the noise space, where each dimension is paired with a data class. When generating a noise vector for a specific class, we adjust the mean of the corresponding dimension to a non-zero value while the other dimensions remain centered around zero. This shift in the mean value implicitly encodes the class information within the noise vector itself, guiding the generator to produce a sample from the intended class.

The process of generating structured noise vectors, denoted as z_c, is facilitated by the reparameterization technique. Initially, a noise vector z is drawn from an N-dimensional Gaussian distribution. The mean of the vector is then altered based on the one-hot encoding of the target class c:

$$z_c = z + \text{one\_hot\_encode}(c) \tag{10.1}$$

This strategy simplifies the sampling process from the structured noise space and enhances the efficiency of GAN training. Figure 10.1 illustrates the modified data pipeline for the generator within the SNS-GAN framework. By supplying the generator with the structured noise vector z_c, it learns to map this input to the corresponding class-specific sample without necessitating alterations to the generator's internal architecture or imposing any constraints on its structure.

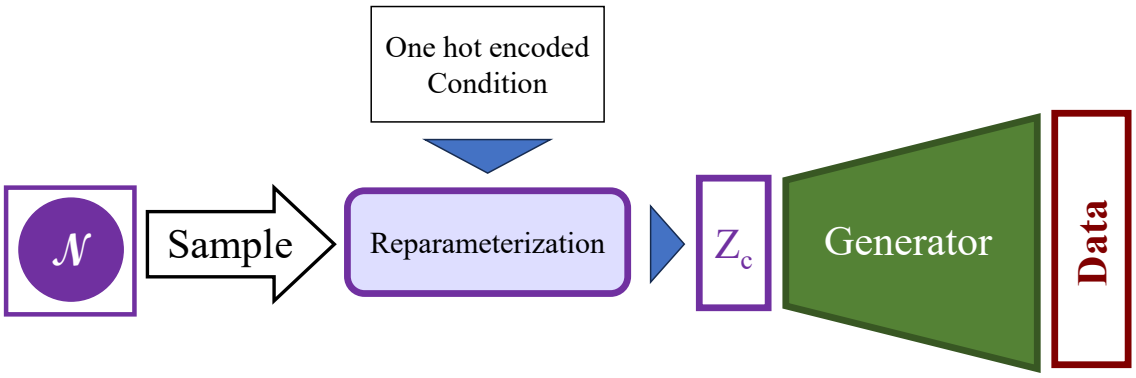


Fig. 10.1 The modified data pipeline for the SNS-GAN generator.

As a result, the generator inherently understands the class conditions and operates effectively

as if it were an unmodified, unconditional GAN. This novel approach not only simplifies the training process but also opens up new possibilities for the application of GANs across various domains where class-specific data generation is essential.

The SNS-GAN is trained using the original GAN objective as outlined in Equation(2.4). In all experiments carried out in this study, the proposed model is trained using Adam optimizer with the learning rate set to 0.0002 and $\beta_1 = 0.5$ and $\beta_2 = 0.99$.

10.2 Experiment I: Proof of Concept on Image Domain

The inception of SNS-GAN is rooted in the need for a robust method to generate time series data. However, the complex and non-intuitive nature of time series data poses significant challenges for qualitative analysis of new generative approaches. Although quantitative measures exist for evaluating GANs, their limitations necessitate a qualitative assessment to gauge the model's generative capabilities fully. Consequently, we first apply SNS-GAN to the image domain, where visual inspection can intuitively validate the model's effectiveness. This experiment serves as a foundation for subsequent applications to time series data.

10.2.1 Datasets

MNIST

To evaluate SNS-GAN, we utilized the MNIST dataset [147], which comprises grayscale images of handwritten digits (0–9). Each image is assigned a label within the range [0–9]. The dataset consists of 60,000 training and 10,000 testing images, providing a standard benchmark for assessing generative models.

CIFAR10

For a more challenging scenario, we employed the CIFAR-10 dataset [148], featuring color images across ten distinct classes, including various natural scenes and objects. With its 50,000 training and 10,000 test images, CIFAR-10 escalates the complexity for generative models and serves as an ideal candidate to test SNS-GAN's capabilities.

10.2.2 Results and Discussion

Figure 10.2 illustrates the architectural configuration of SNS-GAN for both the MNIST and CIFAR-10 datasets, highlighting the generator's transposed convolution operations and the

discriminator's 2D convolutions. The hyperparameters employed in our experiments are detailed in Table 10.1.

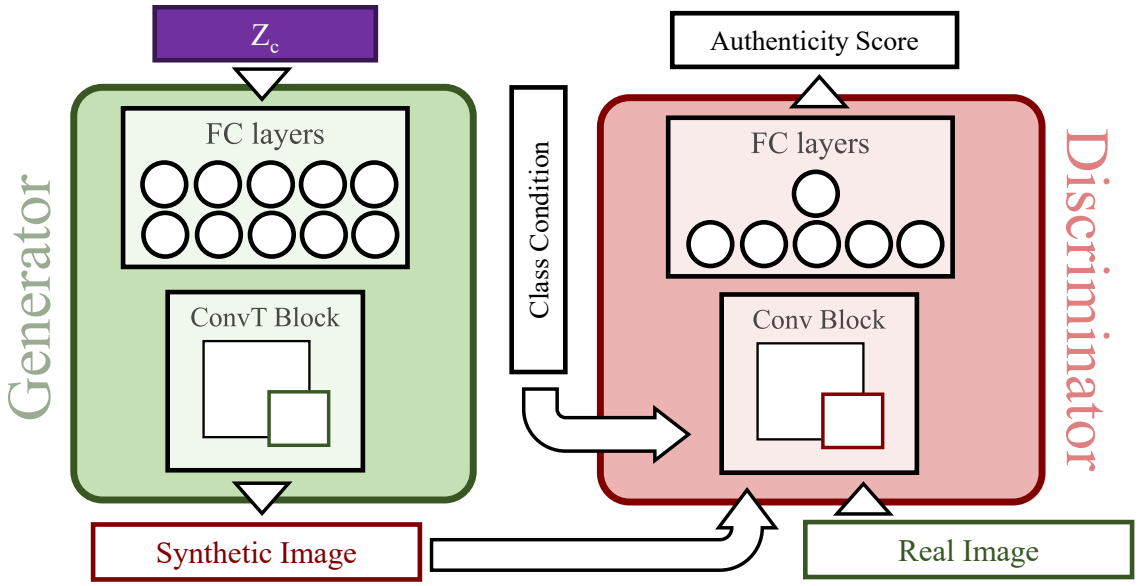


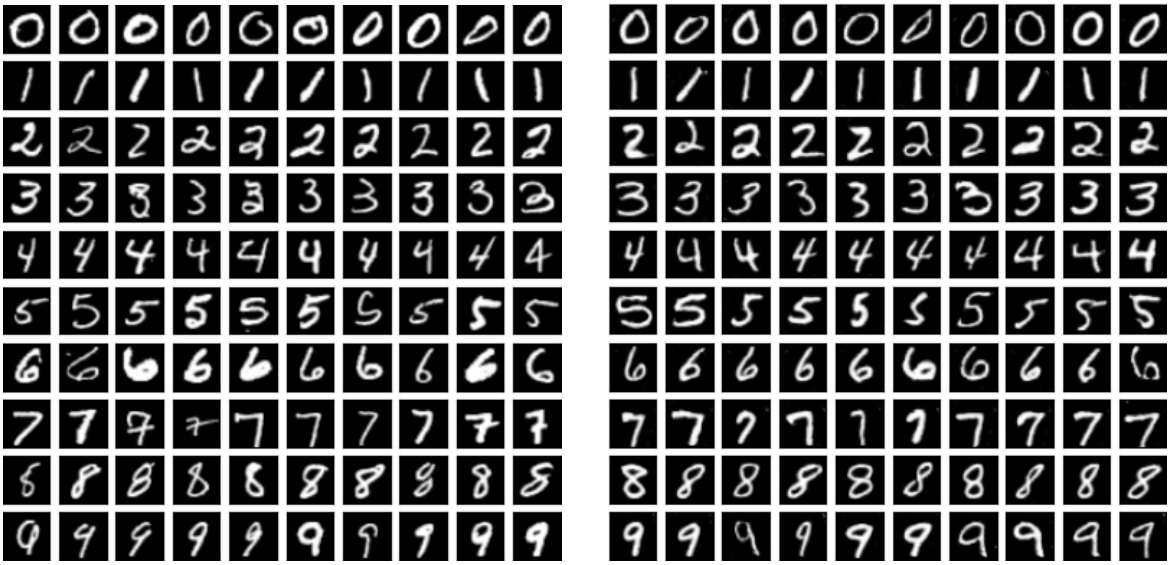
Fig. 10.2 The SNS-GAN architecture for image domain experiments, featuring transposed convolution in the generator and 2D convolution in the discriminator.

Table 10.1 Hyperparameters for SNS-GAN in image domain experiments.

	MNIST	CIFAR10
Noise size	1000 (100×10)	1000 (100×10)
Transposed convolution layers	2	4
Kernel size (Transposed convolution)	4×4	4×4
Stride (Transposed convolution)	2	2
Convolution layers (Discriminator)	2	4
Kernel size (Convolution)	4×4	4×4
Stride (Convolution)	2	2

Visual comparisons between generated samples and real dataset images are depicted in Figures 10.3 and 10.4 for the MNIST and CIFAR-10 datasets, respectively. The results suggest that SNS-GAN successfully enforces class-specific conditions on the generator. The samples display not only high fidelity but also considerable diversity within each class.

Figure 10.4 presents generated CIFAR-10 samples alongside real ones. Despite the complexity introduced by color and natural scene content, SNS-GAN demonstrates proficient



(a) Real MNIST data (b) SNS-GAN generated MNIST data

Fig. 10.3 Comparison of real and SNS-GAN generated samples from the MNIST dataset.

(a) Real CIFAR-10 data (b) SNS-GAN generated CIFAR-10 data

Fig. 10.4 Comparison of real and SNS-GAN generated samples from the CIFAR-10 dataset.

conditional enforcement. Notably, diversity within classes is also preserved. CIFAR-10's low resolution complicates visual fidelity assessment, prompting us to employ Inception Score (IS) [149] and Fréchet Inception Distance (FID) [105] for a quantitative evaluation. These metrics are standard in the image domain for evaluating the quality and diversity of GAN outputs. The comparison with 13 other established GAN models presented in Table 10.2 suggests that SNS-GAN, even with its relatively simple structure, ranks competitively, particularly in terms of FID.

Table 10.2 Quantitative results for CIFAR10 comparing the proposed model with 13 other GAN models.

