UNIVERSIDADE FEDERAL DO CEARÁ
CENTRO DE CIÊNCIAS
DEPARTAMENTO DE COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO
DOUTORADO EM CIÊNCIA DA COMPUTAÇÃO

MADSON LUIZ DANTAS DIAS

TIME SERIES ANOMALY DETECTION AND DIAGNOSIS VIA MANIFOLD LEARNING AND NORMALIZING FLOWS

FORTALEZA
2024

MADSON LUIZ DANTAS DIAS

TIME SERIES ANOMALY DETECTION AND DIAGNOSIS VIA MANIFOLD LEARNING AND NORMALIZING FLOWS

Thesis submitted to the Post-Graduation Program in Computer Science of the Center of Science of the Federal University of Ceará, as a partial requirement for obtaining the title of Doctor in Computer Science. Concentration Area: Theoretical Computer Science.

Supervisor: Prof. Dr. César Lincoln Cavalcante Mattos.

FORTALEZA

2024

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

D533t Dias, Madson Luiz Dantas.
Time series anomaly detection and diagnosis via manifold learning and normalizing flows / Madson Luiz Dantas Dias. – 2024.
97 f. : il. color.

Tese (doutorado) – Universidade Federal do Ceará, Centro de Ciências, Programa de Pós-Graduação em Ciência da Computação , Fortaleza, 2024.
Orientação: Prof. Dr. César Lincoln Cavalcante Mattos.

1. Normalizing flows. 2. Detecção de anomalias. 3. Séries temporais. 4. Manifold learning. I. Título.
CDD 005

MADSON LUIZ DANTAS DIAS

TIME SERIES ANOMALY DETECTION AND DIAGNOSIS VIA MANIFOLD LEARNING AND NORMALIZING FLOWS

Thesis submitted to the Post-Graduation Program in Computer Science of the Center of Science of the Federal University of Ceará, as a partial requirement for obtaining the title of Doctor in Computer Science. Concentration Area: Theoretical Computer Science

Approved on: 21/08/2024.

For my parents, Marconi and Maria, and my fiancée, Priscila, for their unconditional support throughout this period.

ACKNOWLEDGEMENTS

In the life of a graduate student, the main focus is on learning and teaching science. Although it may seem like an easy task, it is not. This is the period when we miss the time when we were children and did not have to worry about Jacobian matrices, gradient descent or likelihood functions. I confess that it was only during this period that I learned the true meaning of the word "*deadline*", and it did not happen in the best way. This was a difficult period and it would certainly have been much more difficult if it were not for the help of several people who supported, guided and encouraged me to make this possible. It is with great pleasure that I recognize the professors, friends and family who supported me in this undertaking.

I believe that I owe my fascination with science to my father, Marconi. The many math classes that I attended under his guidance, since childhood, left lasting marks that may never disappear. I would like to thank him for having participated, even without knowing it, in some of my first troubled "studies" and, later, for his constant encouragement of my more sophisticated undertakings in this area. I would also like to thank my mother, Maria, who not only supported my desire to enroll in a PhD's degree, but also gave me the strength to complete it. I also couldn't forget to thank Priscila, who I met during this period and is now my fiancée, for supporting me, helping me and encouraging me whenever I needed it.

Supervising is a difficult task that requires a lot of patience and a willingness to share. I would like to thank Professor César for having carried out this task with such competence. His constant personal attention, suggestions and support over the years were crucial to my development as a researcher. He always presented himself with a light touch in meetings, but also with great professionalism and love for what he does. I was very lucky to have a supervisor like him.

Although UFC has probably been my second home, spending hours and hours in a lab, implementing and running codes, would not have been as fun without the company of my friends Bustamante, Daniel, Diego, Ernando, Lucas, Thiago, and Renan. Furthermore, the possibility of discussing not only research-related topics, but also more general life problems, gives me the certainty that we are not just fellows.

I would also like to thank the professors of the program, especially Professor João Paulo Pordeus, who was always willing to help. Finally, I would like to thank the financial support of the Fundação de Apoio ao Desenvolvimento Científico e Tecnológico - Ceará (FUNCAP), which provided a scholarship, together with the MDCC.

"The absence of evidence is not the evidence of absence."
(Carl Sagan)

RESUMO

O surgimento de tecnologias de coleta e armazenamento de dados permitiu o acúmulo extenso de dados ao longo do tempo. Consequentemente, conjuntos de dados de séries temporais de alta dimensão tornaram-se predominantes em diversos campos, incluindo redes de sensores, segurança, saúde, manufatura e finanças. Embora represente um desafio substancial, detectar eventos raros em tais conjuntos de dados é uma tarefa fundamental em diversas aplicações, incluindo detecção de intrusão cibernética, análise de defeitos, detecção de falhas, análise de fraude de cartão de crédito, detecção de trajetória suspeita, etc. Vários métodos têm sido empregados para lidar com esse desafio, desde abordagens tradicionais como classificação, agrupamento de dados, métodos baseados em distância, estimação de densidade e técnicas estatísticas bem como soluções mais contemporâneas envolvendo modelos de aprendizado profundo. Nesta tese, fornecemos uma visão geral abrangente do campo, apresentamos métodos do estado-da-arte para detecção e diagnóstico de anomalias em séries temporais multivariadas e introduzimos duas novas abordagens para resolver problemas envolvendo detecção de anomalias em séries temporais. A primeira abordagem, intitulada ag**gr**egated **a**nomaly **d**etection with normaliz**ing** flow**s** (GRADINGS), é um *framwork* para detecção de anomalias em conjuntos de dados de séries temporais, aplicada a dados de trajetórias, baseada na estimativa da densidade de cada segmento de trajetória e na agregação das probabilidades dos segmentos em um único score de anomalia. Essa estratégia permite o tratamento de sequências possivelmente grandes de diferentes comprimentos. A segunda abordagem, chamada **r**obust **an**omaly **d**etection on **m**ultivariate time **s**eries (RANDOMS), utiliza *normalizing flows* e técnicas de *manifold learning* para resolver os problemas de detecção e diagnóstico de anomalias em séries temporais multivariadas. Avaliações extensivas dos métodos propostos foram conduzidas em várias aplicações, comparando-as com diversos modelos. Os resultados dos nossos experimentos computacionais demonstram a eficácia das abordagens propostas, superando consistentemente os métodos de detecção e diagnóstico de anomalias do estado-da-arte existentes em diversos casos.

Palavras-chave: normalizing flows; detecção de anomalias; séries temporais; manifold learning.

