ISSN 1688-2784

Universidad de la República
Facultad de Ingeniería

Deep Generative Models for Time-Series Anomaly Detection

Tesis presentada a la Facultad de Ingeniería de la Universidad de la República por

Gastón García González

en cumplimiento parcial de los requerimientos para la finalización de la carrera de Doctorado en Ingeniería Eléctrica.

Dirección de Tesis

Alicia Fernández ..... Universidad de la República
Pedro Casas ..... Austrian Institute of Technology

Tribunal

Pere Barlet-Ros ..... Universidad Politécnica de Cataluña
Rafael Molina ..... Universidad de Granada
Andrés Ferragut ..... Universidad ORT
Pablo Musé ..... Universidad de la República

Dirección Académica

Alicia Fernández ..... Universidad de la República

Montevideo
martes 29 abril, 2025

Deep Generative Models for Time-Series Anomaly Detection, Gastón García González.

ISSN 1688-2784

Esta tesis fue preparada en LaTeX usando la clase iietesis (v1.2).
Contiene un total de 105 páginas.
Compilada el martes 29 abril, 2025.
`https://github.com/GastonGarciaGonzalez/Tesis-de-doctorado-GGG/tree/main/TESIS`

Acknowledgements

Esta tesis pone cierre a varios años de trabajo, del cual, si bien soy el principal responsable, hubiera sido imposible sin la ayuda de muchas personas e instituciones, así como de amigos y familia.

Primero, quiero expresar mi profundo agradecimiento a mi país por haberme brindado una educación pública, gratuita y de calidad, que es la base para que todas las personas puedan desarrollarse como profesionales, eligiendo y trazando su propio camino. Me considero un hijo de la Escuela Pública, de la Dirección General de Educación Técnico Profesional – UTU y de la Universidad de la República del Uruguay – UdelaR. Gracias a sus maestras, maestros, profesoras y profesores por su dedicación.

También quiero agradecer a las siguientes instituciones: A la Agencia Nacional de Investigación e Innovación (ANII), que, gracias a su programa de becas de doctorado, me permitió desarrollar este trabajo con libertad. A la Comisión Académica de Posgrados (CAP), que, a través de su programa de becas de finalización, me permitió darle el cierre esperado a esta tesis. A la Comisión Sectorial de Investigación Científica (CSIC), cuyo programa de movilidad me permitió participar en diferentes pasantías y congresos en el exterior. A la Asociación Uruguaya de Posgraduandos y Posgraduandas (AUPP), por el trabajo y los logros alcanzados en beneficio de todos nosotros. A la empresa Telefónica Uruguay, que planteó el problema central de este trabajo y proporcionó los datos para su desarrollo. A la empresa UTE, que me permitió extender la aplicación de los métodos desarrollados hacia otras tareas. En el ámbito internacional, quiero agradecer a la Universidad de Granada (UGR) y al Austrian Institute of Technology (AIT) por abrirme sus puertas en más de una ocasión para realizar pasantías en el exterior, tan importantes para el desarrollo de esta investigación. Por último, en lo que respecta a instituciones, quiero agradecer al Instituto de Ingeniería Eléctrica – IIE, que ha sido mi casa profesional. Gracias por permitirme desarrollarme laboralmente, siempre priorizando mi formación académica y personal.

En cuanto a las personas en particular, quiero agradecer en primer lugar a quienes han sido parte fundamental de este doctorado: mis tutores, Alicia Fernández y Pedro Casas. Gracias por preocuparse y ocuparse de que mi crecimiento como investigador alcanzara el mayor nivel posible, permitiéndome llevar adelante mis ideas, haciéndolas suyas y ayudándome a que llegaran a buen puerto. Su vocación, dedicación y profesionalismo han sido y serán siempre un ejemplo para mí. Les estaré eternamente agradecido.

Quiero continuar agradeciendo a otras personas que han tenido una vinculación directa con este trabajo, ya que formaron parte de proyectos que fueron el marco para su desarrollo y aportaron ideas y resultados fundamentales. Ellos son: Gabriel Gómez, Sergio Martínez, José Acuña, Emilio Martínez y Manuel Sánchez. Gracias por apoyar siempre mis ideas y compartir las suyas; son grandes referentes para mí. A Emilio y Manuel, en particular, por adueñarse del código y apoyarme en la realización de diversos experimentos.

También, dentro del IIE, quiero agradecer a todos los compañeros y compañeras que lo integran y que siempre me han apoyado. Para mí es un placer y un honor compartir este instituto con ustedes, un lugar donde aprendo todos los días, tanto de los más experimentados como de los más jóvenes. Un agradecimiento especial a mi compañero y amigo Martín Randall, por haber compartido desde el principio esta aventura de los posgrados. Asimismo, agradezco al personal no docente por su constante disposición y amabilidad.

Siguiendo con mis amigos, quiero agradecerles su apoyo incondicional y la paciencia ante mis ausencias. A Llagu y a Facu, por el gran equipo que formamos y que se ha mantenido a lo largo de los años. A Santi, un amigo que me dio la universidad y que me honró al elegirme padrino de su hija. A mis amigos de Mercedes, Enzo y Marcos, gracias por estar siempre.

Agradezco de corazón a mi familia, mis primeros formadores, que aún hoy me siguen enseñando: mis abuelos Ramón y Aurora, Miguel y Nely, Miguel y Alicia. A mis tías y tíos, en especial a mi tío Sergio y mi tía Esther. A mis primos y primas. A todos ellos, junto a mis padres y mi hermana, les debo una infancia muy feliz.

Por supuesto, a mis padres, Rosario y Darío, que me enseñaron que con trabajo y honestidad se puede llegar lejos, y porque siempre me apoyaron en todos mis emprendimientos.

Finalmente, quiero agradecer a mi gran amor, mi compañera de vida, María Eugenia. Tu amor y apoyo incondicional fueron grandes motores para finalizar este trabajo. En los momentos de flaqueza, nunca me dejaste caer, y en los de alegría, los celebraste como nadie. ¡Gracias!

A mi gran amor María Eugenia
y nuestro gran amor Catalina. A
mis padres y mis abuelos.

Esta página ha sido intencionalmente dejada en blanco.

Abstract

Time series analysis has become a prominent area of study driven by the explosive growth of data generation a trend that continues to accelerate. Real time anomaly detection in time series is a crucial and challenging problem. Behind an anomaly may lie an ongoing system attack, a potential failure that could escalate, or even fraudulent activities. Anomalies are inherently rare, isolated events that are atypical and often unpredictable. They often lack consistent patterns and may evolve over time, further complicating their identification. Additionally,monitoring systems typically handle numerous time series, each with its own unique behavior. In some cases, certain time series may exhibit causal relationships with others, which could contain important information to take into account.

In this thesis, we present a novel and versatile approach for modeling the normal behavior of multivariate and univariate time-series using generative deep learning models. At its core, our methodology leverages Variational Autoencoders (VAEs) to construct robust representations of typical patterns in data, addressing critical challenges in anomaly detection. These challenges include handling limited or incomplete information about anomalies and capturing causal and temporal dependencies across diverse time-series.

A central contribution of this work is the development of the Dilated Convolutional Variational Autoencoder (*DC-VAE*), a lightweight and scalable generative model tailored to capture the distribution of normal behavior within the variables of a system. *DC-VAE* operates effectively in two configurations: a multivariate approach that models all variables of a system as a single multivariate time-series and a global approach that treats individual time-series of the same system independently within one model. By integrating dilated convolutions, *DC-VAE* efficiently models long temporal patterns without compromising training or inference time, maintaining its lightweight design.

This method, tested on the real TELCO dataset, demonstrates superior performance over more time-series than methods that require training or fixing specific models for each individual time-series. It also outperforms other multivariate deep learning methods on datasets that are popular in the community.

To enhance adaptability and extend the utility of *DC-VAE*, we introduce *GenDeX*, a continual learning mechanism that addresses catastrophic forgetting. This mechanism enables the *DC-VAE* model to retain knowledge of previously learned series while seamlessly incorporating new ones, ensuring stable performance in both reconstruction and anomaly detection tasks. *GenDeX* proves effective not only for handling domain changes (such as adding or dropping time-series from the

model) but also for dealing with more common challenges in time-series problems, such as concept drift.

Building upon these foundations, we propose the Foundation Auto-Encoder (*FAE*), a pre-trained global model developed on the UCR'21 dataset, which encompasses a diverse range of time-series from multiple domains. *FAE* demonstrates exceptional zero-shot learning capabilities, achieving competitive anomaly detection performance even without prior exposure to specific series. When applied to the TELCO dataset, *FAE* not only maintains strong reconstruction quality but also highlights its foundational properties, enabling generalization across datasets and tasks.

Different experiments validate the effectiveness of our approach. *DC-VAE* achieves good performance in anomaly detection, while *GenDeX* ensures stability and knowledge retention in dynamic environments. *FAE* showcases the potential of foundation models for time-series analysis, offering a scalable and interpretable solution for monitoring, anomaly detection, and continual learning. These advancements underscore the versatility and practicality of deep generative models in real-world applications.

For the sake of reproducibility and as an additional contribution, we make the TELCO dataset publicly available to the community and openly release the code implementing *DC-VAE*, *GenDeX*, and *FAE*.

Resumen

El análisis de series temporales se ha convertido en un área de estudio relevante, impulsada por un gran crecimiento de la generación de datos, una tendencia que continúa acelerándose. La detección de anomalías en tiempo real en series temporales es un problema crucial y desafiante. Detrás de una anomalía puede estar un ataque continuo al sistema, un fallo potencial que podría escalar o incluso actividades fraudulentas. Las anomalías son eventos aislados, inherentemente raros, atípicos y a menudo impredecibles. Con frecuencia carecen de patrones consistentes y pueden evolucionar con el tiempo, lo que complica aún más su identificación. Además, los sistemas de monitoreo generalmente manejan numerosas series temporales, cada una con su propio comportamiento único. En algunos casos, ciertas series temporales pueden presentar relaciones causales con otras, que podrían contener información importante a tener en cuenta.

En esta tesis, presentamos un enfoque novedoso y versátil para modelar el comportamiento normal de series temporales univariadas y multivariadas utilizando modelos generativos basados en aprendizaje profundo. Como núcleo, nuestra metodología aprovecha las propiedades de los Variational Autoencoders (VAEs) para construir representaciones robustas de los patrones típicos en los datos, de manera de abordar los desafíos críticos que se presentan en la detección de anomalías. Estos desafíos incluyen manejar información limitada o incompleta sobre anomalías y capturar dependencias causales y temporales a través de diversas series temporales.

Una contribución central de este trabajo es el desarrollo del Dilated Convolutional Variational Autoencoder (*DC-VAE*), un modelo generativo liviano y escalable diseñado para capturar la distribución del comportamiento normal dentro de las variables de un sistema. *DC-VAE* opera de manera efectiva en dos configuraciones: un enfoque multivariado que modela todas las variables de un sistema como una única serie temporal multivariada y un enfoque global que trata series temporales individuales del mismo sistema de forma independiente dentro de un solo modelo. Al integrar convoluciones dilatadas, *DC-VAE* modela eficientemente patrones temporales largos sin comprometer el tiempo de entrenamiento o inferencia, manteniendo un diseño liviano.

Este método, probado en el conjunto de datos real TELCO, demuestra un desempeño superior sobre más series temporales que los métodos que requieren entrenar o fijar modelos específicos para cada serie temporal individual. También supera a otros métodos de aprendizaje profundo multivariados en conjuntos de

datos populares en la comunidad.

Para mejorar la adaptabilidad y ampliar la utilidad del *DC-VAE*, se propuso *GenDeX*, un mecanismo de aprendizaje continuo que aborda el problema de olvido catastrófico. Este mecanismo permite que el modelo *DC-VAE* retenga el conocimiento de series aprendidas previamente mientras incorpora nuevas series sin problemas, asegurando un rendimiento estable tanto en tareas de reconstrucción como de detección de anomalías. *GenDeX* demuestra ser efectivo no solo para manejar cambios de dominio (como agregar o eliminar series temporales del modelo), sino también para abordar desafíos más comunes en problemas de series temporales, como el cambio de distribución.

Sobre estas bases, proponemos el Foundation Auto-Encoder (*FAE*), un modelo global pre entrenado en el conjunto de datos UCR'21, que abarca una amplia gama de series temporales de múltiples dominios. *FAE* demuestra una capacidad excepcional de aprendizaje cero (zero-shot learning), logrando un rendimiento competitivo en la detección de anomalías incluso sin haber sido expuesto previamente a series específicas. Al aplicarse al conjunto de datos TELCO, *FAE* no solo mantiene una fuerte calidad de reconstrucción, sino que también resalta sus propiedades fundamentales, lo que permite la generalización a través de conjuntos de datos y tareas.

Diferentes experimentos validan la efectividad de nuestro enfoque. *DC-VAE* logra un buen desempeño en la detección de anomalías, mientras que con *GenDeX* se asegura estabilidad y retención de conocimiento en entornos dinámicos. *FAE* muestra el potencial de los modelos fundamentales para el análisis de series temporales, ofreciendo una solución escalable e interpretable para monitoreo, y detección de anomalías. Estos avances subrayan la versatilidad y la practicidad de los modelos generativos profundos para el análisis de series temporales en aplicaciones del mundo real.

Una contribución adicional de la tesis, es la generación y publicación de la base de datos TELCO, así como el acceso libre al código que implementa *DC-VAE*, *GenDeX* y *FAE*, lo que facilita la reproducibilidad de los experimentos por parte de la comunidad científica.

Table of Contents

Chapter 1

Introduction

In recent decades, time series analysis has emerged as a key field of research, fueled by the rapid and ever-increasing growth of data generation. A time series is defined as a sequence of values, each associated with a timestamp that determines its position within the series. These values are obtained through systematic measurements, enabling the analysis of how certain metrics evolve over time.

Examples of time series span a wide range of domains. In socio-economic contexts, they include variables such as annual inflation rates or currency exchange rates. In climate science, examples include trends in temperature or atmospheric pressure. Industrial applications are equally diverse, with time series derived from sensor readings in *Internet of Things* (IoT) systems monitoring machinery health, variations in power grid voltage, or metrics tracking production line efficiency. Similarly, in the medical field, time series often represent patient vital signs, such as heart rate or blood pressure, as well as biochemical markers like glucose levels or hormone concentrations measured over time.

The primary goal in analyzing time series is to uncover patterns within the data to understand past phenomena or predict future behaviors. Among the wide array of tasks in this domain, time series forecasting is particularly valuable to industry, helping businesses anticipate future trends and make informed decisions. Figure 1.1[1] highlights the ranking of company queries to Google Cloud solutions in 2022, where forecasting solutions take the top position, followed closely by anomaly detection solutions.

In this work, we center our focus on the latter—anomaly detection in time series—exploring innovative approaches and methodologies to enhance performance in identifying and analyzing deviations from expected behavior.

Anomaly detection is a crucial and challenging problem. Behind an anomaly may lie an ongoing system attack, a potential failure that could escalate, or even fraudulent activities. Anomalies are inherently rare, isolated events that are atypical and often unpredictable. They frequently lack consistent patterns and may evolve over time, further complicating their identification.

[1]Talk by Nicolás Loeff presenting [1] on 2022-07-22 at Facultad de Ingeniería, Universidad de la República, Montevideo.

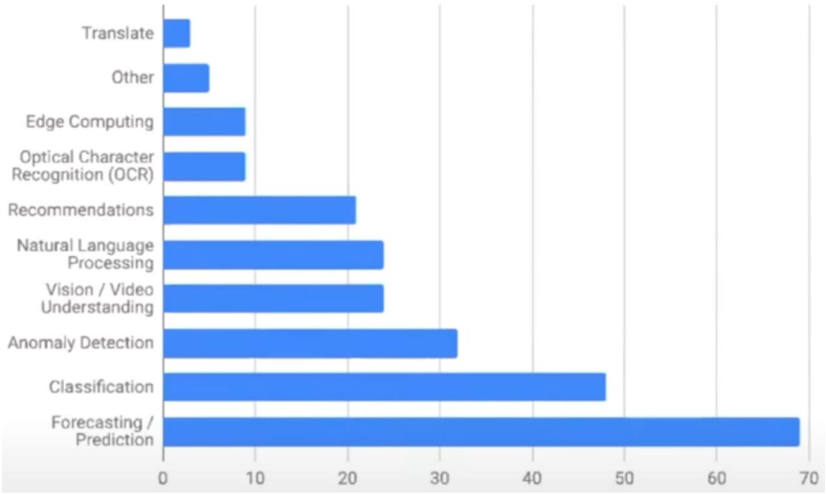


Figure 1.1: Ranking of company queries to Google Cloud solutions in 2022. Image extracted from a talk by Nicolás Loeff, presented on 2022-07-22 at Facultad de Ingeniería, Universidad de la República, Montevideo [1].

In fields such as image, text, or audio processing, deep learning solutions often require substantial computational resources and memory but consistently outperform simpler methods. However, in the domain of time-series analysis, the performance gap between deep learning and traditional methods is less pronounced. Despite this, the interest in anomaly detection for time-series continues to grow compared to other domains, as shown in Figure 1.2. Nevertheless, we aim to demonstrate that neural network-based solutions for anomaly detection offer unique advantages and hold significant potential for future advancements.

This work also addresses additional challenges related to deep learning for time-series analysis. These methods often incur substantial computational and ti-

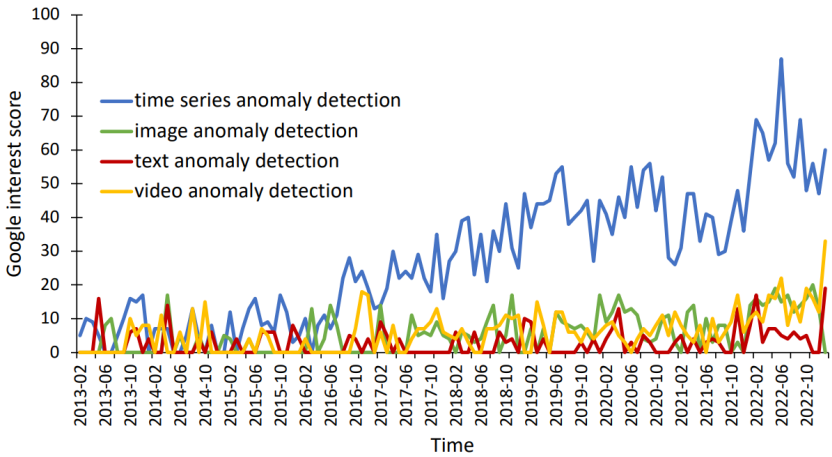


Figure 1.2: The graphic shows the evolution of user interest on Google regarding anomaly detection in time-series, images, text, and video. Image extracted from [2].

me costs to achieve robust detection performance. Consequently, when the normal behavior of time-series data shifts (Concept Drift) or when new series need to be monitored (Domain Change), the method must adapt efficiently without compromising its performance on unchanged data. Importantly, this adaptation process should be less resource-intensive than retraining the model from scratch. To address these challenges, this work explores the application of Continuous Learning in anomaly detection.

Furthermore, with the rise of foundational models—initially developed for text and later extended to time-series data—we investigate the feasibility of creating a model that, once trained on diverse time-series datasets, can be applied to previously unseen series. We also evaluate the trade-offs between the adaptability of such models and their accuracy in anomaly detection, offering insights into their practical applications and limitations.

1.1. Motivation

Anomaly detection is a challenging problem, primarily because we often lack sufficient information about anomalies, making feature extraction difficult. Additionally, monitoring systems typically handle numerous time series, each with its own unique behavior. In some cases, certain time series may exhibit causal relationships with others, which could contain important information to take into account.

One of the most common approaches to anomaly detection is to create a baseline model of normal behavior—the most frequent patterns in the data—and then, during inference, compare each new value in the series to this model. Based on the deviation from the baseline, the system can determine if a value is anomalous or within the normal range.

In another way, deep generative models have shown impressive performance in learning complex distributions for images, audio, and text, producing highly realistic synthetic samples during inference. So, why not use these models for time series as well, establishing them as the baseline for normal behavior? Additionally, instead of developing a separate model for each time series, we could create a single, unified model for all series in the system. Such a model would not only learn temporal correlations but could also capture relationships across the different series, further enhancing its ability to detect anomalies.

Of course, we are not the first to explore this idea; many studies have addressed it. However, in many cases, these works focus solely on anomaly detection performance without evaluating the quality of the normal behavior representation, or they test only on synthetic data or benchmarks with trivial anomalies. Additionally, some solutions present complex architectures, where the motivation for using certain components is unclear.

For all these reasons, we chose to use a deep generative model to learn the normal behavior across many or all time series in the system. Specifically, we employed Variational Autoencoders (VAEs), a generative model that can be trained in an unsupervised or self-supervised manner without convergence issues. We prioritized

designing a model architecture that is fast, lightweight, and easy to understand. As you will be see, this method was tested not only for anomaly detection but also for other tasks, such as continual learning to address catastrophic forgetting and as a foundational model, as we discuss further on.

1.2. Main Contributions of the Thesis

The main contributions of this thesis are the in-depth analysis of generative models, such as VAEs, for the implementation of versatile and accurate time-series anomaly detectors. Throughout this work, we have prioritized practical considerations for implementation, aiming for a lightweight and fast-to-train architecture that does not require extensive hyperparameter tuning to achieve high accuracy, while also providing a simple way to set the operational threshold. Another major concern throughout this study was the model's ability to adapt seamlessly to new scenarios, including distribution shifts in the data, changes in domain, and generalization to unseen conditions.

To address these challenges, we first proposed *DC-VAE*, a method for anomaly detection in both multivariate and univariate time-series. *DC-VAE* is an easy-to-implement detector that demonstrated accurate performance across different types of data. To enhance adaptability to changes in the data and leverage the generative capabilities of our models, we introduced *GenDeX*, a continual learning extension of *DC-VAE* capable of incorporating new information without losing performance on previously learned data. In the context of the growing popularity of foundation models, we demonstrated that *DC-VAE*'s ability to capture diverse behaviors from different time-series sources opens the possibility of developing an anomaly detection method capable of maintaining strong performance even on previously unseen time-series data. We refer to this method as *FAE*.

During the course of this thesis, several research projects were undertaken, where the results contributed significantly to achieving their objectives. Initially, with the project titled *Detección de anomalías en sistemas de telecomunicaciones mediante métodos de aprendizaje continuo*, funded by the Agencia Nacional de Investigación e Inovación (ANII), which was carried out between 2020 and 2022. Later, with the project Generalización y adaptación de dominio en la detección de anomalías en series temporales, funded by the Comisión Sectorial de Investigación Científica (CSIC), during the 2022–2024 period. Throughout both projects, the results also contributed to the final stages of an agreement with the company Telefónica for anomaly detection in time series. This agreement had been developed in different phases since 2016 and was the catalyst for my involvement in this field, as well as the framework for my master's thesis during the 2019–2020 period. Currently, a collaboration agreement is underway with the state-owned energy company UTE for the analysis of time series. The agreement consists of two distinct problems: the first involves anomaly detection in transmission stations, and the second focuses on the use of generative models for generating synthetic time series for simulation in planning. Results from this thesis work have been applied to both problems.

As a contribution, the TELCO dataset is made available, containing seven months of data provided by the owning company, Telefónica. These data include labels created by the company's engineers, and their publication has been appreciated by other experts in the field of anomaly detection. Additionally, all the code used in this thesis is publicly accessible.

This openness has enabled other researchers to test the proposed *DC-VAE* model in their experiments. Notably, an analysis conducted for the European Space Agency [3] compared the performance of our method with another proposed by the National Aeronautics and Space Administration (NASA) [4]. Among other highlights, they specifically noted the ease of determining an operational detection point, the fully convolutional architecture, the availability of public code, and its ease of use. According to their analysis, these features make our code more elegant and better aligned with their use cases compared to the other model.

1.2.1. Available

- TELCO Dataset
  - Our web
  - IEEE - Data Port
- Code
  - *DC-VAE*
  - global *DC-VAE*, and *GenDeX* experiments
  - global *DC-VAE*, and *FAE* experiments

1.2.2. Peer-Reviewed Publications

- G. García González, P. Casas, E. Martínez, A. Fernández (2024). Towards Foundation Auto-Encoders for Time-Series Anomaly Detection. In 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 10th SIGKDD International Workshop on Mining and Learning from Time Series (MILETS) — From Classical Methods to LLMs, Barcelona, Spain.
- González, G. G., Casas, P., Martínez, E., & Fernández, A. (2024, July). On the Quest for Foundation Generative-AI Models for Anomaly Detection in Time-Series Data. In 2024 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW) (pp. 252-260). IEEE.
- González, G. G., Casas, P., Martínez, E., & Fernández, A. (2024, May). Timeless Foundations: Exploring DC-VAEs as Foundation Models for Time Series Analysis. In 2024 8th Network Traffic Measurement and Analysis Conference (TMA) (pp. 1-4). IEEE.

- González, G. G., Tagliafico, S. M., Fernández, A., Gómez, G., & Casas, P. (2023). One Model to Find Them All Deep Learning for Multivariate Time-Series Anomaly Detection in Mobile Network Data. IEEE Transactions on Network and Service Management.
- González, G. G., Casas, P., & Fernández, A. (2023, July). Fake it till you Detect it: Continual Anomaly Detection in Multivariate Time-Series using Generative AI. In 2023 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW) (pp. 558-566). IEEE.
- González, G. G., Casas, P., & Fernández, A. (2023, June). Deep Generative Replay for Multivariate Time-Series Monitoring with Variational Autoencoders. In 2023 7th Network Traffic Measurement and Analysis Conference (TMA) (pp. 1-4). IEEE.
- González, G. G., Casas, P., Fernández, A., & Gómez, G. (2022, October). Steps towards continual learning in multivariate time-series anomaly detection using variational autoencoders. In Proceedings of the 22nd ACM Internet Measurement Conference (pp. 774-775).
- González, G. G., Tagliafico, S. M., Iie-Fing, A. F., Gómez, G., Acuña, J., & Casas, P. (2022, June). Dc-vae, fine-grained anomaly detection in multivariate time-series with dilated convolutions and variational auto encoders. In 2022 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW) (pp. 287-293). IEEE.
- García González, G., Martínez Tagliafico, S., Fernández, A., Gómez, G., Acuña, J., Mariño, C., & Casas, P. (2022). Mining multivariate time-series for anomaly detection in mobile networks on the usage of variational auto encoders and dilated convolutions. In 8th SIGKDD International Workshop on Mining and Learning from Time Series–Deep Forecasting: Models, Interpretability, and Applications, Washington, DC, USA, aug. 15 2022, pp. 1-7.. ACM.
- García González, G., Casas, P., Fernández, A., & Gómez, G. (2021). On the usage of generative models for network anomaly detection in multivariate time-series. ACM SIGMETRICS Performance Evaluation Review, 48(4), 49-52.

1.3. Thesis Structure

- Chapter 2: The proposed detector in this work, *DC-VAE*, is introduced. Its composition, structure, and functioning are explained. Advantages and disadvantages compared to other detectors are analyzed.
- Chapter 3: This chapter focuses on *GenDex*, the application of Generative Replay to the *DC-VAE* method. It explores how this Continual Learning

technique can be applied to address adaptability issues when faced with changes in data distribution or domain shifts.

- Chapter 4: This chapter is dedicated to *FAE*. It explores the use of our model as a Foundation Model, demonstrating its ability to achieve good performance on unseen data.
- Chapter 5: In this final chapter, we summarize the key findings of the thesis and reflect on the contributions made throughout the research. Additionally, we discuss potential avenues for future work.

Esta página ha sido intencionalmente dejada en blanco.

Chapter 2

DC-VAE, anomaly detection in multivariate time-series with dilated convolutions and variational auto encoders

In this chapter we conceive a novel approach for Multivariate Time-Series (MTS) anomaly detection, tackling many of the aforementioned challenges. We introduce *DC-VAE*, a deep-learning-based, unsupervised, and multivariate approach to real-time anomaly detection in MTS, based on popular Variational Auto-Encoders (VAEs) [5]. VAEs are a generative version of classical auto-encoders, with the advantage of producing as output prediction not only an expected value but also the associated standard deviation, corresponding to the distribution the model understands (i.e., has learned) generated the corresponding input. This automatically defines a *normality region* for each independent time-series, which can then be easily exploited for detecting deviations beyond this region. Using VAEs as an underlying approach allows the user to visualize the region of normal behavior in an interpretable way, enabling fine-grained, per univariate time-series anomaly detection.

To exploit the temporal dependencies and characteristics of time-series data in a fast and efficient manner, we take a Dilated Convolutional (DC) Neural Network (NN) as the VAE's encoder and decoder architecture. DCNNs have shown excellent performance for processing sequential data in a causal manner [6], i.e., without relying on recursive architectures, which are generally less time-efficient and more difficult to train (e.g., gradient exploding/vanishing problems). Compared to normal convolutions, dilated convolutions improve time-series modeling by increasing the receptive field of the neural network, reducing computational and memory requirements, and enabling training – and detection – on longer-in-the-past temporal sequences.