	IS	FID
FCGAN	6.41	42.6
BEGAN	5.62	-
PROGAN	8.80	-
LSGAN	6.76	29.5
DCGAN	6.69	42.5
WGAN-GP	8.21	21.5
SN-GAN	8.43	18.8
Geometric GAN	-	27.1
RGAN	-	15.9
ACGAN	8.25	-
BigGAN	**9.22**	**14.7**
RealnessGAN	-	34.6
SS-GAN	-	15.7
SNS-GAN	6.9	14.46

These findings affirm SNS-GAN's utility for implicit class conditioning and pave the way for its application in the time series domain. The following sections will extend the methodology to time series data, underscoring the potential of SNS-GAN for generating class-conditional sequences.

10.3 Experiment II: Time Series Domain

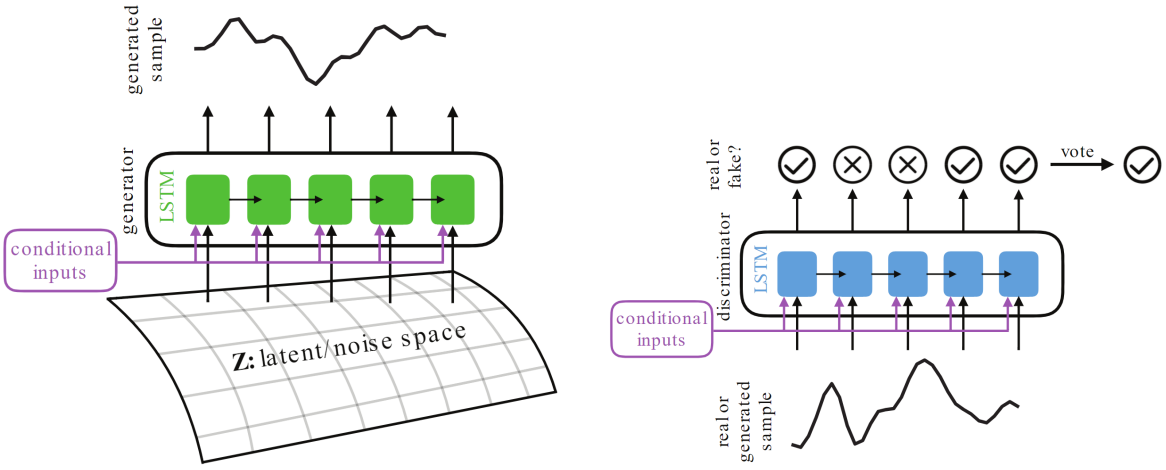
This chapter delves into the application of the SNS-GAN model to the generation of class-conditional time series data. We aim to discern whether the SNS-GAN generator can distinguish between classes in the time series domain without explicit class labels. This inquiry involves outlining the datasets and reference models, detailing the architecture of the proposed framework, and presenting the experimental findings.

Table 10.3 Dataset employed for time series experiments alongside their key characteristics

Feature	Name	Class	Length	Train	Test
low class number low time step length	Smooth Subspace	3	15	150	150
low class number high time step length	Strawberry	2	235	613	370
high class number low time step length	Crop	24	46	7200	16800
high class number high time step length	Fifty Words	50	270	450	455

10.3.1 Datasets

Our evaluation employs four datasets from the UCR benchmark [108], a repository of 128 univariate time series datasets. The selection was motivated by the diversity in the number of classes and the length of time steps, offering a comprehensive examination of model performance under varied conditions. The chosen datasets, enumerated in Table 10.3, represent a range of feature combinations to test the models' adaptability.



(a) The RCGAN generator

(b) The RCGAN discriminator

Fig. 10.5 Schematic of condition imposing pipeline in RCGAN. Figures are adopted from [95]

10.3.2 Baseline

To benchmark SNS-GAN's efficacy, we employ the Recurrent Conditional GAN (RCGAN) [95] as the baseline model. RCGAN leverages recurrent neural networks (RNNs) within the GAN architecture to generate time series data, primarily for medical applications.

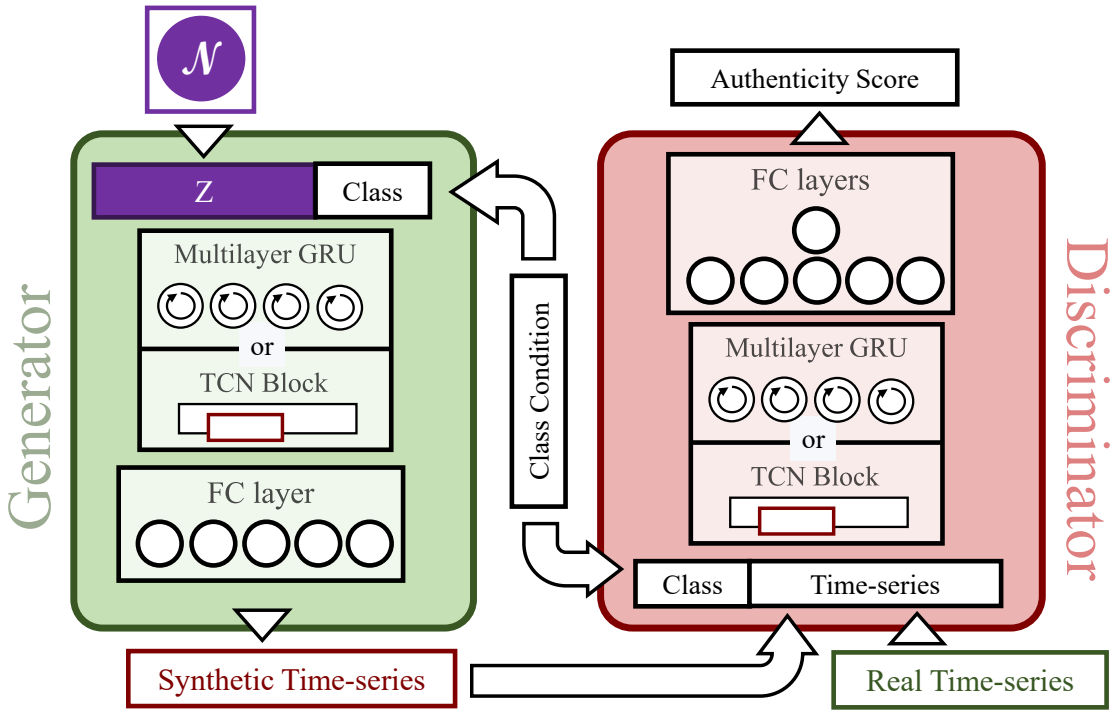


Fig. 10.6 RCGAN architecture employed as the baseline in this study

As depicted in Figure 10.5, RCGAN feeds conditional information explicitly to both the generator and discriminator. The generator processes different noise vectors concatenated with the conditions at each time step (Figure 10.5a), while the discriminator evaluates the authenticity of time series data at each individual step (Figure 10.5b).

We adopt two RCGAN architectures, RCGAN-RNN, which adheres to the original specification using GRUs, and RCGAN-TCN, which substitutes RNN units with Temporal Convolutional Networks (TCNs) for its primary components. Figure 10.6 presents the implemented RCGAN architecture and Table 10.4 details the RCGAN configurations.

Table 10.4 RCGAN hyperparameters

	Variant	Generator	Discriminator
GRU layers	RCGAN-RNN	1	1
GRU hidden size	RCGAN-RNN	256	256
TCN layers	RCGAN-TCN	1	1
TCN kernel size	RCGAN-TCN	8	8

10.3.3 SNS-GAN for Class Conditional Time Series Generation

The SNS-GAN architecture tailored for time series is depicted in Figure 10.7. Three variants are introduced: SNS-GAN-Linear, which relies on fully connected layers; SNS-GAN-RNN, which uses GRUs; and SNS-GAN-TCN, which incorporates TCNs. The Linear and TCN variants generate entire sequences in one pass, whereas the RNN variant employs an autoregressive approach, where the time steps are generated sequentially. The hyperparameters for the SNS-GAN models are specified in Table 10.5.

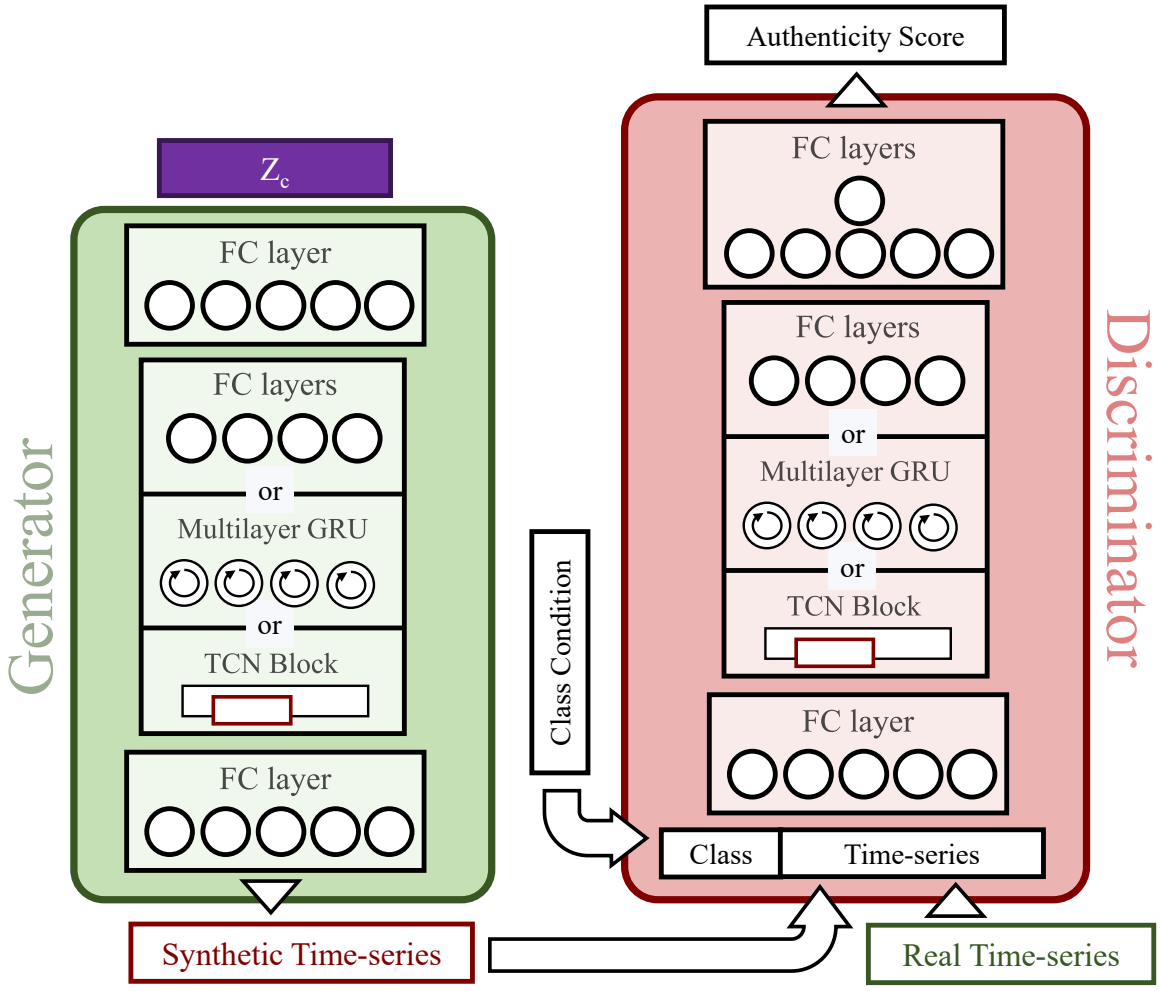


Fig. 10.7 SNS-GAN architecture employed for time series generation

10.3.4 Results and Discussion

To quantify the quality of generated samples in the time series domain, we utilized ITS and FITD as introduced in Chapter 8 this thesis.