ABSTRACT

The emergence of data collection and storage technologies has allowed the accumulation of extensive data over time. Consequently, high-dimensional time series data sets have become prevalent across diverse fields, including sensor networks, security, healthcare, manufacturing, and finances. Although it represents a substantial challenge, detecting rare events in such data sets is a fundamental task in several applications, including cyber-intrusion detection, defect analysis, fault detection, credit card fraud analysis, suspicious trajectory detection, etc. Various methods have been employed to handle this challenge, ranging from traditional approaches such as classification, clustering, distance metrics, density estimation, and statistical techniques to more contemporary solutions involving deep learning models. In this thesis, we provide a comprehensive overview of the field, present state-of-the-art methods of multivariate time series anomaly detection and diagnosis, and introduce two new approaches for solving problems involving anomaly detection in time series. The first one called ag**gr**egated **a**nomaly **d**etection with normaliz**ing** flow**s** (GRADINGS) is a framework for anomaly detection in time series database, applied to trajectory data that is based on estimating the density for each trajectories segments and aggregating segments' likelihoods into a single anomaly score. Such a strategy enables the handling of possibly large sequences of different lengths. The second approach called **r**obust **an**omaly **d**etection on **m**ultivariate time **s**eries (RANDOMS), which uses normalization flows and manifold learning techniques to solve the anomaly detection and diagnosis problems. Extensive evaluations of the proposed methods have been conducted across various applications, comparing them against several models. The results of our computational experiments demonstrate the efficacy of our approaches, consistently outperforming existing state-of-the-art anomaly detection and diagnosis methods in numerous cases.

Keywords: normalizing flows; anomaly detection; time series, manifold learning.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

$\mathcal{M}$-flow	Manifold-learning flows
Actnorm	Activation Normalization
AD	Anomaly Detection
AF	Ambient Flow
AUC	Area Under the Curve
AUCP	Area Under the Curve Percentage Thresholder
CAE-M	Convolutional Auto-Encoding Memory Network
DAGMM	Deep Auto-encoding Gaussian Mixture Model
DNF	Denoising normalizing flows
EVT	Extreme Value Theory
FOM	Flow on a prescribed manifold
GMM	Gaussian Mixture Model
GND	Graph deviation network
GRADINGS	Aggregated Anomaly Detection with Normalizing Flows
GRU	Gated Recurrent Unit
ID	In distribution
KL	Kullback-Leibler Divergence
LOE	latent outlier exposure
LOF	Local Outlier Factor
LSTM	Long Short-Term Memory
MAD-DAN	Multi-scale Anomaly Detection using Generative Adversarial Networks
MADE	Masked Autoencoder for Distribution Estimation
MAF	Masked Autoregressive Flow
MBA	MIT-BIH Supra-ventricular Arrhythmia Database
MCC	Matthews Correlation Coefficient
Moglow	Motion Glow
MSCRED	Multi-Scale Convolutional Recurrent Encoder-Decoder
MSDS	Multi-Source Distributed System
MSL	Mars Science Laboratory
MTS	Multivariate Time Series

ND	Novelty Detection
NDCG	Normalized Discounted Cumulative Gain
NDT	Non-parametric Dynamic error Thresholding
NF	Normalizing Flows
OD	Outlier Detection
ODE	Ordinary Differential Equation
OOD	Out-of-Distribution Detection
OSR	Open Set Recognition
PDF	Probability Density Function
PIE	Pseudo invertible encoding
RANDOMS	**R**obust **An**omaly **D**etection on **M**ultivariate time **S**eries
RealNVP	Real-valued Non-Volume Preserving
RNN	Recurrent Neural Network
SF	SoftFlow
SMAP	Soil Moisture Active Passive
SNF	Spread normalizing flows
SWaT	Secure Water Treatment
TranAD	Transformer-based Anomaly Detection
TSAD	Time Series Anomaly Detection
USAD	Unsupervised Anomaly Detection for Multivariate Time Series
WADI	Water Distribution

LIST OF SYMBOLS

$\boldsymbol{x}$	Observation or data point
$\boldsymbol{x}_t$	Observation at time t
$\boldsymbol{x}_{t,d}$	Observation at time t for dimension d
$\mathbf{X}_t$	Random variable at time t
$\mathbf{X}$	Multivariate time series
$\mathbf{X}'$	Test multivariate time series
$\mathbf{X}_{t,d}$	Value of dimension d at time t
$\mathbb{R}^D$	D-dimensional real-valued space
T	Length of a time series
D	Number of dimensions (variables)
t, t'	Time indices
$S(\cdot)$	Anomaly scoring function
$H(\mathbf{X}, \tau)$	Decision function for anomaly detection
τ	Threshold for anomaly classification
$A(x)$	Anomaly score for data point x
$A(T)$	Aggregated anomaly score for trajectory T
$\alpha(S_i^{(m)})$	Anomaly score for segment i of trajectory m
ϕ	Aggregation function
$\delta(\cdot)$	Flattening function to vector space
$q_l^{(m)}$	Location point in trajectory m
$q_{l,j}^{(m)}$	Coordinate j of point l in trajectory m
μ_k	Mean of component k in GMM
Σ_k	Covariance matrix of component k in GMM
π_k	Weight of component k in GMM
$p(\cdot)$	Probability density function
$\mathcal{N}(\mu, \sigma^2)$	Gaussian distribution

$RD(x,u)$	Reachability distance of x w.r.t. u
$LRD(x)$	Local reachability density of x
K	Number of neighbors or GMM components
β	Weighting factor in F-score
P	Precision
R	Recall
F_β	F-score
MCC	Matthews correlation coefficient
AUC	Area under ROC curve
$\log$	Natural logarithm operator

CONTENTS

1 INTRODUCTION

"An approximate answer to the right problem is worth a good deal more than an exact answer to an approximate problem."
(John Tukey)

A data series comprises an ordered sequence of data elements, which provide descriptions of various attributes or characteristics related to an object or process and are typically represented through continuous measurements (PARZEN, 1961). When this ordering is based specifically on the time dimension, it is commonly denoted as a time series (WEI, 2013). These measurements can belong to diverse domains, ranging from finance and healthcare to environmental monitoring and industrial processes. In the context of machine learning, several tasks are associated with time series data, such as forecasting (LIM; ZOHREN, 2020), classification (FAWAZ *et al.*, 2019), clustering (LIAO, 2005), transformation (SALLES *et al.*, 2019), and detection of rare events (TENG, 2010).

Detecting rare events within such complex data is a challenge that has received considerable attention from many researchers in the most diverse areas such as defect analysis in industrial machines (SCHMIDT; SIMIC, 2019), cyber-intrusion detection (BUCZAK; GUVEN, 2016), video anomaly detection (KUMAR *et al.*, 2020), suspicious trajectory detection (DIAS *et al.*, 2020), and sensor networks (ZHANG *et al.*, 2023b). Furthermore, rare event detection encompasses the concepts of AD, OD, ND, OSR, and OOD, which, despite being objects of study with a very similar essence, they present subtle differences (YANG *et al.*, 2021). These differences and similarities will be discussed in more detail in the next chapters.

In this thesis, we focus on the task of anomaly detection in multivariate time series, whose main objective is to detect discrepant events in a data set not seen in a training stage (ZHAO *et al.*, 2020). More specifically, we are interested in solving three specific problems in this context that are related to the granularity and type of such anomalies, namely: (*i*) anomaly detection in time series database, (*ii*) anomaly detection within a given time series, and (iii) anomaly diagnosis. The last one can be considered as a subsequent problem of the anomaly detection within a given time series (GARG *et al.*, 2022).