The main properties and contributions of *DC-VAE* can be summarized as follows: (i) **single model for MTS analysis:** *DC-VAE* learns the behavior of

the complete MTS process within a single model parametrization, avoiding per-time-series learning and fitting, and further exploiting the richness of the multidimensional process; (ii) **real-time operation:** the model architecture is fully causal, and provides instantaneous predictions for each independent time-series at each new time-step, using a sliding window of past measurements; (iii) **efficient temporal-memory representation:** the VAE encoder/decoder architecture based on dilated convolutions permits to efficiently process temporal sequences of longer length, making detection more robust; (iv) **self-supervised baseline modeling:** by conception, auto-encoders are self-supervised models, because the model trains itself to learn the main features of the input from the very same input samples, and ground-truth labels are only needed for tighter calibration of detection thresholds – nevertheless, in the absence of ground-truth, *DC-VAE* still estimates a normal operation region, indirectly providing a detection threshold; (v) **compact deep-learning architecture:** the structure and number of layers in *DC-VAE*'s architecture is defined by a single parameter T, representing the length of the temporal sliding-window of past measurements used as input; (vi) **independent, per time-series detection:** VAEs provide an estimation of the expected value and its associated standard deviation for each independent time-series, which provides further flexibility and detail to the monitoring process; (vii) **detection results are visually interpretable:** predictions provided by *DC-VAE* define a continual and dynamically adapted normality region, independently for each time-series, making it visually easy to interpret the occurrence of an anomaly.

We apply *DC-VAE* to a MTS dataset arising from the monitoring of an operational mobile ISP, detecting anomalies of very different structural properties. Referred to as the TELCO dataset [7], this *large-scale* – about 750 thousand samples, *long time-span* – seven months' worth of measurements collected at a five-minutes scale, *multi-dimensional* – twelve different metrics (time-series), network monitoring dataset includes ground-truth labels for anomalous events at each individual time-series, manually labeled by the experts of the network operation center (NOC) managing the mobile ISP. We benchmark *DC-VAE* against a broad set of 18 different time-series anomaly detectors coming from the signal processing and machine learning domains, individually testing on each time-series – to keep the scope of the comparative analysis, 15 of these traditional models are combined into a powerful ensemble detector. In addition, we evaluate *DC-VAE* in an open, publicly available dataset commonly used in the literature – the SWaT dataset [8], and compare its performance against other MTS anomaly detectors based on deep learning generative models, which have become very popular in recent years. For the sake of reproducibility and as an additional contribution, we make the TELCO dataset publicly available to the community, and openly release the *DC-VAE*'s code (`https://github.com/GastonGarciaGonzalez/DC-VAE`).

2.1. Related Work

There are multiple surveys on general-domain anomaly detection techniques [9–11]. The diversity of data characteristics and types of anomalies results in a lack

of universal anomaly detection models. The temporal nature of a very large spectrum of data problems has led to a strong development of the particular field of time-series anomaly detection [9, 12]. It is common to find open-source libraries implementing traditional approaches from the literature—a notable example used in this study is the Python library ADTK[1]. Other libraries, such as Darts[2], TimeEval[3] and Ruptures [4], provide a variety of models, ranging from classic ones like AutoRegressive Integrated Moving Average (ARIMA) models to deep neural networks. As noted in [9], most of the methods for unsupervised anomaly detection in univariate and multivariate time-series consist of predicting an expected value based on past information and finding a decision threshold to decide whether the prediction matches the observation. The automatic and adaptive computation of detection thresholds remains an open research problem.

Modern approaches to time-series anomaly detection based on deep learning technology have flourished in recent years [13–15]. Due to their data-driven nature and achieved performance in multiple domains, generative models such as VAEs [5] and Generative Adversarial Networks (GANs) [16] have gained relevance in the anomaly detection field [17–23]. VAEs [5, 24, 25] represent a powerful and widely used class of models to learn complex data distributions. A potential limitation of VAEs is the prior assumption that latent sample representations are independent and identically distributed. While this is the most common assumption followed in the literature, there is ongoing research on the benefits of accounting for covariances between samples in time and between time-series to improve model performance [26–29]. For example, while the original work [5] assumes that the priors over the parameters and latent variables are centered isotropic Gaussians, and that the true posteriors are approximately Gaussian with roughly diagonal covariance, [28] proposes an alternative approximation that captures temporal correlations by introducing a Gaussian process prior in the latent space.

Modeling data sequences through a combination of variational inference and deep learning architectures has been vastly researched in other domains in recent years, mostly by extending VAEs to Recurrent Neural Networks (RNNs), with architectures such as STORN [30], VRNN [31], OmniAnomaly [32], and Bi-LSTM [33] among others. Convolutional layers with dilation have also been incorporated into some of these approaches [34–36], allowing to speed up the training process based on the possibilities of parallelization offered by these architectures. Transformers [37] is another popular architecture recently showing great performance in sequential data processing; previous work on anomaly detection using transformers and VAEs [38] improves training speed as compared to the state of the art, additionally outperforming standard baseline methods. In particular, the paper improves over [32], considered a reference work in the area. Transformer-based anomaly detection in MTS data is indeed a promising research direction.

Few papers on deep learning-based detectors have addressed the problem of

[1] `https://pythonrepo.com/repo/arundo-adtk-python-machine-learning`
[2] `https://unit8co.github.io/darts/`
[3] `https://timeeval.readthedocs.io/en/latest/`
[4] `https://centre-borelli.github.io/ruptures-docs/`

real-time detection. In [39], the authors consider the alert delay in detecting so-called *range-anomalies* – i.e., contiguous anomaly segments, and evaluate their models based both on $F1$ scores and on average alert delay. The idea of range-anomaly detection is appealing in practice; in real-world applications, the operator generally does not care about point-wise anomalies, and it is acceptable for an algorithm to trigger an alert for any sample in a contiguous anomaly segment, as far as the detection delay is bounded to a certain max-delay threshold. The work in [40] generalizes the classic measures of Recall, Precision, and F1-score for range-anomalies. We consider these extended performance metrics when evaluating our method on a local telecommunications company dataset.

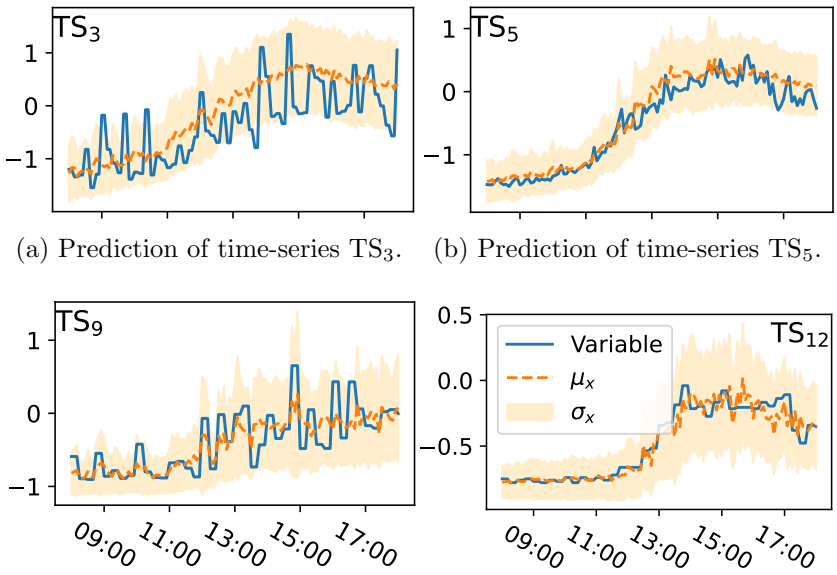
The last topic we overview relates to evaluating and benchmarking model performance through in-the-wild data time-series, using expert domain knowledge for data labeling. Most proposals in the literature have been analyzed on public datasets, such as the well-known Yahoo [41], Numenta [42, 43], NASA [44], or others, where operating conditions are unrealistic, anomalies might be trivial, and labels are poorly assigned in the labeling process [45]. Getting access to datasets labeled by domain experts in an operational environment is irreplaceable for the realistic evaluation of algorithms.

2.2. Anomaly Detection with DC-VAE

Sequential data such as time-series is generally processed through sliding windows, condensing the information of the most recent T measurements. Let us define $\boldsymbol{x}$ as a matrix in $\mathbb{R}^{M\times T}$, where M is the number of variables in the multivariate time-series (MTS) process, which defines the dimension of the problem. We also define $x(t) \in \mathbb{R}^{M\times 1}$ as an M-dimensional vector, representing the MTS at a certain time t, and $x_m(t)$, with $m \in \{1, \ldots, M\}$, as the value of the m-th time-series at time t.

As depicted in Figure 2.2, for a given input $\boldsymbol{x}$, the trained VAE model produces two different predictions, $\boldsymbol{\mu}_x$ and $\boldsymbol{\sigma}_x$ – matrices in $\mathbb{R}^{M\times T}$, corresponding to the parametrization of the Gaussian probability distribution which better represents the given input. If the VAE model was trained (mainly) with data describing the normal behavior of the monitored system, then the output for a non-anomalous input would not deviate from the mean $\boldsymbol{\mu}_x$ more than a specific integer $\boldsymbol{\alpha}$ times the standard deviation $\boldsymbol{\sigma}_x$. On the contrary, if the input presents an anomaly, the output would not belong to the region determined by the predicted mean and standard deviation. For reference, Figure 2.1 presents the main ideas behind the usage of VAEs for time-series anomaly detection, in this case portraying the results obtained in the analysis of the TELCO dataset, which is fully described in Section 2.3. For each of the displayed time-series TS_i – the TELCO dataset corresponds to twelve time-series TS_1 to TS_{12}, its real value $x_m(t)$, along with the outputs of the VAE $\mu_{x_m}(t)$ and $\sigma_{x_m}(t)$, are reported.

In the VAE model, observations $\boldsymbol{x}$ are assumed to depend on a variable $\boldsymbol{z}$ that comes from a lower-dimensional *latent* space. The objective is to maximize $ln\ P(\boldsymbol{x})$, the logarithm of the marginal distribution of the observations through the model.



(a) Prediction of time-series TS_3. (b) Prediction of time-series TS_5.

(c) Prediction of time-series TS_9. (d) Prediction of time-series TS_{12}.

Figure 2.1: Example of time-series analysis through *DC-VAE*, for the TELCO dataset. The normal-operation region is defined by $\boldsymbol{\mu_x}$ and $\boldsymbol{\sigma_x}$.

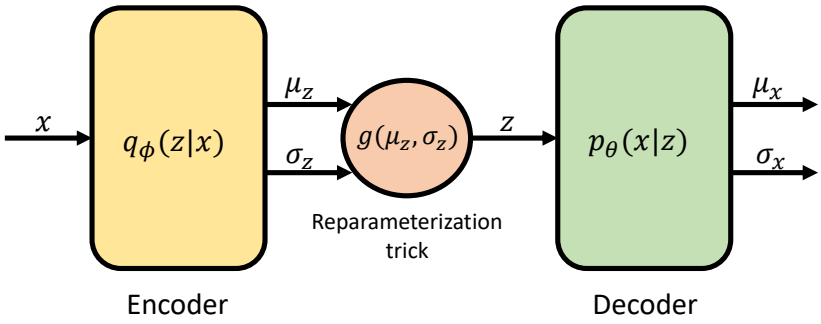


Figure 2.2: Variational autoencoder and the reparameterization trick.

For DC-VAE, similar to $\boldsymbol{x}$, $\boldsymbol{z}$ will also be a sequence of length T, but with a smaller number of dimensions $J < M$, $\boldsymbol{z} \in \mathbb{R}^{J \times T}$. In formal terms, given an input sample $\boldsymbol{x}$ characterized by an unknown probability density $P(\boldsymbol{x})$, the objective is to model or approximate the data's true density P using a parametrized distribution p_θ with parameters θ. Let $\boldsymbol{z}$ be a random vector jointly-distributed with $\boldsymbol{x}$, representing the latent encoding of $\boldsymbol{x}$. We can express $p_\theta(\boldsymbol{x})$ as:

$$p_\theta(\boldsymbol{x}) \quad = \quad \int_{\boldsymbol{z}} p_\theta(\boldsymbol{x}, \boldsymbol{z})\, d\boldsymbol{z}, \tag{2.1}$$

where $p_\theta(\boldsymbol{x}, \boldsymbol{z})$ represents the joint distribution under p_θ of the observable data $\boldsymbol{x}$ and its latent representation or encoding $\boldsymbol{z}$. According to the chain rule (probability), the equation can be rewritten as:

$$p_\theta(\boldsymbol{x}) = \int_{\boldsymbol{z}} p_\theta(\boldsymbol{x}|\boldsymbol{z}) p_\theta(\boldsymbol{z})\, d\boldsymbol{z}. \tag{2.2}$$

In the vanilla VAE, $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ is considered a Gaussian distribution, and therefore, $p_\theta(\boldsymbol{x})$ is a mixture of Gaussian distributions. The computation of $p_\theta(\boldsymbol{x})$ is very expensive and, in most cases, even intractable. To speed up training and make it feasible, it is necessary to introduce a further function to approximate the posterior distribution $p_\theta(\boldsymbol{z}|\boldsymbol{x})$, in the form of $q_\phi(\boldsymbol{z}|\boldsymbol{x}) \approx p_\theta(\boldsymbol{z}|\boldsymbol{x})$. In this way, the overall problem can be easily translated into the autoencoder domain, in which the conditional likelihood distribution $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ is performed by the *probabilistic* decoder. In contrast, the approximated posterior distribution $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ is computed by the *probabilistic* encoder, cf. Figure 2.2.

As in every deep-learning problem, it is necessary to define a differentiable loss function to update the network weights through backpropagation. In VAEs, the idea is to jointly optimize the generative model parameters θ to reduce the reconstruction error between the input and the output of the network and the parameters ϕ of the approximated posterior distribution to have $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ as close as possible to the real posterior $p_\theta(\boldsymbol{z}|\boldsymbol{x})$. The Evidence Lower Bound (ELBO) loss function is generally considered for this task. In the case of VAEs, the ELBO loss function $L_{\theta,\phi}$ can be written as follows:

$$\begin{aligned} L_{\theta,\phi} &= -\log(p_\theta(\boldsymbol{x})) + D_{KL}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x}) \parallel p_\theta(\boldsymbol{z}|\boldsymbol{x})\right) \\ &= -\mathbb{E}_{\boldsymbol{z}\sim q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})\right] + D_{KL}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x}) \parallel p_\theta(\boldsymbol{z})\right), \end{aligned} \tag{2.3}$$

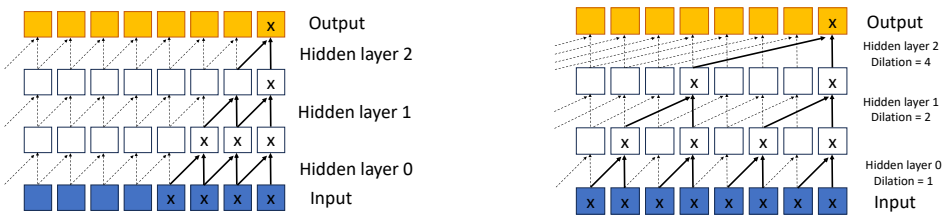
where D_{KL} is the Kullback-Leibler divergence, which here basically measures the information loss when using q to approximate p. To train the autoencoder and make the application of backpropagation feasible, a so-called *reparameterization trick* is generally introduced. The main assumption on the latent space is that it can be considered as a set of multivariate Gaussian distributions, and therefore, $\boldsymbol{z} \sim q_\phi(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{\mu_z}, \boldsymbol{\sigma_z^2})$. Given a random matrix $\boldsymbol{\varepsilon} \sim \mathcal{N}(0, \boldsymbol{I})$ and $\odot$ defined as the element-wise product, the reparameterization trick permits to explicitly define $\boldsymbol{z} = g(\boldsymbol{\mu_z}, \boldsymbol{\sigma_z}) = \boldsymbol{\mu_z} + \boldsymbol{\sigma_z} \odot \boldsymbol{\varepsilon}$. Thanks to this transformation, the variational autoencoder is trainable. The probabilistic encoder has to learn how to map a compressed representation of the input into the two latent vectors $\boldsymbol{\mu_z}$ and $\boldsymbol{\sigma_z}$. At the same time, the stochasticity remains excluded from the updating process and is injected in the latent space as an external input through $\boldsymbol{\varepsilon}$. Under the Gaussian assumption, the ELBO loss function $L_{\theta,\phi}$ can be explicitly re-written as:

$$
\begin{aligned}
L_{\theta,\phi} \;=\; & \frac{1}{2 \times T \times N} \sum_{n=1}^{N} \sum_{t=1}^{T} \Bigg[\\
& \sum_{m=1}^{M} \left(\frac{\left(x_m(t)^{(n)} - \mu_{x_m}(t)^{(n)}\right)^2}{(\sigma^2_{x_m}(t)^{(n)})} + \log(\sigma^2_{x_m}(t)^{(n)}) \right) \\
- \; & \sum_{j=1}^{J} \left(1 + \log(\sigma^2_{z_j}(t)^{(n)}) - (\mu^2_{z_j}(t)^{(n)}) - \sigma^2_{z_j}(t)^{(n)} \right) \Bigg]
\end{aligned}
\tag{2.4}
$$

At each iteration, the loss is calculated for a batch of size N; recall that m indicates the variable (time-series) in the space of $\boldsymbol{x}$, and j the variable in the space of $\boldsymbol{z}$, whereas t represents the specific time instant.

To exploit the temporal dimension of the input time-series, we proposed an encoder/decoder architecture based on popular CNNs, using Dilated Convolutions (DCs) [6]. DC is a technique that expands the input by inserting gaps between its consecutive samples. In simpler terms, it is the same as a normal convolution, but it involves skipping samples so as to cover a larger area of the input. Figure 2.3 explains the basic idea behind DCs. The convolutions must be causal, so that detection can be implemented in real-time. Because such architectures do not have recurrent connections, they are often much faster to train than RNNs and do not suffer from complex-to-tame gradient exploding/vanishing problems. Using DCs instead of standard convolutions has several advantages for real-time analysis: (i) they increase the so-called receptive field, meaning that longer-in-the-past information can be fed into the detection; (ii) DCs are computationally more efficient, as they provide larger coverage at the same computation cost; (iii) by using DC, the pooling steps are omitted, thus resulting in lesser memory consumption; (iv) finally, for the same temporary receptive field, the resulting network architecture is much more compact.

Figure 2.4 depicts the encoder architecture used in *DC-VAE*. The network architecture must be such that the output values depend on all previous input



(a) Normal convolution. (b) Dilated convolution.

Figure 2.3: Figure taken from the original WaveNet paper [6]. Using CNNs with causal filters requires large filters or many layers to learn from long sequences. Dilated convolutions improve time-series modeling by increasing the receptive field of the neural network, reducing computational and memory requirements, enabling training on long sequences.

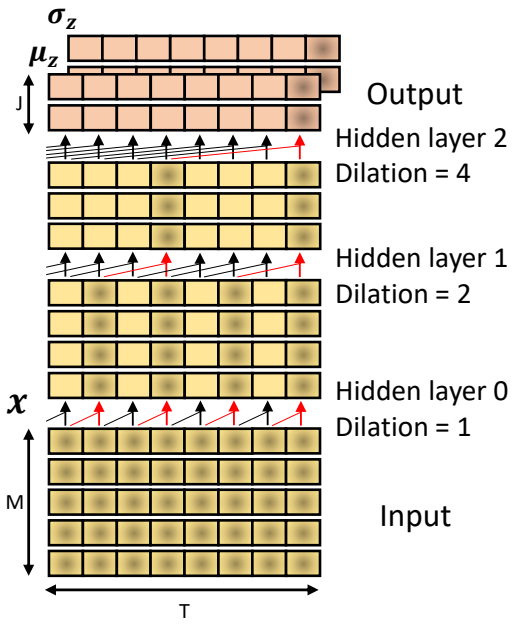


Figure 2.4: Encoder architecture using causal dilated convolutions, implemented through a stack of 1D convolutional layers.

values. The length T of the sliding window plays a key role here, as it must ensure that the output at t depends on the input at that time and at $\{t-T+1, t-T+2, \ldots, t-1\}$. The simplest way to achieve this is to use filters of length $F = 2$ and DCs with dilatation factor $d = F^h$, which grow exponentially with the layer depth $h \in [0, H-1]$, where H is the number of layers of the network. Subsequently, H is the minimum value that verifies: $T \leq 2 \cdot F^{H-1}$. In the example, the window length is $T = 8$, and the target is achieved by taking $H = 3$ layers. This direct relationship between T and the network architecture has a strong practical impact, making it easy to construct the encoder/decoder based on the desired temporal-depth of the analysis.

Note that the dilation process allows doubling T with each added layer. Consequently, a large temporal receptive field of past measurements can be achieved without further deepening the network. The encoder and decoder are symmetric in architecture, both in the number of filters and applied dilations. In the encoder model, the idea is to reduce or maintain layer output dimensions with network depth. The opposite for the decoder is increasing or maintaining the dimension until reaching the observations' dimension. In both cases, the sequence length T is always maintained.

Model training is conducted on top of normal-operation data to capture the baseline for anomaly detection. Once trained, the detection process runs continually, rolling the sliding window of length T by a unitary-time step. At each time t, the *DC-VAE* model takes as input the matrix $\boldsymbol{x} \in \mathbb{R}^{M \times T}$, constructed out of the last T samples observed in the MTS, and produces as output matrices $\boldsymbol{\mu_x}$ and $\boldsymbol{\sigma_x}$ – for notation brevity, we define $\boldsymbol{\mu} = \boldsymbol{\mu_x}$ and $\boldsymbol{\sigma} = \boldsymbol{\sigma_x}$. From these two output

matrices, the anomaly detection only considers their values at time t, corresponding to two vectors $\mu(t)$ and $\sigma(t)$. For each of the univariate time-series m, an anomaly is detected at time t if its value $x_m(t)$ falls outside the normal-operation region, defined by $\mu_m(t)$ and $\sigma_m(t)$. More precisely, an anomaly in time-series m is declared at time t if:

$$|x_m(t) - \mu_m(t)| > \alpha_m \times \sigma_m(t), \tag{2.5}$$

where $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_m, \ldots, \alpha_M)$ is a vector of M detection sensitivity threshold, where each α_m can be set independently for each time-series, allowing for fine-grained, per time-series calibration of the detection process.

Regarding the calibration of $\boldsymbol{\alpha}$, and despite being *DC-VAE* an unsupervised system, we acknowledge that these thresholds are set relying on annotated anomalies. Inevitably in any anomaly detection problem, it is necessary to set an operating point. This must be set by an expert operator in the system, who knows the behavior of the data and the cost of false detections, both positive and negative. In all sets for anomaly detection, this knowledge is in the labels provided by the experts. There are different techniques to define thresholds automatically from the data [46], but all are applicable for the detection of outliers (i.e., values far from normal behavior). In the problem we are dealing with, the interest is to detect anomalies, which are often difficult to differentiate from normal behavior, so the calibration stage inevitably must be supervised.

2.3. Dataset Descriptions

2.3.1. TELCO – A New Open Dataset Released to the Community

A recent study [45] alerts on the limitations of evaluating anomaly detection algorithms on popular time-series datasets such as Yahoo, Numenta, or NASA, among others. In particular, these datasets are noted to suffer from known flaws such as trivial anomalies, unrealistic anomaly density, mislabeled ground truth, and run-to-failure bias. For this reason, we decided to evaluate *DC-VAE* in a proprietary MTS dataset, corresponding to real measurements collected at an operation mobile ISP – note that we are publicly releasing this dataset to the community [5]. The TELCO dataset [7] corresponds to twelve different time-series, with a temporal granularity of five minutes per sample, collected and manually labeled for a period of seven months between January 1 and July 31, 2021. This temporal length is seldom available in other publicly available datasets of this nature and is highly relevant and useful to allow for long-term seasonal behavior analysis.

Each time-series corresponds to aggregated data from different sources; to keep business confidentiality, we do not specify the exact data type reflected by each time-series. The twelve time-series are typical data monitored in a mobile ISP,

[5] `https://iie.fing.edu.uy/investigacion/grupos/anomalias/en/telco-dataset-2/downloads/`

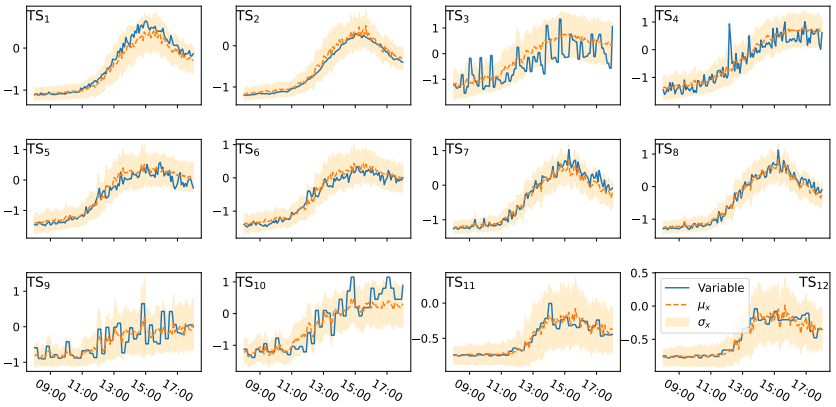


Figure 2.5: Snapshots of the TELCO MTS. For each time-series, the region of normal operation is depicted, as estimated from *DC-VAE* predictions $\boldsymbol{\mu}_x$ and $\boldsymbol{\sigma}_x$.

including the number and amount of prepaid data transfer fees, number and cost of calls, the volume of data traffic, number of SMS, and more.

Figure 2.5 depicts daily snapshots of the complete TELCO MTS. For each time-series, the region of normal operation is depicted, as estimated from *DC-VAE* predictions $\mu_x(t)$ and $\sigma_x(t)$. Different time-series expose different behaviors, e.g., some of them are noisier (TS_3), others have lower dynamic ranges (TS_{11}), and some others show a smoother evolution (TS_2). To appreciate the strong seasonality component of the time-series, Figure 2.6 depicts the TELCO MTS for a period of four days, covering weekdays and weekends.

Table 2.1 presents the main details of the dataset. Note in particular, how

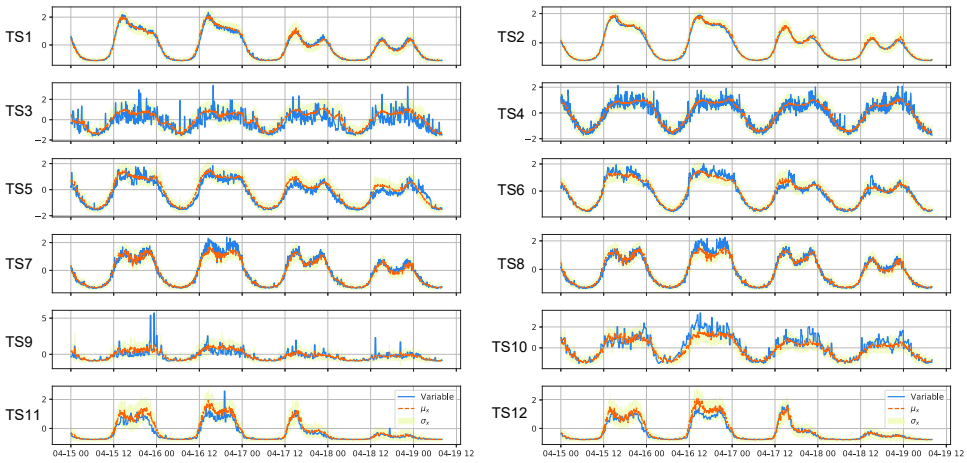


Figure 2.6: TELCO dataset time-series, for four days, along with the corresponding *DC-VAE* estimations. The temporary receptive field – i.e., length of the rolling time-window, is $T = 512$ samples, spanning about two days of past measurements.

Table 2.1: TELCO dataset. Seven-months worth of measurements was manually labeled for twelve different metrics.

Dataset	**# Samples**	**Duration**	**# Anomalous Samples**
Training	310,974	3 months	5,672 (**1.8 %**)
Validation	103,680	1 month	385 (**0.4 %**)
Testing	317,953	3 months	3,080 (**1.0 %**)
Total	732,607	7 months	9,137 (**1.2 %**)

strongly imbalanced the dataset is in terms of normal-operation and anomalous samples, which is the typical case for real network measurements in operational deployments. By definition, anomalies are rare events. We split the full dataset in three independent, time-ordered sub-sets, using measurements from January to March for model training, April for model validation, and May to July for testing purposes. For the sake of completeness, Table 2.2 reports normal-operation and anomalous samples per individual time-series, for the training, validation, and testing sub-sets. The share of anomaly samples is low and significantly different for some of the time-series, adding richness and complexity to the dataset; for example, time series TS_1, TS_4, TS_9, and TS_{10} have a total share of anomaly samples above 2 % or 3 %.