Table 10.5 SNS-GAN hyperparameters

	Variant	Generator	Discriminator
FC layers	SNS-GAN-Linear	1	1
FC hidden size	SNS-GAN-Linear	500	500
GRU layers	SNS-GAN-RNN	1	1
GRU hidden size	SNS-GAN-RNN	512	512
TCN layers	SNS-GAN-TCN	1	1
TCN kernel size	SNS-GAN-TCN	8	8

Table 10.6 The quantitative results of class conditional time series generation

	Smooth Subspace ITS/FITD	Strawberry ITS/FITD	Crop ITS/FITD	Fifty Words ITS/FITD
Real Data	**2.93/0.097**	**1.98/0.039**	**22.76/0.0004**	**37.45/0.15**
SNS-GAN-Linear	**2.88/2.04**	**1.967/0.30**	**19.19/0.05**	**35.08/1.65**
SNS-GAN-TCN	2.83/2.79	1.961/0.82	11.92/0.63	10.77/4.48
SNS-GAN-RNN	2.66/3.73	1.91/2.15	18.88/0.07	13.65/3.32
RCGAN-TCN	2.70/3.38	1.95/1.17	9.83/0.74	8.49/23.49
RCGAN-RNN	2.59/4.45	1.93/1.37	15.86/0.14	20.23/2.51

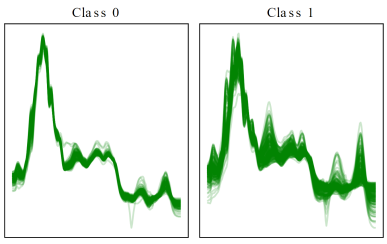
The experimental outcomes are summarized in Table 10.6. The SNS-GAN-Linear model stands out, demonstrating its capability to capture temporal dependencies across all time steps effectively. Figure 10.8 showcases qualitative samples from this model, evidencing its accuracy in class representation and dynamic replication.

While ITS scores are comparable for models on datasets with fewer classes, significant divergence is observed on datasets with a larger class spectrum. RNN-based models excel on the Crop dataset; however, their performance declines on the Fifty words dataset due to error accumulation in long series generation—contrasted by their proficiency with shorter series as depicted in Figure 10.9.

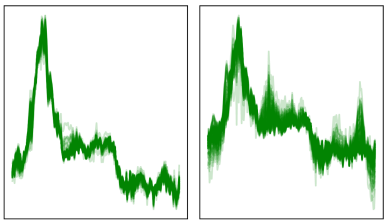
The TCN-based model, however, does not distinctly excel on any dataset and is prone to mode collapse, as demonstrated in Figure 10.10.

In conclusion, the SNS-GAN-Linear variant emerges as the most adept at fulfilling condition imposition and generating authentic time series. It is best suited for fixed-size time series

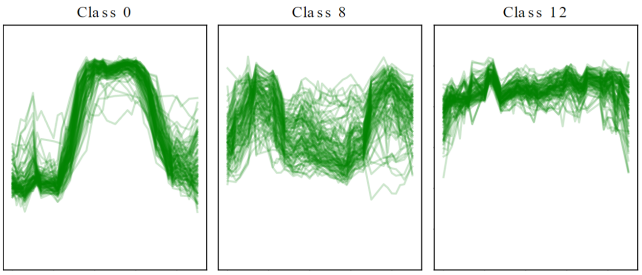
generation, while the RNN variant offers flexibility in length at the cost of diminished quality for longer sequences.



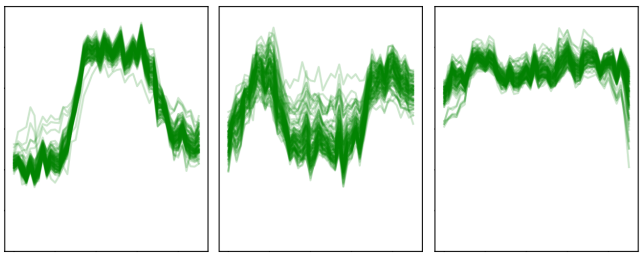
(a) Real samples of Strawberry dataset



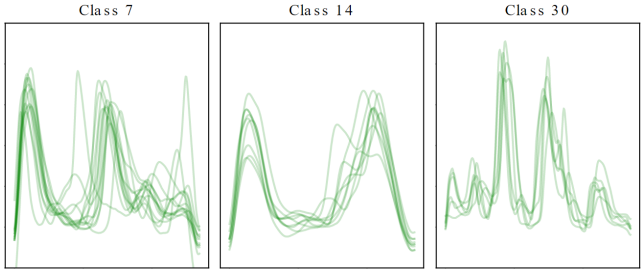
(b) Generated samples for Strawberry dataset



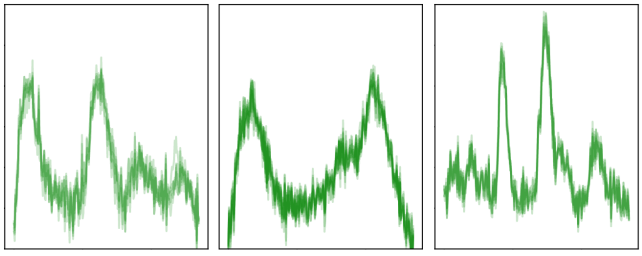
(c) Real samples of Crop dataset



(d) Generated samples of Crop dataset

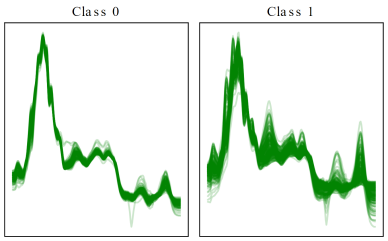


(e) Real samples of Fifty words dataset

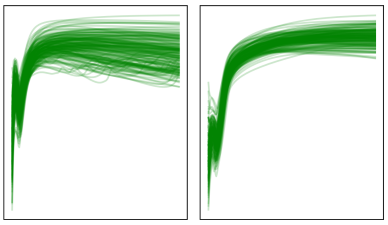


(f) Generated samples of Fifty words dataset

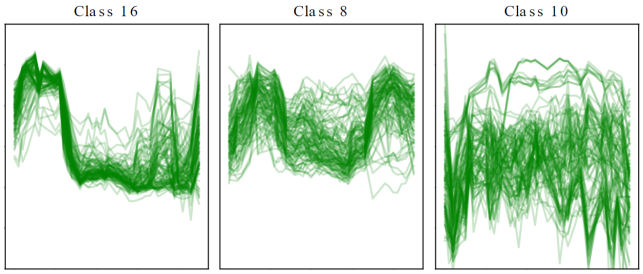
Fig. 10.8 Qualitative results for SNS-GAN-Linear model



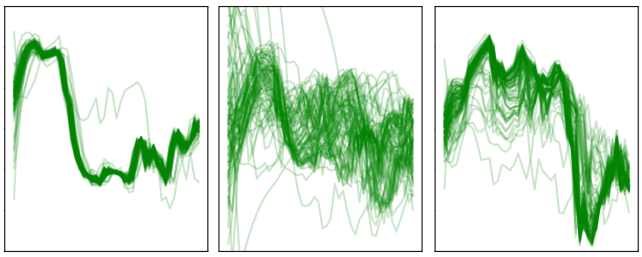
(a) Real samples of Strawberry dataset



(b) Generated samples for Strawberry dataset

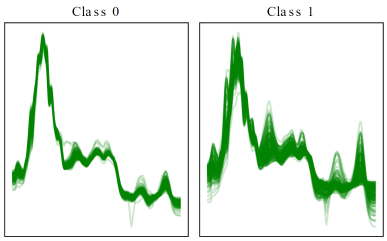


(c) Real samples of Crop dataset

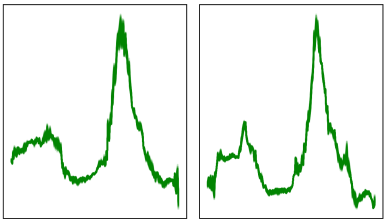


(d) Generated samples of Crop dataset

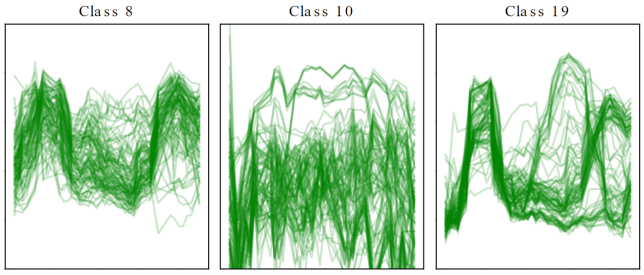
Fig. 10.9 Qualitative results for SNS-GAN-RNN model



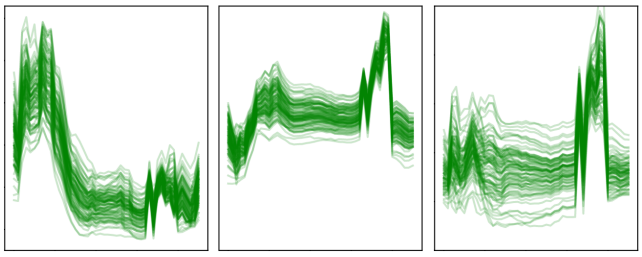
(a) Real samples of Strawberry dataset



(b) Generated samples for Strawberry dataset



(c) Real samples of Crop dataset



(d) Generated samples of Crop dataset

Fig. 10.10 Qualitative results for SNS-GAN-TCN model

Part IV

Conclusion and Future Works

Chapter 11

Conclusion and Future Works

In this thesis, we set out with the overarching objective of enhancing decision-making capabilities through improved probabilistic forecasting. Recognizing the dual necessity of both robust models and extensive datasets for effective machine learning, our research has contributed significant advancements on both these fronts. By developing innovative models and methodologies for data augmentation, we have aimed to bolster the entire pipeline of probabilistic forecasting, thereby facilitating more informed and precise decision-making processes in various fields.