The anomaly detection in time series database typically involves, for a given training set $\mathcal{X} = \{\mathbf{X}_n\}_{n=1}^{N}$ of time series, identifying which time series of the test set $\mathcal{X}' = \{\mathbf{X}'_m\}_{m=1}^{M}$ exhibits unusual patterns or behavior. This can be challenging because the data set may contain a large number of sequences, and it may not be immediately apparent which sequences are anomalous, formally:

Problem 1 (Anomaly detection in time series database) *Given a training set $\mathcal{X} = \{\mathbf{X}_n\}_{n=1}^{N}$ containing N sequences, mostly non-anomalous, where each sequence $\mathbf{X}_n$ have D dimensions and length $T^{(n)}$, i.e., $\mathbf{X}_n \in \mathbb{R}^{T^{(n)} \times D}$. The problem of anomaly detection in a time series database is to learn an anomaly scoring function $S(\cdot)$, that can help to determine if a sample $\mathbf{X}_m$ of the test set is a normal or abnormal instance by using the following decision system:*

$$H(\mathbf{X}, \tau) = \begin{cases} +1 \textit{ (normal)} & \textit{if} \quad S(\mathbf{X}) < \tau \\ -1 \textit{ (abnormal)} & \textit{if} \quad \textit{otherwise} \end{cases} \tag{1.1}$$

where τ is a predefined threshold that separates abnormal from normal data samples.

On the other hand, anomalies within a given time series refer to individual data points or segments that deviate significantly from the expected pattern (GUPTA *et al.*, 2014). As discussed above, this can be caused by various factors, such as measurement errors, changes in the underlying process generating the sequences, or even intentional manipulation. We can handle an anomaly detection within a given time series problem using the following formulation:

Problem 2 (Anomaly detection within a given time series) *Given a time series $\mathbf{X} = \{\boldsymbol{x}_t\}_{t=1}^{T}$ containing T observations of D variables, i.e., $\boldsymbol{x}_t \in \mathbb{R}^D$. For any unseen data point $\boldsymbol{x}_{t'}$ with $t' > T$ of a test time series $\mathbf{X}'$ the problem of anomaly detection within a given time series is to learn a decision function $H : \mathbb{R}^D \to \{\pm 1\}$ to classify if the point at t'-th timestamp of the test set is anomalous, where 1 denotes an anomaly data point.*

Subsequently, to the problem of detecting anomalies within a given time series, when dealing with multivariate series, we can perform a diagnosis at points related to such anomalies in order to identify which variables were most influential. Also known as anomaly interpretation (ZHAO *et al.*, 2020), can be formally defined as:

Problem 3 (Anomaly diagnosis) *Given a data point $\boldsymbol{x}_{t'}$ already identified as an anomaly, the problem of anomaly diagnosis is to create a function $G : \mathbb{R}^D \to \mathbb{R}^D$ that identify the contribution order of the channels to this point be considered as an anomaly.*

The main difference between the first two problems lies in the granularity of anomaly treatment. In the Anomaly detection in time series database problem (Problem 1), we focus on identifying anomalies pertaining to an entire sequence. However, in Problem 2, our concern shifts towards detecting anomalies within individual data points or sequences of points (GARG *et al.*, 2022).

In order to solve the aforementioned problems, some studies have proposed approaches that employ a wide range of methodologies, encompassing classification, clustering, distance-based techniques, and reconstruction methods (MENG *et al.*, 2019). Moreover, in the last years, many studies have adopted the use of deep learning and modern generative models for this purpose (Blázquez-García *et al.*, 2022; ZHONG *et al.*, 2023). Within this context, the concept of Normalizing Flows (NF) (REZENDE; MOHAMED, 2015), a class of probabilistic generative models, has emerged as a particularly promising paradigm for modeling complex data distributions (PAPAMAKARIOS *et al.*, 2021).

NFs are a family of likelihood-based models for estimating expressive probability distributions by using a chain of bidirectional transformations to move between a complex probability density and a simple one (REZENDE; MOHAMED, 2015). Each transformation, i.e., a flow, is parametrized by (possibly deep) neural networks. One of the main advantages of flow-based approaches is the available exact model log-likelihood, which is used as an objective function to optimize all the model parameters jointly. Furthermore, we can compute exact log-likelihood values for new data points, which we will then apply as a coherent anomaly score (KINGMA; DHARIWAL, 2018; CHO *et al.*, 2022).

Unfortunately, a pure likelihood scoring only provides an anomaly rate for each point that can be used to solve the anomaly detection within a given time series (Problem 2). Still, it does not allow anomaly diagnosis (Problem 3) or anomaly detection in time series database (Problem 1). In order to solve the Problem 1, some authors have proposed solutions by using some aggregation statistic function like average over the NF log-likelihood scores (GUDOVSKIY *et al.*, 2022; CHO *et al.*, 2022) that it still does not help in solving the problem of anomaly diagnosis, since we would need a score for each dimension of the data.

Autonomously, several recent studies have proposed solutions using NFs to maximize the likelihood while learning the data manifold (BEITLER *et al.*, 2021; CUNNINGHAM *et al.*, 2022; FLOURIS; KONUKOGLU, 2023; HORVAT; PFISTER, 2023; GEMICI *et al.*, 2016; KIM *et al.*, 2020; CUNNINGHAM *et al.*, 2020; ZHANG *et al.*, 2023a). This approach, in addition to ensuring that better likelihoods will be learned from high-dimensional data, helps us solve the problem of anomaly diagnosis since we can now calculate a per-dimension reconstruction error associated with each point and use it as an anomaly score, as mentioned in Brehmer e Cranmer (2020) and Razavi *et al.* (2025).

In this work, we propose two NF-based approaches to solve the above-mentioned problems. The first approach was applied to detect anomalies in time series database and was evaluated by using trajectory data, a type of multivariate time series that can be high-dimensional due to the presence of several measured points within a single trajectory. Besides, distinct data examples can have different lengths, which cannot be straightforwardly compared. We propose a methodology that tackles both issues by considering fixed-size segments of the available trajectories. An NF model is then used to estimate the probability density of such segments, and a single anomaly score for a new trajectory is computed from its segments by using an aggregation function. We name our approach Aggregated Anomaly Detection with Normalizing Flows (GRADINGS). We evaluate the proposed GRADINGS approach using real-world trajectories. The obtained experimental results indicate the feasibility of our solution.

For the last two problems, following the studies of Henter *et al.* (2020), Kingma e Dhariwal (2018), and Razavi *et al.* (2025), we propose an unsupervised anomaly detection and diagnosis model called **R**obust **An**omaly **D**etection on **M**ultivariate time **S**eries (RANDOMS) that uses Recurrent Neural Network (RNN) to model the information of temporal context and a series of transformations, which includes an activation normalization (referred as actnorm) (KINGMA; DHARIWAL, 2018), a linear transformation, and an affine coupling layer (HENTER *et al.*, 2020). Our model also uses a manifold learning approach to create anomaly scores that combine the negative log-likelihood and the reconstruction error (manifold score). Our computational experiments' outcomes showcase our approach's potential, consistently surpassing the performance of established *state-of-the-art* AD methods across multiple instances.