While the TELCO dataset used in this paper and released to the community has a seven-month time span, we acknowledge that the complete dataset we have collected has almost two years of duration. We have decided to work only on these seven months because it corresponds to the the data for which expert operator an-

Table 2.2: Distribution of anomaly samples in the TELCO dataset, per time-series and per training, validation, and testing sub-sets.
The share of anomaly samples is low, and significantly different for some of the time-series.

	Training			**Validation**			**Testing**			**Total**		
ID	**Norm**	**Anom**	**%**	**Norm**	**Anom**	**%**	**Norm**	**Anom**	**%**	**Norm**	**Anom**	**%**
TS_1	24,731	1,183	4.6	8,628	12	0.14	26,084	412	1.6	59,443	1,607	**2.6**
TS_2	25,713	201	0.8	8,629	11	0.13	25,995	501	1.9	60,337	713	**1.2**
TS_3	25,784	130	0.5	8,636	4	0.05	26,358	138	0.5	60,778	272	**0.4**
TS_4	24,464	1,450	5.6	8,636	4	0.05	26,317	179	0.7	59,417	1,633	**2.7**
TS_5	25,840	74	0.3	8,637	3	0.03	26,390	106	0.4	60,867	183	**0.3**
TS_6	25,850	64	0.2	8,639	1	0.01	26,390	107	0.4	60,879	172	**0.3**
TS_7	25,793	127	0.5	8,638	2	0.02	26,227	269	1.0	60,658	398	**0.7**
TS_8	25,787	127	0.5	8,640	0	–	26,229	267	1.0	60,656	394	**0.6**
TS_9	25,287	627	2.4	8,508	132	1.53	25,932	564	2.1	59,727	1,323	**2.2**
TS_{10}	24,558	1,356	5.2	8,463	177	2.05	25,995	501	1.9	59,016	2,034	**3.3**
TS_{11}	25,725	189	0.7	8,601	39	0.45	26,475	21	0.1	60,801	249	**0.4**
TS_{12}	25,770	144	0.6	8,640	0	–	26,481	15	0.1	60,891	159	**0.3**

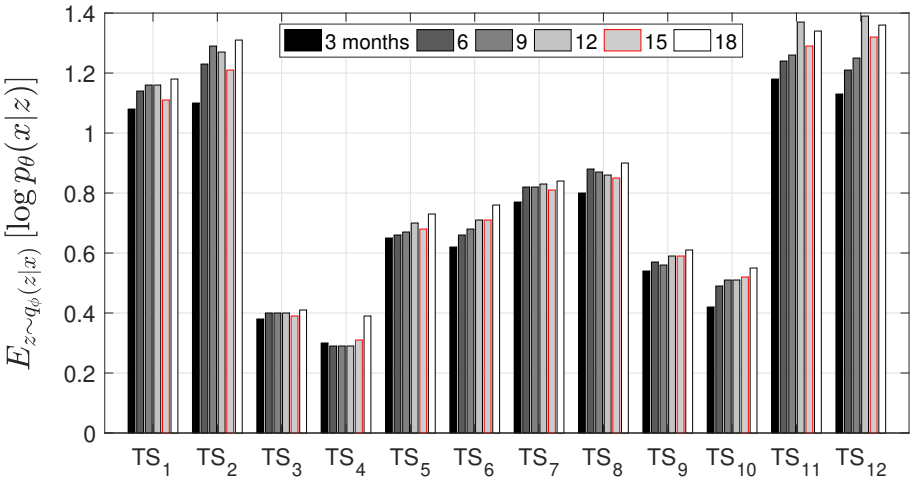


Figure 2.7: Average log-likelihood $\mathbb{E}_{\boldsymbol{z}\sim q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})\right]$ in the reconstruction of TELCO in the testing dataset, using different temporal spans (3 to 18 months) for self-supervised model training.

notated labels are available. Although *DC-VAE* trains in a self-supervised fashion, a fair comparison with supervised methods as the one we do in the evaluations requires that all methods share the same training, validation, and test sets.

Nevertheless, and for the sake of completeness, we investigated the impact on *DC-VAE*'s baseline modeling performance when training with longer time-spans, without labels. Figure 2.7 reports the average log-likelihood $\mathbb{E}_{\boldsymbol{z}\sim q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})\right]$ in the reconstruction of TELCO in the testing dataset, using different temporal spans for self-supervised model training. Interestingly, improvements are rather marginal when considering up to 18 months of training data, suggesting that manually labeling a longer time-span for TELCO might not actually provide a richer dataset.

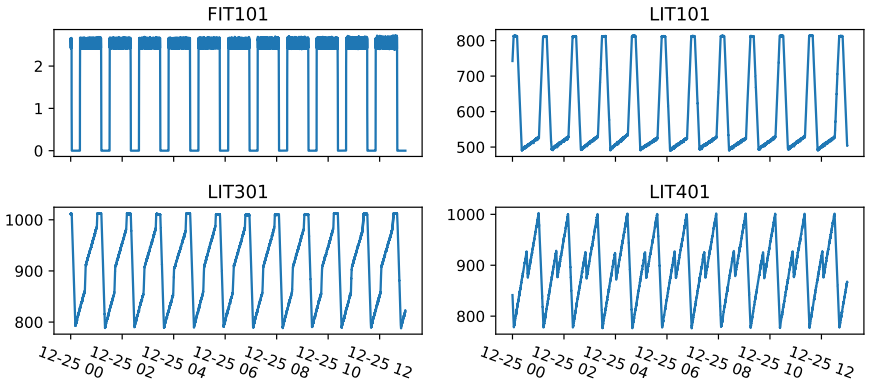


Figure 2.8: SWaT – the four time-series represent normal operation. Anomaly labels in SWaT correspond to 36 temporal ranges when attacks were executed.

2.3.2. The SWaT Open Dataset for Cybersecurity Analysis

While the core of the evaluations and benchmarking is conducted on the TELCO dataset, we also evaluate *DC-VAE* in the Secure Water Treatment (SWaT) dataset [8], an open, publicly available dataset commonly used in the literature for cybersecurity analysis. The SWaT dataset consists of 51 time-series of data collected over eleven days in 2015-2016, on a water treatment operational test-bed, which represents a small-scale version of a large modern cyber-physical system. The dataset contains two sub-sets temporally split; the first week is anomaly free and is considered as the training dataset, whereas the last four days of data contain 36 attacks of different nature and duration (from a few minutes to an hour), and is meant for testing purposes. The total number of anomaly samples accounts for about 5.8 % of the total measurements. As an example of the kind of patterns observed in the SWaT MTS, Figure 2.8 depicts four of the time-series under normal operation. Different from TELCO, which represents a real operational network and anomaly labels are provided by manual inspection on individual time-series, anomaly labels in SWaT correspond to temporal ranges in which the attacks were executed under a controlled environment.

We acknowledge that the SWaT dataset is far from representing a real cyber-physical system and is not perfect as benchmark for anomaly detection, presenting significant trivial anomalies and unrealistic anomaly density, as well as some mislabeled ground truth and missed anomalies in the data (`https://mlad.kaspersky.com/swat-testbed/`). Nevertheless, there are two main reasons for testing *DC-VAE* in SWaT: (i) firstly, despite its deficiencies, the SWaT dataset is widely used in the state of the art as benchmark for multivariate time-series anomaly detection, and this allows us showing that *DC-VAE* provides similar, or even better performance, than other similar systems in a well-known dataset; (ii) secondly, using SWaT lets us testing the modeling capabilities of *DC-VAE* in a dataset with a broader variety of variables – 51 time series in this case.

2.4. *DC-VAE* Evaluation and Benchmarking

2.4.1. DC-VAE Architecture Calibration

The first step before evaluation of *DC-VAE* is to calibrate the model. As explained in Section 2.2, the length T of the sliding window plays a major role in the architecture of *DC-VAE*. Given the usage of the dilated convolutions, T determines the number of encoder and decoder layers (cf. Figure 2.4). The dimension J of the latent space is the other relevant parameter to set; while it must be smaller than the MTS dimension M, it must also be large enough to capture the most relevant information of the MTS process. We test different values for the sequence length T to show how this affects the performance of the model. In particular, we test $T = 1, 8, 16, 32, 64, 128, 256, 512, 1024$ samples, considering the average of the mean absolute error (MAE) between x_m and μ_{x_m}, for each time-series m. Sequence length $T = 1$ corresponds to a standard VAE model with only snapshot-like inputs;

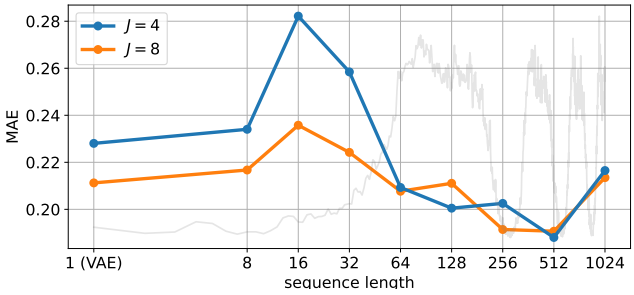
to avoid an excessively compressed model for this sequence length, we consider here an architecture with three fully-connected layers.

Besides the reconstruction MAE, we also compute the so-called explained variance or variance score $\text{Var}_{\text{score}}$, which compares the variance of the reconstruction error and the variance of the input signal:

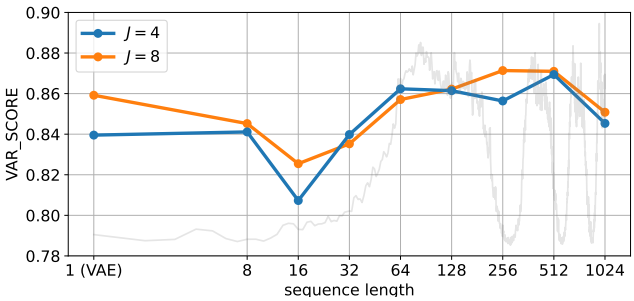
$$\text{Var}_{\text{score}}\left(\boldsymbol{x}(t), \boldsymbol{\mu}_{\boldsymbol{x}}(t)\right) \quad = \quad 1 - \frac{\text{Var}\left(\boldsymbol{x}(t) - \boldsymbol{\mu}_{\boldsymbol{x}}(t)\right)}{\text{Var}(\boldsymbol{x}(t))} \tag{2.6}$$

The value of $\text{Var}_{\text{score}}$ is between $[0, 1]$, where 1 represents the ideal case. Figure 2.9 reports the (a) MAE and (b) $\text{Var}_{\text{score}}$ for each sequence length T and corresponding model architecture, in both cases obtained as the average value across all the time-series, for the TELCO validation set. Latent space dimensions $J = 4$, and $J = 8$ are considered in the analysis. The MAE varies considerably for the proposed range, with $T = 512$ providing the smallest reconstruction error, almost identical for both latent space dimensions. Similarly, for the $\text{Var}_{\text{score}}$, $T = 512$ results in the highest score, for both latent space dimensions.

Another relevant hyperparameter is the number of filters f for each hidden convolutional layer, which together with the number of layers and the input and output dimensions define the size of the architecture in terms of the number of trainable parameters. Also, hyperparameters typical of the training stage, such as the learning rate γ and the mini-batch size m, are key to find the optimal solution.



(a) Reconstruction error.



(b) Variance score.

Figure 2.9: Calibration of *DC-VAE* in TELCO. $T = 512$ provides the smallest reconstruction error and the highest variance score.

Table 2.3: Grid of hyperparameters used in the model calibration.

Hyperparameter	**Grid**	**Best**
T	$\{8, 16, 32, 64, 128, 256, 512, 1024\}$	512
J	$\{1, 2, 4, 8\}$	4
γ	$\{10^{-3}, 10^{-4}\}$	10^{-3}
m	$\{32, 64\}$	32
f	$\{8, 16, 32\}$	16

Table 2.4: Temporal complexity for architecture optimization and model training (hardware reference: GPU Nvidia GTX 1060).

	DC-VAE	**RNN**
Hyperparameter search (hours)	**15**	37
Training best model (minutes)	**10**	15

To find the best combination of these hyperparameters, we use the Tree-structured Parzen Estimator (TPE) approach [47]. In total, 50 attempts were tested on the grid shown in table 2.3, where the hyperparameters for which the model showed the smallest validation loss are reported in the last column.

The hyperparameter search stage for a deep learning model is one of the most important and most expensive steps, since it involves training many models until the optimal values are found. Therefore, lowering the times for this stage is paramount. To evaluate the time gained by using a fully parallelizable compact architecture such as the one proposed in *DC-VAE*, as compared to traditional recurrent architectures, we created another architecture by replacing all layers with RNNs. To search for the hyperparameters, we define another grid that includes the previous one, adding the number of hidden layers: $h = \{2, 4\}$. It is worth recalling that for *DC-VAE*, defining the length of the T sequences automatically sets the number of layers, and thus this value varies between $[3, 10]$. Gated Recurrent Units (GRU) were the type of layer used in the RNNs, as they showed the highest convergence stability in terms of vanilla RNN and LSTM models.

Table 2.4 reports the comparative times taken for hyperparameter search and model training for both architectures, i.e., *DC-VAE* and the RNN-based one. The tests are performed on standard GPU hardware, using a Nvidia GTX 1060 GPU. The fully causal architecture proposed by *DC-VAE* is more compact and can be optimized and trained much faster than traditional recursive architectures. In particular, the hyperparameter search takes less than half the time, and the model training is at least 33 % faster.

2.4.2. Anomaly Detection Results in TELCO

We go back to Figure 2.6 to show *DC-VAE* in action, using a sliding-window of length $T = 512$ samples. *DC-VAE* can properly track different types of behavior in the time-series, including the strong seasonal daily component, but also the operation during weekdays and weekends, clearly visible in TS_2 and TS_{11}, among others. In this example, time-series TS_3 and TS_9 are noisier than time-series TS_5 and TS_{12}, which justifies the need for different sensitivity thresholds α_m to address the underlying nature of each monitored metric. Note in addition how different periods of time-series variability result in more or less tight normal-operation regions estimated by *DC-VAE*, as defined by $\sigma(t)$. Figure 2.10 extends the predictions of *DC-VAE* to a longer time-span, considering two weeks of measurements, for time-series TS_2 and TS_{11}. While both time-series have a strong seasonal component, with marked differences in behavior on weekdays and weekends, TS_{11} has a decreasing trend on the second week, which can be properly tracked by *DC-VAE*.

To apply *DC-VAE* for anomaly detection, we have to calibrate the sensitivity thresholds $\boldsymbol{\alpha}$, which is usually done in a supervised manner, relying on the labeled anomalies available in the training and validation datasets. This step is the only one that requires a certain level of "supervision" (in the sense of ground-truth availability), but could also be done in a self-supervised manner, by labeling anomalies through outlier detection techniques. In our specific problem, each sensitivity threshold α_m is calibrated on a per time-series basis, by maximizing the $F1$ score over the training and validation datasets, doing a grid-search of integer values from 1 to 5. In summary, we decide how many standard deviations σ_m shall be considered as tolerances for the normal-operation variability of the data.

Figure 2.11 reports some examples of real (i.e., labeled) anomalies present in the TELCO dataset, in particular for time-series TS_2, TS_4 TS_6 and TS_9, along with their corresponding identification by *DC-VAE*, where sensitivity thresholds $\boldsymbol{\alpha}$ were calibrated as mentioned before. *DC-VAE* can detect different types of anomalies present in the data, of a more transient and spiky nature in the case of TS_6 and TS_9, or on a more structural basis in the case of TS_2 and TS_4. Note also how some of the actual measurements fall significantly outside the normal-operation region – e.g. in Figure 2.11(c), but still these were not labeled as anomalous by the expert operator. Whether this is a false-positive produced by *DC-VAE*, or a non-labeled anomaly missed by the expert operator is difficult to know. It is important to

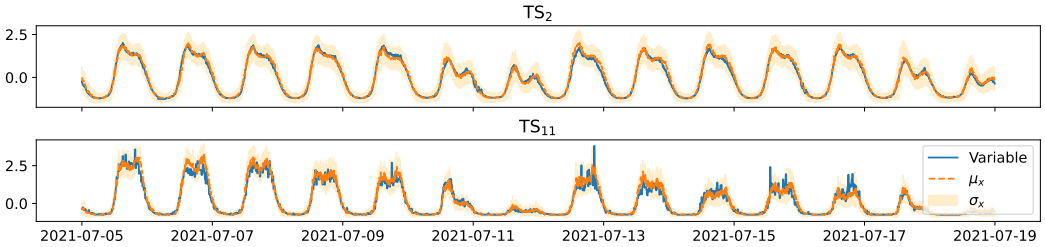
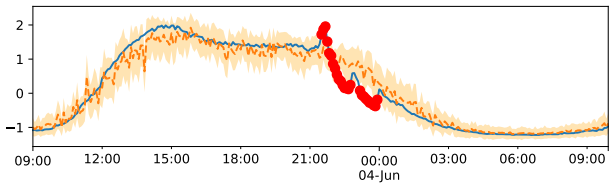
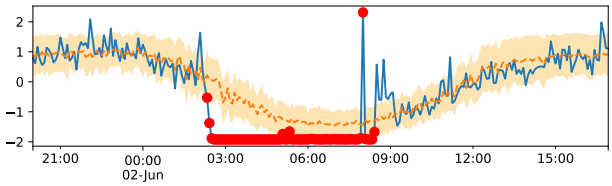


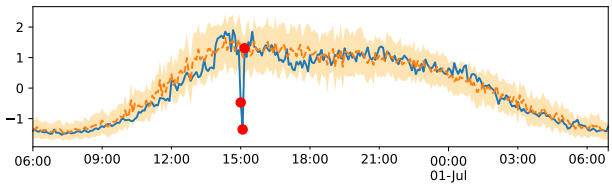
Figure 2.10: *DC-VAE* operation for time-series with stationary behavior. Weekly seasonality is identified, with variations between weekdays and weekends.



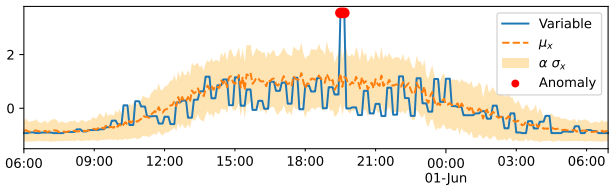
(a) Example of real anomalies in TS_2.



(b) Example of real anomalies in TS_4.



(c) Example of real anomalies in TS_6.



(d) Example of real anomalies in TS_9.

Figure 2.11: Examples of real anomalies present in the analyzed dataset, and their identification by *DC-VAE*.

note that anomalies in real, operational measurements, as labeled by the expert operator, do not always translate into clear outliers in the data; the contrary is also true, meaning that typical outliers in the data might not correspond to actual anomalies in the eyes of the expert operator. Manual data labeling by experts is prone to human error, many times due to a lack of conclusive information for the operator to take a proper decision. These observations are paramount when evaluating anomaly detectors with real, in-the-wild data.

We run a quantitative performance analysis of *DC-VAE* in the testing dataset (cf. Table 2.1), benchmarking its performance against a broad set of more traditional detectors. As performance metrics, we consider an elaborated version of the traditionally used, per-sample evaluation metrics, to consider a more natural and practical approach for real anomaly detection applications, evaluating detection performance in the form of anomaly temporal-ranges. Traditional metrics can make sense for point anomalies where a true positive corresponds to a correct

detection at the precise point in time. However, as shown for example in Figure 2.11(b), many anomalies occur in the form of multiple, consecutive point anomalies, defining an anomaly range. In such scenarios, it could be already enough to have a partial overlap between the real anomaly range and the predicted anomaly interval to consider a correct detection. Previous papers have considered these observations [39, 40, 42], defining new metrics which prioritize early or delayed detection, or focusing mainly on range anomalies. Therefore, we take the extended definitions of recall and precision as defined in [40] to generalize for ranges of anomalies, considering a correct detection if at least one of the samples between the start and the end of the actual anomaly is flagged by the model. We refer to these extended, range-based metrics as R_r, P_r, and $F1_r$, for recall, precision, and F1-score, respectively. More precisely, given a set of λ Real Anomaly ranges $RA = RA_1 \dots RA_\lambda$ and a set of δ Predicted Anomaly ranges $PA = PA_1 \dots PA_\delta$:

$$R_r(RA, PA) = \frac{\sum_{j=1}^{\lambda} R_r(RA_i, PA)}{\lambda} \tag{2.7}$$

$$P_r(RA, PA) = \frac{\sum_{j=1}^{\delta} P_r(RA, PA_i)}{\delta} \tag{2.8}$$

$$F1_r = 2 \times \frac{R_r \times P_r}{R_r + P_r} \tag{2.9}$$

In a nutshell, an intersection between an anomaly interval and the whole set of predictions is enough to set $R_r(RA_i, PA)$ to one. $P_r(RA, PA_i)$ is determined in its dual form. To consider the manual labeling uncertainty in the real anomaly location [48], we run a preprocessing on the real anomaly regions, convolving the series with a rectangular window, to obtain better-defined anomaly ranges.

Table 2.5 summarizes the different anomaly detection approaches considered in the benchmark against *DC-VAE*. Most of these approaches correspond to univariate detection methods (except S-VAE), largely studied in the signal processing domain. A broad set of 15 univariate detectors are integrated into a single ensemble detector, referred to as ENS-15. The ensemble includes regression models, change-point detectors, outliers detectors, dimensionality reduction, clustering, and more. The aggregation corresponds to a majority voting strategy, where each detector is independently calibrated in the training and validation datasets, and a voting threshold maximizing $F1$ validation scores is computed. In TELCO, ENS-15 detects an anomaly if at least four ensemble models detect it. We also consider well-established time-series detectors, such as Seasonal Exponential Smoothing (S-EXPS) and the standard Auto-Regressive Integrated Moving Average (ARIMA) model. These approaches base the detection on the prediction of μ_x and σ_x for each time instant, making them particularly interesting to compare against *DC-VAE*. To show the advantages of *DC-VAE* as compared to the usage of standard, vanilla VAEs for anomaly detection in time-series, we define the Standard-VAE (S-VAE)

Table 2.5: Set of benchmark time-series anomaly detectors used in TELCO against *DC-VAE*.

ENS-15	Local Outlier Factor (LOF)
	Isolation Forest (IF)
	Double Roll. Aggregate with Interquartile Range (DRA-IR)
	Quantile Detector (QQ)
	Interquartile Range Detector (IR)
	Generalized Extreme Studentized Deviate Test (G-ESDT)
	DRA with Single Change-Point Detection (DRA-CP)
	Level Shift Detector (LS)
	Volatility Shift Detector (VS)
	Seasonal Decomposition with Exp. Smoothing (SD-ETS)
	Time-Series Seasonality Detector (TSS)
	Autoregressive Detector (AR)
	Linear Regression Detector (LR)
	PCA Detector (PCA)
	K-means Clustering Detector (K-means)
S-EXPS	Seasonal Exponential Smoothing
ARIMA	Auto Regressive Integrated Moving Average
S-VAE	Standard vanilla VAE, equivalent to *DC-VAE* with $T = 1$

as a snapshot-input-based anomaly detection model, where the encoder/decoder architecture is based on a standard 3-layers, fully connected feed-forward neural network, and the input corresponds to the MTS at the specific time of detection – i.e., $T = 1$ in S-VAE. The comparison against S-VAE serves to demonstrate the advantages of *DC-VAE* temporal-aware architecture, through the dilated convolutions. Finally, evaluations are reported independently for each to the twelve time-series TS_m in the TELCO dataset.

Table 2.6 reports the corresponding results in the testing dataset, independently for each time-series, and as an average value. The first observation is that achieved results are in general rather poor, achieving $F1_r$ scores around 60 % for eight out of the twelve time-series, and below for the rest. This is highly in contrast with the high $F1$ scores usually reported in the literature, when dealing with simulated or flawed datasets [45]. Indeed, as we explained before, dealing with in-the-wild measurements and human-labeled, highly-imbalanced datasets is more complex than what the results in the literature usually report – real, in practice MTS anomaly detection is highly complex. Performance is significantly different for some of the time-series, which corresponds to the different nature and underlying behavior (cf. Figure 2.6) and the fraction of anomalies (cf. Table 2.2). While *DC-VAE*'s performance as compared to S-VAE is outstanding, results show that no single approach is superior to the rest in all the time-series. *DC-VAE*'s performance is similar, on average, to S-EXPS and ARIMA. Still, among those already mentioned, the main advantage of *DC-VAE* remains its multivariate operation and

Table 2.6: Anomaly detection performance benchmarking in TELCO, comparing *DC-VAE* against S-EXPS, ARIMA, S-VAE, and an ensemble of 15 traditional detectors (ENS-15). First and second highest $F1$ scores are marked in red and blue, respectively.

%	**ENS-15**			**S-EXPS**			**ARIMA**			**S-VAE**			***DC-VAE***		
TS ID	R_r	P_r	$F1_r$	R_r	P_r	$F1_r$	R_r	P_r	$F1_r$	R_r	P_r	$F1_r$	R_r	P_r	$F1_r$
TS_1	45	50	48	45	88	60	64	92	**75**	23	56	32	58	71	**64**
TS_2	37	100	54	70	96	**81**	59	95	**73**	16	92	27	74	20	67
TS_3	78	33	47	78	58	**67**	78	46	58	71	50	59	86	47	**60**
TS_4	75	59	**66**	67	41	**51**	58	38	46	63	25	36	63	21	32
TS_5	73	73	**73**	45	63	53	64	64	**64**	50	20	29	75	50	60
TS_6	88	62	**72**	63	63	63	75	50	60	14	100	25	57	83	68
TS_7	77	63	**69**	69	53	60	69	46	56	45	100	63	72	90	**80**
TS_8	67	44	53	56	36	43	56	56	**56**	57	35	43	44	80	**57**
TS_9	10	17	**12**	5	5	5	19	9	12	6	4	4	17	11	**13**
TS_{10}	8	18	11	48	44	46	48	38	42	39	81	**52**	52	59	**55**
TS_{11}	58	21	31	50	32	**39**	67	26	37	67	17	27	100	25	**40**
TS_{12}	0	0	0	100	67	**80**	100	24	**38**	0	0	0	100	11	22
mean	51	45	45	58	54	**54**	63	49	51	38	48	33	67	47	**52**
median	63	47	51	60	55	**57**	64	46	56	42	43	31	68	49	**59**

the overall MTS modeling within a single learning step.

2.4.3. Benchmarking DC-VAE in the SWaT Open Dataset

For the sake of completeness and to provide a stronger and more comprehensive benchmarking, we compare *DC-VAE* against other deep-learning-based MTS anomaly detectors in SWaT. As discussed in the related work, GAN-based MTS detectors are very popular in the literature, given their flexibility to model a complex MTS process without making any assumptions on the underlying distributions. GANs are a powerful approach to learning the underlying distributions of data samples, in a purely data-driven, model-agnostic manner. Such models can be used in the practice to construct better normal-operation baselines, improving the identification of instances that deviate from this baseline. We, therefore, compare *DC-VAE* against three GAN-based detectors proposed in recent years, including EGAN [49], MAD-GAN [21], and our previous work on GAN-based MTS anomaly detection, referred to as NET-GAN [23].

To train *DC-VAE* in SWaT, we take an architecture using $J = 16$ as the dimension of the latent space, and a sequence length $T = 128$, both parameters calibrated in the same way we did it in TELCO (cf. Figure 2.9). We train both *DC-VAE* and NET-GAN in the SWaT training dataset, using a small share of samples from the attacks for calibration. Regarding EGAN and MAD-GAN, we decided to report here the results obtained by the authors in [21], which would generally correspond to the best performance which could be achieved by these methods. Finally, we also include a standard Auto Encoder (AE) model as the

Table 2.7: Anomaly detection performance benchmarking against deep-learning generative models in SWaT.

Detector	R	P	$F1$
Auto Encoder	53	73	61
EGAN	68	41	51
NET-GAN-(G)enerator	65	98	**78**
NET-GAN-(D)iscriminator	65	29	40
MAD-GAN-P (best precision)	55	100	70
MAD-GAN-R (best recall)	100	12	22
MAD-GAN-F1 (best F1 score)	64	99	**77**
DC-VAE	67	94	**78**

simplest approach comparable to *DC-VAE*.

Table 2.7 reports the results obtained in the testing dataset in terms of recall, precision, and $F1$ scores. We fall back to the standard evaluation on point anomalies instead of range anomalies, to be consistent with the results obtained in SWaT as reported in the literature. We consider two variations of NET-GAN detectors [23], one using the generator function (NET-GAN-G), and the other one the discriminator function (NET-GAN-D). We also consider three different variations of MAD-GAN, optimized for best precision (MAD-GAN-P), recall (MAD-GAN-R), and $F1$ score (MAD-GAN-F1). *DC-VAE* results are comparable to those obtained with NET-GAN-G and MAD-GAN-F1, and significantly better than EGAN or the AE model. In addition, absolute results are also significantly better than those obtained in TELCO, helping us demonstrate that anomaly detection in real data as the one in TELCO, dealing with the error-prone process of human labeling, is much more complex than what the literature usually reports on such benchmarks. To sum-up, we can claim that *DC-VAE* realizes state-of-the-art detection performance, while again, flagging its underlying advantages.