11.1 Probabilistic Forecasting modeling

Throughout this research, we have emphasized the importance of innovative approaches in the realm of probabilistic forecasting. By introducing and refining novel methodologies such as ForGAN, VAEneu, and ProbCast, we have pushed the boundaries of what is possible with Generative models in forecasting scenarios. These methods represent a significant leap forward, not only in their technical sophistication but also in their practical applicability to real-world forecasting problems.

11.1.1 Discrimination Ability of Scoring Rules

An in-depth empirical analysis of established methodologies for evaluating probabilistic forecasters was undertaken (see Chapter 4). This included a critical evaluation of the newly introduced CRPS-Sum scoring rule. Our findings highlight significant drawbacks in how CRPS-Sum reflects a model's performance, leading to the recommendation against its use due to potential misrepresentation of model effectiveness.

Although this thesis pointed out the unreliability of CRPS-Sum however, the task of unbiased assessment of probabilistic forecasting on multivariate time series remains an open problem. A potential future direction can be proposing a new strictly proper scoring rule for multivariate data. Another approach is the study of how well a collection of proper scoring rules together can objectively reflect the performance of probabilistic forecasters for multivariate time series.

11.1.2 One-step-ahead Probabilistic Forecasting on Univariate Time series

Chapter 5 introduced ForGAN, a groundbreaking probabilistic forecaster utilizing GANs. Additionally, we developed the CRPS Loss function, a novel training loss specifically tailored for probabilistic forecasting models. This led to the creation of VAEneu, a Variational Auto-Encoder-based probabilistic forecaster. Furthermore, this thesis suggests adopting the CRPS Loss as an auxiliary loss function to enhance the stability and accuracy of ForGAN. Our comprehensive study across 13 datasets established the superiority of these models over the baseline model, particularly highlighting the effectiveness of VAEneu in terms of accuracy and training efficiency. The CRPS Loss, in particular, demonstrated its effectiveness in enhancing the performance of ForGAN and in training VAEneu.

11.1.3 Multi-step-ahead Probabilistic Forecasting on Univariate Time series

In Chapter 6, this thesis extends the one-step-ahead forecasting models with the Auto-Regression technique to the realm of multi-step-ahead forecasting. Furthermore, this thesis has taken these models a step further by integrating an attention-based sequence-to-sequence (seq2seq) structure. This integration equips both ForGAN and VAEneu with the capability to generate forecasts across the entire horizon of the target forecast, thus enhancing their versatility and applicability.

A comprehensive empirical analysis spans 12 datasets is conducted and involves a comparative assessment against 11 well-established probabilistic forecasting models. The results from these extensive experiments are quite revealing and underscore the superior performance of the proposed models. In 9 out of the 12 datasets, the proposed models outperformed the baselines and, in certain instances, improved the CRPS of the state-of-the-art by more than 50%. This remarkable achievement highlights the potency of the proposed methodologies in

delivering highly accurate forecasts.

Among the various models developed, the Auto-Regressive ForGAN model emerged as the frontrunner, showcasing superior performance across all datasets. However, the other models, including the attention-based variants of ForGAN and VAEneu, also demonstrated commendable performance, closely trailing the leading model.

An interesting aspect of these proposed models is that they are constructed without explicit hyperparameter tuning, relying instead on heuristics for hyperparameter determination. The exceptional performance achieved under these conditions further validates the robustness and adaptability of the proposed models in different forecasting scenarios, irrespective of the hyperparameter settings.

Further investigations into these models revealed several key strengths. Not only did they exhibit unparalleled accuracy in probabilistic forecasting, but they also demonstrated rapid convergence to the most optimal model parameters during the training phase. Additionally, their ability to efficiently generate forecasts during inference makes them highly suitable for real-time forecasting scenarios.

In conclusion, the multi-step-ahead probabilistic forecasting models presented in this thesis are not just incremental improvements over existing methodologies; they represent a paradigm shift in the field of probabilistic forecasting. These models, with their advanced capabilities and robust performance, have effectively set a new benchmark, propelling the field of probabilistic forecasting forward and opening up new possibilities for future research and applications.

The emergence of ForGAN and VAEneu opens up various fronts for further investigations. For instance, this thesis has primarily explored specific architectural components such as Recurrent Neural Networks (RNNs), Temporal Convolutional Networks (TCNs), and Sequence-to-Sequence (Seq2Seq) models for proposed models. However, the potential for incorporating additional architectural elements, particularly the emerging and powerful transformers, presents an exciting opportunity to enhance model accuracy and efficiency further.

Another key area for future exploration is the comprehensive analysis of the effect of hyperparameters on proposed models' performance. This thesis did not exhaustively investigate

the complicated dynamics of hyperparameters, especially for attention-based models. This gap in the research opens up an intriguing avenue for future studies to delve deeper into optimizing these parameters, thereby unlocking the full potential of the models.

11.1.4 One-Step-Ahead Probabilistic Forecasting on Multivariate Times series

The exploration of multivariate time series forecasting, as undertaken in Chapter 7 of this thesis, addresses the inherent complexity of this domain compared to its univariate counterpart. The chapter introduces **ProbCast**, a novel framework conceptualized to seamlessly transition deterministic models into the probabilistic forecasting paradigm using GANs with minimal structural alterations.

Empirical validation of the ProbCast framework was conducted using two publicly available datasets. These experiments were designed to test the framework's efficacy in transforming deterministic models into their probabilistic equivalents. The results of these experiments were revealing; they demonstrated that ProbCast could not only successfully perform this transformation but also enhance the accuracy of the resulting probabilistic models. This improvement in accuracy highlights the framework's potential to deliver more reliable and precise forecasts, especially in complex multivariate scenarios.

In the domain of multivariate time series forecasting, extending the proposed models for multistep-ahead forecasting via auto-regression represents a valuable line of inquiry. Additionally, modifying the CRPS Loss for multivariate time series will enable the VAEneu model to effectively handle and forecast across a broader spectrum of multivariate time series datasets, thereby significantly expanding its utility and relevance.

11.2 Time series Data Augmentation

The thesis acknowledges the critical role of data in machine learning; we have made substantial contributions to the field of data augmentation. The development and implementation of techniques such as novel assessment methods for time series generative models, Rare Event-aware generative learning with GANs, and novel class conditioning for GANs exemplify our commitment to addressing the data scarcity challenge. These advancements not only aid in improving the models' training efficiency and effectiveness but also enhance the models' ability to generalize and perform in diverse and challenging scenarios.

11.2.1 Novel Methods for Gauging Performance of Generative Model on Time series Domain

A significant barrier to the progression of generative models in the time series domain has been the absence of a universally accepted method for evaluating their performance. This thesis addresses this gap by introducing two innovative assessment methods: the InceptionTime Score (ITS) and the Fréchet Inception Time Distance (FITD). These metrics are designed specifically to evaluate the effectiveness of generative models in the context of time series data.

An extensive empirical study forms the core of Chapter 8, where ITS and FITD are applied across a diverse set of 80 datasets. This comprehensive evaluation not only validates the effectiveness of these metrics in assessing the performance of generative models in the time series domain but also leads to the development of interpretive guidelines. These guidelines aid researchers and practitioners in understanding the nuances of ITS and FITD scores, facilitating a deeper insight into the capabilities and limitations of generative models in the time series domain.

These metrics provide much-needed tools for objectively measuring the performance of generative models, paving the way for more rigorous development and comparison in this critical area of machine learning research.

11.2.2 Extreme Event Aware Time series Generative Model

The unique challenge of data scarcity in time series analysis often manifests in the infrequent occurrence of specific patterns, particularly those representing extreme events. These patterns, while rare, can be of critical importance in various applications. For instance, in meteorological datasets, patterns representing heavy rainfall events are rare but crucial for accurate forecasting and risk management. Recognizing this, Chapter 9 of this thesis introduces a specialized generative model designed to enhance the representation of such extreme events within datasets. The method employs GAN as the foundation, and an auxiliary loss function is integrated into the generator's training process. This loss function is specifically designed to target the tail of the data distribution, where extreme events are situated.

The enhanced generative model's practical effectiveness is demonstrated through its application in forecasting scenarios. In a real-world dataset, where extreme events are underrepresented, supplementing the training data with synthetically generated samples that accurately

reflect these extreme occurrences leads to a marked improvement in forecasting performance. The model's ability to augment datasets with high-quality representations of extreme events proves invaluable, particularly in tasks where predicting such events accurately is paramount.

An intriguing avenue for extending this study lies in conducting more expansive experiments, incorporating a wider array of datasets. Such an approach would further validate and solidify the promise of the proposed model. Additionally, a detailed analysis of the tail-focused auxiliary loss function's hyperparameters presents another compelling research direction. Investigating these parameters could unravel insights into how they influence the model's behavior, particularly in terms of controlling the frequency and representation of extreme events in the generated data.

11.2.3 Class Conditional Time series Generation

Class conditional generative models represent a significant stride in the field of machine learning, offering the capability to generate data samples based on specific class labels. This ability not only grants greater control over the data generation process but also facilitates targeted dataset augmentation, especially in scenarios where enhancing data representation for certain classes is crucial. However, in the time series domain, conditioning generative models on class labels have traditionally been limited to more rudimentary approaches.

Recognizing this gap, Chapter 10 of this thesis introduces the Structured Noise Space GAN (SNS-GAN), a novel approach to conditioning GAN's generator without necessitating alterations in its structural design or imposing specific architectural requirements. The SNS-GAN method innovatively leverages the concept of structured noise space to embed class conditions within the generator's noise space. This technique avoids the complexities and limitations associated with direct conditioning methods, enabling the generator to discern and respond to class-specific information intrinsically present within the noise vector.

Empirical studies reveal that while SNS-GAN exhibits commendable performance in generating image samples, it is particularly effective in the time series domain. The model's ability to accurately and effectively impose class conditions on the generator is highlighted by its outstanding performance on 4 distinct datasets. Results indicate a considerable margin of superiority over baseline models based on both FITD and ITS.

The introduction of Structured Noise Space GAN (SNS-GAN) in this thesis represents a pioneering step in the realm of GAN conditioning, particularly focusing on the time series

domain. This novel approach, which embeds class conditions within the noise space, lays the groundwork for a plethora of research possibilities and enhancements in this field. The flexibility of SNS-GAN, highlighted by its ability to adapt to various generative architectures without necessitating structural modifications, is a particularly promising aspect. This feature enables the potential application of SNS-GAN across a more diverse range of generative models.