1.1 Objectives

The main objective of this thesis consists of developing and evaluating effective anomaly detection and diagnosis frameworks for multivariate time series data using manifold learning and normalizing flows, demonstrating its applicability and performance across diverse real-world domains. The specific objectives of our work are listed below:

1. To provide a comprehensive review of key components, terminology, and their roles in the time series anomaly detection process.
2. To propose a novel NF-based framework specifically designed to work on anomaly detec-

tion on time series database;

3. To propose a novel manifold learning and NF-based framework specifically designed to work on anomaly detection and diagnosis within a given time series;
4. To evaluate all the proposed models in their specific tasks by performing a comparative analysis over several data sets.

1.2 List of publication

1. ROCHA, A.; MONTEIRO, M.; MATTOS, C.; **DIAS, M. L. D.**; SOARES, J.; MAGALHÃES, R.; MACEDO, J. Edge ai for internet of medical things: A literature review. *Computers and Electrical Engineering, v. 116, p. 109202, 2024.* ISSN 0045-7906. Available in: <https://www.sciencedirect.com/science/article/pii/S0045790624001307>.
2. **DIAS, M. L. D.** *et al.* Anomaly detection in trajectory data with normalizing flows. In: *2020 International Joint Conference on Neural Networks, IJCNN 2020, Glasgow, United Kingdom, July 19-24, 2020.* IEEE, 2020. p. 1–8. Available in: <https://doi.org/10.1109/IJCNN48605.2020.9206939>.
3. **DIAS, M. L. D.** *et al.* Parsimonious minimal learning machine via multiresponse sparse regression. *International Journal of Neural Systems, v. 30, n. 5, 2020.* Available in: <https://doi.org/10.1142/S0129065720500239>.
4. **DIAS, M. L. D.** *et al.* Sparse minimal learning machine using a diversity measure minimization. In: *27th European Symposium on Artificial Neural Networks, ESANN 2019, Bruges, Belgium, April 24-26, 2019*, 2019. Available in: <https://www.esann.org/sites/default/files/proceedings/legacy/es2019-178.pdf>.
5. **DIAS, M. L. D.** *et al.* Sparse minimal learning machines via $L_{1/2}$ norm regularization. *7th Brazilian Conference on Intelligent Systems, BRACIS 2018, São Paulo, Brazil, October 22-25, 2018.* IEEE Computer Society, 2018. p. 206–211. Available in: <https://doi.org/10.1109/BRACIS.2018.00043>.
6. MAIA, Á; **DIAS, M. L. D.**; GOMES, J; NETO, A. Optimally selected minimal learning machine. In: YIN, H. *et al.* (Ed.). *Intelligent Data Engineering and Automated Learning - IDEAL 2018 - 19th International Conference, Madrid, Spain, November 21-23, 2018*, Proceedings, Part I. Springer, 2018. (Lecture Notes in Computer Science, v. 11314), p. 670–678. Available in: <https://doi.org/10.1007/978-3-030-03493-1_70>.

1.3 Outline

The remaining of this document is organized as follows:

- In Chapter 2, we introduce fundamental concepts, including a taxonomy of rare events, anomaly detection, and time series anomaly detection.
- Chapter 3 introduces normalizing flows, first detailing its modeling based on the variable change rule, exploring transformations and their determinants, and presenting the main flow architectures.
- In Chapter 4, we present GRADINGS, a method for anomaly detection in time series databases, exploring the benefits of NFs and creating an anomaly score that uses log-likelihood and aggregation functions. We also present a set of experiments to check the viability of applying this technique to address the problem.
- Chapter 5 is dedicated to our main proposal, called RANDOMS, for solving problems of anomaly detection and diagnosis. It reviews related works for sequential modeling and flow-based anomaly detection methods. The chapter then outlines the methodology, including motivation, flow step, manifold learning, and anomaly inference. A comprehensive empirical evaluation is also presented.
- Chapter 6 presents the conclusions, summarizing the main contributions of the thesis. It discusses the effectiveness and limitations of the proposed methods, including GRADINGS and RANDOMS, in addressing the challenges of anomaly detection and diagnosis in time series data. The chapter also outlines future research directions, suggesting potential improvements and applications for the developed techniques.

2 ANOMALY DETECTION

“Every science is made up entirely
of anomalies rearranged to fit.”
(Raphael Aloysius Lafferty)

The concepts of anomaly detection, outlier detection, novelty detection, open-set recognition, and out-of-distribution are objects of study with a very similar essence, since all of them have as main objective to *uncovering patterns in data that do not conform to the expected behavior of normality* (RUFF *et al.*, 2021). Therefore, they share the notion of *in-distribution* and have as main objective to discern instances *out-of-distribution*, under the assumption that a model be designed to operate in an environment where it encounters not only the data it was trained on, but also data it has never seen before, i.e. the *open world* assumption (YANG *et al.*, 2021). The difference lies in where the detection will be carried out, the presence or absence of anomalies in the training set, and the type of distribution shift existing in the data. Understanding such differences can help us identify which concept to use according to the problem to be treated, thus improving the understanding of choice and creating more adequate models for different applications.

In addition, we should also pay attention to the differences existing within the context of anomaly detection in time series since we can detect anomalies from different perspectives, including *point anomalies*, *subsequence anomalies*, *contextual anomalies*, and *anomaly time series*. Understanding which type of anomaly is being treated is essential to choosing the right type of model to use (YANG *et al.*, 2021).

To enhance understanding in the upcoming discussions and maintain clarity and consistency in terminology and symbols, we aim to start this chapter by revisiting fundamental concepts and introducing the notations that will persist throughout the document. This chapter comprises a presentation of a taxonomy associated with identifying rare events and comprehensive definitions of AD, including elucidation of diverse anomaly types. Additionally, the chapter offers clear explanations of AD in the context of time series data.

2.1 Preliminary

Before we discuss the differences mentioned above, we will begin by introducing the notion of *distribution shift*, which, although it has become a common topic in recent years (WI-

LES *et al.*, 2022; FANG *et al.*, 2020), its study dates back to as early as 1986 (SCHLIMMER; GRANGER, 1986). This concept refers to situations wherein a machine learning model encounters test data originating from a distribution distinct from the one it was trained on. In essence, the characteristics of the test data differ from those of the training data (MORENO-TORRES *et al.*, 2012).

Consider a data set with training examples denoted as $\mathcal{D} = \{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1}^{N}$, sampled from the joint distribution $P(X, Y)$ over the Cartesian product of the input (sensory) space $\mathcal{X} = \mathbb{R}^D$ and the label (semantic) space $\mathcal{Y}$. Here, X and Y represent random variables associated with the input and label spaces, respectively. Distribution shift can manifest itself both in the marginal distribution $P(X)$ and jointly in $P(X)$ and $P(Y)$[1]. According to on Kelly *et al.* (1999), we can more commonly find three types of shifts, namely, distribution shift, covariate shift, and concept shift.