2.4.4. *DC-VAE* for Satellite Telemetry

The code was made publicly available to encourage the community to adopt and adapt it for their own applications.

The European Space Agency Anomaly Detection Dataset for Satellite Telemetry (ESA-ADB) [3] is the result of close collaboration between European Space Agency (ESA) spacecraft operations engineers and machine learning experts. This new ESA anomaly dataset contains real, annotated telemetry data from three different ESA missions, two of which are included in ESA-ADB. The results of typical anomaly detection algorithms, evaluated through their novel hierarchical evaluation process, highlight the need for new approaches to better address the needs of operators.

Real-world satellite telemetry presents an especially challenging example of multivariate time series data, with numerous specific problems and complexities

related to its high dimensionality and volume (years of recordings from up to thousands of channels per satellite), complex network of dependencies between channels, and diverse characteristics (e.g., varying sampling frequencies across time and channels, data gaps caused by idle states and communication issues, trends linked to the degradation of spacecraft components, and concept drifts associated with different operational modes and mission phases). Additionally, the data includes diverse channel types (e.g., a wide variety of physical measurements, categorical status flags, counters, and binary telecommands) and is affected by noise and measurement errors due to the harsh space environment.

The dataset[6] includes 76 channels from Mission 1 and 100 channels from Mission 2. The number of data points exceeds 700 million for each mission, resulting in more than 7 gigabytes (GB) of compressed data in total. This is orders of magnitude larger than any other publicly available satellite telemetry dataset.

For this extensive benchmark on anomaly detection in satellite telemetry, one of the methods selected was our proposed method, *DC-VAE*[7]. According to their hierarchical evaluation of the results, Telemanom [4], a deep learning-based semi-supervised algorithm designed for satellite telemetry anomaly detection, achieved the best performance for Mission 1. It obtained the highest F-measure and alarming precision, thanks to its dynamic thresholding scheme (NDT), which merges adjacent detections. This highlights the importance of proper thresholding and postprocessing methods as part of an anomaly detection algorithm.

While Telemanom had the lowest Anomaly Detection Timing Quality Curve (ADTQC)—a novel metric designed to assess the accuracy of anomaly start-time identification from the perspective of spacecraft operations engineers—*DC-VAE* achieved the highest ADTQC, sometimes outperforming Telemanom. According to their evaluation, this suggests that more advanced thresholding or postprocessing techniques could significantly improve event-wise performance scores.

It is important to note that they did not perform model selection using validation instances. Instead, they tested only two fixed α configurations ($\alpha = 3$ and $\alpha = 5$) across all channels, rather than optimizing α for each specific channel.

Telemanom vs. *DC-VAE*

After the publication of their work, the authors contacted us to provide feedback on their experience using *DC-VAE*. They wrote the following:

"Even if DC-VAE does not give the best results in terms of the specific benchmark metrics, we see it as a promising and elegant solution that only needs some improvements in postprocessing/thresholding to outperform NASA's Telemanom and to be implemented in the real operational environment of ESA. Hence, we are building on top of DC-VAE in a few ongoing projects."

They also outlined several points highlighting the weaknesses of Telemanom and the strengths of *DC-VAE*, with some of these being identified in their work [3].

Telemanom Disadvantages:

[6] `https://zenodo.org/records/12528696`

[7] `https://github.com/kplabs-pl/ESA-ADB`

- Only supports a single output channel.
- The input (historical samples) and output (forecast) windows differ in size and location, limiting the applicability of typical autoencoder-based methods.
- The dynamic thresholding mechanism (NDT) involves complex parameters, and recent findings suggest the presence of problematic hardcoded values ("magic numbers") that hinder anomaly detection in channels with small signal values.
- LSTM layers are challenging to accelerate on space-enabled hardware.
- The latent space is not regularized, making it difficult to explore as a generative model.

***DC-VAE* Advantages**:

- Supports multiple output channels.
- The input and output windows have the same size and location, following a typical autoencoder structure.
- Thresholding is simpler because the network outputs the standard deviation for each sample, allowing thresholds to be set based on standard deviations from the mean, adding interpretability.
- Fully convolutional architecture, making it more suitable for efficient processing.
- The variational bottleneck enables the exploration of the latent space, opening new possibilities for generative tasks.
- The code is publicly available, easy to run, and easy to modify.
- The approach is considered more elegant and better aligned with ESA's operational use cases.

2.5. Temporal and Spatial Response of *DC-VAE*

The visual exploration of *DC-VAE* predictions and detections in TELCO revealed certain behaviors of the model when confronted with different temporal and/or spatial patterns which are worth studying. In particular, the impact of the sequence length T on the reaction of the model to certain phenomena is relevant. Next, we present different prototypical examples of simulated anomalies and their impact on *DC-VAE* predictions, using S-VAE and the ARIMA models for comparison, when applicable.

Impact of strong outliers

The processing of the complete MTS simultaneously has evidenced, and in particular for simpler versions of the model with shorter sequence lengths T, that coarse outliers affecting a single time-series can affect the predictions for other time-series, generating false detections. Figure 2.12 shows how a major outlier in TS_{11} strongly perturbates predictions for TS_4, especially for sequence length below 32 in this example. This effect can be partially mitigated by taking longer sequences at the input. As a lesson learned, using longer sequences improves the filtering of strong outliers from the data.

Multivariate model properties

Besides being more scalable in production, having a single model for the analysis of the complete MTS also improves detection. Figure 2.13(a) shows S-VAE model predictions for two highly correlated time-series, TS_1 and TS_2. An artificial univariate anomaly in TS_1, emulating a period where the time-series is constant (e.g., no incoming measurements), has a contained impact on the rest of the time-series predictions, as reflected in the predictions of $\boldsymbol{\mu}_x$ and $\boldsymbol{\sigma}_x$ for TS_2. As the S-VAE model has no temporal information (i.e., $T = 1$), predictions are influenced by the fact that the rest of the time-series remained unchanged. Nevertheless, in this example, the anomaly introduced in TS_1 would be clearly detected.

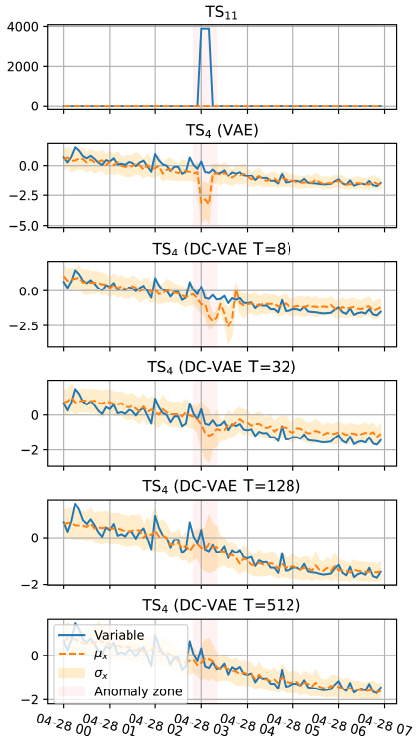
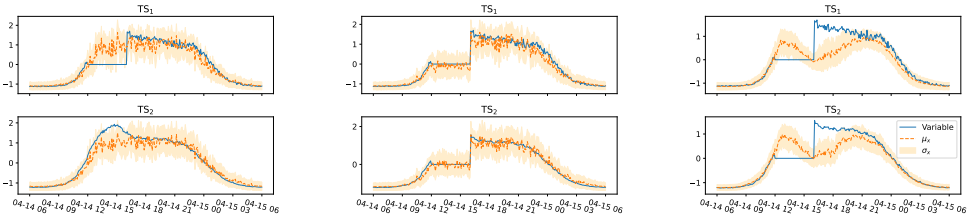


Figure 2.12: A strong outlier in TS_{11} results in poor prediction for TS_4, with sequence length $T = 32$. This effect is mitigated with longer lengths T.



(a) S-VAE under univariate anomalies. (b) S-VAE under multivariate anomalies. (c) *DC-VAE* under multivariate anomalies.

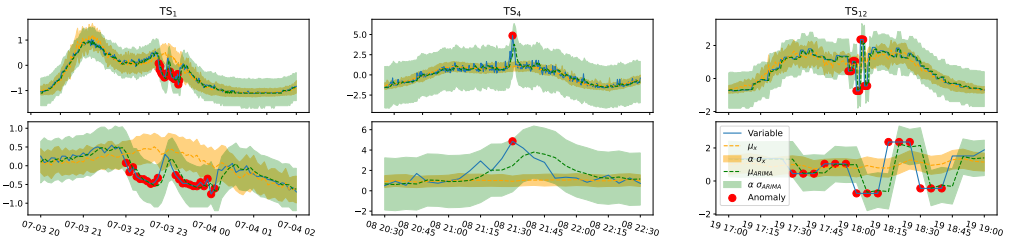
Figure 2.13: S-VAE and *DC-VAE* response to univariate and multivariate anomalies. The simultaneous modeling of the full MTS process adds regularity and stability to the detection.

Temporal model properties

We now apply the previous anomaly to all the time-series in the same period and verify how the VAE-based models exploit temporal correlations among time-series. Figure 2.13(b) shows that this time, the S-VAE model predictions perfectly follow the anomaly, making it go completely undetected. The result is totally different for *DC-VAE*; as shown in Figure 2.13(c), the predictions of a *DC-VAE* model with a sequence length of $T = 512$ tend to follow the past behavior, and take longer to track the anomaly pattern, effectively detecting it.

Similar to *DC-VAE*, the ARIMA detection model enables the visualization of the normal-operation region. However, as we show in Figure 2.14, being univariate and with a small temporal window makes ARIMA less robust for MTS anomaly detection. In the figure, model predictions are depicted in green for ARIMA and in orange for *DC-VAE*, and red dots indicate real (i.e., labeled) anomalies. Figures 2.14(a) and 2.14(b) show that the value of σ_x for the ARIMA model is constant over time, but dynamically adapts in *DC-VAE*, providing a better, more accurate normal-operation region. This is a strong advantage of *DC-VAE*, since it adapts to the noise variations that these time-series generally present.

The same happens to the estimations of μ_x. While the estimation of the signal through the ARIMA model closely follows the time-series, even in the occurrence of real anomalies – and thus the model misses detection, the estimation provided



(a) Range anomaly. (b) Point anomaly. (c) Fluctuating anomaly.

Figure 2.14: *DC-VAE* and ARIMA response to range and point anomalies. The lower image is always a close-up view of the upper one. Being univariate and with a small temporal window makes ARIMA less robust for MTS anomaly detection, and missing anomalies.

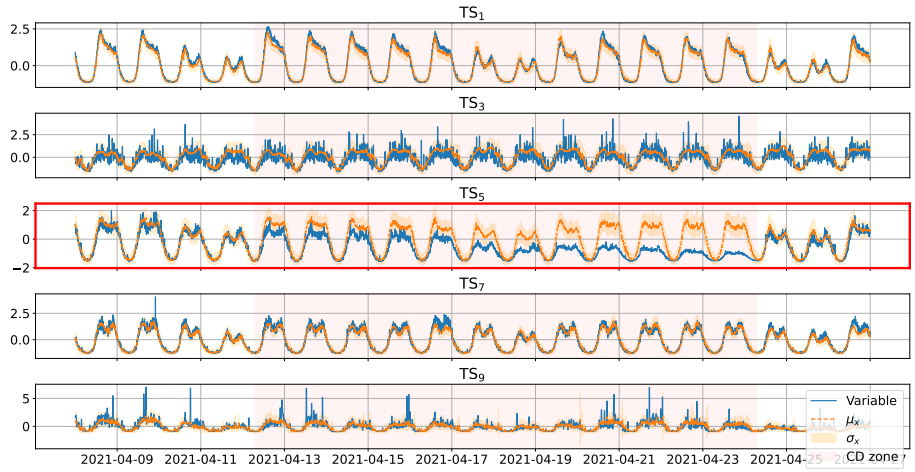


Figure 2.15: *DC-VAE* response to univariate concept-drift: a gradual linear fall of the values during the day without affecting night behavior. While the drift does not affect the predictions on the other time-series, it becomes easily detectable at the corresponding time-series.

by *DC-VAE* maintains a normal behavior in the face of the anomalies, allowing to properly detect them. The largest spatial (M) and temporal (T) ranges of *DC-VAE* add robustness to the anomaly detection process.

Concept drift response

The ability to detect Concept Drift (CD) in time-series data is a paramount property [50]. The CD can manifest itself as a shift in the mean, an increase or decrease in the variance, or both changes simultaneously, which may be imperceptible for many methods [51]. These CD changes may be related to important trends in the data, requiring proper detection. We simulate a univariate CD in one of the time-series, and check the outputs of *DC-VAE*. Figure 2.15 shows an example of CD, where a gradual change in the interval indicated as the CD zone is simulated in TS_5. The daily values of the time-series are reduced linearly, starting at 80 % (beginning of the CD zone) up to 40 % (end of the CD zone). This change does not only affect the mean value of the time-series, but also its variance. Interestingly, predictions of the *DC-VAE* follow the past behavior learned as normal, allowing the CD event to be detected.

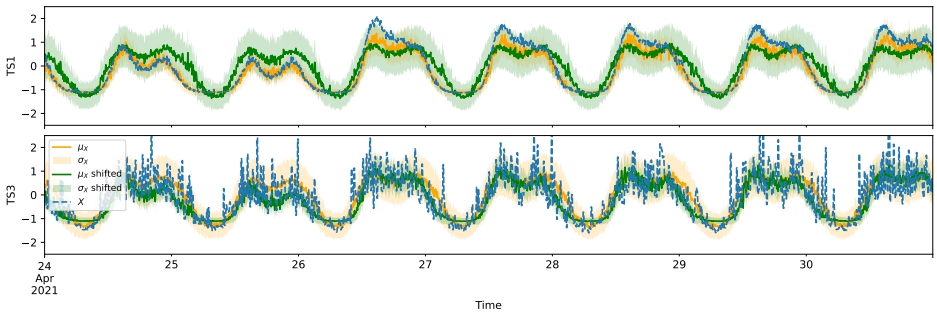
2.6. Limitations of the Multivariate Approach

The multivariate approach has both advantages and disadvantages. In this section, we present experiments that highlight its limitations in two specific situations that could arise in real contexts.

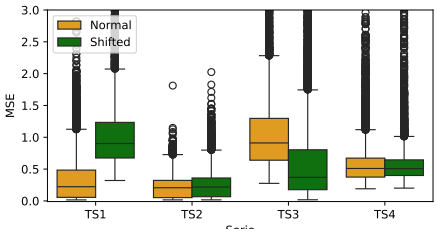
The first evaluation aims to assess what happens when the order of the variables in the input is not respected. For example, if we train a model with the TELCO dataset where the variables are ordered from top to bottom as TS_1, TS_2,

..., TS_{12}, following the numerical sequence of the series, it is essential to maintain this order during inference. However, in a real-world scenario, it is possible that an operator might change this order, which could affect the model's performance. After testing different combinations of variable order changes, we observed that the model tends to return predictions similar to those obtained when the original order used during training is preserved, making it difficult to disrupt the model's learned patterns. As an example, we swapped the variables TS_1 and TS_3 and compared the reconstruction output with the result obtained when the original order was maintained. In Figure 2.16(a), we present both reconstructions over the real values. As shown, the *Shifted* (green) reconstruction for TS_1 retains the same shape as TS_3, the position it occupied in the input. This is particularly noticeable in the size of the sigma values, the shorter valleys, and the lack of distinction between workdays and weekends. The same behavior, but in reverse, is observed in the reconstruction of TS_3.

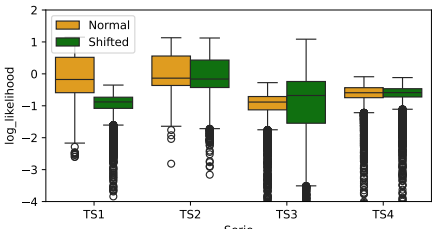
To provide a quantitative comparison between *DC-VAE* models, we used the metrics $MSE_{\mu,\sigma}(x)$ and $LL_{\mu,\sigma}(x)$, defined in equations 2.10 and 2.11, respectively. Both metrics yield a value for each time-series data point when x, μ_x, and σ_x are replaced by $x_m(t)$, $\mu_{x_m}(t)$, and $\sigma_{x_m}(t)$ in the equations. Equation 2.10 represents the result obtained when, given μ_x and σ_x, we generate a considerable number of samples and compute the MSE for all these values with respect to x.



(a) Comparison of reconstruction.



(b) Boxplot of MSE.



(c) Boxplot of log-likelihood.

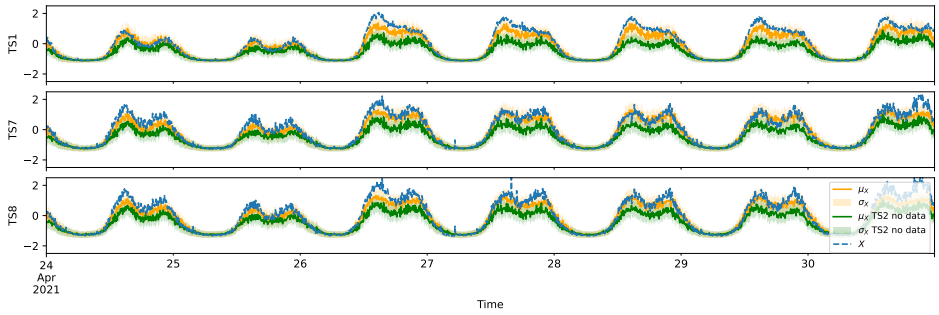
Figure 2.16: In (a), a comparison of the reconstruction for an input in the same order as the training (orange) and an input with the variables TS1 and TS3 shifted (green) is shown, over the real values (blue). In (b), the distribution of the MSE values for each configuration is presented, and in (c), the same is shown for the log-likelihood values.

$$MSE_{\mu,\sigma}(x) = (x - \mu_x)^2 + \sigma_x^2 \tag{2.10}$$

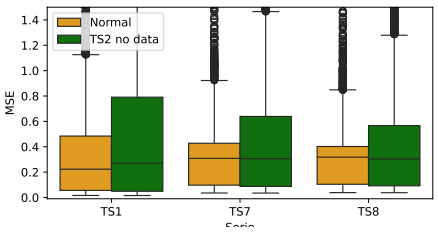
$$LL_{\mu,\sigma}(x) = -\frac{1}{2}\log(2\pi) - \frac{1}{2}\log(\sigma_x^2) - \frac{(x-\mu_x)^2}{2\sigma_x^2} \tag{2.11}$$

The equation 2.11 is the logarithm of the distribution function given μ_x and σ_x, which indicates how closely the real values align with the model. Larger values suggest a closer match. Figure 2.16(b) shows the distribution of MSE values. For TS_1, the difference observed in Figure 2.16(a) is reflected, where the MSE distribution for the *Normal* inference is significantly lower than for the *Shifted* configuration. A similar trend is evident in the log-likelihood metric, where the *Normal* reconstruction outperforms the *Shifted* one. However, this pattern does not hold for TS_3, where the noisiness of the values reduces the relevance of the mean reconstruction, and the σ_x values play a more dominant role. Nevertheless, the performance change remains evident.

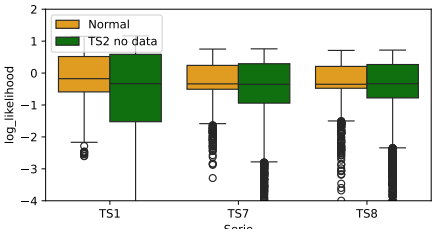
Figures 2.16(b) and (c) also present the same distributions for TS_2 and TS_4, as these variables are correlated with TS_1 and TS_3, respectively. While a slight performance difference can be observed, it is not significant, leading to the conclusion that the most affected variables are those whose positions in the input order



(a) Comparison of reconstruction.



(b) Boxplot of MSE.



(c) Boxplot of log-likelihood.

Figure 2.17: In (a), a comparison of the reconstruction for an input identical to the dataset (orange) and an input where the variable TS_2 is flattened to a constant value (green), simulating the absence of data, is shown over the real values (blue). In (b), the distribution of the MSE values for each configuration is presented, and in (c), the same distribution is shown for the log-likelihood values. The variables shown: TS_1, TS_7, and TS_8, are the most affected by the absence of data in TS_2.

differ from those used during training.

The second evaluation examines how the absence of data in one variable affects the reconstruction of the others. To assess this, we fix one of the variables to a constant value, simulating missing data, and compare the reconstruction of the remaining variables to the results obtained using the normal input.

Figure 2.17 presents an example where the *Normal* inference, as in the previous case, corresponds to the input as it appears in the dataset. The data for the case labeled TS_2 *no data* is the same, except that the variable TS_2 has been replaced with a constant value, simulating the absence of data. In this example, we show the reconstruction performance for the variables most affected—those that are more correlated with TS_2.

Observing the reconstructions in Figure 2.17(a), particularly around the peaks, we can see that the absence of data in TS_2 (green reconstruction) causes the μ_x values to struggle in accurately following the real values x (blue line). Even for TS_1, there are noticeable issues with the reconstruction's shape compared to both the real values and the reconstruction when TS_2 contains data (orange reconstruction).

In Figures 2.17(b–c), we observe a clear degradation in the reconstruction metrics when data is absent. Specifically, the distributions of the MSEμ, σ and LLμ, σ values become wider, indicating increased variability in reconstruction errors and likelihood.

This result highlights how the advantage of a multivariate model—leveraging information across variables—can also become a disadvantage when data from one variable is missing, as it negatively affects the reconstruction of other correlated variables.

2.7. Global *DC-VAE*: A New Approach for Better Adaptability

A multivariate model has several advantages, as we observed earlier, such as leveraging the relationships between variables, providing predictions for all variables simultaneously, and being faster to train. However, it also presents a rigidity that can pose challenges in specific scenarios.

The first issue is that during inference, the order of the variables in the multivariate time series must remain consistent with the model's expected input. Changing the order of some variables affects the reconstruction performance for those variables. Similarly, the absence of data in one variable can degrade the performance of other related variables.

Another challenge arises from the need to add or remove variables in certain systems. In such cases, the input and output layers would need to be retrained from scratch. For example, in the case of a telecommunication company as Telefónica, as we'll see after, where variables are obtained from database queries, the monitored variables may change dynamically based on user requirements. These challenges make it difficult to reuse the model across different scenarios, limiting its flexibility and generalizability.

For this reason, we propose developing a new version of the *DC-VAE*, shifting from a multivariate approach to a global approach. A global model in time-series analysis is typically a univariate model trained on multiple time-series simultaneously. This means the global model learns patterns that are shared across multiple series, allowing it to generalize better and handle variability between series. In contrast to a multivariate model that operates on a fixed set of variables, a global model can dynamically adapt to the inclusion or exclusion of variables or series. This flexibility not only addresses the rigidity of multivariate models but also enables the model to be reused across different scenarios and datasets with minimal retraining.

The architecture and detection process remain the same in the global approach. However, two significant aspects change: the shape of the input and output of the VAE model, and the structure of the latent space. The latter change arises because, in the original *DC-VAE*, the latent space samples $\mathbf{z}$ have the same length T as the input samples $\mathbf{x}$. In the global approach, since the inputs are univariate, maintaining this shape in the latent space is not feasible in an Auto-Encoder (AE). This necessitates adjustments to the latent space structure to accommodate the univariate nature of the inputs while preserving the model's functionality.

To address this, we leverage the dilated convolutional architecture of the encoder, which ensures that the last vector of the output sequence captures information from the entire input sequence, as illustrated in Figure 2.4 and Figure 2.18. Therefore, we use only this final vector for the latent space representation. With this approach, the shape of the latent space depends solely on J, making $\mathbf{z} \in R^J$ as shown in the figure 2.18. In this figure, the complete structure of the global *DC-*

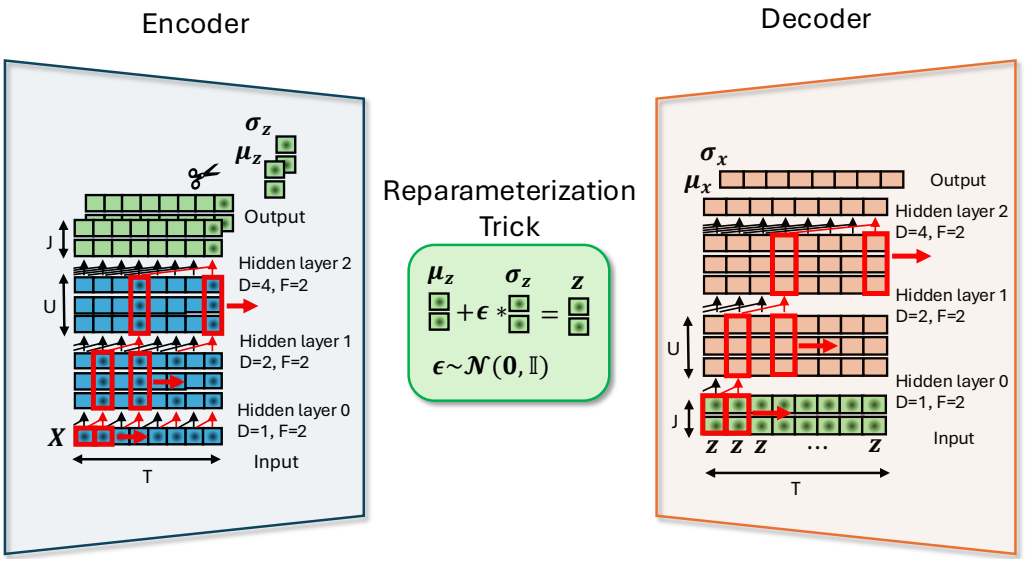


Figure 2.18: Scheme of the global DC-VAE. The three main aspects that change are: the input and output of the encoder and decoder, which are now univariate; the shape of the latent space, which now depends only on the hyperparameter J; and the input to the decoder, which is the repetition of the vector $\mathbf{z}$ T times.

VAE is represented. It highlights changes in the input and output of the encoder and decoder, as well as the dimension of the latent space. One aspect that may draw attention is the input to the decoder, which consists of repeating the vector $\mathbf{z}$, obtained from the reparameterization trick, T times.

This decision was made, firstly, to preserve the decoder architecture from the original *DC-VAE* and, secondly, to maintain symmetry between the encoder and decoder in terms of the number of parameters. An alternative approach could involve using the $\mathbf{z}$ vector followed by a dense layer as the decoder's input layer, but this would result in a decoder with more parameters than the encoder. Another option could involve using transpose convolutions, but we observed that this approach made it more difficult for the model to converge. One negative aspect of this approach is that, at the beginning of the decoder, many layer outputs are identical for different time instances. However, due to the fact that only a portion of the filters are active at the start of the sequence, combined with the dilations, these values quickly evolve into distinct outputs as the sequence progresses.

2.7.1. Global *DC-VAE* Analysis

The first step for developing a global model was to determine the appropriate hyperparameters, as the architecture changes slightly in this approach. To achieve this, we performed a hyperparameter search using a predefined grid, as shown in Table 2.8. The search was conducted while ensuring that the number of trainable parameters did not exceed 90 % of the total number of samples (time-series windows). In the column *Best*, we present the selected hyperparameters. Notably, the value of T was 128, approximately half a day for the TELCO dataset (288 values per day), while J was set to half of this value, 64. The total number of parameters amounted to 174,560, which represents 56 % of the total samples (309,516).

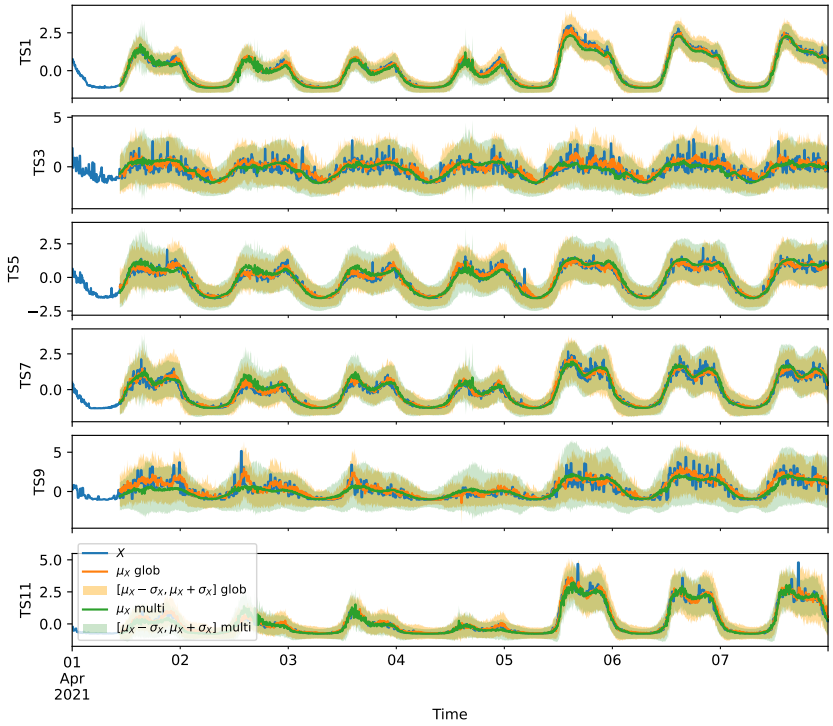
Reconstruction Capability

The first aspect to analyze is the model's capability to reconstruct the data accurately. As a reference, we also trained a model with the same architecture but with a multivariate input. The only differences were in the input and output configurations of the model, which incorporated slightly more trainable parameters (179,840), while the latent space remained the same as in the global approach.