An immediate extension of this work could involve exploring a broader array of datasets and generative architectures in both the image and time series domains. Such explorations would provide a more comprehensive understanding of SNS-GAN's strengths and limitations, enhancing its applicability and performance. Furthermore, the versatility of SNS-GAN suggests its potential utility in domains beyond traditional image and time series analysis. Investigating its application in areas such as music generation or synthetic voice creation could unveil new dimensions of this model, contributing significantly to the advancement of generative modeling techniques across various types of data. These future directions not only highlight the innovative nature of SNS-GAN but also underscore its potential to revolutionize the field of generative modeling.

11.3 Broad Impact and Contributions to Probabilistic Forecasting

This thesis represents a comprehensive and impactful contribution to the field of probabilistic forecasting, with a focus on enhancing decision-making processes across various domains. The novel methodologies introduced not only showcase technical innovation but also practical utility, especially in real-world scenarios.

In Chapter 5, the thesis presents a patented application of the ForGAN model within the automotive industry. This application demonstrates the model's utility in time series forecasting, particularly in predicting and optimizing engine performance. By implementing the ForGAN model, we can significantly extend the lifespan of car engine parts, enable predictive maintenance strategies, and contribute to environmental conservation by reducing air pollution. This is achieved through the model's ability to facilitate engines running at optimal conditions, highlighting the real-world impact and industrial relevance of the research.

Chapter 9 explores the application of the proposed models for data augmentation in the water management domain. This involves generating synthetic time series data, focusing on extreme weather conditions like heavy rainfall. By enriching datasets with these rare event scenarios, the research significantly strengthens the performance of downstream forecasting tasks. This contribution is particularly important in enhancing the preparedness and responsiveness to extreme weather events, which is vital for effective water management.

In summary, this thesis represents a comprehensive effort to strengthen the entire pipeline of probabilistic forecasting. By marrying advanced modeling techniques with innovative approaches to data augmentation, we have laid a foundation for more accurate, reliable, and effective decision-making tools. This, we believe, will have a lasting impact on various domains where predictive accuracy is paramount.

References

[1] David Foster. *Generative Deep Learning: Teaching Machines to Paint, Write, Compose, and Play*. en. "O'Reilly Media, Inc.", June 2019. ISBN: 978-1-4920-4189-4.

[2] Ian Goodfellow. *NIPS 2016 Tutorial: Generative Adversarial Networks*. Apr. 2017. DOI: 10.48550/arXiv.1701.00160. URL: http://arxiv.org/abs/1701.00160.

[3] David Salinas et al. "DeepAR: Probabilistic forecasting with autoregressive recurrent networks". In: *International Journal of Forecasting* 36.3 (2020), pp. 1181–1191.

[4] Aaron van den Oord et al. *WaveNet: A Generative Model for Raw Audio*. Sept. 2016. DOI: 10.48550/arXiv.1609.03499. URL: http://arxiv.org/abs/1609.03499.

[5] Ian Goodfellow et al. "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems*. Vol. 27. Curran Associates, Inc., 2014, pp. 2672–2681. URL: https://papers.nips.cc/paper_files/paper/2014/hash/5ca3e9b122f61f8f06494c97b1afccf3-Abstract.html.

[6] Ishaan Gulrajani et al. "Improved training of wasserstein gans". In: *Advances in neural information processing systems* 30 (2017).

[7] Cédric Villani. *Optimal transport: old and new*. Vol. 338. Springer, 2009.

[8] Xudong Mao et al. "Least Squares Generative Adversarial Networks". In: 2017, pp. 2794–2802. URL: https://openaccess.thecvf.com/content_iccv_2017/html/Mao_Least_Squares_Generative_ICCV_2017_paper.html.

[9] Mehdi Mirza and Simon Osindero. "Conditional generative adversarial nets". In: *arXiv preprint arXiv:1411.1784* (2014).

[10] G. E. Hinton and R. R. Salakhutdinov. "Reducing the Dimensionality of Data with Neural Networks". In: *Science* 313.5786 (July 2006), pp. 504–507. DOI: 10.1126/science.1127647. URL: https://www.science.org/doi/10.1126/science.1127647.

[11] Diederik P Kingma and Max Welling. "Auto-encoding variational bayes". In: *arXiv preprint arXiv:1312.6114* (2013).

[12] Alireza Makhzani et al. *Adversarial Autoencoders*. May 2016. DOI: 10.48550/arXiv.1511.05644. URL: http://arxiv.org/abs/1511.05644.

[13] G. Udny Yule. "On a Method of Investigating Periodicities in Disturbed Series, with Special Reference to Wolfer's Sunspot Numbers". In: *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character* 226 (1927), pp. 267–298. ISSN: 02643952. URL: http://www.jstor.org/stable/91170.

[14] Herman O. A. Wold. "On Prediction in Stationary Time Series". In: *The Annals of Mathematical Statistics* 19.4 (Dec. 1948), pp. 558–567. ISSN: 0003-4851, 2168-8990. DOI: 10.1214/aoms/1177730151. URL: https://projecteuclid.org/journals/annals-of-mathematical-statistics/volume-19/issue-4/On-Prediction-in-Stationary-Time-Series/10.1214/aoms/1177730151.full.

[15] George Edward Pelham Box and Gwilym Jenkins. *Time Series Analysis, Forecasting and Control*. San Francisco, CA, USA: Holden-Day, Inc., 1990. ISBN: 0-8162-1104-3.

[16] O. L. Davies and Robert G. Brown. "Statistical Forecasting for Inventory Control." In: *Journal of the Royal Statistical Society. Series A (General)*. Vol. 123. 1960, p. 348. DOI: 10.2307/2342487. URL: https://www.jstor.org/stable/10.2307/2342487?origin=crossref.

[17] Rob Hyndman et al. *Forecasting with Exponential Smoothing*. Springer Series in Statistics. Berlin, Heidelberg: Springer, 2008. ISBN: 978-3-540-71916-8. DOI: 10.1007/978-3-540-71918-2. URL: http://link.springer.com/10.1007/978-3-540-71918-2.

[18] Andrew C. Harvey. *Forecasting, Structural Time Series Models and the Kalman Filter*. en. Cambridge University Press, 1990. ISBN: 978-0-521-40573-7.

[19] Zhe Chen and Emery N. Brown. "State space model". en. In: *Scholarpedia* 8.3 (Mar. 2013), p. 30868. ISSN: 1941-6016. DOI: 10.4249/scholarpedia.30868. URL: http://www.scholarpedia.org/article/State_space_model.

[20] Trevor J. Hastie. "Generalized Additive Models". In: *Statistical Models in S*. Routledge, 1992. ISBN: 978-0-203-73853-5.

[21] Robert F Engle. "Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation". In: *Econometrica: Journal of the econometric society* (1982), pp. 987–1007.

[22] Tim Bollerslev. "Generalized autoregressive conditional heteroskedasticity". In: *Journal of econometrics* 31.3 (1986), pp. 307–327.

[23] Daniel B. Nelson. "Conditional Heteroskedasticity in Asset Returns: A New Approach". In: *Econometrica* 59.2 (1991), pp. 347–370. ISSN: 0012-9682. DOI: 10.2307/2938260. URL: https://www.jstor.org/stable/2938260.

[24] Jean-Michel Zakoian. "Threshold heteroskedastic models". en. In: *Journal of Economic Dynamics and Control* 18.5 (Sept. 1994), pp. 931–955. ISSN: 0165-1889. DOI: 10.1016/0165-1889(94)90039-6. URL: https://www.sciencedirect.com/science/article/pii/0165188994900396.

[25] Lawrence R. Glosten, Ravi Jagannathan, and David E. Runkle. "On the Relation between the Expected Value and the Volatility of the Nominal Excess Return on Stocks". en. In: *The Journal of Finance* 48.5 (1993), pp. 1779–1801. ISSN: 1540-6261. DOI: 10.1111/j.1540-6261.1993.tb05128.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1540-6261.1993.tb05128.x.

[26] Emmanouil A. Platanios and Sotirios P. Chatzis. "Gaussian Process-Mixture Conditional Heteroscedasticity". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.5 (May 2014), pp. 888–900. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2013.183.

[27] Tilmann Gneiting et al. "Calibrated Probabilistic Forecasting Using Ensemble Model Output Statistics and Minimum CRPS Estimation". EN. In: *Monthly Weather Review* 133.5 (May 2005), pp. 1098–1118. ISSN: 1520-0493, 0027-0644. DOI: 10.1175/MWR2904.1. URL: https://journals.ametsoc.org/view/journals/mwre/133/5/mwr2904.1.xml.

[28] Adrian E. Raftery et al. "Using Bayesian Model Averaging to Calibrate Forecast Ensembles". EN. In: *Monthly Weather Review* 133.5 (May 2005), pp. 1155–1174. ISSN: 1520-0493, 0027-0644. DOI: 10.1175/MWR2906.1. URL: https://journals.ametsoc.org/view/journals/mwre/133/5/mwr2906.1.xml.

[29] David T. Frazier et al. "Approximate Bayesian forecasting". en. In: *International Journal of Forecasting* 35.2 (Apr. 2019), pp. 521–539. ISSN: 0169-2070. DOI: 10.1016/j.ijforecast.2018.08.003. URL: https://www.sciencedirect.com/science/article/pii/S016920701830147X.

[30] Ruben Loaiza-Maya, Gael M. Martin, and David T. Frazier. "Focused Bayesian prediction". en. In: *Journal of Applied Econometrics* 36.5 (2021), pp. 517–543. ISSN: 1099-1255. DOI: 10.1002/jae.2810. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/jae.2810.

[31] Michael P. Clements and Jae H. Kim. "Bootstrap prediction intervals for autoregressive time series". en. In: *Computational Statistics & Data Analysis* 51.7 (Apr. 2007), pp. 3580–3594. ISSN: 0167-9473. DOI: 10.1016/j.csda.2006.09.012. URL: https://www.sciencedirect.com/science/article/pii/S0167947306003331.

[32] Richard A. Davis and Mikkel S. Nielsen. "Modeling of time series using random forests: Theoretical developments". In: *Electronic Journal of Statistics* 14.2 (Jan. 2020), pp. 3644–3671. ISSN: 1935-7524, 1935-7524. DOI: 10.1214/20-EJS1758. URL: https://projecteuclid.org/journals/electronic-journal-of-statistics/volume-14/issue-2/Modeling-of-time-series-using-random-forests-Theoretical-developments/10.1214/20-EJS1758.full.

[33] Giorgio Corani, Alessio Benavoli, and Marco Zaffalon. "Time Series Forecasting with Gaussian Processes Needs Priors". en. In: *Machine Learning and Knowledge Discovery in Databases. Applied Data Science Track*. Ed. by Yuxiao Dong et al. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2021, pp. 103–117. ISBN: 978-3-030-86514-6. DOI: 10.1007/978-3-030-86514-6_7.