The *distribution shift*, also referred as *dataset shift* occurs when, for a given test set $\mathcal{D}' = \{\boldsymbol{x}_m, \boldsymbol{y}_m\}_{m=1}^{M}$, the training joint distribution is not equal to the testing joint distribution, i.e., $P'(Y, X) \neq P(Y, X)$. In the *covariate shift*, also known as *population drift*, there is a discrepancy in the marginal distribution $P(X)$ of the covariates $\boldsymbol{x}$, between the training and test sets, i.e., $P'(X) \neq P(X)$, while the conditional distribution $P'(Y|X) = P(Y|X)$ remains consistent. On the other hand, the *semantic* or *concept shift*, or even *concept drift*, occurs when there is a disparity in the output distribution of $P(Y)$. In this case, $P'(Y|X) \neq P(Y|X)$ in $\mathcal{D}'$, while $P'(X) = P(X)$, indicating that the input distribution of the test data mirrors that of the training data (MORENO-TORRES *et al.*, 2012). Tian *et al.* (2021) proposes a modification in this equality to $P'(X) \approx P(X)$, since it is rare that this change of concept occurs without changes in $P(x)$.

The above concepts are strongly related to rare event detection, especially concerning the anomaly detection task, since AD involves identifying patterns within data that diverge from expected behavior during the testing phase (inductive learning), specifically in the $\mathcal{D}'$ set. In this scenario, we can have both covariate or semantic shift, which leads to two sub-tasks, sensory AD and semantic AD. Outlier detection, on the other hand, aims to find regions with lower density in the training data $\mathcal{D}$, i.e., transductive learning (BIANCHINI *et al.*, 2016; AHMED; COURVILLE, 2020).

[1] It is crucial to emphasize that the label space $\mathcal{Y}$ may not necessarily be connected to whether the instance is an anomaly or not. In certain scenarios, these labels may pertain to a classification problem distinct from the AD issue.

In the task of novelty detection, in turn, aims to detect patterns in the test set that do not belong to any of the categories present in the training set, i.e., $P'(Y) \neq P(Y)$, a change that occurs through a semantic shift, that matches with semantic AD (RUFF *et al.*, 2021). It is essential to highlight that both in the case of AD and in the case of novelty detection, we can have two configurations for the training labels, one class or multi-class. Still, regardless, AD and ND are formulated as binary classification tasks, considering whether the instances are anomalies or not (SALEHI *et al.*, 2022).

Another two concepts closely related with AD, OD, and ND are OSR, and OOD detection. Both share the notion of In distribution (ID) and have as main objective to discern instances *out-of-distribution*, under the assumption that a model be designed to operate in an environment where it encounters not only the data it was trained on, but also data it has never seen before, i.e., the open world assumption (YANG *et al.*, 2021). However, both OSR and OOD detection requires the multi-class classifier to simultaneously: (i) accurately classify test samples of "known classes" and (ii) detect test samples of "unknown unknown classes". The main difference between OSR and OOD detection tasks is that they have distinct benchmark settings: OSR typically divides a single multi-class classification data set into class-based ID and OOD portions, while OOD detection uses one data set as ID and fetches other data sets as OOD, ensuring non-overlapping categories between them (RUFF *et al.*, 2021).

As we described, despite the close relationship between the concepts above, they differ subtly at some points, and understanding such differences can help us identify which concept to use according to the problem to be addressed, thus improving the choice as well as the creation of models most suitable for different applications. In Figure 1, we illustrate the aforementioned concepts and their relations, divided by classification tasks. In this work, we will address the task of sensory AD in multivariate time series, that is, the task related to detecting changes in the input distribution of test data $P'(X)$. In the next sections, we will formally describe the problem addressed.

2.2 TSAD

Time series are sequential data naturally ordered by time. This definition includes the possibility of irregularly sampled time series or dimensions (variables), i.e., with varying temporal granularities (SHUMWAY; STOFFER, 2000). We can formally define a Multivariate Time Series (MTS) as follows:

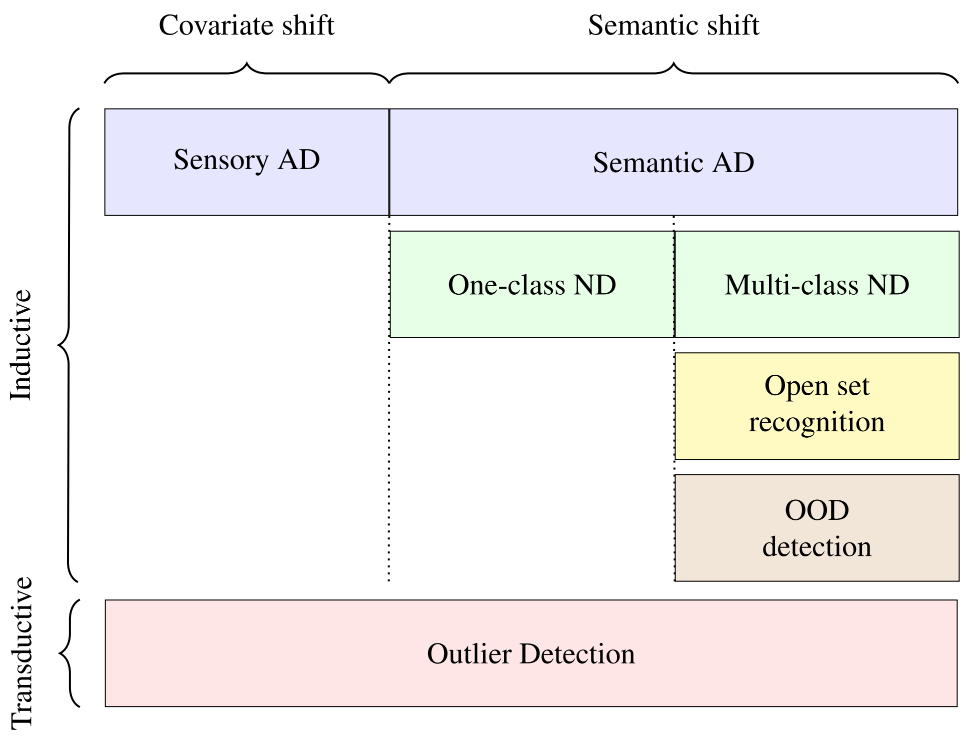


Figure 1 – Taxonomy of rare event detection tasks. This diagram classifies rare event detection tasks based on the type of learning (inductive vs. transductive) and the type of distribution shift (covariate shift vs. semantic shift). In inductive learning, sensory and semantic AD focus on detecting anomalies in sensory or target data, typically under covariate or semantic changes, respectively. ND detects new, previously unseen classes using single or multiple classes for training. In contrast, OSR and OOD detection uses a multi-class setting and recognizes known classes while identifying deviant patterns. In transductive learning, OD identifies data points in the training set that deviate significantly from the majority of the data. (Modified from (YANG *et al.*, 2021))