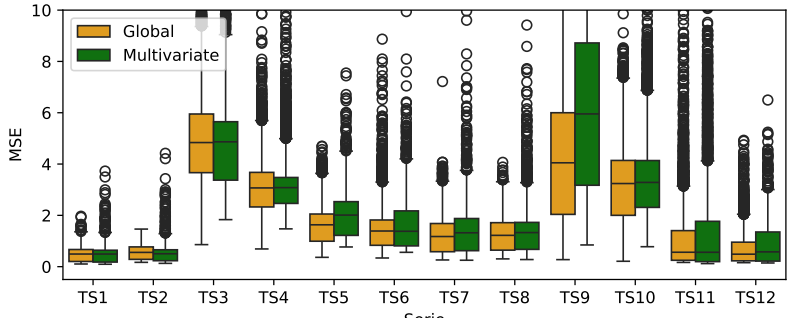
Table 2.8: Grid of hyperparameters used in the model calibration of the global *DC-VAE*.

Hyperparameter	**Grid Search Ranges**	**Best**
T - sequence length	$\{128 - 2048\}$, step=128	128
J - latent dimension	$\{64 - 3T/4\}$, step=32	64
γ - learning rate	$\{1e^{-5} - 5e^{-4}\}$	$4e^{-4}$
m - mini-batch size	$[32, 64, 128]$	32
U - number of filters	$\{64 - 128\}$, step=16	80

In Figure 2.19, we present both qualitative and quantitative comparisons between the two approaches. In Figure 2.19(a), the reconstructions are shown alongside the actual series values, revealing that both approaches produce similar results. However, the multivariate model exhibits slightly smoother reconstructions in noisier series. Figure 2.19(b) displays boxplots of the MSE for each TELCO time series, indicating comparable performance across most series. Notably, for TS_9, the global model achieves better reconstruction accuracy.



(a) Reconstruction comparison for some time-series of TELCO.



(b) Boxplot of reconstruction MSE for all time-series.

Figure 2.19: Comparison of MSE reconstructions between both approaches, global and multivariate, where the figure shows that there isn't a significant difference between them.

Anomaly Detection Performance

Then, if the reconstruction is similar between both approaches, we would expect similar anomaly detection performance. In Table 2.9, the performance of *DC-VAE* is shown, consistent with Table 2.6, where the color coding remains the same. On the right side, the performance of the global model with the selected hyperparameters is presented. As observed, the global approach outperforms in $F1_r$ for only 1/3 of the variables. However, the mean values are comparable, with a decrease of 10 points in R_r but a gain of 6 points in P_r. The median is considerably lower.

In conclusion, the global approach is unaffected by the order of variables in the input or by missing data influencing other variables, as it operates with univariate inputs and outputs. In addition, while its anomaly detection performance may be slightly lower than that of the multivariate approach, the difference is not significant.

Completely Unsupervised Detection using p-values

Up to this point, the operational points of all evaluated *DC-VAE* models were selected in a supervised manner by choosing the α_i values that maximize the $F1_r$ metric on a validation set, which requires a labeled dataset. This approach allows for a fair comparison of *DC-VAE* models with others under the same conditions. However, given that our models follow a Gaussian distribution, it is possible to set the α values using specific criteria based on the properties of the Gaussian distribution.

When we define a value as an anomaly if its distance to the predicted μ_x exceeds α times σ_x, this is equivalent to establishing symmetric bounds around μ_x, where any value inside these bounds is considered normal, and any value outside

Table 2.9: Anomaly detection performance benchmarking in TELCO, comparing *DC-VAE* against global *DC-VAE*.

	DC-VAE			***global DC-VAE***		
TS ID	R_r	P_r	$F1_r$	R_r	P_r	$F1_r$
TS_1	58	71	**64**	29	60	39
TS_2	74	20	67	71	80	**75**
TS_3	86	47	**60**	71	29	42
TS_4	63	21	32	50	35	**41**
TS_5	75	50	60	50	67	57
TS_6	57	83	68	57	100	**72**
TS_7	72	90	**80**	50	71	59
TS_8	44	80	**57**	38	50	43
TS_9	17	11	**13**	0	0	0
TS_{10}	52	59	**55**	60	24	34
TS_{11}	100	25	**40**	100	21	35
TS_{12}	100	11	22	100	100	**100**
mean	67	47	**52**	56	53	50
median	68	49	**59**	54	55	42

is considered anomalous. The area under the Gaussian distribution within these bounds represents the probability of normality, P_N. This enables us to link P_N with α. Let the lower bound be γ (where $\gamma < \mu_x$). Since the area under both tails is symmetric, we can calculate P_N as follows:

$$P_N = 1-2P_{\mu_x,\sigma_x}(x < \gamma) = 1-2\int_{-\infty}^{\gamma} \frac{1}{\sqrt{2\pi\sigma_x^2}} e^{-\frac{(x-\mu_x)^2}{2\sigma_x^2}}\, dx = 1-2\Phi_{\mu_x,\sigma_x}(\gamma). \quad (2.12)$$

The function $\Phi_{\mu_x,\sigma_x}(\gamma)$ can be expressed in terms of the error function, $erf()$, which has an inverse function:

$$\Phi_{\mu_x,\sigma_x}(\gamma) = \frac{1}{2}\left[1 + erf\left(\frac{\gamma-\mu_x}{\sigma_x\sqrt{2}}\right)\right]. \quad (2.13)$$

By combining Equations 2.12 and 2.13, we can solve for γ in terms of P_N:

$$\gamma = \mu_x + \sigma_x\sqrt{2}\; erf^{-1}(-P_N). \quad (2.14)$$

Considering that all values satisfying $|x - \mu_x| > \mu_x - \gamma$ are anomalies, we can rewrite the inequality and derive α in terms of P_N, that is,

$$|x-\mu_x| > \mu_x - \gamma \quad (2.15)$$
$$|x-\mu_x| > -\sigma_x\sqrt{2}\; erf^{-1}(-P_N) \quad (2.16)$$
$$\frac{|x-\mu_x|}{\sigma_x} > -\sqrt{2}\; erf^{-1}(-P_N). \quad (2.17)$$

Thus, the α value is given by,

$$\alpha = -\sqrt{2}\; erf^{-1}(-P_N). \quad (2.18)$$

If we set a criterion, such as requiring P_N to be more than: $[99\,\%, 90\,\%, 70\,\%]$, the corresponding α values are [2.58, 1.64, 1.03].

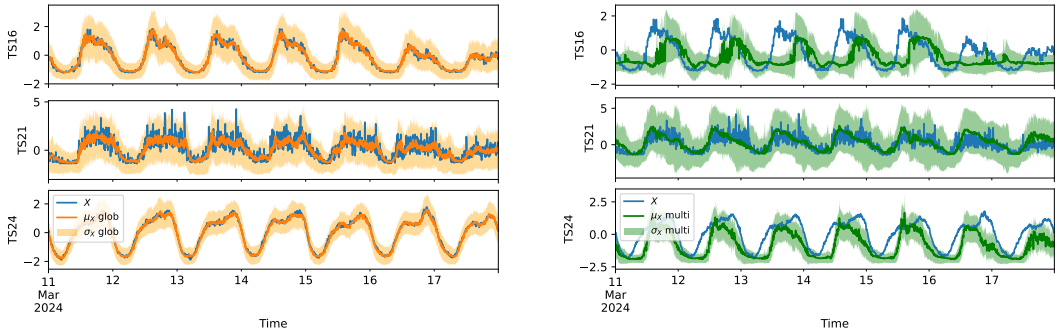
In Table 2.10, we present the results for these P_N values applied to the Global *DC-VAE*. As observed, α decreases as P_N decreases. Starting with a high P_N value, we see high P_r values and low R_r values. As P_N decreases, P_r begins to decline, while R_r improves. This trend is evident for most time-series in Table 2.10.

An interesting observation is that the mean and median of R_r, P_r, and $F1_r$ for $P_N = 70\,\%$ are not significantly different from those reported in Table 2.9 for the same model. This suggests that *DC-VAE*, as a Gaussian-based model, enables the definition of an operational point using a fixed criterion for all time series. This approach, which does not rely on labeled data, can yield results comparable to those obtained using a supervised criterion.

Moreover, we must consider that linking performance directly to the values of P_N is not straightforward. First, the values may not originate from a Gaussian distribution. Even if they did, *DC-VAE* is trained to maximize the ELBO, and since this is a relative maximization, it does not necessarily coincide with the maximization of the likelihood. Additionally, performance depends on the quality of the labeled data, and throughout this work, performance has been measured using range metrics, whereas this probability is a point estimate.

Table 2.10: Global *DC-VAE* performance in a completely unsupervised approach for different values of P_N.

Global DC-VAE	$\mathbf{P}_N = 99\,\%$			$\mathbf{P}_N = 90\,\%$			$\mathbf{P}_N = 70\,\%$		
TS ID	R_r	P_r	$F1_r$	R_r	P_r	$F1_r$	R_r	P_r	$F1_r$
TS_1	14	100	25	21	100	35	28	60	**39**
TS_2	0	0	0	8	100	15	13	100	**22**
TS_3	43	60	**50**	57	44	**50**	86	10	17
TS_4	25	100	40	38	50	**43**	75	25	38
TS_5	25	67	36	38	80	51	63	71	**67**
TS_6	0	0	0	0	0	0	57	83	**68**
TS_7	40	100	57	50	100	**67**	50	56	53
TS_8	13	50	20	35	67	36	38	43	**40**
TS_9	5	20	9	33	15	**21**	50	8	13
TS_{10}	0	0	0	10	100	18	45	68	**54**
TS_{11}	33	100	**50**	67	33	44	100	19	32
TS_{12}	0	0	0	0	0	0	100	100	**100**
mean	17	50	24	29	57	32	59	54	**45**
median	13	55	23	29	58	33	53	58	**39**



(a) Reconstruction with global approach. (b) Reconstruction with multivariate approach.

Figure 2.20: Time-series reconstruction on TELCO2 using the global and multivariate models trained on TELCO.

Zero-shot Reconstruction

To test the generalization capability of the global model, we performed an inference on a different dataset using the same model trained on TELCO from the previous experiment. This new dataset, which we call TELCO2, also contains 12 distinct time-series sampled every 5 minutes but originates from a different source than TELCO. The series exhibit similar seasonality, with valleys at night and peaks during the day, but with different shapes. In Figure 2.20(a), an example of the global model's inference on TELCO2 is shown. As observed, the reconstructions are as accurate as those obtained with TELCO. For comparison, Figure 2.20 (b) presents the inference results using the multivariate model. As expected, given the previous results, the reconstructions show poorer performance.

This result demonstrates that the global approach can generalize not only within the domain where the model was trained but also in a different domain,

proving that this approach can work for zero-shot inference. Although the domains of TELCO and TELCO2 are not drastically different, in a later section, we will show that a global approach can be useful for time series from very different domains as well. Additionally, we will explore whether this zero-shot capability can be applied to anomaly detection tasks.

2.8. Conclusions

DC-VAE is an anomaly detection method based on variational autoencoders (VAE), featuring a fully convolutional architecture with dilations. This combination makes the method simple and fast to train while avoiding convergence issues. Additionally, it requires only a few hyperparameters, most of which are practically determined once the sliding window length is selected, simplifying the search for the optimal model. Regarding training data, the method benefits from semi-supervised training (excluding anomalies), but its autoencoder nature also allows for unsupervised training (using raw data) without compromising performance.

For detection, although the method's inputs and outputs can be either multivariate or univariate, anomaly detection is performed in a univariate manner, with a normality region defined for each individual time series. The operating point is determined by a single parameter, α. Given the model's Gaussian assumption, it is possible to establish α values using unsupervised criteria. Thanks to its fully causal and lightweight architecture, anomaly detection can be performed in near real-time.

When applied to real-world data, such as the TELCO dataset (provided in this work), the original multivariate version of *DC-VAE* proved to be the most effective anomaly detection method across a greater number of time series compared to other approaches. It outperformed a combination of machine learning, statistical, and recursive methods (ENS-15), as well as established models for this task, such as state-space-based approaches (S-EXPS, ARIMA), and even a standard VAE model (S-VAE). It also demonstrated superiority over its own variations, including the global *DC-VAE* and *FAE*, as will be discussed later. Furthermore, its application to the widely used multivariate dataset SWaT showed that the model is competitive with more complex state-of-the-art methods such as EGAN, MAD-GAN, and NET-GAN (previously proposed by us). *DC-VAE* was also evaluated on satellite telemetry data, benchmarked against different anomaly detection methods. It demonstrated strong performance in the early detection of anomalies in one of the missions, outperforming Telemanom, a specialized method designed for this type of data. Additionally, the authors of the benchmarking study highlighted the versatility and ease of use of *DC-VAE* compared to Telemanom.

Later, the global *DC-VAE* variant addressed several adaptability limitations caused by multivariate inputs, such as the ordering of time series or the absence of data for certain variables, without significantly compromising detection performance. Since it retains nearly the same architecture, all the original properties are preserved. Additionally, it showed signs of being capable of adapting to new

domains without requiring prior exposure during training. This observation led to the development of the foundational version of *DC-VAE*, known as *FAE*, which will be discussed in Chapter 4.

Esta página ha sido intencionalmente dejada en blanco.

Chapter 3

Continual Anomaly Detection in Time-Series using Generative AI

One of the main limitations faced by *DC-VAE*, and AI/ML-driven approaches for anomaly detection in general, is their inability to effectively handle *Concept Drift* (CD) and *Domain Change* (DC). Concept drift refers to events where the statistical properties of the target variable, or the relationships between input features and the target variable, change over time. As a result, the patterns and rules that an AI/ML model has learned from historical data may no longer hold for current data, requiring the model to be updated to adapt to these changes. Domain change, on the other hand, occurs when the environment in which the model operates differs from the one it was trained on. For example, if the system monitored by the anomaly detector adds one or more new time series after the model has been trained, these new variables need to be incorporated into the model. CD and DC are closely related to another phenomenon that impacts and degrades the performance of AI/ML models, known as *catastrophic forgetting*. While catastrophic forgetting is a distinct issue, it is related in that it occurs when an AI/ML model trained on a set of tasks or data samples forgets previously learned information after learning new tasks or samples. Under catastrophic forgetting, the model's performance on earlier tasks deteriorates significantly, even if the old and new tasks are related. CD and DC are strongly linked to catastrophic forgetting because they involve changes in the data distribution that can render an AI/ML model outdated or inaccurate. Both problems require methods that enable models to adapt to evolving data distributions, typically through retraining. In its simplest and most effective form, retraining an AI/ML model with newly acquired data—whether due to CD, DC, or new related tasks—requires access to all previously used training data. However, this traditional retraining approach is often limited by the availability of past data and constrained by memory and computational resource requirements.

We resort to the *continual learning* paradigm [52] to address the continual model adaptation and retraining of *DC-VAE*. Continual learning enables a model to learn from a stream of evolving data, without forgetting previously learned knowledge. It involves updating the model's parameters and architecture as new data

arrives, while also preserving knowledge learned from previous data, representing a promising approach to deal with CD and DC. We extend *DC-VAE* to a continual learning setup, leveraging the generative AI properties of the underlying VAE model to remember past data. By conception, once the encoder-decoder VAE model has been trained, the decoding function is capable to synthesize new "fake" data mimicking the characteristics of the time-series training datasets, using as input only Gaussian noise. As such, the decoder acts as a lossy compression of the data used for training. We combine *DC-VAE* and its generative decoder into *GenDeX*, an approach to continual learning for anomaly detection in time-series network measurements. In a nutshell, when *DC-VAE* is confronted with concept drifts, or is applied to a new time-serie dataset – e.g., measurements collected at a different network or representing a different process – *GenDeX* uses the previously trained decoder to synthesize past time-serie measurements, and combines them with the new time-serie data to retrain the underlying VAE model. *GenDeX* follows a *Deep Generative Replay* (DGR) [53] paradigm for continual learning, where a generative model produces synthetic data which replays old memories during training, augmenting the heterogeneity and expressiveness of the retraining. The rationale behind *GenDeX* is that *DC-VAE* continually improves its tracking and baselining capabilities as it processes new measurements with different underlying statistical characteristics, improving as such its generalization and anomaly detection capabilities with time.

In this chapter, we study in depth the generative capabilities of *DC-VAE*, investigating the characteristics of the resulting latent space and refining it for fine-grained temporal data generation. Finally, we demonstrate how the trained decoder can generate synthetic time series from Gaussian noise, successfully capturing the patterns of each individual time series in the process, despite their differing characteristics. Additionally, we evaluate how *GenDeX* addresses catastrophic forgetting, maintaining performance on both new and previously learned tasks in the context of both concept drift (CD) and domain change (DC).

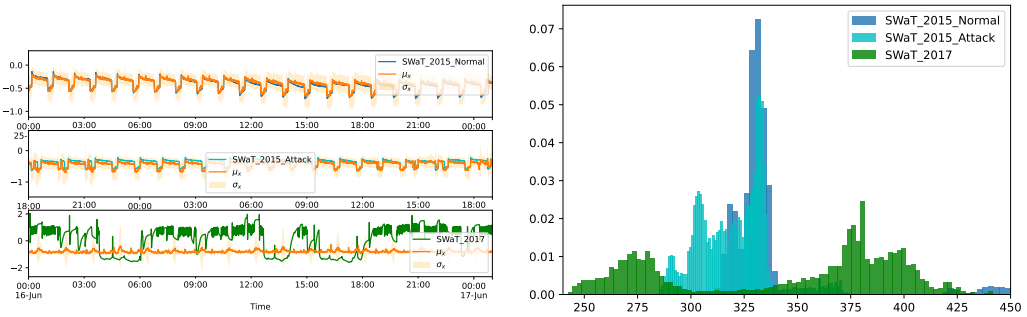
3.1. Related Work

Continual Learning (CL) enables a model to learn from a stream of evolving data, without forgetting previously learned knowledge. There are various approaches to CL, including *Regularization Techniques* (RT) [52], *Generative Replay* (GR) [53], and *Dynamic Architecture* (DA) [54]. RT involves penalizing the model's parameters to reduce the impact of new data on previously learned knowledge. One such technique is *Elastic Weight Consolidation* (EWC) [52], which uses a quadratic penalty term to constrain the neural network's weights during training to protect important parameters from forgetting. GR involves generating synthetic data that is similar to previously observed data to reinforce old memories. *Deep Generative Replay* (DGR) is an example of this approach, which uses a generative model to produce synthetic data that is similar to previously observed data. The synthetic data is used to replay old memories during training to prevent forgetting. Similar to DGR, BooVAE [55] generates new data to augment the training

set. However, unlike generative replay, BooVAE generates new samples by perturbing the existing data rather than directly generating new samples from scratch, (in theory) preserving the statistical properties of the original data distribution. Dynamic architecture involves expanding or shrinking the model's architecture to accommodate new knowledge or discard outdated knowledge. *Progressive Neural Networks* (PNN) [54] is a notable approach to dynamic architectures, which dynamically expands the neural network architecture to incorporate new knowledge while retaining previous knowledge. PNN can achieve high accuracy on sequential learning tasks without forgetting previously learned knowledge.

3.2. GenDeX - Continual Learning for *DC-VAE*

A Concept Drift (CD) can manifest itself as a shift in the mean, an increase or decrease in the variance, or even as complete data modifications. Such changes may be related to important trends in the data or to measurements collected in a different setup, requiring proper detection and retraining. Figure 2.15 in Section 2.5 shows an example of *DC-VAE* operation under a concept drift, where a gradual change in the interval indicated as the CD zone is simulated in a single time-series (TS_5), leaving the other series untouched. *DC-VAE* is not capable to track this individual drift, given its multivariate nature – the complete MTS process introduces an hysteresis effect in the reaction of the model. Note in particular how the model can perfectly track the non-modified time-series, and how the estimation for TS_5 follows the pre-CD pattern. Once the induced drift is over, and the MTS process returns to previous statistical behavior, *DC-VAE*'s tracking for TS_5 becomes again accurate. Figure 3.1 shows *DC-VAE* under a more drastic concept drift, in this case considering data from different years (2015 and 2017) from the open SWaT dataset [8] – commonly used for detection of cyber-attacks in cyber-physical systems. Figure 3.1(a) shows the tracking of *DC-VAE* in (top) the 2015 normal operation dataset used for training, (middle) the 2015 attack dataset used for testing, and (bottom) the 2017 dataset. *DC-VAE* performs accurately in the testing dataset, as the underlying empirical distributions of both training and testing datasets significantly overlap, as evidenced in Figure 3.1(b). However,



(a) *DC-VAE* under strong concept drift. (b) Subsets S_{2015-N}, S_{2015-A}, and S_{2017}.

Figure 3.1: Strong subset changes requires retraining.

the model totally fails to capture the SWaT dataset in 2017, as the underlying distributions of the corresponding data are significantly different.

We therefore explore an approach to cope with the described concept drifts, in particular exploiting the generative nature of the *DC-VAE* model for continual learning. In a continual learning framework, we assume a continually evolving stream of data, represented as a sequence of subsets S_j, each characterized by a specific underlying distribution. We define a sequence of λ_∞ subsets $S_1, \ldots, S_{\lambda_\infty}$ arriving sequentially and assume access to only the data in the current subset S_t, with $t \leq \lambda_\infty$. We consider a CD occurring at time t, and thus assume that the underlying distributions of $S_1, \ldots, S_{t-1}$ are similar among them, but significantly different from S_t. An initial *DC-VAE* model is trained using S_1 data, which performs accurately until time t. We refer to this model as $DC\text{-}VAE_0 = \{q^0_\phi, p^0_\theta\} = \{E^0_\phi, D^0_\theta\}$, where E and D represent the encoding and decoding functions, respectively.

GenDeX follows the principles behind DGR to adapt $DC\text{-}VAE_0$ to the new data S_t, without forgetting the parameterization learned from S_1, valid for $S_1, \ldots, S_{t-1}$. Figure 3.2 explains the *GenDeX* approach. The decoding function D^0_θ acts as generator, and it is used to synthesize a new dataset $F_{1\to(t-1)}$ out of Gaussian noise, which mimics former training examples in S_1 and its underlying distribution. We say D^0_θ acts as the *teacher* model. Then, the new *student* model $DC\text{-}VAE_1$ is trained on joint synthetic data F and new data S_t. This approach is conceptually simple, model-agnostic and overcomes catastrophic forgetting, as the updated model $DC\text{-}VAE_1$ is now capable to handle pre- and post-concept drift data distributions. The challenging part in *GenDeX* is to tame the latent space of *DC-VAE* to actually generate an time-serie process which reliably reproduces the data initially used for training.

Recall that for the original multivariate *DC-VAE* latent space $\boldsymbol{z} \in \mathbb{R}^{J\times T}$ can be potentially huge, e.g., in the examples we showed in Section 2.2, $J = 4$ and $T = 512$, so we have to deal with a 2048-dimensional space, and thus, sampling Gaussian noise of such dimensionality might not generate the desired outcome. Therefore, as mentioned before and as reflected by the architecture of *DC-VAE* in Figure 2.4, we trim the latent space dimensionality and focus exclusively on $\boldsymbol{z}$ at T as show in Figure 2.18, resulting in a vector $\boldsymbol{z} \in \mathbb{R}^J$. Realizing a latent

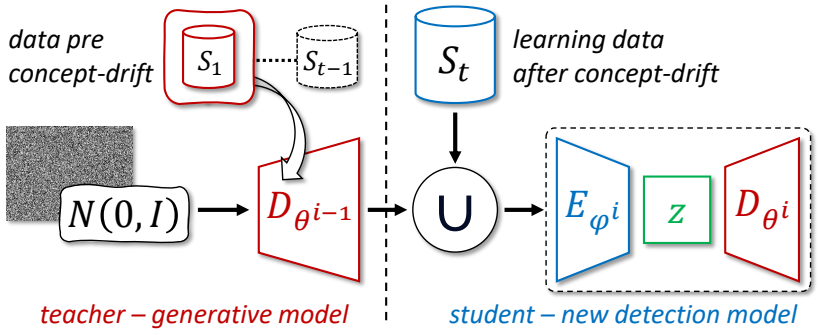
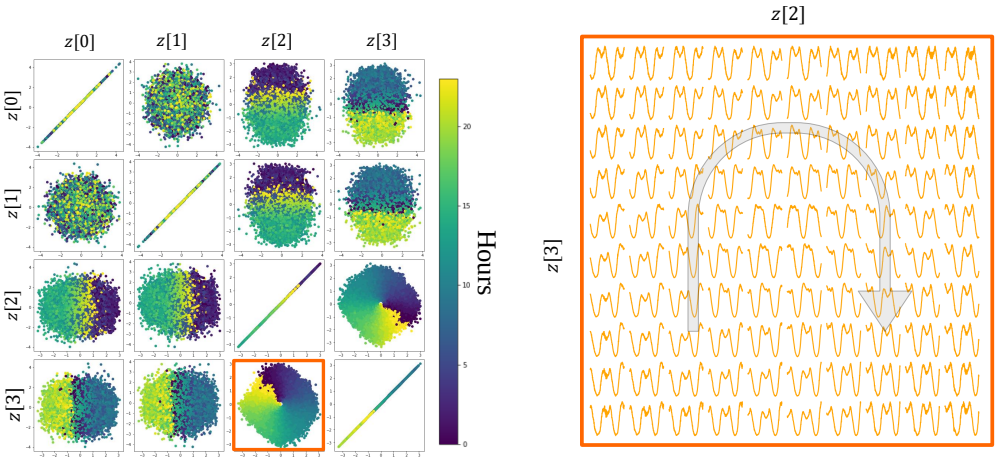


Figure 3.2: The GenDeX generative replay approach. At time t, a concept drift significantly modifying the underlying distribution of S_t triggers a model retraining event i.

space where the sample distribution approaches a zero-one normal distribution, as the VAE hypothesis states, helps the generative part of the VAE model, i.e., the decoder, to generate samples that resemble the real ones, by simply drawing inputs from such a Gaussian distribution. Next, we demonstrate how to realize the generating function in practice, exploring the latent space and reporting the results obtained in the synthetic generation of MTS data from the TELCO time-series dataset.

3.3. Latent space and generative feature of *DC-VAE*

We now focus on the generative properties of *DC-VAE*, firstly by analyzing the latent space generated by the encoding function E_ϕ, and then by exploring the generative capabilities of the generative model as represented by the trained decoding function D_θ. The dimension of the latent space in a VAE model is one of the hyper-parameters to define during model evaluation. These dimensions are restricted by the dimensions of the input samples $\boldsymbol{x}$ space, as for the model to only capture the relevant information or energy of the samples, there must be a dimension reduction. By conception and hypothesis, the distribution of the samples $\boldsymbol{z}$ living in the latent space must be a normal distribution with zero mean and an identity covariance matrix. This is enforced during training with the second term of the ELBO loss function 2.3.



(a) Latent space representation. (b) Synthetic time-series generation.

Figure 3.3: *DC-VAE* latent space representation. Latent space $\boldsymbol{z}$ with $J = 4$. The colors correspond to the hours of the day. Grid of samples generated from uniform sampling on dimensions $z[2]$ and $z[3]$ of the $\boldsymbol{z}$ latent space. If the figure is traversed clockwise, it is possible to see how the generated time-series evolve over time.

3.3.1. Analysis for the multivariate approach

To evaluate the behavior of the encoder E_ϕ, a representation of the latent space is shown for a trained *DC-VAE* multivariate model using TELCO data. We use $J = 4$ in the architecture adopted for the global approach, as illustrated in Figure 2.7, where the latent space consists of vectors of dimension J. Therefore, each latent representation is given by $\boldsymbol{z} = z[0], z[1], z[2], z[3]$. Figure 3.3(a) shows the resulting latent representation, projected onto each bi-dimensional combination of the dimensions $z[i]$. Each point in the figure corresponds to the projection of a sample from the validation set.