[34] Samya Tajmouati, Bouazza EL Wahbi, and Mohamed Dakkon. "Applying regression conformal prediction with nearest neighbors to time series data". In: *Communications in Statistics - Simulation and Computation* 0.0 (Apr. 2022), pp. 1–11. ISSN: 0361-0918. DOI: 10.1080/03610918.2022.2057538. URL: https://doi.org/10.1080/03610918.2022.2057538.

[35] Tim Januschowski et al. "Forecasting with trees". en. In: *International Journal of Forecasting*. Special Issue: M5 competition 38.4 (Oct. 2022), pp. 1473–1481. ISSN: 0169-2070. DOI: 10.1016/j.ijforecast.2021.10.004. URL: https://www.sciencedirect.com/science/article/pii/S0169207021001679.

[36] Abbas Khosravi, Saeid Nahavandi, and Doug Creighton. "A neural network-GARCH-based method for construction of Prediction Intervals". en. In: *Electric Power Systems Research* 96 (Mar. 2013), pp. 185–193. ISSN: 0378-7796. DOI: 10.1016/j.epsr.2012.11.007. URL: https://www.sciencedirect.com/science/article/pii/S0378779612003409.

[37] Syama Sundar Rangapuram et al. "Deep State Space Models for Time Series Forecasting". In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio et al. Vol. 31. Curran Associates, Inc., 2018. URL: https://proceedings.neurips.cc/paper_files/paper/2018/file/5cf68969fb67aa6082363a6d4e6468e2-Paper.pdf.

[38] Stephan Rasp and Sebastian Lerch. "Neural Networks for Postprocessing Ensemble Weather Forecasts". EN. In: *Monthly Weather Review* 146.11 (Nov. 2018), pp. 3885–3900. ISSN: 1520-0493, 0027-0644. DOI: 10.1175/MWR-D-18-0187.1. URL: https://journals.ametsoc.org/view/journals/mwre/146/11/mwr-d-18-0187.1.xml.

[39] Sean J. Taylor and Benjamin Letham. "Forecasting at Scale". In: *The American Statistician* 72.1 (Jan. 2018), pp. 37–45. ISSN: 0003-1305. DOI: 10.1080/00031305.2017.1380080. URL: https://doi.org/10.1080/00031305.2017.1380080.

[40] Ruofeng Wen et al. *A Multi-Horizon Quantile Recurrent Forecaster*. June 2018. DOI: 10.48550/arXiv.1711.11053. URL: http://arxiv.org/abs/1711.11053.

[41] Alex Graves. *Generating Sequences With Recurrent Neural Networks*. June 2014. DOI: 10.48550/arXiv.1308.0850. URL: http://arxiv.org/abs/1308.0850.

[42] Jan Gasthaus et al. "Probabilistic Forecasting with Spline Quantile Function RNNs". en. In: *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*. PMLR, Apr. 2019, pp. 1901–1910. URL: https://proceedings.mlr.press/v89/gasthaus19a.html.

[43] Adèle Gouttes et al. *Probabilistic Time Series Forecasting with Implicit Quantile Networks*. July 2021. DOI: 10.48550/arXiv.2107.03743. URL: http://arxiv.org/abs/2107.03743.

[44] Will Dabney et al. "Implicit Quantile Networks for Distributional Reinforcement Learning". en. In: *Proceedings of the 35th International Conference on Machine Learning*. PMLR, July 2018, pp. 1096–1105. URL: https://proceedings.mlr.press/v80/dabney18a.html.

[45] Bryan Lim et al. "Temporal Fusion Transformers for interpretable multi-horizon time series forecasting". en. In: *International Journal of Forecasting* 37.4 (Oct. 2021), pp. 1748–1764. ISSN: 0169-2070. DOI: 10.1016/j.ijforecast.2021.03.012. URL: https://www.sciencedirect.com/science/article/pii/S0169207021000637.

[46] Kelvin Kan et al. *Multivariate Quantile Function Forecaster*. Dec. 2022. DOI: 10.48550/arXiv.2202.11316. URL: http://arxiv.org/abs/2202.11316.

[47] Brandon Amos, Lei Xu, and J. Zico Kolter. "Input Convex Neural Networks". en. In: *Proceedings of the 34th International Conference on Machine Learning*. PMLR, July 2017, pp. 146–155. URL: https://proceedings.mlr.press/v70/amos17b.html.

[48] Qifa Xu et al. "QRNN-MIDAS: A novel quantile regression neural network for mixed sampling frequency data". en. In: *Neurocomputing* 457 (Oct. 2021), pp. 84–105. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2021.06.006. URL: https://www.sciencedirect.com/science/article/pii/S0925231221009012.

[49] David Salinas et al. "High-dimensional multivariate forecasting with low-rank Gaussian Copula Processes". In: *Advances in Neural Information Processing Systems*. Vol. 32. Curran Associates, Inc., 2019. URL: https://proceedings.neurips.cc/paper/2019/hash/0b105cf1504c4e241fcc6d519ea962fb-Abstract.html.

[50] Yuyang Wang et al. “Deep Factors for Forecasting”. en. In: *Proceedings of the 36th International Conference on Machine Learning*. PMLR, May 2019, pp. 6607–6617. URL: https://proceedings.mlr.press/v97/wang19k.html.

[51] Yitian Chen et al. “Probabilistic forecasting with temporal convolutional neural network”. en. In: *Neurocomputing* 399 (July 2020), pp. 491–501. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2020.03.011. URL: https://www.sciencedirect.com/science/article/pii/S0925231220303441.

[52] Emmanuel de Bézenac et al. “Normalizing Kalman Filters for Multivariate Time Series Analysis”. In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., 2020, pp. 2995–3007. URL: https://proceedings.neurips.cc/paper/2020/hash/1f47cef5e38c952f94c5d61726027439-Abstract.html.

[53] Danilo Rezende and Shakir Mohamed. “Variational Inference with Normalizing Flows”. en. In: *Proceedings of the 32nd International Conference on Machine Learning*. PMLR, June 2015, pp. 1530–1538. URL: https://proceedings.mlr.press/v37/rezende15.html.

[54] Kashif Rasul et al. “Multi-variate probabilistic time series forecasting via conditioned normalizing flows”. In: *arXiv preprint arXiv:2002.06103* (2020).

[55] Hilaf Hasson et al. “Probabilistic Forecasting: A Level-Set Approach”. en. In: Nov. 2021. URL: https://openreview.net/forum?id=VD3TMzyxKK.

[56] Kashif Rasul et al. “Autoregressive Denoising Diffusion Models for Multivariate Probabilistic Time Series Forecasting”. en. In: *Proceedings of the 38th International Conference on Machine Learning*. PMLR, July 2021, pp. 8857–8868. URL: https://proceedings.mlr.press/v139/rasul21a.html.

[57] Ali Caner Türkmen et al. “Forecasting intermittent and sparse time series: A unified probabilistic framework via deep renewal processes”. en. In: *PLOS ONE* 16.11 (Nov. 2021), e0259764. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0259764. URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0259764.

[58] J. D. Croston. “Forecasting and Stock Control for Intermittent Demands”. In: *Journal of the Operational Research Society* 23.3 (Sept. 1972), pp. 289–303. ISSN: 0160-5682. DOI: 10.1057/jors.1972.50. URL: https://doi.org/10.1057/jors.1972.50.

[59] Xingyu Zhou et al. “Stock Market Prediction on High-Frequency Data Using Generative Adversarial Nets”. en. In: *Mathematical Problems in Engineering* 2018 (Apr. 2018), e4907423. ISSN: 1024-123X. DOI: 10.1155/2018/4907423. URL: https://www.hindawi.com/journals/mpe/2018/4907423/.

[60] Kang Zhang et al. “Stock Market Prediction Based on Generative Adversarial Network”. en. In: *Procedia Computer Science*. 2018 International Conference on Identification, Information and Knowledge in the Internet of Things 147 (Jan. 2019), pp. 400–406. ISSN: 1877-0509. DOI: 10.1016/j.procs.2019.01.256. URL: https://www.sciencedirect.com/science/article/pii/S1877050919302789.

[61] Yilun Lin et al. “Pattern Sensitive Prediction of Traffic Flow Based on Generative Adversarial Framework”. In: *IEEE Transactions on Intelligent Transportation Systems* 20.6 (June 2019), pp. 2395–2400. ISSN: 1558-0016. DOI: 10.1109/TITS.2018.2857224.

[62] H. M. Dipu Kabir et al. “Partial Adversarial Training for Neural Network-Based Uncertainty Quantification”. In: *IEEE Transactions on Emerging Topics in Computational Intelligence* 5.4 (Aug. 2021), pp. 595–606. ISSN: 2471-285X. DOI: 10.1109/TETCI.2019.2936546.

[63] Xiang Yin et al. “VAECGAN: a generating framework for long-term prediction in multivariate time series”. In: *Cybersecurity* 4.1 (July 2021), p. 22. ISSN: 2523-3246. DOI: 10.1186/s42400-021-00090-w. URL: https://doi.org/10.1186/s42400-021-00090-w.

[64] Alexander Alexandrov et al. “GluonTS: Probabilistic and Neural Time Series Modeling in Python”. In: *Journal of Machine Learning Research* 21.116 (2020), pp. 1–6. URL: http://jmlr.org/papers/v21/19-820.html.

[65] Ashish Vaswani et al. “Attention is All you Need”. In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc., 2017. URL: https://papers.nips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

[66] Tilmann Gneiting and Adrian E Raftery. “Strictly proper scoring rules, prediction, and estimation”. In: *Journal of the American statistical Association* 102.477 (2007), pp. 359–378.

[67] Ludwig Baringhaus and Carsten Franz. “On a new multivariate two-sample test”. In: *Journal of multivariate analysis* 88.1 (2004), pp. 190–206.

[68] Gábor J Székely and Maria L Rizzo. “A new test for multivariate normality”. In: *Journal of Multivariate Analysis* 93.1 (2005), pp. 58–80.

[69] Pierre Pinson and Julija Tastu. *Discrimination ability of the energy score*. DTU Informatics, 2013.

[70] Florian Ziel and Kevin Berk. “Multivariate forecasting evaluation: On sensitive and strictly proper scoring rules”. In: *arXiv preprint arXiv:1910.07325* (2019).

[71] Alireza Koochali et al. “Random Noise vs. State-of-the-Art Probabilistic Forecasting Methods: A Case Study on CRPS-Sum Discrimination Ability”. In: *Applied Sciences* 12.10 (2022). ISSN: 2076-3417. DOI: 10.3390/app12105104. URL: https://www.mdpi.com/2076-3417/12/10/5104.

[72] Guokun Lai et al. “Modeling long-and short-term temporal patterns with deep neural networks”. In: *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. 2018, pp. 95–104.

[73] *NYC Taxi and Limousine Commission. TLC trip record data.* 2015. URL: https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page.