Definition 1 (Multivariate time series) *Denoted as* $\mathbf{X} = \{\boldsymbol{x}_t\}_{t=1}^{T}$*, a multivariate time series is defined as a sequentially ordered collection of* D*-dimensional vectors, each of them recorded at a specific time* t *and comprising* D *real-valued observations, represented as* $\boldsymbol{x}_t = (x_{t,1}, x_{t,2}, \ldots, x_{t,D})$*.*

Each individual $\boldsymbol{x}_t$ of an MTS is referred to as a "*point*", and this is the minimum information of a time series. A special case of multivariate time series occurs when $D = 1$, in which the series is called *univariate time series*. In this context, each *point* is a scalar, and the time series is represented as a vector of size T. It is also important to note that for any specific dimension $d \in \{1, \ldots, D\}$ in an MTS, $\boldsymbol{x}^{(d)} = \{x_{t,d}\}_{t=1}^{T}$ consists in a univariate time series. Each observation, represented as $x_{t,d}$ in the vector $\boldsymbol{x}_t$ is an observed value of a random, time-dependent variable $X_{d,t}$ in $\boldsymbol{X}_t = (X_{t,1}, \ldots, X_{t,D})$. Given this scenario, each variable could exhibit dependencies not only on its historical values but also on other time-dependent variables.

Sometimes, we must also analyze specific contiguous subsets of points for a given time series. These subsets are called subsequences, pattern fragments, segments, sliding windows, motifs, or n-grams (GUPTA *et al.*, 2014), and can be formally defined by:

Definition 2 (Subsequence) *A subsequence* $\mathbf{S}_{t_1:t_2} \subseteq \mathbf{X}$ *of the time series* $\mathbf{X}$ *with length* $L = t_2 - t_1 > 0$ *is given by* $\mathbf{S}_{t_1:t_2} = \{\boldsymbol{x}_t\}_{t=t_1}^{t_2} = (\boldsymbol{x}_{t_1}, \boldsymbol{x}_{t_1+1}, \ldots, \boldsymbol{x}_{t_2})$*, where* $1 \leq t_1 \leq t_2 \leq T$*.*

Before starting the discussion about time series anomaly detection, we need to define the concept of "anomaly" in a more general way, without specifying the type of data. Given a training set $\mathcal{X} \subseteq \mathbb{R}^D$ and considering $P^+(X)$ on $\mathcal{X}$ as a distribution that represents the concept of normality, in other words, the majority behavior of the data, we can define the set of anomalies using:

Definition 3 (Anomaly set) *Assuming* $p^+(x)$ *as the corresponding probability function of the ground-truth law of normal behavior* $P^+(X)$*, the set of anomalies* $\mathcal{A}$ *can be defined as a set of data points* $\boldsymbol{x}$ *on* $\mathcal{X}$ *that appears in regions of low probability of* $P^+(X)$ *and can be written as*

$$\mathcal{A} = \{\boldsymbol{x} \in \mathcal{X} \mid p^+(\boldsymbol{x}) \leq \tau\}, \quad \tau \geq 0, \tag{2.1}$$

where τ *is a threshold such that the probability of* $\mathcal{A}$ *under* $P^+(X)$ *is sufficiently small.*

In the context of time series analysis, we can distinguish between various types of anomalies, including *point anomalies*, *subsequence anomalies*, *contextual anomalies*, and *anomaly time series*. Understanding which type of anomaly is being treated is essential to choosing the right type of model to use (YANG *et al.*, 2021).

A *point anomaly* is an instance that exhibits unusual behavior at a specific time instant when compared with the rest of the time series (global anomaly) or to its neighboring points (local anomaly). These anomalies stand out as isolated irregularities, displaying behaviors that are markedly different from the surrounding time points and are characterized by their solitary nature and distinct deviation from the general trends, making them relatively easier to detect and categorize (Blázquez-García *et al.*, 2022).

Definition 4 (Point anomaly) *Given a multivariate* D*-dimensional time series* $\mathbf{X}$ *with size* T*, a point anomaly is defined as a point* $\boldsymbol{x}_t \in \mathcal{A}$ *with covariates differs significantly from all the points in the interval* $[t-k, t+k]$*,* $k \in \mathbb{N}^+$ *and* k *is sufficient large.*

A *group*, *collective*, or *subsequence anomaly* consists of a subsequence of interconnected points anomalies. Unlike point anomalies, which focus on individual instances, group anomalies focus on the collective behavior of multiple data points. These anomalies can be challenging to detect since the deviation might not be apparent when considering individual data points alone. Group anomalies could indicate complex interactions or events that influence a subset of the data in unique ways (CHEN *et al.*, 2021).

Definition 5 (Collective anomaly) *For a given multivariate time series X with size T, a collective anomaly is a subsequence in an interval from t_1 and t_2, where $t_1 < t_2 < T$, such that all of the points within this segment are considered point anomalies, i.e.:*

$$\mathbf{S}_{t_1:t_2} = \{x_t \mid t_1 \leq t \leq t_2 \textit{ and } x_t \in \mathcal{A}\}. \tag{2.2}$$

Contextual, *local*, or *conditional anomalies* are data points that appear normal when considered in one context but become anomalous when examined within a different context (GUPTA *et al.*, 2014). These anomalies are sensitive to specific conditions or contextual factors, meaning that their abnormality is dependent on the circumstances under which they are observed. Contextual anomalies can help uncover hidden relationships or dynamics that might not be apparent when considering the data points in isolation (Blázquez-García *et al.*, 2022).

Definition 6 (Contextual anomaly) *A conditional or contextual anomaly is a data point that, in a specific context of time, lies in a low probability region of $P^+(X)$. In this case, the normal law is a conditional distribution $P^+(X) \equiv P^+(X|T)$ that depends of the time variable, a contextual variable is a point that $\boldsymbol{x}_t$ where $p^+(\boldsymbol{x} \mid t) \leq \tau$, where τ is a sufficiently small threshold.*

It is important to notice that all of the anomaly types presented above can impact one (univariate) or more (multivariate) time-dependent variables (CHOI *et al.*, 2021). Furthermore, an entire univariate time series can be considered an anomaly. Also referred as a *metric anomaly* (ZHONG *et al.*, 2023), a *time series anomaly* arises when a specific dimension within a multivariate time series exhibits unusual behavior in comparison to the remaining dimensions of the time series (Blázquez-García *et al.*, 2022).

Finally, a yet under-explored variant of the metric anomaly is the *multiple time series anomaly* or anomaly in time series database (GUPTA *et al.*, 2014). In this case, the data is a set of multiple multivariate time series $\mathcal{X} = \{\mathbf{X}_n\}_{n=1}^{N}$ and an anomaly is an entire multivariate time series from this set that significantly deviate from the majority of the other time series.