The Gaussian property of the latent space distribution is essential for the time-series generation process, as there are no input samples $\boldsymbol{x}$ in *DC-VAE* to use as reference, thus samples need to be generated from input noise. Besides the shape of the realized distribution, and to reflect the temporal dimension of the MTS data, Figure 3.3(a) depicts the coded samples in colors, each color representing a different hour of the day. More specifically, each sample color corresponds to the discretized hourly values of the newest sample-value within the input sequence, at time t. If we consider the bi-dimensional latent space $\{z[2], z[3]\}$, we observe how each hour of the day maps to a different angular area in the data distribution. To

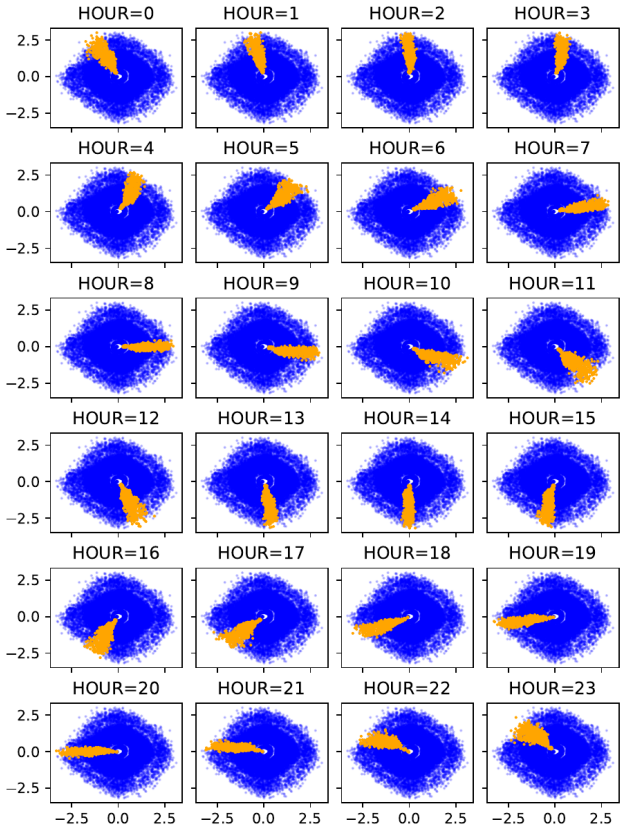


Figure 3.4: *DC-VAE* latent space representation, in an hourly basis. Sampling the latent space at different angles results in different times of the day in the generated time-series.

better appreciate this effect, Figure 3.4 shows the same encoding, but this time highlighting the values of $\{z[2], z[3]\}$ for each hour. Interestingly, each hour has a particular range of angles, and these are sequentially arranged, ordered continuously by hour of the day. Under this setup, it is enough to feed the decoder D_θ with samples drawn from a zero-one normal distribution to generate synthetic time-series samples out of noise. Figure 3.3(b) displays a series of synthetically generated μ_x windows for one of the twelve variables of TELCO, obtained by uniformly sampling the dimensions $z[2]$ and $z[3]$. If the figure is traversed clockwise, it is possible to appreciate how the generated time-series evolve over time.

We now move on to the generation of synthetic time-serie data, for the twelve time-series in TELCO, using D_θ. Figure 3.5 shows two examples per time-series generated out of noise, along with real time-series included in the original validation set, for two days worth of time series duration. The trend of the twelve time-series is perfectly captured by the synthetically μ_x generated examples, with the paramount advantage of these being synthetically generated by D_θ. The twelve time-series are properly generated, despite having different types of behavior and variability.

To evaluate the generative power of *DC-VAE* more broadly, we generate the same number of samples (windows) as those in the validation set for each time-series, and compare them with the real time-series values in the validation set. Figure 3.6 reports, for each time-series, the distribution of the generated and real values, in the form of a histogram. Each pair of distributions have strong overlapping, especially for non-spiky values. Time-series TS_3, TS_9, and TS_{10} show a rather variable behavior, with values strongly deviating from the baseline, which cannot be tracked by the generated baseline values, as shown in the corresponding histograms. Recall that we are using *DC-VAE* to track the form and trends of the time-series, by generating μ_x, which would naturally not capture spiky behaviors.

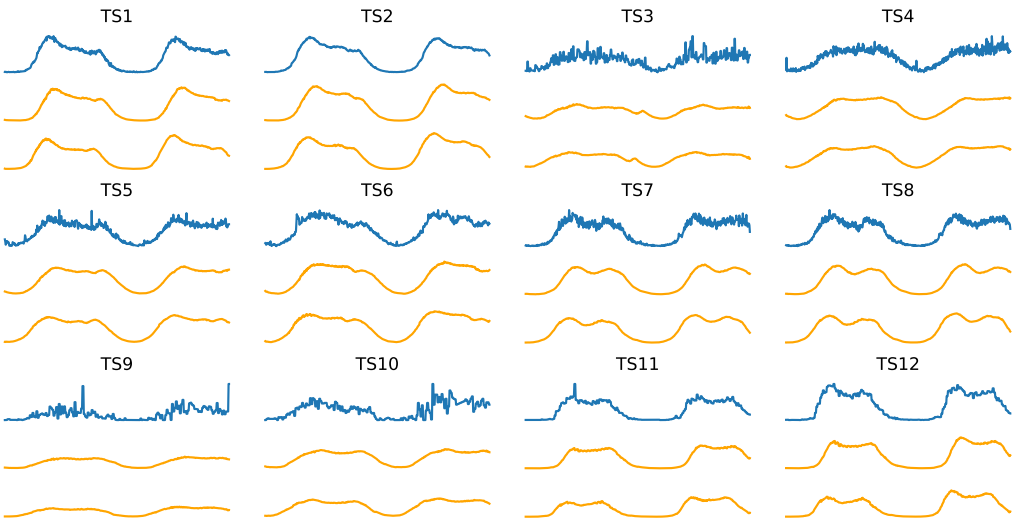


Figure 3.5: Synthetic MTS data generated through multivariate *DC-VAE*. For each time-series in TELCO, two examples of time-series window generated from noise are depicted. The trend of the twelve time-series is perfectly captured by the synthetically generated examples.

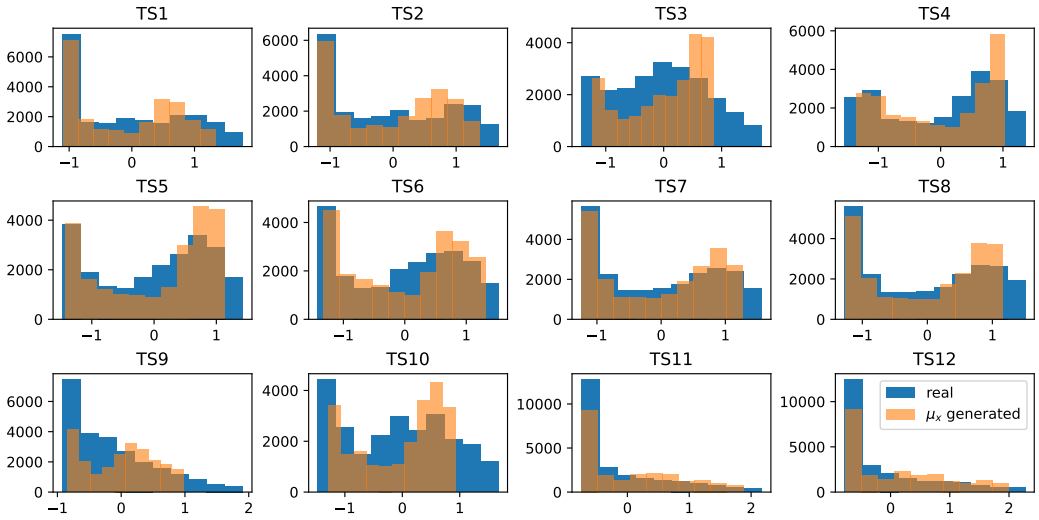


Figure 3.6: Synthetic MTS data generated through DC-VAE. Histograms of samples (μ_x) generated from noise for each time-series of the TELCO dataset. The same number of samples as those in the validation set are generated for each time-series.

Indeed, we are interested in adapting the baselines for anomaly detection, to enable a proper detection of deviations from these baselines.

3.3.2. Analysis for the global approach

In a multivariate approach, the differences between the samples correspond to the specific time period covered by each window. As shown in Figure 3.4, the time of day is a key factor that the encoder E_ϕ needs to capture, forming distinct clusters so that the decoder D_θ can accurately reconstruct the data without delays or distortions. Another factor could be the day of the week, as some series in

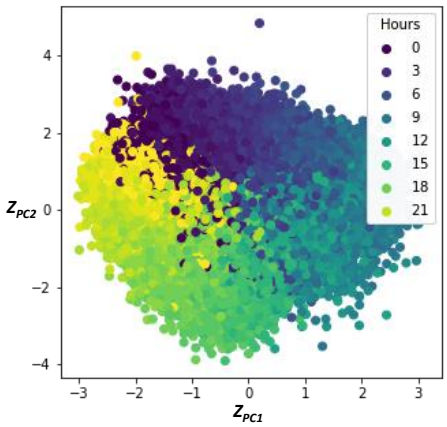


Figure 3.7: Latent space representation in two dimensions for a global model – temporal evolution.

TELCO exhibit different patterns on weekdays compared to weekends. However, beyond temporal characteristics, the encoder does not need to encode additional information, since all series occupy the same positions in the input.

In a global approach, the encoder must capture both, temporal features and the specific type of time-series it needs to reconstruct, as each time-series can have different shapes and trends. To better understand the modeling capabilities of the global *DC-VAE*, we focus on the analysis of the latent space for the different time-series and times within the analysis period. Recall that the latent space dimension for *DC-VAE*, as shown in Table 2.8, is set to $J = 64$. To facilitate visualization of $\boldsymbol{z}$ in a two- or three-dimensional space, we apply the standard PCA and study the top two and the top three principal components, denoted $\boldsymbol{z_{PCi,...i=1,2,3}}$.

To compare the encoding of temporal features with multivariate encoding, we analyzed the entire test dataset to observe how the global model encodes different windows across all time series with respect to the hour of the day. Figure 3.7 shows the latent representations of each test sample $\boldsymbol{x}_t$ for all 12 time series, with each point color-coded according to a three-hour period of the day. When plotting the first two principal components, a pattern similar to that observed in Figure 3.3 for the multivariate approach emerges, where the direction in the latent space aligns with the natural progression of hours on a clock, continuously ordered by the time of day.

To analyze how the model encodes information from different time series to accurately decode each with its distinct shape and behavior, we examined the latent space for four different time series from the TELCO dataset. Figure 3.8 shows the first three principal components of the latent representations, highlighting the latent space for TS1, TS4, TS8, and TS12. As observed, *DC-VAE* maps each time series to a distinct region of the latent space, demonstrating its ability to effectively differentiate between the characteristics of different time series. Samples from different time series that are located close to each other in the latent space exhibit similar behaviors in the time-series domain. For example, TS1 and TS12 are positioned closer to the center of the latent space, both showing marked differences between weekends and weekdays. In contrast, TS4 and TS8, which do not exhibit significant variations between weekdays and weekends, are grouped together and

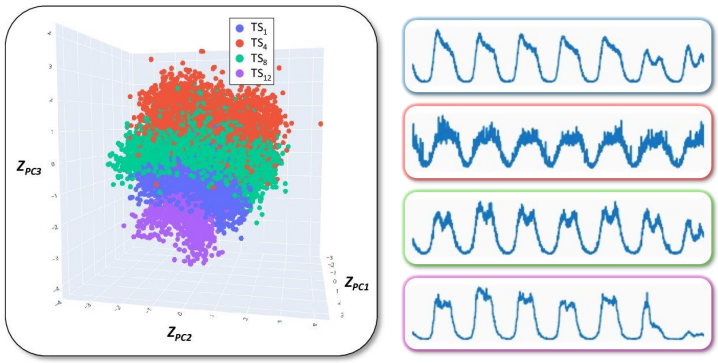


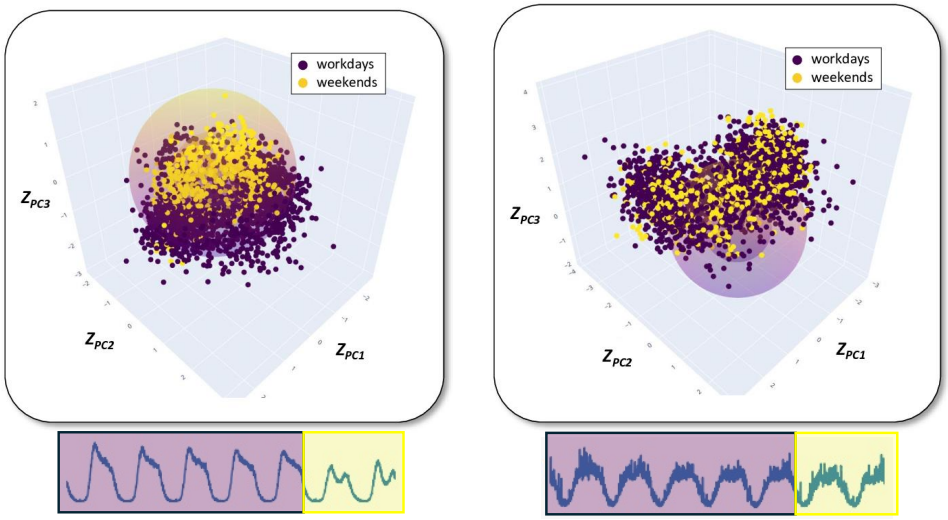
Figure 3.8: Latent space representation per different time-series: TS_1, TS_4, TS_8, TS_{12}.

positioned farther from the center of the latent space.

Since the global model allows us to differentiate and analyze each time series individually, we can explore how *DC-VAE* encodes temporal characteristics that are not shared across all TELCO time series. One such characteristic is the difference between weekdays and weekends, which is present in some series but absent in others. To investigate this, we selected two examples—one with and one without this feature—to analyze their latent space encodings. Figures 3.9(a, b) show the latent representations, where workday samples are displayed in purple and weekend samples in yellow. In (a), we present TS1, which exhibits a clear difference between weekdays and weekends, and in (b), TS4, which does not. For reference, both plots include two spheres with radii of one and two, centered at the origin, along with an example week from each time series displayed below the plots. As observed, TS1 shows a distinct separation: weekend samples cluster closer to the center of the latent space, while workday samples are distributed near the borders of the spheres. This clear distinction between workdays and weekends is not present in TS4, where the samples are more uniformly distributed, reflecting the absence of significant temporal variation between weekdays and weekends.

Finally, another feature present in only some time series is the variation across days of the month. Time series TS11 and TS12 exhibit a downward trend throughout the month, a behavior that can also be observed in their latent space representations. Figure 3.10 shows the latent representation of TS_{12}, with each point color-coded according to the day of the month, ranging from day 1 in purple to day 31 in yellow. As the month progresses, the latent representations shift from the outer borders of the space towards the center, reflecting the series' gradual downtrend over time.

To conclude these evaluations, we observe that *DC-VAE* effectively captures



(a) Latent space representation of TS1. (b) Latent space representation of TS4.

Figure 3.9: Latent space representation specifically for TS_1 and TS_4 in a temporal basis, considering workdays (purple) and weekends (yellow).

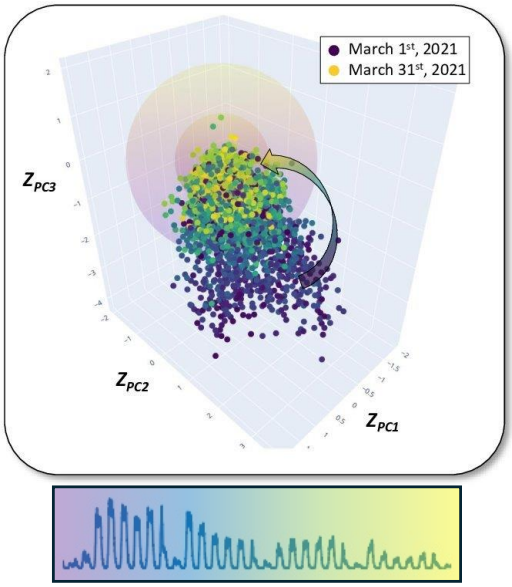


Figure 3.10: Latent space representation for TS_{12}, in a daily basis – from day 1 in purple to day 31 in yellow, for the full month of March 2021.

and differentiates various time-series behaviors and temporal features present in the training data. This indicates that the model and its architecture are sufficiently expressive to handle large, heterogeneous time-series datasets. Additionally, the visual analysis of the latent representations highlights how VAEs—despite their generative nature—operate in a relatively transparent manner, facilitating interpretation and analysis for human understanding. This characteristic represents a significant advantage of VAEs, positioning them as powerful yet explainable generative AI models.

Furthermore, we observe that the latent encodings for all time series—or subsets of them—tend to approximate a standard normal distribution. This implies that sampling from an isotropic normal distribution with zero mean and an identity covariance matrix, and feeding these samples into the decoder D_θ, will result in the generation of all time series (in the global approach) and all time instances (in both approaches). This property is fundamental for the application of *GenDeX*, the continual learning extension of *DC-VAE*.

3.3.3. *GenDeX* analysis

As we explained before, *GenDeX* is an anomaly detection model capable of incorporating new information into the base model without losing performance on the rest of the time-series that make up the system. For example, if we want to start monitoring a new time-series added to the system but lack sufficient data to train a model from scratch, we can perform fine-tuning on a previously trained *DC-VAE*. This allows us to leverage a model likely trained on a substantial amount of data from the same system.

However, if fine-tuning is not handled carefully, it can lead to overfitting on the new data, where the model weights are adjusted specifically for the new series, causing catastrophic forgetting of the other time-series. The same issue arises when one of the time-series already included in the model experiences concept drift, making the base model's representation of normal behavior for that specific series obsolete. Therefore, it is essential to incorporate this new behavior without compromising the performance on the remaining series.

This is where the use of a generative model as the base model becomes advantageous for continual learning, as it allows to the generation of synthetic data to preserve prior knowledge without the need to store large volumes of historical data.

GenDeX for Concept Drift

The first analysis compares the behavior of the model when a single time series undergoes concept drift (CD) and fine-tuning is required to update the model. In Figure 3.11, we present the time series example used for this experiment. As observed, the time series experiences three consecutive CDs, where both the mean and standard deviation increase over time. Each CD lasts for three months, with the first two months used for fine-tuning and the remaining month reserved as a validation set to evaluate the reconstruction using the MSE. For this experiment, we use the same twelve time-series from TELCO but from different months outside the published dataset, where the affected time series is TS_1.

To make the application of generative replay (GR) in this experiment more explicit and to complement Figure 3.2, Figure 3.12 presents a diagram illustrating the use of *GenDeX* in this scenario. It shows how the input data was composed for each fine-tuning performed on the model during each CD. Initially, we have $Model_{t-1}$, which represents the model trained with all the time series in the dataset and is in operation when the first concept drift CD_t occurs. At this point, the current decoder (D_{t-1}) is used as a generator to produce synthetic data representing the base model learned by $Model_{t-1}$. This synthetic data is combined with the data from CD_t. The combined dataset is then used to fine-tune the model, resulting in $Model_t$, which captures both the pre-CD data distribution and the newly introduced distribution. This process is applied consecutively for subsequent CDs.

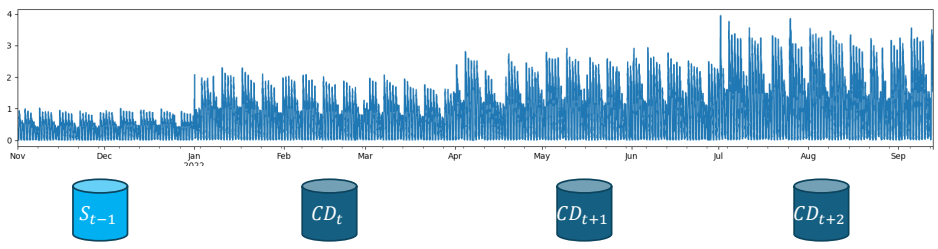


Figure 3.11: Time series experiencing three consecutive concept drifts. S_{t-i} represents the previous values where the models were trained, while CD_t, CD_{t+1}, and CD_{t+2} represent the consecutive concept drifts.

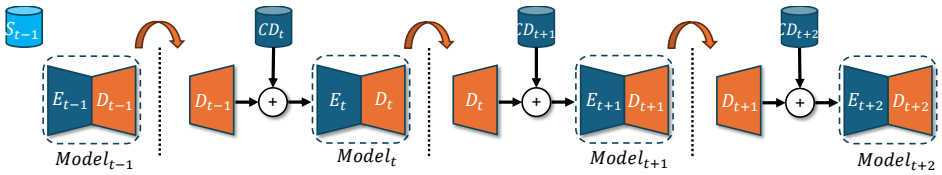


Figure 3.12: Diagram of the application of *GenDeX* in the concept drift example. It shows that for each CD, the decoder of the model in operation is used to generate the synthetic data required for the update.

The objective of this experiment is to compare how the model adapts to each instance of concept drift (CD) and how this adaptation affects the reconstruction of the previous data distribution when using *GenDeX* versus not using it. The application of *GenDeX* is illustrated in Figure 3.12, while the non-application scenario involves fine-tuning the model using only the data from each CD without incorporating synthetic data.

Figure 3.13 presents the results of this comparison for the affected time series.

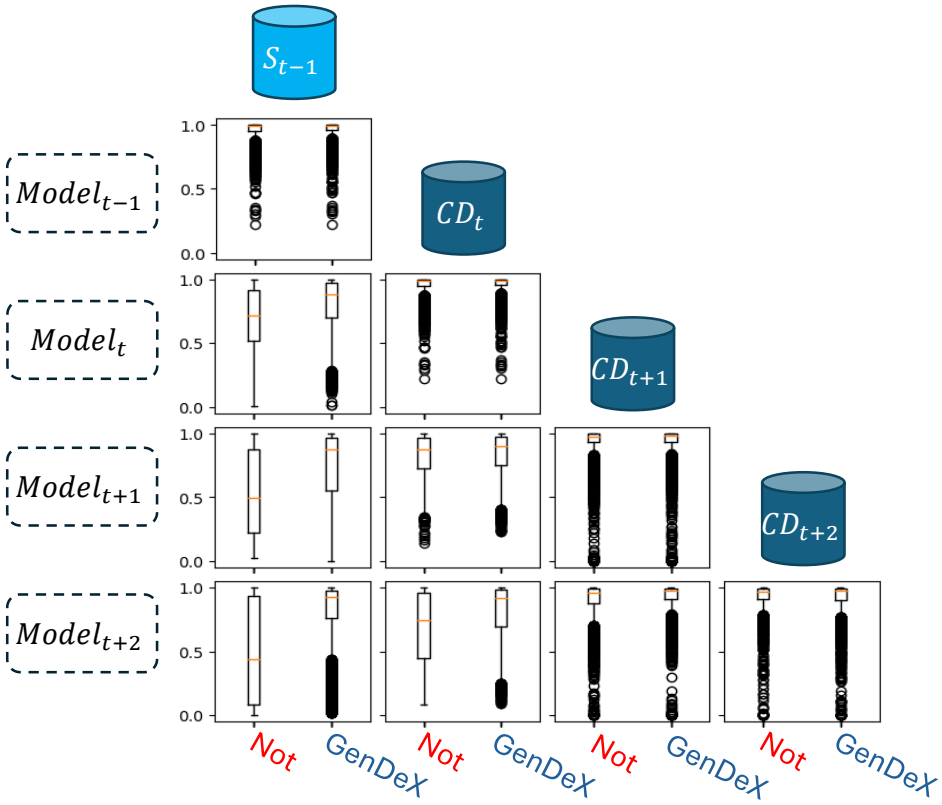


Figure 3.13: Boxplot comparison of squared z-score exponent values across different data distributions and models. Columns indicate the evaluation data distribution, rows indicate the evaluated model. Left boxplot (*Not*) shows results without *GenDeX*, and right boxplot (*GenDeX*) shows results with *GenDeX* applied. Values near 1 indicate better reconstruction quality.

The columns represent the data distributions used for evaluation, while the rows correspond to the models being evaluated. Each cell contains a boxplot illustrating the distribution of squared z-score exponent values, where values close to 1 indicate high reconstruction quality, and values near zero indicate poor reconstruction. In each box, the left boxplot corresponds to results without *GenDeX* (labeled *Not*), and the right boxplot represents results with *GenDeX* applied.

$$z\text{-}score = \frac{|\mathbf{x} - \mu_{\mathbf{x}}|}{\sigma_{\mathbf{x}}} \tag{3.1}$$

The diagonal of the figure shows the results of models fine-tuned and evaluated on the same data distribution used for the update. As observed, both approaches achieve good performance in this scenario. This demonstrates that the combination of synthetic data generated from the previous decoder, together with the CD data in the case of *GenDeX*, does not negatively impact performance on the new data compared to a model fine-tuned exclusively with the CD data.

On the other hand, to assess the impact of catastrophic forgetting, the key observations lie in the first two columns, corresponding to the base distribution S_{t-1} and the first CD (CD_t). Here, models fine-tuned on subsequent CDs are evaluated on the original and earlier data distributions. It is evident that performance degrades for both approaches as models continue to be updated. However, comparing the boxplots reveals that the degradation is more pronounced when *GenDeX* is not used (left boxplots). In contrast, models using *GenDeX* better preserve performance on the original data distribution and earlier CDs during continual updates.

GenDeX Adaptation to Domain Changes

The next experiment aims to demonstrate how *GenDeX* performs when handling domain changes, specifically in scenarios where the base model needs to be updated to incorporate new time series for monitoring. Using the TELCO dataset, we trained a base model with the first six time series, $\mathrm{TS}_{1..,6}$, designed to detect anomalies within this system of six series. The remaining time series were treated as new data sources to be integrated into the base model, one at a time. For these new series, we utilized the validation set to simulate a realistic scenario where only a limited amount of data is available for model updates. The process of applying *GenDeX* in this context is illustrated in the diagram in Figure 3.14.

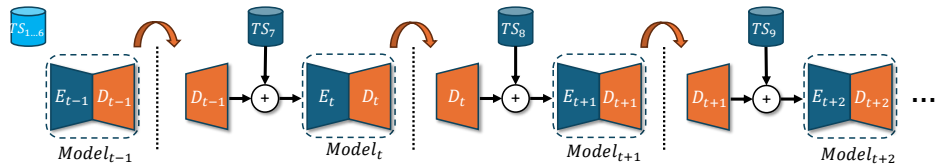


Figure 3.14: Diagram of the application of GenDeX in the domain change example. It illustrates how, for each time-series incorporation, the decoder of the model in operation is used to generate the synthetic data required for model updating.

Regarding the synthetic data generation process, given its speed and scalability, we could generate as many synthetic samples as needed. However, for consistency and comparability, we generated a number of synthetic samples equal to the number of real samples multiplied by the number of time series already present in the model prior to the update.

For comparison, we also evaluated a baseline approach where fine-tuning was conducted using only the data from the newly introduced time series, without incorporating any synthetic data generated by *GenDeX*.

Figure 3.15 presents the MSE results for the reconstruction of the time-series originally present in the base model $Model_{t-1}$, highlighting how the reconstruction error evolves as new time-series are incorporated through standard fine-tuning (left) and with *GenDeX* (right). The first row shows the results for TS_1 and TS_2. The blue boxplot represents the reconstruction error using $Model_{t-1}$, while the subsequent boxplots (from left to right) represent the error evolution after incrementally adding new time-series, from TS_7 to TS_{11}.

Without *GenDeX* (left), the reconstruction error increases as more time series are added, indicating a degradation in performance for the original series TS_1 and TS_2. Conversely, when using *GenDeX* (right), the error decreases, demonstrating improved performance after the update. For TS_3 (second row), the incorporation of new time series has minimal impact on reconstruction quality in both approaches, as the error remains stable across the boxplots. A similar trend is observed for TS_4 without *GenDeX*; however, when *GenDeX* is applied, the error decreases compared to the initial boxplot. The third row mirrors the behavior of the first, where the use of *GenDeX* consistently prevents performance degradation and, in

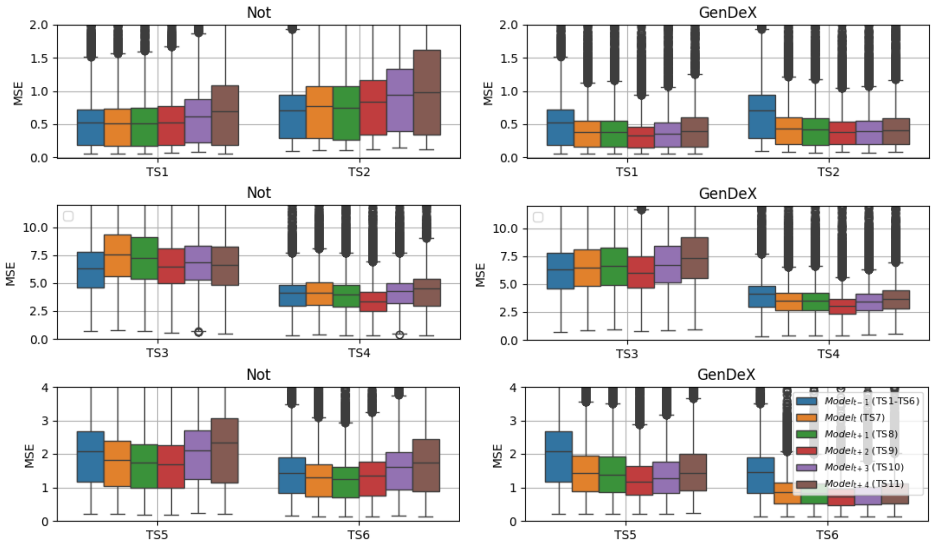


Figure 3.15: Results of the MSE over the series $TS_{1..,6}$, used to train the base model prior to the domain changes. The results without *GenDeX* (*Not*) are shown on the left, and with *GenDeX* on the right. The blue boxplot represents the values of the base model, while the remaining boxplots (from left to right) show the results after updating the model with TS_7 through TS_{11}.

some cases, even improves reconstruction quality.