[74] Frank White. *Viscous Fluid Flow*. English. Boston: McGraw-Hill Education, Apr. 2005. ISBN: 978-0-07-124493-0.

[75] Subrahmanyan Chandrasekhar. *Hydrodynamic and hydromagnetic stability*. Courier Corporation, 2013.

[76] Colin Sparrow. *The Lorenz equations: bifurcations, chaos, and strange attractors*. Vol. 41. Springer Science & Business Media, 2012.

[77] Stephen H Kellert. *In the wake of chaos: Unpredictable order in dynamical systems*. University of Chicago press, 1993.

[78] Georges Hebrail and Alice Berard. *Individual household electric power consumption*. 2012. URL: https://doi.org/10.24432/C58K54.

[79] Paulo Cortez et al. “Multi-scale Internet traffic forecasting using neural networks and time series methods”. In: *Expert Systems* 29.2 (2012), pp. 143–155.

[80] Michael C Mackey and Leon Glass. “Oscillation and chaos in physiological control systems”. In: *Science* 197.4300 (1977), pp. 287–289.

[81] Eduardo Méndez, Omar Lugo, and Patricia Melin. “A competitive modular neural network for long-term time series forecasting”. In: *Nature-Inspired Design of Hybrid Intelligent Systems*. Springer, 2017, pp. 243–254.

[82] R. I. Ogie et al. “Optimal placement of water-level sensors to facilitate data-driven management of hydrological infrastructure assets in coastal mega-cities of developing nations”. In: *Sustainable Cities and Society* 35 (2017), pp. 385–395. ISSN: 22106707. DOI: 10.1016/j.scs.2017.08.019.

[83] Rakshitha Godahewa et al. “Monash Time Series Forecasting Archive”. In: *Neural Information Processing Systems Track on Datasets and Benchmarks*. 2021.

[84] Randall Pruim, Daniel Kaplan, and Nicholas Horton. *mosaicData: Project MOSAIC Data Sets*. Sept. 2022. URL: https://cran.r-project.org/web/packages/mosaicData/index.html.

[85] Adam Paszke et al. “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[86] Tijmen Tieleman, Geoffrey Hinton, et al. “Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude”. In: *COURSERA: Neural networks for machine learning* 4.2 (2012), pp. 26–31.

[87] Mohsin Munir et al. “DeepAnT: A Deep Learning Approach for Unsupervised Anomaly Detection in Time Series”. In: *IEEE Access* 7 (2019), pp. 1991–2005. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2018.2886457. URL: https://ieeexplore.ieee.org/abstract/document/8581424.

[88] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. “Sequence to Sequence Learning with Neural Networks”. In: *Advances in Neural Information Processing Systems*. Vol. 27. Curran Associates, Inc., 2014. URL: https://proceedings.neurips.cc/paper/2014/hash/a14ac55a4f27472c5d894ec1c3c743d2-Abstract.html.

[89] Tilmann Gneiting and Matthias Katzfuss. “Probabilistic forecasting”. In: *Annual Review of Statistics and Its Application* 1 (2014), pp. 125–151.

[90] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. en. MIT Press, Nov. 2016. ISBN: 978-0-262-33737-3.

[91] Kyunghyun Cho et al. “Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1724–1734. DOI: 10.3115/v1/D14-1179. URL: https://aclanthology.org/D14-1179.

[92] Adam Paszke et al. “Automatic Differentiation in PyTorch”. In: *NIPS 2017 Workshop on Autodiff*. 2017. URL: https://openreview.net/forum?id=BJJsrmfCZ.

[93] Ali Borji. “Pros and cons of gan evaluation measures”. In: *Computer Vision and Image Understanding* 179 (2019), pp. 41–65.

[94] Ali Borji. "Pros and cons of GAN evaluation measures: New developments". In: *Computer Vision and Image Understanding* 215 (2022), p. 103329.

[95] Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch. "Real-valued (medical) time series generation with recurrent conditional GANs". In: *arXiv preprint arXiv:1706.02633* (2017).

[96] Magnus Wiese et al. *Deep Hedging: Learning to Simulate Equity Option Markets*. en. SSRN Scholarly Paper. Rochester, NY, Oct. 2019. DOI: 10.2139/ssrn.3470756. URL: https://papers.ssrn.com/abstract=3470756.

[97] Hao Ni et al. "Conditional sig-wasserstein gans for time series generation". In: *arXiv preprint arXiv:2006.05421* (2020).

[98] Federico Gatta et al. "Neural networks generative models for time series". en. In: *Journal of King Saud University - Computer and Information Sciences* 34.10, Part A (Nov. 2022), pp. 7920–7939. ISSN: 1319-1578. DOI: 10.1016/j.jksuci.2022.07.010. URL: https://www.sciencedirect.com/science/article/pii/S1319157822002361.

[99] S. Kullback and R. A. Leibler. "On Information and Sufficiency". In: *The Annals of Mathematical Statistics* 22.1 (Mar. 1951), pp. 79–86. DOI: 10.1214/aoms/1177729694. URL: https://doi.org/10.1214/aoms/1177729694.

[100] Jinsung Yoon, Daniel Jarrett, and Mihaela van der Schaar. "Time-series Generative Adversarial Networks". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*. Ed. by Hanna M. Wallach et al. 2019, pp. 5509–5519. URL: http://papers.nips.cc/paper/8789-time-series-generative-adversarial-networks.

[101] Konstantin Shmelkov, Cordelia Schmid, and Karteek Alahari. "How good is my GAN?" In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 213–229.

[102] Suman V. Ravuri and Oriol Vinyals. "Classification Accuracy Score for Conditional Generative Models". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*. Ed. by Hanna M. Wallach et al. 2019, pp. 12247–12258. URL: http://papers.nips.cc/paper/9393-classification-accuracy-score-for-conditional-generative-models.

[103] Xiaomin Li et al. "Tts-gan: A transformer-based time-series generative adversarial network". In: *Artificial Intelligence in Medicine: 20th International Conference on Artificial Intelligence in Medicine, AIME 2022, Halifax, NS, Canada, June 14–17, 2022, Proceedings*. Springer, 2022, pp. 133–143.

[104] Kevin Lin et al. "Adversarial ranking for language generation". In: *Advances in Neural Information Processing Systems*. 2017, pp. 3155–3165.

[105] Martin Heusel et al. "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium". In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. Ed. by Isabelle Guyon et al. 2017, pp. 6626–6637. URL: https://proceedings.neurips.cc/paper/2017/hash/8a1d694707eb0fefe65871369074926d-Abstract.html.

[106] Kaleb E Smith and Anthony O Smith. "Conditional GAN for timeseries generation". In: *arXiv preprint arXiv:2006.16477* (2020).

[107] Hassan Ismail Fawaz et al. "Inceptiontime: Finding alexnet for time series classification". In: *Data Mining and Knowledge Discovery* 34.6 (2020), pp. 1936–1962.

[108] Yanping Chen et al. *The UCR Time Series Classification Archive*. July 2015.

[109] Sanjeev Arora et al. "Generalization and Equilibrium in Generative Adversarial Nets (GANs)". en. In: *Proceedings of the 34th International Conference on Machine Learning*. PMLR, July 2017, pp. 224–232. URL: https://proceedings.mlr.press/v70/arora17a.html.

[110] Sven Eggimann et al. "The Potential of Knowing More: A Review of Data-Driven Urban Water Management". In: *Environmental science & technology* 51.5 (2017), pp. 2538–2553. ISSN: 1520-5851. DOI: 10.1021/acs.est.6b04267.

[111] Guangtao Fu et al. "The role of deep learning in urban water management: A critical review". In: *Water Research* (2022), p. 118973.

[112] Liliane Manny. "Socio-technical challenges towards data-driven and integrated urban water management: A socio-technical network approach". In: *Sustainable Cities and Society* 90 (2023), p. 104360. ISSN: 22106707. DOI: 10.1016/j.scs.2022.104360.

[113] Frank Blumensaat et al. "How Urban Storm- and Wastewater Management Prepares for Emerging Opportunities and Threats: Digital Transformation, Ubiquitous Sensing, New Data Sources, and Beyond - A Horizon Scan". In: *Environmental science & technology* 53.15 (2019), pp. 8488–8498. DOI: 10.1021/acs.est.8b06481.

[114] Abdelkader Dairi et al. "Deep learning approach for sustainable WWTP operation: A case study on data-driven influent conditions monitoring". In: *Sustainable Cities and Society* 50 (2019), p. 101670. ISSN: 22106707. DOI: 10.1016/j.scs.2019.101670.

[115] Constantine E. Kontokosta and Rishee K. Jain. "Modeling the determinants of large-scale building water use: Implications for data-driven urban sustainability policy". In: *Sustainable Cities and Society* 18 (2015), pp. 44–55. ISSN: 22106707. DOI: 10.1016/j.scs.2015.05.007.

[116] Sheng Miao et al. "Applying machine learning in intelligent sewage treatment: A case study of chemical plant in sustainable cities". In: *Sustainable Cities and Society* 72 (2021), p. 103009. ISSN: 22106707. DOI: 10.1016/j.scs.2021.103009.

[117] David Weinberg et al. "A Review of Reinforcement Learning for Controlling Building Energy Systems From a Computer Science Perspective". In: *Sustainable Cities and Society* 89 (2023), p. 104351. ISSN: 22106707. DOI: 10.1016/j.scs.2022.104351.

[118] Sushil Kumar Singh, Young-Sik Jeong, and Jong Hyuk Park. "A deep learning-based IoT-oriented infrastructure for secure smart City". In: *Sustainable Cities and Society* 60 (2020), p. 102252. ISSN: 22106707. DOI: 10.1016/j.scs.2020.102252.

[119] Liliane Manny et al. "Barriers to the digital transformation of infrastructure sectors". In: *Policy sciences* 54.4 (2021), pp. 943–983. ISSN: 0032-2687. DOI: 10.1007/s11077-021-09438-y.

[120] Liliane Manny. "Socio-technical challenges towards smart urban water systems". PhD Thesis. ETH Zurich, 2022. DOI: 10.3929/ethz-b-000578852.

[121] Marcia L. Baptista and Elsa M. P. Henriques. "1D-DGAN-PHM: A 1-D denoising GAN for Prognostics and Health Management with an application to turbofan". In: *Applied Soft Computing* 131 (2022), p. 109785. ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2022.109785. URL: https://www.sciencedirect.com/science/article/pii/S1568494622008341.

[122] Vinicius L. S. Silva et al. "Data Assimilation Predictive GAN (DA-PredGAN): applied to determine the spread of COVID-19". In: *Journal of Scientific Computing* 94.1 (Jan. 2023), p. 25. ISSN: 0885-7474, 1573-7691. DOI: 10.1007/s10915-022-02078-1. URL: http://arxiv.org/abs/2105.07729.