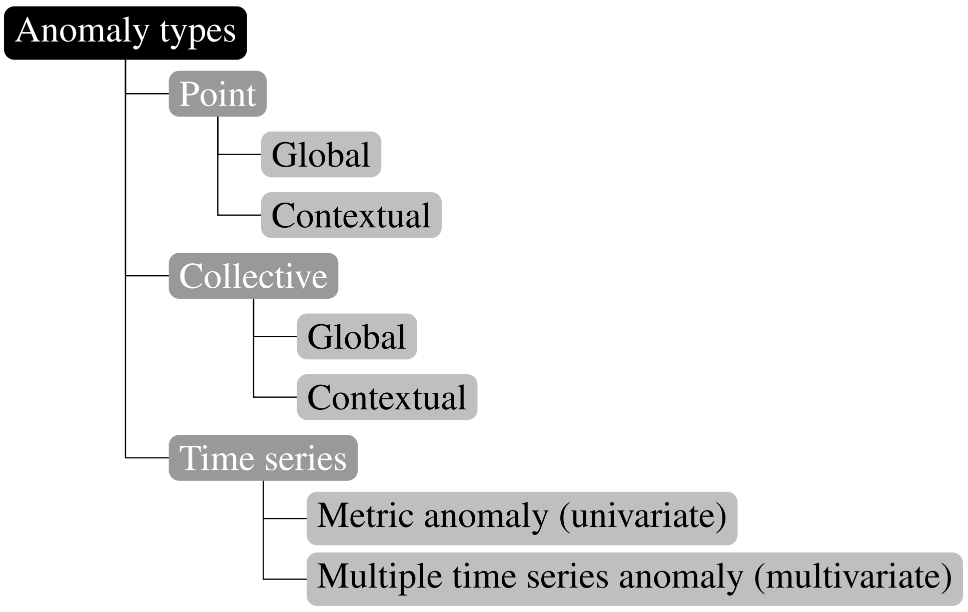


Figure 2 – Anomaly Types Based on Granularity. This diagram categorizes anomalies into three primary types: Point, Collective, and Time Series. Point and collective anomalies are divided into Global (deviations that are outliers considering the entire dataset) and Contextual (anomalies only within a specific context or subset of the data.). Time Series anomalies are classified into Metric anomalies (univariate) and Multiple time series anomalies (multivariate).

In Figure 2 we present an illustration that represents the relations of all types of anomaly presented. Furthermore, alternative perspectives may further subdivide anomalies into more specific categories, such as trend, seasonal, or drift anomalies (DARBAN *et al.*, 2025; TANG *et al.*, 2019; BAO *et al.*, 2019).

2.3 TSAD tasks definition

The previous discussion regarding anomaly detection in time series and the problem statement in the Introduction chapter allow us to define tasks involving this context formally. It is important to define the tasks so that we can understand the role of each method in solving this problem. The problem of detecting anomalies in time series can be solved using two main tasks: (i) anomaly scoring and (ii) score thresholding. Anomaly scoring is related to create some function in order to, for a given point, evaluate how close to abnormal it is, formally:

Task 1 (Anomaly scoring) *The task of anomaly scoring is to learn a scoring function $S : \mathbb{R}^D \rightarrow \mathbb{R}$ such that for any two scores $s_{t'} = S(\boldsymbol{x}'_t)$ and $s_t = S(\boldsymbol{x}_t)$, it must be true that if $s_{t'} > s_t$, then $\boldsymbol{x}_{t'}$ is more anomalous than $\boldsymbol{x}_t$.*

After creating scores for a set of points, we need to state which of these points can be considered anomalies. So, we need to define a threshold that separates anomalies from normal points for a given test set. This task can be defined as follows:

Task 2 (Score thresholding) *For a given anomaly scoring function S, the task of score thresholding is to select a threshold τ in order maximize the performance of the anomaly classification in the labeled test set $\mathbf{X}' = \{(\boldsymbol{x}_{t'}, y_{t'})\}_{t'=T+1}^{T'}$, i.e, minimize type I (false-positives) and type II (false-negatives) errors.*

These two tasks are usually solved separately by two different methods. In this thesis, we will consider these two tasks and focus on the first one.

2.4 Conclusion

In this chapter, we explore anomaly detection in time series, starting with discussing the taxonomy of rare events and the importance of understanding their differences to select the most appropriate methods for each situation. Next, we focus specifically on anomaly detection in multivariate time series, detailing the different types of anomalies, such as point, contextual, and collective, and their variants, and we finish by presenting the main related tasks. Through this analysis, we highlight the importance of a judicious approach when choosing detection techniques to ensure accurate and effective identification of anomalous events in complex time series.

3 NORMALIZING FLOWS

"Everything flows and nothing abides;
everything gives way and nothing stays fixed ."
(Heraclitus of Ephesus)

Initially studied by Agnelli *et al.* (2010), Tabak e Vanden-Eijnden (2010) and popularized by Rezende e Mohamed (2015), normalizing flows are generative models that leverage an invertible and differentiable nonlinear transformation to map samples from a simple, fixed base (or latent) distribution to a new, more complex distribution. NF models are particularly powerful for estimating complicated probability densities, offering exact inference and log-likelihood evaluation (KINGMA; DHARIWAL, 2018), making them valuable tools for AD (GUDOVSKIY *et al.*, 2022; DIAS *et al.*, 2020; YU *et al.*, 2021).

In this chapter, we describe the general NF framework, focusing on its application to anomaly detection. We start by exploring the evolution from the variable change rule to the concept of NFs, highlighting the fundamental principles and details of this method. We then describe the main transformations used in NFs. Additionally, we introduce conditional NF, which extends the basic NF framework to handle conditional distributions, providing flexibility in modeling data with varying contexts or conditions. Finally, we discuss about manifold learning in the context of NFs, examining how NFs can be leveraged to learn the underlying manifold of high-dimensional data, facilitating more effective anomaly detection by capturing the intrinsic structure of the data space.

3.1 From change of variables to normalizing flows

The *change of variables* is a mathematical technique used to simplify problems by substituting one set of variables with another. In calculus and differential equations, this method is commonly employed to transform integrals or equations into more manageable forms. For integrals, if you have an expression involving x, you can substitute x with a new variable u and express dx in terms of du using the chain rule. This substitution helps in rewriting the integral in terms of the new variable, often making it easier to solve. When making a substitution, it is essential to choose an appropriate substitution that simplifies the problem and to adjust the bounds of integration if you are dealing with definite integrals (KAPLAN, 2002).

Given a random variable u with a known Probability Density Function (PDF) $u \sim$

$p_U(u)$ and an observed random variable x with unknown PDF $x \sim p_X(x)$, we can use the definition of probability to relate these two variables through $\int p_X(x)dx = \int p_U(u)du = 1$ if, and only if $p_X(x)dx = p_U(u)du$. Using the change of variables technique we can find $p_X(x)$ by considering x as a function of u using a mapping such that $x = f(u)$ and $u = f^{-1}(x)$, thus the above expression can be rewritten as

$$p_X(x)dx = p_U(f^{-1}(x))df^{-1}(x) \;\Leftrightarrow\; p_X(x) = p_U(f^{-1}(x))\left|\frac{d}{dx}f^{-1}(x)\right|, \tag{3.1}$$

where $|\cdot|$ represents the absolute value, inserted in order to avoid negative values for $p_X(x)$, since the expression $\frac{d}{dx}f^{-1}(x)$ can generate negative values. An example of the utilization of this method is presented in Figure 3.