These results demonstrate that for this example, *GenDeX* effectively mitigates the risk of catastrophic forgetting when integrating new time-series, preserving or even improved the reconstruction quality for previously learned data while supporting continuous model updates.

Another important consideration when applying a continual learning method like *GenDeX* is whether it could interfere with the performance of the newly incorporated information. On one hand, fine-tuning modifies the model specifically to adapt to the new data, while on the other, *GenDeX* mixes the new information with previously learned knowledge to preserve prior performance.

In Figure 3.16, we present the MSE results for the reconstruction of the time-series $\mathrm{TS}_{7..,11}$, which were sequentially added to the base model as new time-series. Regarding catastrophic forgetting, the results are consistent with those observed for the previously included series ($TS_{1..,6}$), where *GenDeX* effectively mitigates performance degradation—except for a minor deviation in TS_9.

Additionally, when examining the first boxplot for each time-series (representing performance immediately after its incorporation) and comparing results with and without *GenDeX*, the performance remains comparable. This indicates that the application of *GenDeX* does not compromise the model's ability to effectively integrate new information while still preserving the performance on previously learned data.

While in both examples—the Concept Drift and Domain Change problem—the application of *GenDeX* clearly demonstrates its ability to mitigate the catastrophic forgetting problem without losing performance on the newly incorporated information regarding reconstruction, which is the foundation for the anomaly detection. Consequently, the next logical step is to evaluate solely the anomaly detection performance. For this evaluation, we specifically focus on the last example because it utilizes the original TELCO dataset without alterations and includes the

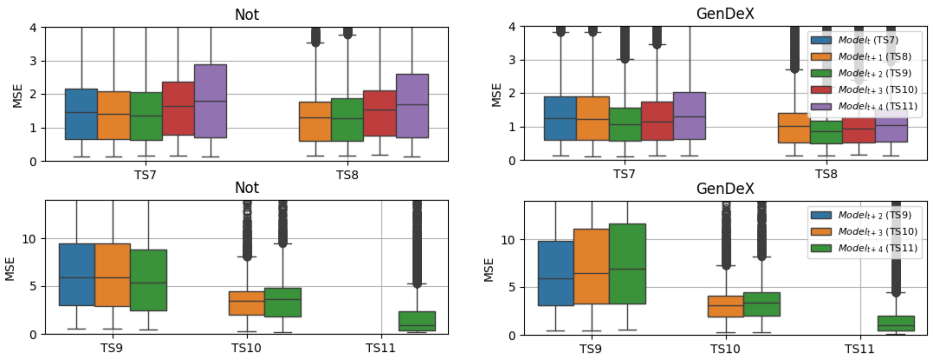


Figure 3.16: Results of the MSE over the series $TS_{7..,11}$, which were incrementally incorporated into the model using fine-tuning. The results without GenDeX ("Not") are shown on the left, while those with GenDeX are shown on the right. Each time-series starts with the model's performance immediately after its incorporation (first boxplot for each series), while the subsequent boxplots to the right represent the reconstruction error for the same series after the addition of further time-series.

corresponding labels.

To begin, we fix the operational points (α's) for $TS_{1..,6}$ using $Model_{t-1}$ on a validation set. Following this, we evaluate the subsequent models—after incorporating their respective series—on the test set of $TS_{1..,6}$. As shown in Figure 3.17, the $F1_r$ and the area under the curve of P_rR_r (PR_AUC_r) values are presented in a bar graph format. In line with Figure 3.15, the blue bar represents the performance of $Model_{t-1}$, whereas the subsequent bars illustrate the performance of $Model_{t,...t+4}$.

If we analyze the results series by series, starting with TS1, we observe that, regardless of whether *GenDeX* is applied, the performance in both metrics, $F1_r$ and PR_AUC_r, deteriorates with the incorporation of most of the new time-series. For TS2, when the last two time-series, TS10 and TS11, are incorporated, the $F1_r$ decreases when only fine-tuning is used. However, with *GenDeX*, the value increases consistently for all incorporations. In the case of TS_3, the performance remains unchanged for both approaches. For TS_4, a clear improvement in $F1_r$ is observed with the use of *GenDeX*, although the opposite occurs with PR_AUC_r. TS_5, like TS_3, shows identical performance for both approaches. Finally, for TS_6, similar to TS_2, there is a noticeable improvement with the use of *GenDeX*. In conclusion, for this example, the anomaly detection performance does not exhibit substantial degradation when fine-tuning is applied to new time-series, which *GenDeX* could improve. This shows that, for this example, the degradation in the model's reconstruction is not significant enough to affect anomaly detection performance, though this does not rule out the possibility that it could happen in other cases.

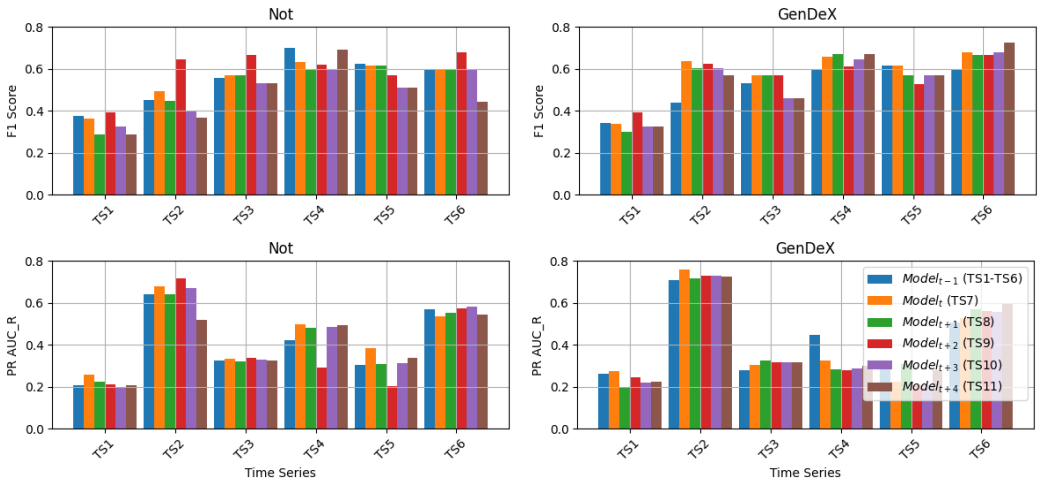


Figure 3.17: Results of the $F1_r$ and PR_AUC_r over the series $TS_{1..,6}$, used to train the base model prior to the domain changes. The results without *GenDeX* (*Not*) are shown on the left, and with *GenDeX* on the right. The blue boxplot represents the values of the base model, while the remaining boxplots (from left to right) show the results after updating the model with TS_7 through TS_{11}.

3.4. Conclusions

DC-VAE is a promising approach for anomaly detection in time-series data. However, like other learning-based methods, it requires retraining when faced with concept drift (adapting to continually evolving data) or domain changes (incorporating new time series). To address these challenges, we have extended *DC-VAE* into a continual learning framework, leveraging the generative AI capabilities of the underlying model.

Through *GenDeX*, *DC-VAE* can be retrained efficiently without requiring access to historical time-series data, maintaining performance while mitigating the effects of catastrophic forgetting. The key idea behind *GenDeX* is that *DC-VAE* can continually enhance its tracking and baselining capabilities as it encounters new data with different statistical characteristics, thereby improving its generalization and anomaly detection performance over time.

In this chapter, we investigate the generative capabilities of *DC-VAE*, exploring its latent space and demonstrating how it can be harnessed to generate synthetic time-series data. Using real ISP measurements, we show that *DC-VAE* can generate synthetic time-series samples that accurately replicate the behavior and trends of the original data used for training, drawing these samples from simple Gaussian noise.

When evaluating reconstruction performance in concept drift and domain adaptation tasks, *GenDeX* outperforms simple fine-tuning. It effectively addresses the limitations of catastrophic forgetting, enables retraining without access to past data, and maintains strong performance when incorporating new information.

Chapter 4

Foundation Models for Time-Serie Anomaly Detection

In this chapter, we focus on devising a Generative AI model capable of matching or even surpassing the performance of conventional time-series modeling methods without the need for training on the specific target dataset - a concept known as Zero-Shot Learning (ZSL). ZSL is a problem setup in deep learning where, at test time, a learner observes samples from classes which were not observed during training, and needs to predict the class that they belong to. The ZSL concept is powerful and appealing for anomaly detection applications, and such a foundational model could be utilized with limited, or even without specific fine-tuning on the downstream data typically used by other models. The zero-shot approach offers several inherent advantages: firstly, it simplifies the application of the model for time-series modeling, eliminating the requirement for specialized knowledge of fine-tuning techniques and the significant computational resources associated with them; secondly, it naturally aligns with scenarios characterized by limited data availability, where training or fine-tuning data is limited; lastly, by harnessing the comprehensive pattern extrapolation capabilities of extensively pre-trained models, it circumvents the substantial time, effort, and domain-specific expertise typically demanded for crafting dedicated time-series models.

We therefore investigate if a model pre-trained on multiple time-series data can learn temporal patterns useful for accurate reconstruction on previously unseen time-series. For doing so, we use as starting point our former *DC-VAE* model. VAEs are generative AI models that learn the underlying distribution of the data and can generate new samples from this distribution. In the context of time-series data, VAEs can capture latent representations of temporal patterns and generate sequences that exhibit similar characteristics, making them powerful for generalization and ZSL. VAEs learn a low-dimensional latent space representation of the input data, which captures the underlying structure of the data in a compressed form. By learning meaningful representations, VAEs can generalize well to unseen data points that lie within the same distribution as the training data, supporting generalization to new instances. As generative models, VAEs can generate new samples from the learned latent space distribution, potentially enabling zero-shot

learning (ZSL) by producing samples that belong to unseen classes or categories without explicitly being trained on them. By sampling from the latent space, VAEs can generate diverse and realistic data points even for classes not present in the training set.

We introduce and investigate *FAE* (Foundational Auto-Encoders), a foundational generative-AI model for anomaly detection in time-series data, based on in the previouse introduced global *DC-VAE*.

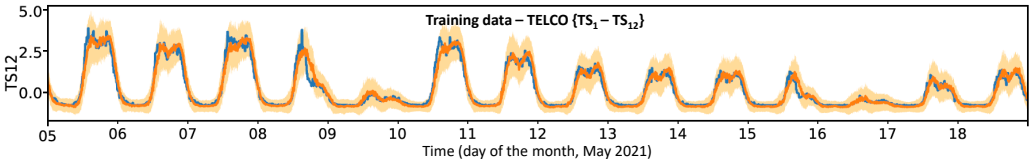
4.1. Related Work

Transformer-based models [56] are gaining popularity in recent years for time-series analysis, given their remarkable performance in large-scale settings, such as long sequence time-series forecasting (LSTF). LSTF requires capturing long-range dependencies between input and output efficiently. Earlier examples include the TFT interpretable model [57] and the MQTransformer model [58]. The Informer model [59] introduced Transformers for long sequence forecasting through sparse self-attention mechanisms. This concept has since been further refined through various forms of inductive bias and attention mechanisms in models like the Autoformer [60] and the FEDformer [61].

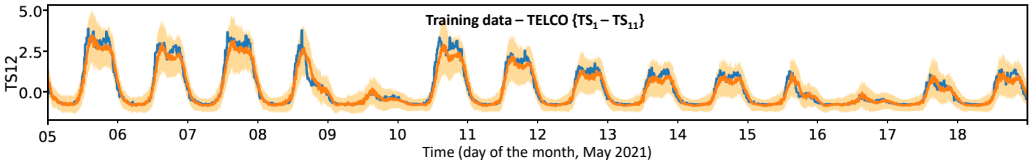
Finally, there is a recent surge in papers targeting the conception of foundation models for time-series data, capable of generating accurate predictions for diverse datasets not seen during training. The underlying concept of these models is to rely on highly expressive, large-scale architectures which are trained on millions or billions of time-series data points, coming from very diverse domains and having high heterogeneity in terms of temporal behaviors and characteristics. TimeGPT-1 [62], PromptCast [63], LLMTime [64], TimesFM [65], Lag-Llama [66], and Time-LLM [67] are all examples of novel foundation models for time-series forecasting, which target a *Zero-Shot Learning* (ZSL) application.

4.2. Preliminary Analysis of Zero-Shot Learning over TELCO

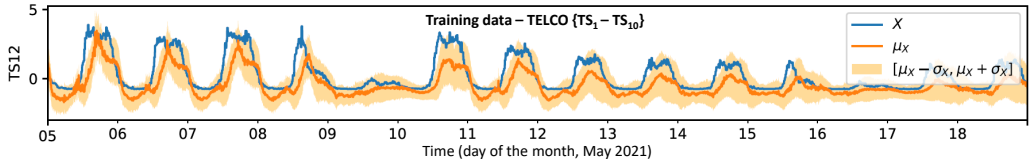
We now present *FAE* in a zero-shot setting, evaluating the model on time series that were not seen during training. We focus our analysis on TS_{12} due to its combined seasonality and distinctive temporal trend, as well as its strong correlation with TS_{11}. To conduct this evaluation, we train three *DC-VAE* models on three different TELCO training datasets, each considering a different number of time series TS_i, representing three distinct *FAE* models. The first model uses all 12 time-series – we refer to it as *full-FAE*; the second model considers a zero-shot setting for TS_{12}, with a training dataset which includes time-series TS_1 to TS_{11}, leaving out all samples from TS_{12}; given the strong temporal correlation between TS_{12} to TS_{11}, we also train a third model leaving out all samples from TS_{11} and TS_{12}, i.e., training on time-series TS_1 to TS_{10}. The *full-FAE* model



(a) *FAE* predictions for TS_{12}, with *full-FAE* (12 time-series).



(b) *FAE* predictions for TS_{12}, with *FAE* trained without TS_{12}.

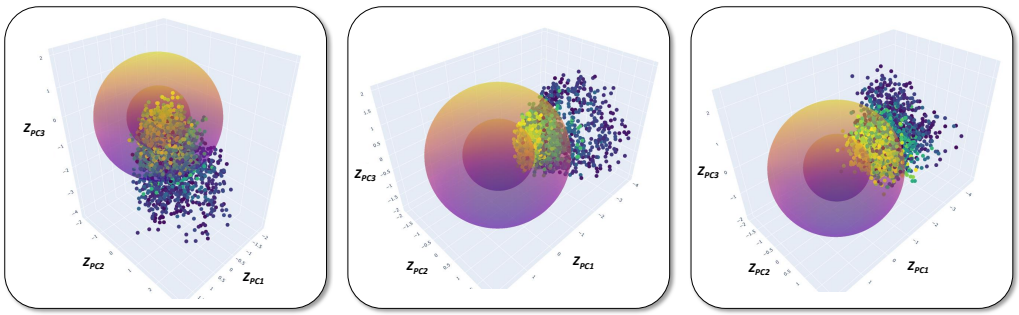


(c) *FAE* predictions for TS_{12}, with *FAE* trained without TS_{11} and TS_{12}.

Figure 4.1: Zero-shot modeling experimentation, predicting TS_{12} for two weeks in the testing dataset (May 2021). (a) *FAE* is trained on the full, 12 time-series training set – modeling performance is optimal. (b) *FAE* is trained on 11 time-series, leaving out TS_{12} – performance remains almost unchanged. (c) *FAE* is trained on 10 time-series, leaving out TS_{11} and TS_{12} – modeling performance is impacted.

mimics a situation where we pre-train with a sufficiently large and heterogeneous dataset which covers the statistical behavior of the downstream data – i.e., a model that *has seen it all*. The other two models mimic two different levels of zero-shot learning: the former represents a pure zero-shot setting for TS_{12}, where the pre-trained model has nevertheless observed a similar statistical behavior in a different time-series, i.e., TS_{11} – in particular, it has seen both the seasonality and the monthly trend behaviors; the latter represents a more challenging setting, where the pre-trained model has not seen the monthly trend behavior, which is not present in TS_1 to TS_{10}.

Figure 4.1 presents the prediction performance of the three models, when applied to two weeks of TS_{12} samples. In Figure 4.1(a), the modeling performance for *full-FAE* is optimal, as it can properly track the different behaviors and patterns in the time-series. A similar performance is observed in Figure 4.1(b) for the second model, which learns the characteristics of TS_{12} at training time, from TS_{11}. Not surprisingly, the performance of the third model in Figure 4.1(c) is significantly worse than for the other two models, given the lack of a similar temporal pattern in the training data. To some extent, there is an identification with the patterns observed in time-series TS_1 – note how the daily sharp peaks are exacerbated – which is coherent with their close representations in the latent space (cf.



(a) (b) (c)

Figure 4.2: Latent space representation for TS_{12}, with different FAE pre-trained models. Colors represent the different days of the analysis window, going from day 5 in purple to day 19 in yellow, for the full month of March 2021. (a) is the result for the Full-FAE, (b) for FAE trained without TS_{11}, and (c) for FAE trained without TS_{11} and TS_{12}

Figure 3.8). Nevertheless, it somehow manages to capture and track the monthly downtrend, even without previous evidence of it.

To conclude, Figure 4.2 shows the latent representation of the TS_{12} test samples for the three *FAE* pre-trained models, where colors represent the different days of the analysis window, going from day 5 in purple to day 19 in yellow. *Full-FAE* encoded samples form a sort of cone in the latent space in Figure 4.2(a), where the base (purple and blue) represents the first days of the month and the tip – pointing towards the center of the latent space – represents the days towards end of the month. Figure 4.2(b) shows a similar cone-shape for the samples encoded by the second pre-trained model, but this time, the tip of the cone has moved away from the center. Finally, while Figure 4.2(c) shows a similar distribution of samples, with yellow and clearer colors closer to the center and darker ones at the periphery of the central sphere, the regular cone-shape observed before is no longer well-defined, evidencing a different mapping behavior of the model.

4.3. The pre-trained model for *FAE*

To achieve the desired diversity of time-series data necessary for obtaining a robust pre-trained model—the foundation of our foundational model—we selected a dataset that meets these criteria: UCR'21, presented in [68].

4.3.1. The UCR dataset

The UCR'21 dataset includes 250 time-series from various sources, such as electric power systems, medical applications, telecommunications, and more. This dataset reflects over 20 years of work surveying time-series anomaly detection literature and consolidating datasets into a comprehensive collection.

Figure 4.3 illustrates 12 different examples of time-series from UCR. As observed, the dataset showcases significant variability in shapes and frequencies. This

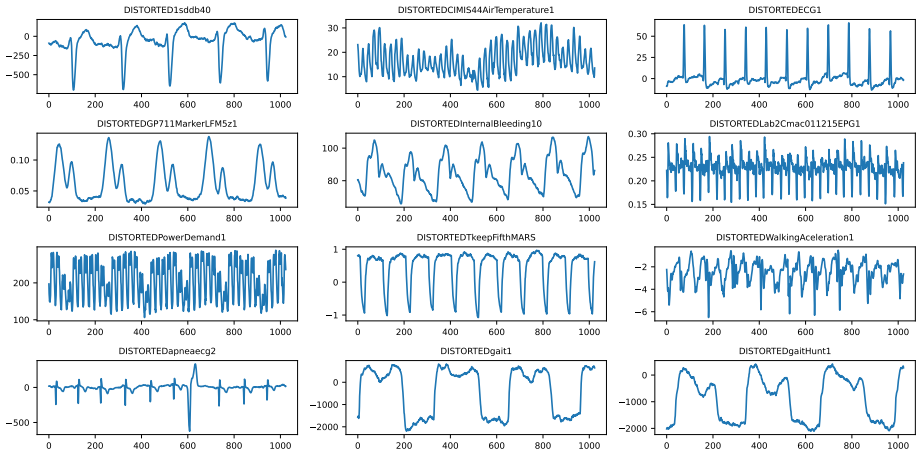


Figure 4.3: Twelve different examples of time-series from the UCR '21 dataset. Each plot displays the first 1024 values, highlighting the diversity of frequencies and periods present in the dataset.

diversity is further emphasized as each plot represents the first 1024 samples of a series, revealing different numbers of periods across the examples.

This diversity, combined with the size of the UCR dataset—over 5 million data points in the training set alone—posed a significant challenge at the beginning. It was the first time we applied an architecture like the global *DC-VAE* to a dataset with such characteristics. In comparison, TELCO, with fewer than 100,000 data points in its training dataset, offered a much more uniform structure: all time series originated from the same system, shared the same sampling frequency (5 minutes), and followed the same daily seasonality.

Due to these differences, it became necessary to enhance our architecture with additional tools to effectively handle the complexity of the UCR dataset.

4.3.2. Giving flexibility to the architecture

Inspired by the WaveNet architecture [6], which influenced the use of dilated convolutions in *DC-VAE*, we introduced gates, residual connections, and skip connections into the model. These additions were carefully chosen for two main reasons.

The first and most important reason lies in the diversity of sampling frequencies and periods present in the UCR'21 dataset. The combination of gates, residual connections, and skip connections enhances the model's flexibility. Gates allow the model to suppress specific parts of the hidden layer outputs, passing only the relevant information along the time and filter axes. Residual connections enable the model to preserve the input information from one layer to the next, providing the possibility to skip certain layers that may not be relevant for a given input. This is particularly useful in our architecture with dilated convolutions, where each layer captures correlations at different temporal scales. Finally, skip connections summarize information from all layer outputs, benefiting both the encoder and

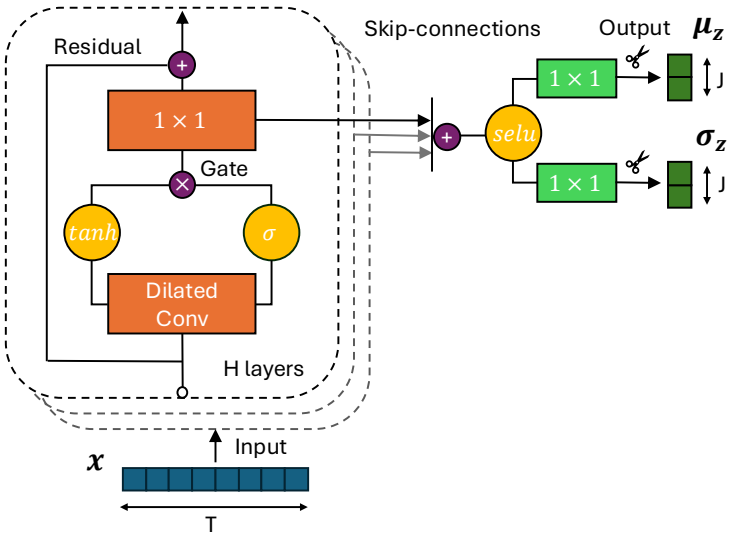


Figure 4.4: Encoder architecture of the *FAE*. This diagram shows the new components incorporated into the previously presented *DC-VAE*. Unlike the latter, each dilated convolutional layer includes a gate, and residual connections are added. At the end, a skip connection summarizes the outputs of each residual block. The decoder is symmetric, so the same diagram can represent the full architecture.

decoder. In Figure 4.4, we can observe a diagram of the architecture connections for the encoder. As mentioned before, this is similar to the structure presented in [6], but our version includes some modifications in the output. The *Dilated Conv* block represents a dilated convolution layer, which is the same as the hidden layer shown in Figures 2.18 and 2.4.

The second reason is that these mechanisms do not introduce many additional trainable parameters. For example, a gate is simply the element-wise multiplication of two different nonlinear transformations applied to the same output of the dilated convolution: a hyperbolic tangent (tanh()), which processes preliminary information to pass to the next level, and a sigmoid function ($\sigma()$), which acts as a gating mechanism. Similarly, a residual connection is a simple sum where the layer's input is added to its output, while skip connections aggregate all layer outputs at the end of the encoder or decoder block. Additionally, these mechanisms accelerate convergence, helping the model train more efficiently. The only part where we add more parameters is in the 1×1 convolution layers, which were added to maintain the same dimensions between the input and output of the main block to enable the residual connection sum. By incorporating these techniques, we maintain a lightweight and scalable architecture, aligning with our goal of designing an efficient model.

4.3.3. Pre-trained model

For the pre-trained model on UCR'21, we first performed a hyperparameter search to identify the best model configuration. Notably, the modifications to the

Table 4.1: Grid of hyperparameters used in the model calibration of the pre-trained model for *FAE*.

Hyperparameter	**Grid Search Ranges**	**Best**
T - sequence length	$\{128 - 2048\}$, step=128	128
J - latent dimension	$\{64 - 3T/4\}$, step=32	96
γ - learning rate	$\{1e^{-5} - 5e^{-6}\}$	$1{,}8e^{-5}$
m - mini-batch size	$[32, 64, 128]$	128
U - number of filters	$\{64 - 512\}$, step=16	256

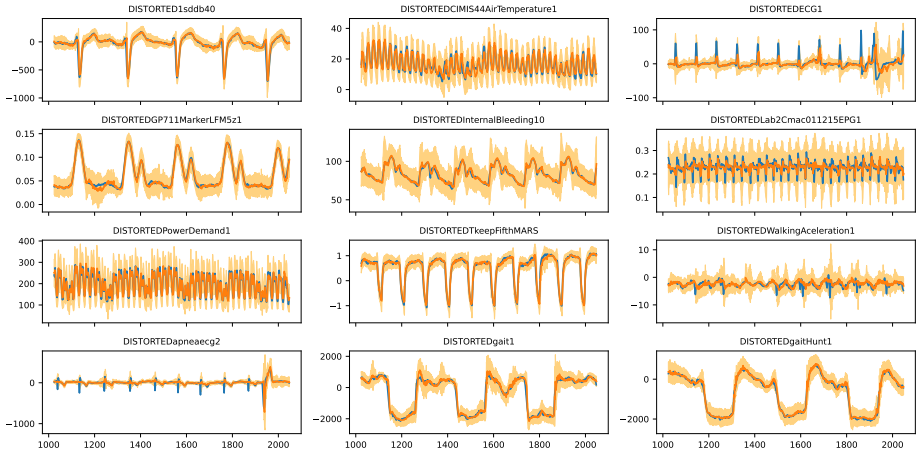


Figure 4.5: Reconstruction examples over the values of the same time-series from the UCR dataset as in Figure 4.3, but using the test set.

architecture did not introduce any additional hyperparameters. In Table 4.1, we present the search grid and the selected values for the pre-trained model.

Due to the increase in the number of filters per layer ($U = 256$) and the size of the latent space ($J = 96$) compared to previous configurations, this model has 2.6 million trainable parameters, significantly fewer than the more than 5 million samples in the UCR'21 training set.

A portion of the reconstruction results on the test set can be observed in Figure 4.5. This figure demonstrates that, despite the variability of the time series, the model was able to learn a well-structured latent space representation. As a result, the decoder successfully generates accurate reconstructions of the diverse behaviors present in the dataset.

4.4. Zero-Shot Evaluation on TELCO

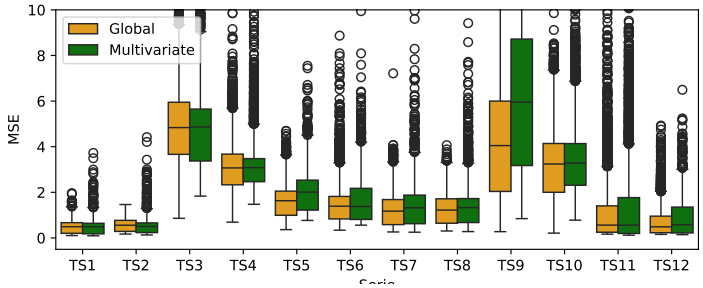
After pre-training the *FAE* model, we evaluated its foundational properties on the TELCO dataset. We expect that the features learned from the diverse time series in UCR'21 will help the model encode the samples in a way that

represents their characteristics, even if they were not seen before, allowing the decoder to reconstruct the samples accurately. As in previous experiments, we began by comparing reconstruction metrics.