[123] Mohammad Hassan Sharifinasab, Mohammad Emami Niri, and Milad Masroor. "Developing GAN-boosted Artificial Neural Networks to model the rate of drilling bit penetration". In: *Applied Soft Computing* 136 (2023), p. 110067. ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2023.110067. URL: https://www.sciencedirect.com/science/article/pii/S1568494623000856.

[124] Zhong-Sheng Chen et al. "A virtual sample generation approach based on a modified conditional GAN and centroidal Voronoi tessellation sampling to cope with small sample size problems: Application to soft sensing for chemical process". In: *Applied Soft Computing* 101 (2021), p. 107070. ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2020.107070. URL: https://www.sciencedirect.com/science/article/pii/S1568494620310085.

[125] Jincheng Chen et al. "Global temperature reconstruction of equipment based on the local temperature image using TRe-GAN". In: *Applied Soft Computing* 128 (2022), p. 109498. ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2022.109498. URL: https://www.sciencedirect.com/science/article/pii/S1568494622005907.

[126] TR Rosin et al. "A committee evolutionary neural network for the prediction of combined sewer overflows". In: *Water Resources Management* 35.4 (2021), pp. 1273–1289.

[127] Duo Zhang, Geir Lindholm, and Harsha Ratnaweera. "Use long short-term memory to enhance Internet of Things for combined sewer overflow monitoring". In: *Journal of Hydrology* 556 (2018), pp. 409–418. ISSN: 00221694. DOI: 10.1016/j.jhydrol.2017.11.018.

[128] Zolal Ayazpour, Amin E Bakhshipour, and Ulrich Dittmer. "Combined sewer flow prediction using hybrid wavelet artificial neural network model". In: *International Conference on Urban Drainage Modelling*. Springer, 2018, pp. 693–698.

[129] Rocco Palmitessa et al. "Soft sensing of water depth in combined sewers using LSTM neural networks with missing observations". In: *Journal of Hydro-environment Research* (2021). ISSN: 15706443. DOI: 10.1016/j.jher.2021.01.006.

[130] S. R. Mounce et al. "Predicting combined sewer overflows chamber depth using artificial neural networks with rainfall radar data". In: *Water Science and Technology* 69.6 (2014), pp. 1326–1333. ISSN: 0273-1223. DOI: 10.2166/wst.2014.024.

[131] Stefan Falkner, Aaron Klein, and Frank Hutter. "BOHB: Robust and efficient hyperparameter optimization at scale". In: *International Conference on Machine Learning*. PMLR, 2018, pp. 1437–1446.

[132] Richard Liaw et al. "Tune: A Research Platform for Distributed Model Selection and Training". In: *arXiv preprint arXiv:1807.05118* (2018).

[133] Lisha Li et al. “Hyperband: A novel bandit-based approach to hyperparameter optimization”. In: *The Journal of Machine Learning Research* 18.1 (2017), pp. 6765–6816.

[134] Kevin Jamieson and Ameet Talwalkar. “Non-stochastic best arm identification and hyperparameter optimization”. In: *Artificial intelligence and statistics*. PMLR, 2016, pp. 240–248.

[135] Alireza Koochali et al. “Quantifying quality of class-conditional generative models in time series domain”. In: *Applied Intelligence* (July 2023). ISSN: 1573-7497. DOI: 10.1007/s10489-023-04644-y. URL: https://doi.org/10.1007/s10489-023-04644-y.

[136] Augustus Odena, Christopher Olah, and Jonathon Shlens. “Conditional Image Synthesis with Auxiliary Classifier GANs”. en. In: *Proceedings of the 34th International Conference on Machine Learning*. PMLR, July 2017, pp. 2642–2651. URL: https://proceedings.mlr.press/v70/odena17a.html.

[137] Mingming Gong et al. “Twin Auxilary Classifiers GAN”. In: *Advances in Neural Information Processing Systems*. Vol. 32. Curran Associates, Inc., 2019. URL: https://papers.nips.cc/paper_files/paper/2019/hash/4ea06fbc83cdd0a06020c35d50e1e89a-Abstract.html.

[138] Minguk Kang and Jaesik Park. “ContraGAN: Contrastive Learning for Conditional Image Generation”. In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., 2020, pp. 21357–21369. URL: https://proceedings.neurips.cc/paper/2020/hash/f490c742cd8318b8ee6dca10af2a163f-Abstract.html.

[139] Peng Zhou et al. *Omni-GAN: On the Secrets of cGANs and Beyond*. Mar. 2021. DOI: 10.48550/arXiv.2011.13074. URL: http://arxiv.org/abs/2011.13074.

[140] Liang Hou et al. *Conditional GANs with Auxiliary Discriminative Classifier*. June 2022. DOI: 10.48550/arXiv.2107.10060. URL: http://arxiv.org/abs/2107.10060.

[141] Minguk Kang et al. *Rebooting ACGAN: Auxiliary Classifier GANs with Stable Training*. Nov. 2021. DOI: 10.48550/arXiv.2111.01118. URL: http://arxiv.org/abs/2111.01118.

[142] Takeru Miyato et al. *Spectral Normalization for Generative Adversarial Networks*. Feb. 2018. DOI: 10.48550/arXiv.1802.05957. URL: http://arxiv.org/abs/1802.05957.

[143] Andrew Brock, Jeff Donahue, and Karen Simonyan. *Large Scale GAN Training for High Fidelity Natural Image Synthesis*. Feb. 2019. DOI: 10.48550/arXiv.1809.11096. URL: http://arxiv.org/abs/1809.11096.

[144] Takeru Miyato and Masanori Koyama. *cGANs with Projection Discriminator*. Aug. 2018. DOI: 10.48550/arXiv.1802.05637. URL: http://arxiv.org/abs/1802.05637.

[145] Ligong Han et al. *Dual Projection Generative Adversarial Networks for Conditional Image Generation*. Nov. 2021. DOI: 10.48550/arXiv.2108.09016. URL: http://arxiv.org/abs/2108.09016.

[146] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. *A Learned Representation For Artistic Style*. Feb. 2017. DOI: 10.48550/arXiv.1610.07629. URL: http://arxiv.org/abs/1610.07629.

[147] Yann LeCun and Corinna Cortes. “MNIST handwritten digit database”. In: (2010). URL: http://yann.lecun.com/exdb/mnist/.

[148] Alex Krizhevsky. “Learning Multiple Layers of Features from Tiny Images”. In: (2009), pp. 32–33. URL: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

[149] Tim Salimans et al. “Improved Techniques for Training GANs”. In: *Advances in Neural Information Processing Systems*. Vol. 29. Curran Associates, Inc., 2016. URL: https://proceedings.neurips.cc/paper/2016/hash/8a3363abe792db2d8761d6403605aeb7-Abstract.html.

Alireza Koochali

Data Scientist

Linkedin : www.linkedin.com/in/alireza-koochali-6033715a/

Experiences

2018 - Present — **IAV GmbH, Germany**
Data Scientist

- Developed advanced predictive models using Generative AI, enhancing combustion engine efficiency and introducing probabilistic forecasting for risk-aware adjustments.
- Secured a patent for a novel forecasting method, underscoring the innovative impact of my work.
- Gained familiarity with Docker, Kubernetes, and CI/CD pipelines through collaboration with deployment teams.
- Utilized Python, PyTorch, and TensorFlow to build and optimize forecasting models.
- Demonstrated strong problem-solving skills for data science challenges.
- Communicated complex technical concepts to non-technical stakeholders effectively.

2015 - 2018 — **German Research Center for Artificial Intelligence (DFKI), Germany**
Research Assistant

- Managed backend development for a website using Elasticsearch, providing live search and analysis capabilities for the YFCC100M dataset.
- Played a vital role in creating publications that significantly enhanced data accessibility and research capabilities.

Education

2018 - 2024 — **University of Kaiserslautern-Landau, Germany**
Ph.D. – Computer Science
Topic: Decision-Making in the Face of Uncertainty: Harnessing GANs for Probabilistic Forecasting
Area: Time series Analysis, Probabilistic Forecasting, Generative Models

2014 - 2017 — **University of Kaiserslautern-Landau, Germany**
Master of Science – Computer Science
Topic: Multimodal Sentiment Analysis with Deep Neural Networks
Area: Natural Language Processing, Computer Vision, Sentiment Analysis

Selected Publication

2023 — Quantifying Quality of Class-Conditional Generative Models in Time Series Domain
Alireza Koochali, Maria Walch, Sankrutyayan Thota, Peter Schichtel, Andreas Dengel, and Sheraz Ahmed. *Applied Intelligence*

2022 — Method and Device for the Probabilistic Prediction of Sensor Data
Alireza Koochali, Matthias Schultalbers, Peter Schichtel, and Sheraz Ahmed.
US patent US20220187772A1

2022 — Random Noise vs. State-of-the-Art Probabilistic Forecasting Methods: A Case Study on CRPS-Sum Discrimination Ability
Alireza Koochali, Peter Schichtel, Andreas Dengel, and Sheraz Ahmed
Applied Sciences

2022 — A Bayesian Generative Adversarial Network (GAN) to Generate Synthetic Time-Series Data, Application in Combined Sewer Flow Prediction
Amin Ebrahim Bakhshipour, **Alireza Koochali**, Ulrich Dittmer, Ali Haghighi, Sheraz Ahmad, and Andreas Dengel.
2nd International Joint Conference on Water Distribution Systems Analysis and Computing and Control in the Water Industry, Valencia, Spain

2021 If You Like It, GAN It—Probabilistic Multivariate Times Series Forecast with GAN
Alireza Koochali, Andreas Dengel, and Sheraz Ahmed.
Engineering Proceedings

2019 Probabilistic Forecasting of Sensory Data With Generative Adversarial Networks – ForGAN
Alireza Koochali, Peter Schichtel, Andreas Dengel, and Sheraz Ahmed.
IEEE Access

Technical Skills

Programming : Python, Java, C++, SQL
Machine Learning : Statistics, Deep Learning, Time series Analysis, NLP, Generative models
Libraries : Pytorch, Keras, Tensorflow, Scipy, scikit-learn, Numpy, Pandas
Project Management : AWS, Git, CI/CD, Docker, Kubernetes, AB testing
Data Visualization : Mathplotlib, plotly, Seaborn

Soft Skills

Effective Communicator : Skilled in simplifying complex scientific concepts for diverse audiences
Problem Solver & Creative Thinker : Pioneered the use of GANs in probabilistic time series forecasting
Fast Adopter : Quickly adapted techniques and approaches in response to project-specific challenges.
Leader & Team Player : Supervised and mentored multiple Master's students, leading seminars, projects, and theses that consistently produced novel results.

Language Proficiency

English : Full professional proficiency TOEFL ibt= 100
German : Limited working proficiency B1 Certificate