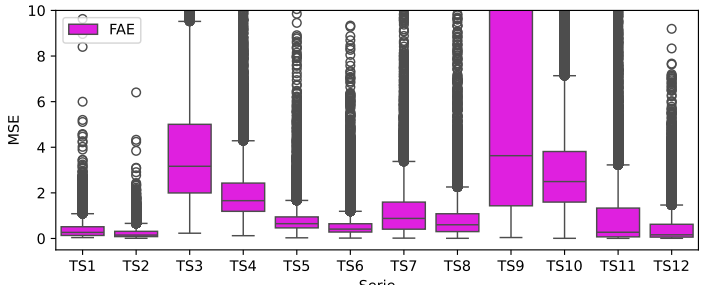
In Figure 4.6(a), we compare the MSE of a global *DC-VAE* and a multivariate *DC-VAE* trained directly on TELCO (previously shown in Figure 2.19(b)). Meanwhile, Figure 4.6(b) presents the reconstruction performance of *FAE* in a zero-shot setting. Notably, the boxplots for the *DC-VAE* models and *FAE* are comparable, except for TS_9, where the MSE boxplot is wider for the *FAE* reconstruction.

Additionally, in Figure 4.7, we visualize the zero-shot reconstruction for all time series in TELCO. As observed, the values of μ_x consistently track the real series, demonstrating performance comparable to previous reconstruction examples with *DC-VAE* models trained on TELCO (e.g., Figures 2.1, 2.5, 2.6, and 2.10). However, when examining the σ_x values, we notice noisier and less meaningful results compared to previous cases, which poses a problem for anomaly detection.

To compare the anomaly detection performance of zero-shot *FAE* against multivariate and global *DC-VAE* models, we present the results on the TELCO test set in Table 4.3. As shown in the *FAE* column, the majority of the $F1_r$ values show worse performance compared to the *DC-VAE* models, except for TS_4 and



(a) Multivariate and global DC-VAE reconstruction results.



(b) FAE zero-shot reconstruction results.

Figure 4.6: MSE results for reconstruction predictions: (a) Results for multivariate and global *DC-VAE* models trained on TELCO. (b) Results for zero-shot inference using the pre-trained *FAE*.

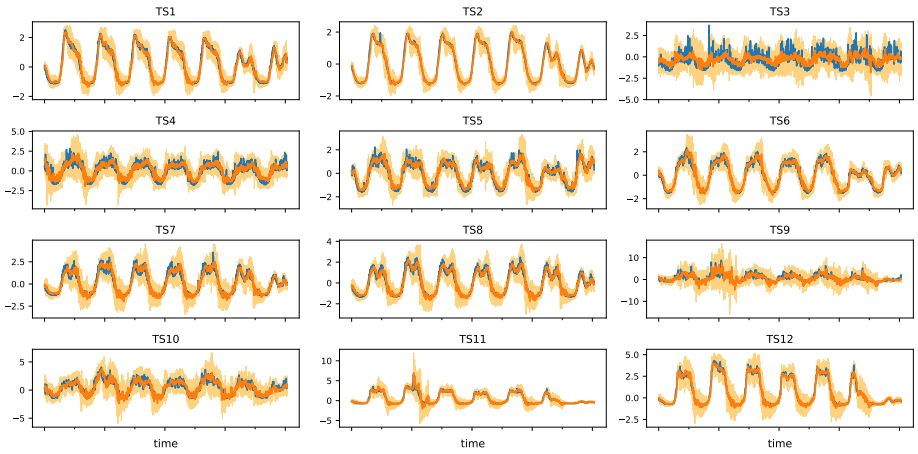


Figure 4.7: Reconstruction examples over the TELCO dataset using the *FAE model*.

Table 4.2: *DC-VAE* with new results obtained using *FAE*. For the latter, two columns are presented: the first, labeled simply as *FAE*, shows results calculated in the same manner as for the *DC-VAE* models. In the FAE_{μ_x} column, the z-score calculation was performed using only the values of μ_x.

	DC-VAE			***global DC-VAE***			***FAE***			$\boldsymbol{FAE_{\mu_x}}$		
TS ID	R_r	P_r	$F1_r$	R_r	P_r	$F1_r$	R_r	P_r	$F1_r$	R_r	P_r	$F1_r$
TS_1	58	71	**64**	29	60	39	14	67	24	21	100	41
TS_2	74	20	67	71	80	**75**	8	100	15	41	100	59
TS_3	86	47	**60**	71	29	42	43	30	35	71	45	55
TS_4	63	21	32	50	35	41	50	50	**50**	38	56	45
TS_5	75	50	60	50	67	57	38	38	38	63	56	59
TS_6	57	83	68	57	100	**72**	29	100	44	71	50	59
TS_7	72	90	**80**	50	71	59	40	44	42	50	100	67
TS_8	44	80	**57**	38	50	43	25	40	31	50	80	**62**
TS_9	17	11	**13**	0	0	0	17	19	18	17	33	**22**
TS_{10}	52	59	**55**	60	24	34	35	15	21	30	26	28
TS_{11}	100	25	**40**	100	21	35	33	8	13	33	7	12
TS_{12}	100	11	22	100	100	**100**	0	0	0	0	0	0
mean	67	47	**52**	56	53	50	28	43	28	40	54	41
median	68	49	**59**	54	55	42	31	39	27	40	53	50

TS_9, where *FAE* performs better. This is mainly due to a decrease in $Recall_r$ values.

As observed in the reconstruction evaluation, the σ_x values do not appear to be good representations of the real data, and as previously mentioned, this affects the anomaly detection (AD) performance. Although σ_x is a fundamental part of our approach to creating a normal environment for AD, we decided to further investigate the performance using only the μ_x values for scoring. In this approach, the σ_x values were excluded, and the z-score was calculated using a rolling mean and standard deviation with a window size equal to T, based on the absolute error between x and μ_x.

First, we calculated the absolute error for each time series ($e = |x - \mu_x|$). Then, for each time point t in the series e, we computed the mean and standard deviation, denoted as μ_{e_t} and σ_{e_t}, respectively, using the previous T values. This is shown in Equations 4.2 and 4.3:

$$\mu_{e_t} = \frac{1}{T}\sum_{j=0}^{T-1} e_{t-j} \tag{4.1}$$

$$\sigma_{e_t} = \sqrt{\frac{1}{T}\sum_{j=0}^{T-1}(e_{t-j} - \mu_{e_t})^2} \tag{4.2}$$

$$\textbf{z-score}_t = \frac{|e_t - \mu e_t|}{\sigma_{e_t}} \tag{4.3}$$

For each value of the time series x, excluding the first T values, we calculated the z-score as shown in Equation 4.3.

As with the previous scoring method, we first selected the α values using the validation set and then performed detections on the test set. The results are shown in the FAE_{μ_x} column.

As observed, this adjustment significantly improves performance. The majority of the $F1_r$ values increase compared to the original *FAE*, except for TS4, TS11, and TS_{12}, which remain the same. When observing the mean and median in the last rows, both are better than the *FAE* values, with the median even showing an improvement of 8 points over the global *DC-VAE*.

Although it does not surpass the *DC-VAE* models trained directly on TELCO in most cases, the results are comparable.

4.5. Zero-Shot Learning against *Lag-Llama*

To compare the *FAE* with another foundation model, we chose a recently proposed algorithm, *Lag-Llama* [66]. *Lag-Llama* is a general-purpose foundation model for univariate probabilistic time series forecasting, based on a decoder-only transformer architecture that uses lags as covariates.

Lag-Llama is pretrained on a large corpus of diverse time series data from several domains and demonstrates strong zero-shot generalization capabilities compared to a wide range of forecasting models on downstream datasets across domains. Moreover, when fine-tuned on relatively small fractions of such previously unseen datasets, *Lag-Llama* achieves state-of-the-art performance, outperforming prior deep learning approaches, and emerging as the best general-purpose model on average.

Unlike *FAE*, which is designed for anomaly detection, *Lag-Llama* is a forecasting method. To use it for anomaly detection, we configured it with a context size of 128 values to make it comparable with the T values used in *FAE*. Additionally, we set the method to output five predictions for the next value and return the mean of these predictions. Using this configuration, we performed zero-shot inference on

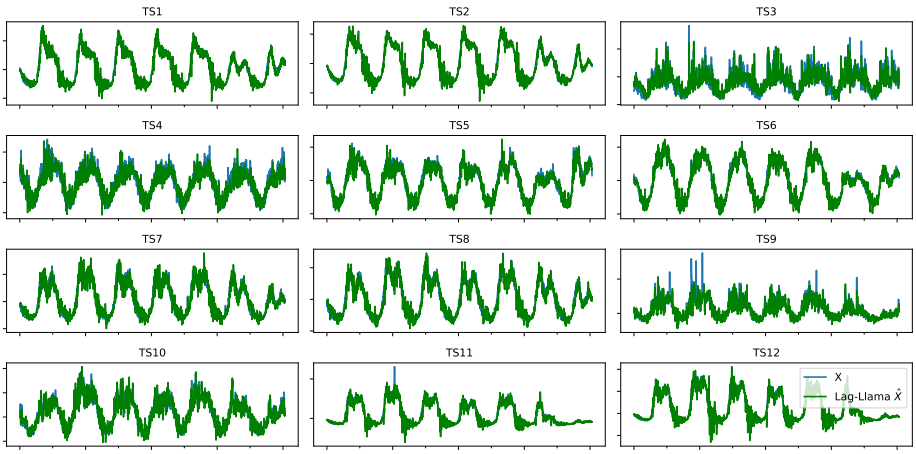


Figure 4.8: Reconstruction examples on the TELCO dataset using zero-shot inference with the *Lag-Llama* model.

the concatenated train and validation sets of TELCO to calculate the α values for each time series, and then applied the model to the test set. One key observation is that *Lag-Llama* required more than 20 minutes to generate predictions for each set, whereas *FAE* produced its predictions in under 1 minute.

To illustrate the predictions compared to the real values, Figure 4.8 presents an example from a portion of the TELCO test set. As observed, the zero-shot predictions closely follow the real values; however, they appear noisier than the μ_x values shown in Figure 4.7.

Then, applying the same rolling z-score used for *FAE*μ_x, we evaluated the performance of zero-shot anomaly detection using *Lag-Llama*. In Table 4.3, we

Table 4.3: Comparison of zero-shot anomaly detection on TELCO using *FAE*, *FAE*$_{\mu_x}$, and *Lag-Llama*. The best $F1_r$ values are highlighted.

	FAE			***FAE*$_{\mu_x}$**			***Lag-Llama***		
TS ID	R_r	P_r	$F1_r$	R_r	P_r	$F1_r$	R_r	P_r	$F1_r$
TS_1	14	67	24	21	100	41	29	100	**44**
TS_2	8	100	15	41	100	**59**	4	13	6
TS_3	43	30	35	71	45	**55**	29	29	29
TS_4	50	50	**50**	38	56	45	50	19	27
TS_5	38	38	38	63	56	**59**	25	67	36
TS_6	29	100	44	71	50	**59**	29	67	40
TS_7	40	44	42	50	100	**67**	50	50	50
TS_8	25	40	31	50	80	**62**	25	33	29
TS_9	17	19	18	17	33	**22**	11	12	11
TS_{10}	35	15	21	30	26	**28**	15	20	17
TS_{11}	33	8	13	33	7	12	67	9	**15**
TS_{12}	0	0	0	0	0	0	0	0	0
mean	28	43	28	40	54	**41**	27	35	25
median	31	39	27	40	53	**50**	27	24	28

compare the results of *FAE* and *FAE*μ_x with those of *Lag-Llama*. As observed, the performance of *Lag-Llama* is comparable to that of *FAE*, but for the majority of the time series, *FAE*$_{\mu_x}$ performs better, except for TS1 and TS11, where *Lag-Llama* is superior. This shows that our method, *FAE*, although it does not achieve a good estimation of σ_x to be compared with a *DC-VAE* model trained on TELCO, can achieve a better representation of μ_x than the more complex and sophisticated pre-trained method, *Lag-Llama*, for better zero-shot anomaly detection.

4.6. Conclusions

We have introduced *FAE*, a novel approach for time-series modeling, inspired by the success of large pre-trained foundation models in different domains. *FAE* focuses on detecting anomalies in univariate time-series data, leveraging *DC-VAE* for pre-training on large-scale, heterogeneous time-series datasets. This pre-training potentially enables it to model and track a baseline for normal operation, even on previously unseen datasets.

The assessment of *FAE*'s performance has shown promising results. In particular, we have demonstrated *FAE*'s ability to capture and distinguish various temporal behaviors within the training time-series, highlighting its capacity to model large and heterogeneous datasets effectively.

Our exploration extended to the zero-shot learning scenario, where *FAE*'s performance on unseen time-series was evaluated. To enhance flexibility in handling different sampling frequencies and seasonality patterns, we incorporated additional architectural components into *DC-VAE*, including gating mechanisms, residual connections, and skip connections. These modifications allow *FAE* to adapt more effectively while ensuring fast convergence without compromising the model's efficiency.

We trained a diverse pre-trained model for *FAE* using the UCR'21 dataset, demonstrating its ability to track various behaviors and patterns in time-series data. The zero-shot capability of this model was tested on TELCO, comparing its reconstruction and anomaly detection performance against models selected through hyperparameter search and extensively trained on the target data distribution. Notably, *FAE*, pre-trained on a diverse time-series distribution that excluded TELCO data, was able to track TELCO's time-series effectively using μ_x. However, unlike the *DC-VAE* model, *FAE* exhibited noisier σ_x values, which negatively impacted anomaly detection performance. By adopting a new scoring approach that leveraged the strong performance of μ_x, *FAE* achieved results approximately comparable to those of the *DC-VAE* models.

To further benchmark the zero-shot detection capabilities of our model, we implemented an anomaly detector based on the time-series foundation forecasting model *Lag-Llama*. While *Lag-Llama* successfully tracked TELCO time-series, its predictions were noisier compared to *FAE*'s μ_x values, resulting in poorer overall performance compared to our *FAE*.

These findings underscore *FAE*'s potential as a viable foundation model for time-series analysis.

Chapter 5

Concluding Remarks

In this work, we have presented —and progressively extended— the *DC-VAE* framework, demonstrating its effectiveness as an anomaly detection method for time-series data through three key developments: the original *DC-VAE*, its continual learning extension *GenDeX*, and the foundational model *FAE*.

The original *DC-VAE* combines the strengths of variational autoencoders with a fully convolutional architecture featuring dilations. This design ensures efficient training, stable convergence, and near real-time anomaly detection capabilities. Its simplicity, characterized by a small set of hyperparameters and Gaussian-based thresholding, facilitates both supervised and unsupervised training. When evaluated on real-world datasets such as TELCO and SWaT, *DC-VAE* consistently outperformed traditional statistical models, machine learning approaches, and even other deep learning-based methods, showcasing its robustness in diverse operational environments.

To address the challenges posed by concept drift and domain adaptation, we introduced *GenDeX*, a continual learning framework built on top of *DC-VAE*. Leveraging the model's generative capabilities, *GenDeX* enables efficient retraining without the need for historical data, mitigating catastrophic forgetting while maintaining strong anomaly detection performance. Through synthetic data generation and robust adaptation to evolving statistical characteristics, *GenDeX* demonstrated superior performance over conventional fine-tuning techniques, particularly in dynamic environments.

Further extending the *DC-VAE* framework, we developed *FAE*, inspired by the success of foundation models in other domains. *FAE* applies pretrained knowledge from large, heterogeneous time-series datasets to anomaly detection tasks, even in zero-shot scenarios. Its architecture, enhanced with gating mechanisms and residual connections, facilitates fast convergence and adaptability to new data distributions. Despite some challenges in modeling uncertainty σ_x for unseen datasets, *FAE* achieved competitive results in comparison to extensively fine-tuned models and outperformed state-of-the-art foundation models like Lag-Llama.

Overall, the evolution from *DC-VAE* to *GenDeX* and *FAE* illustrates a comprehensive approach to time-series anomaly detection, balancing model simplicity, adaptability, and performance. These advancements highlight the potential of com-

bining variational inference with continual learning and foundation modeling principles, paving the way for future research in scalable, interpretable, and resilient time-series analysis.

5.1. Future Directions

In this section, we outline some potential lines of future work that could be based on this research. The complete convolutional architecture allows for flexibility with respect to the window size of the input samples. Specifically, if we train a *DC-VAE* by fixing a maximum window size T, but generate a training dataset with varying window sizes, the model could be capable of producing reconstructions at different resolutions. We believe that this approach empowers the model, during inference, to detect anomalies at multiple time scales. Due to the fast inference speed of *DC-VAE*, this process could be performed efficiently. By sweeping between smaller and larger resolutions, we may be able to identify anomalies that could be difficult to detect with a single resolution.

In addition, in this work, all the proposed models rely solely on the values of the time series, without considering the information provided by timestamps. Preliminary tests were conducted, and incorporating this information resulted in highly similar predictions. This suggests that the model prioritizes the countable and finite aspects of the timestamp data over the stochastic nature of the series. However, effectively leveraging timestamp information could lead to a more robust base model, potentially reducing false positives on specific dates. Furthermore, it could provide greater control over the samples generated by the decoder. This capability would enable the model to generate complete time series, making it useful for creating different scenarios or filling gaps in missing data.

References

[1] Bryan Lim, Sercan Ö Arık, Nicolas Loeff, and Tomas Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4):1748–1764, 2021.

[2] Paul Boniol, John Paparrizos, and Themis Palpanas. An interactive dive into time-series anomaly detection. In *ICDE 2020-40th International Conference on Data Engineering*, 2024.

[3] Krzysztof Kotowski, Christoph Haskamp, Jacek Andrzejewski, Bogdan Ruszczak, Jakub Nalepa, Daniel Lakey, Peter Collins, Aybike Kolmas, Mauro Bartesaghi, Jose Martinez-Heras, et al. European space agency benchmark for anomaly detection in satellite telemetry. *arXiv preprint arXiv:2406.17826*, 2024.

[4] Tom Soderstrom, Chris Mattmann, Ian Colwell, Chris Laporte, Connor Francis, Kyle Hundman, and Valentino Constantinou. Telemanom: An extensible framework for time-series anomaly detection. 2019.

[5] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[6] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[7] Gastón García González, Sergio Martínez Tagliafico, Alicia Fernández, Gabriel Gómez, José Acuña, and Pedro Casas. TELCO – a new Multivariate Time-Series Dataset for Anomaly Detection in Mobile Networks, 2023.

[8] A. P. Mathur and N. O. Tippenhauer. SWaT: A Water Treatment Testbed for Research and Training on ICS Security. In *IEEE International Workshop on Cyber-physical Systems for Smart Water Networks (CySWater)*, pages 31–36, 2016.

[9] Ane Blázquez-García, Angel Conde, Usue Mori, and Jose A. Lozano. A review on outlier/anomaly detection in time series data. *ACM Comput. Surv.*, 54(3), April 2021.

[10] Manish Gupta, Jing Gao, Charu Aggarwal, and Jiawei Han. Outlier detection for temporal data. *Synthesis Lectures on Data Mining and Knowledge Discovery*, 5(1):1–129, 2014.

[11] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3), July 2009.

[12] Mohammad Braei and Sebastian Wagner. Anomaly Detection in Univariate Time-series: A Survey on the State-of-the-Art. *CoRR*, abs/2004.00433, 2020.

[13] D. Baessler, T. Kortus, and G. Guehring. Unsupervised anomaly detection in multivariate time series with online evolving spiking neural networks. *Machine Learning*, 111:1377–1408, 2022.

[14] Kukjin Choi, Jihun Yi, Changhwa Park, and Sungroh Yoon. Deep Learning for Anomaly Detection in Time-Series Data: Review, Analysis, and Guidelines. *IEEE Access*, 9, 2021.

[15] Guansong Pang, Chunhua Shen, Longbing Cao, and Anton Van Den Hengel. Deep learning for anomaly detection: A review. *ACM Comput. Surv.*, 54(2), March 2021.

[16] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*, volume 27, 2014.

[17] Sultan Zavrak and Murat Iskefiyeli. Anomaly-based intrusion detection from network flow features using variational autoencoder. *IEEE Access*, 8:108346–108358, 2020.

[18] Houssam Zenati, Chuan Sheng Foo, Bruno Lecouat, Gaurav Manek, and Vijay Ramaseshan Chandrasekhar. Efficient GAN-based anomaly detection. *arXiv preprint arXiv:1802.06222*, 2018.

[19] Run-Qing Chen, Guang-Hui Shi, Wanlei Zhao, and Chang-Hui Liang. A joint model for IT operation series prediction and anomaly detection. *Neurocomputing*, 448:130–139, 2021.

[20] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. In *International Conference on Learning Representations*, 2017.

[21] Dan Li, Dacheng Chen, Baihong Jin, Lei Shi, Jonathan Goh, and See-Kiong Ng. MAD-GAN: Multivariate anomaly detection for time series data with generative adversarial networks. In *International Conference on Artificial Neural Networks*, pages 703–716. Springer, 2019.

[22] Alexander Geiger, Dongyu Liu, Sarah Alnegheimish, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. TadGAN: Time series anomaly detection using generative adversarial networks. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 33–43. IEEE, 2020.

[23] Gastón García González, Pedro Casas, Alicia Fernández, and Gabriel Gómez. On the usage of generative models for network anomaly detection in multivariate time-series. *SIGMETRICS Perform. Eval. Rev.*, 48(4):49–52, may 2021.

[24] Carl Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.

[25] Diederik P Kingma and Max Welling. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392, 2019.

[26] Francesco Paolo Casale, Adrian V. Dalca, Luca Saglietti, Jennifer Listgarten, and Nicoló Fusi. Gaussian Process Prior Variational Autoencoders. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 10390–10401, 2018.

[27] Laurent Girin, Fanny Roche, Thomas Hueber, and Simon Leglaive. Notes on the use of variational autoencoders for speech and audio spectrogram modeling. In *DAFx 2019-22nd International Conference on Digital Audio Effects*, pages 1–8, 2019.

[28] Vincent Fortuin, Dmitry Baranchuk, Gunnar Rätsch, and Stephan Mandt. GP-VAE: Deep Probabilistic Time Series Imputation. In *International conference on artificial intelligence and statistics*, pages 1651–1661. PMLR, 2020.

[29] Siddharth Ramchandran, Gleb Tikhonov, Kalle Kujanpää, Miika Koskinen, and Harri Lähdesmäki. Longitudinal variational autoencoder. In *International Conference on Artificial Intelligence and Statistics*, pages 3898–3906. PMLR, 2021.

[30] Justin Bayer and Christian Osendorfer. Learning stochastic recurrent networks. In *NIPS 2014 Workshop on Advances in Variational Inference*, 2014.

[31] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. A recurrent latent variable model for sequential data. *Advances in neural information processing systems*, 28, 2015.

[32] Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2828–2837, 2019.

[33] Samira Shabanian, Devansh Arpit, Adam Trischler, and Yoshua Bengio. Variational bi-LSTMs. *arXiv preprint arXiv:1711.05717*, 2017.

[34] Zichao Yang, Zhiting Hu, Ruslan Salakhutdinov, and Taylor Berg-Kirkpatrick. Improved variational autoencoders for text modeling using dilated convolutions. In *International conference on machine learning*, pages 3881–3890. PMLR, 2017.

[35] Guokun Lai, Bohan Li, Guoqing Zheng, and Yiming Yang. Stochastic wavenet: A generative latent variable model for sequential data. *arXiv preprint arXiv:1806.06116*, 2018.

[36] Chao Meng, Xue Song Jiang, Xiu Mei Wei, and Tao Wei. A time convolutional network based outlier detection for multidimensional time series in cyber-physical-social systems. *IEEE Access*, 8:74933–74942, 2020.

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[38] Hongwei Zhang, Yuanqing Xia, Tijin Yan, and Guiyang Liu. Unsupervised anomaly detection in multivariate time series through transformer-based variational autoencoder. In *2021 33rd Chinese Control and Decision Conference (CCDC)*, pages 281–286. IEEE, 2021.

[39] Haowen Xu, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, Dan Pei, Yang Feng, et al. Unsupervised anomaly detection via variational auto-encoder for seasonal KPIs in web applications. In *Proceedings of the 2018 World Wide Web Conference*, pages 187–196, 2018.

[40] Nesime Tatbul, Tae Jun Lee, Stan Zdonik, Mejbah Alam, and Justin Gottschlich. Precision and Recall for Time Series. *32nd Conference on Neural Information Processing Systems (NeurIPS)*, 2018.

[41] N. Laptev, S. Amizadeh, and Y. Billawala. S5 - a labeled anomaly detection dataset. 2015.

[42] Alexander Lavin and Subutai Ahmad. Evaluating real-time anomaly detection algorithms–the numenta anomaly benchmark. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 38–44. IEEE, 2015.

[43] Subutai Ahmad, Alexander Lavin, Scott Purdy, and Zuha Agha. Unsupervised Real-time Anomaly Detection for Streaming Data. *Neurocomputing*, 262:134–147, 2017.

[44] Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 387–395, 2018.

[45] Renjie Wu and Eamonn Keogh. Current Time Series Anomaly Detection Benchmarks are Flawed and are Creating the Illusion of Progress. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1, 2021.

[46] J. Beirlant, Y. Goegebeur, J. Segers, and J. L. Teugels. Statistics of extremes: theory and applications. 558, 2004.

[47] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. *Advances in neural information processing systems*, 24, 2011.

[48] Arsalan Shahid, Gary White, Jaroslaw Diuwe, Alexandros Agapitos, and Owen O'Brien. SLMAD: Statistical Learning-Based Metric Anomaly Detection. In *International Conference on Service-Oriented Computing*, pages 252–263. Springer, 2020.

[49] Houssam Zenati, Chuan Sheng Foo, Bruno Lecouat, Gaurav Manek, and Vijay Ramaseshan Chandrasekhar. Efficient GAN-Based Anomaly Detection. *CoRR*, abs/1802.06222, 2018.

[50] João Gama, Indrundefined Žliobaitundefined, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *Association for Computing Machinery*, 46, 2014.

[51] Gustavo H. F. M. Oliveira, Rodolfo C. Cavalcante, George G. Cabral, Leandro L. Minku, and Adriano L. I. Oliveira. Time series forecasting in the presence of concept drift: A pso-based approach. In *2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 239–246, 2017.

[52] James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming Catastrophic Forgetting in Neural Networks. *CoRR*, abs/1612.00796, 2016.

[53] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual Learning with Deep Generative Replay. *CoRR*, abs/1705.08690, 2017.

[54] Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *CoRR*, abs/1606.04671, 2016.

[55] Anna Kuzina, Evgenii Egorov, and Evgeny Burnaev. Boovae: Boosting approach for continual learning of vae. *Advances in Neural Information Processing Systems*, 35, 2021.

[56] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[57] Bryan Lim, Sercan Arik, Nicolas Loeff, and Tomas Pfister. Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting. *International Journal of Forecasting*, 37(4):1748–1764, 2021.

[58] Kevin C. Chen, Lee Dicker, Carson Eisenach, and Dhruv Madeka. MQTransformer: Multi-horizon Forecasts with Context Dependent Attention and Optimal Bregman Volatility. In *KDD 2022 Workshop on Mining and Learning*

from Time Series – Deep Forecasting: Models, Interpretability, and Applications, 2022.

[59] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. In *Thirty-Fifth AAAI Conference on Artificial Intelligence*, pages 11106–11115. AAAI Press, 2021.

[60] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting. *CoRR*, abs/2106.13008, 2021.

[61] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency Enhanced Decomposed Transformer for Long-term Series Forecasting. *CoRR*, abs/2201.12740, 2022.

[62] Azul Garza and Max Mergenthaler-Canseco. TimeGPT-1, 2023.

[63] Hao Xue and Flora D. Salim. PromptCast: A New Prompt-Based Learning Paradigm for Time Series Forecasting. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–14, 2023.

[64] Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew Gordon Wilson. Large Language Models Are Zero-Shot Time Series Forecasters, 2023.

[65] Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A Decoder-only Foundation Model for Time-series Forecasting, 2024.

[66] Kashif Rasul, Arjun Ashok, Andrew Robert Williams, Hena Ghonia, Rishika Bhagwatkar, Arian Khorasani, Mohammad Javad Darvishi Bayazi, George Adamopoulos, Roland Riachi, Nadhir Hassen, Marin Biloš, Sahil Garg, Anderson Schneider, Nicolas Chapados, Alexandre Drouin, Valentina Zantedeschi, Yuriy Nevmyvaka, and Irina Rish. Lag-Llama: Towards Foundation Models for Probabilistic Time Series Forecasting, 2024.

[67] Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, and Qingsong Wen. Time-LLM: Time series forecasting by reprogramming large language models. In *International Conference on Learning Representations (ICLR)*, 2024.

[68] Renjie Wu and Eamonn J Keogh. Current time series anomaly detection benchmarks are flawed and are creating the illusion of progress. *IEEE transactions on knowledge and data engineering*, 35(3):2421–2429, 2021.

List of Tables

Esta página ha sido intencionalmente dejada en blanco.

List of Figures

Esta es la última página.
Compilado el martes 29 abril, 2025.
https://github.com/GastonGarciaGonzalez/Tesis-de-doctorado-GGG/tree/main/TESIS