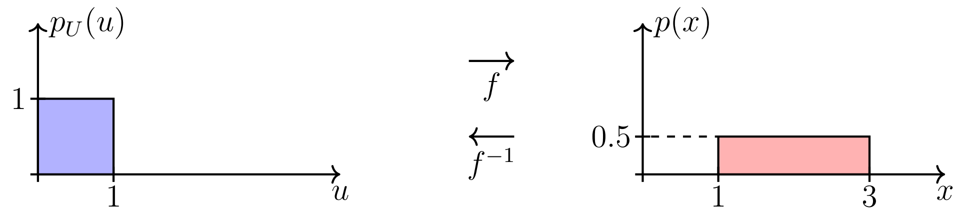


Figure 3 – Illustration of using change of variables formula to find an unknown distribution of a random variable x. Consider $u \sim \mathcal{U}(0,1)$, a random variable with uniform distribution between 0 and 1 and $x = f(u) = 2u + 1$ a linear transformation of the origin variable u, with inverse given by $f^{-1}(x) = \frac{x-1}{2}$ we then have $p_X(x) = 1\left|\frac{1}{2}\right| = 0.5$.

In the bi-variate case, considering two random variables $\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^2$ and $\boldsymbol{u} \in \mathcal{U} \subseteq \mathbb{R}^2$ with $\boldsymbol{x} \sim p_X(\boldsymbol{x})$ and $\boldsymbol{u} \sim p_U(\boldsymbol{u})$, related by a linear mapping $f : \mathbb{R}^2 \to \mathbb{R}^2$ such that $\boldsymbol{x} = f(\boldsymbol{u})$, $\boldsymbol{u} = f^{-1}(\boldsymbol{x}) = \mathbf{A}\boldsymbol{x} + \boldsymbol{b}$, where $\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$ and, for simplicity, $\boldsymbol{b} = [0, 0]$. This mapping transforms a 1×1 square in $\mathcal{X}$ to a parallelogram in $\mathcal{U}$ with area equals to $ad - bc$, as you can see in Figure 4. This change of area is equal to the determinant of the $\mathbf{A}$.

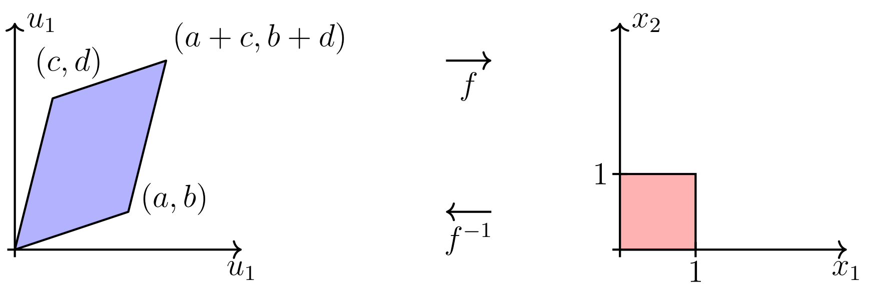


Figure 4 – Illustration of changing in area on a bi-variate variable in a linear mapping.

For the multivariate case, we can also use the intuition presented above for linear mappings without any modifications, but for non-linear transformations, the computation of the local linear change in volume is given through the determinant of the Jacobian matrix of the function f. Then, given two random variables $\boldsymbol{x} \in \mathbb{R}^D$ and $\boldsymbol{x} \in \mathbb{R}^D$ with $\boldsymbol{x} \sim p_X(\boldsymbol{x})$ and $\boldsymbol{u} \sim p_U(\boldsymbol{u})$, related by a mapping $f : \mathbb{R}^D \to \mathbb{R}^D$ such that $\boldsymbol{x} = f(\boldsymbol{u})$, and $f^{-1}(\boldsymbol{x}) = \boldsymbol{u}$, we can write that

$$p_X(\boldsymbol{x}) = p_U(f^{-1}(\boldsymbol{x})) \left|\det \mathbf{J}_{f^{-1}}(\boldsymbol{x})\right|, \tag{3.2}$$

where $\mathbf{J}_f(\boldsymbol{x})$ is the Jacobian matrix of the function f, a $D \times D$ matrix of all partial derivatives of f given by:

$$\mathbf{J}_f(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_D} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_D}{\partial x_1} & \cdots & \frac{\partial f_D}{\partial x_D} \end{bmatrix}.$$

An important property of invertible and differentiable mappings is that we can create a transformation simply by chaining multiple mappings (PAPAMAKARIOS *et al.*, 2021). Given K mappings $f_1, f_2, \ldots, f_K$, their composition $f_1 \circ f_2 \circ \cdots \circ f_k \circ \cdots \circ f_K$ is also a transformation, where $f_k(\boldsymbol{z}_{k-1}) = \boldsymbol{z}_k$, and $f_k^{-1}(\boldsymbol{z}_k) = \boldsymbol{z}_{k-1}$. Its inverse and Jacobian-determinant are given by:

$$(f_1 \circ f_2 \circ \cdots \circ f_k \circ \cdots \circ f_K)^{-1} = f_1^{-1} \circ f_2^{-1} \circ \cdots \circ f_k^{-1} \circ \cdots \circ f_K^{-1}, \tag{3.3}$$

$$\det \mathbf{J}_{(f_1 \circ f_2 \circ \cdots \circ f_k \circ \cdots \circ f_K)}(\boldsymbol{x}) = \prod_{k=1}^{K} \det \mathbf{J}_{f_k}(\boldsymbol{z}_{k-1}). \tag{3.4}$$

As a result of the above equation, we can establish a relationship between two random variables through their distributions, with invertible and differentiable transformations. furthermore, given our understanding that we can link together a sequence of transformations, we can craft what we refer to as *normalizing flows*.

An alternative strategy is constructing flows by chaining a number of mappings tending to infinite. Known as *infinitesimal flow* or *continuous-time flow*, in this case, the flow is constructed by defining an Ordinary Differential Equation (ODE) that describes how the flow evolves over time. In this document, we just consider discrete flows.

3.1.1 Normalizing Flows

Let $\boldsymbol{x} \sim p_X(\boldsymbol{x})$ and $\boldsymbol{u} \sim p_U(\boldsymbol{u})$ be two D-dimensional random variables, where $p_X(\boldsymbol{x})$ is an unknown distribution and $p_U(\boldsymbol{u})$ is a simple and known distribution, such as a multivariate Gaussian, and $f_1, f_2, \ldots, f_K$ be a composition of K successive invertible and differentiable mappings, which is called *bijective*. The generative process in the most comprehensive flow-based model is defined as follows:

$$\boldsymbol{x} = f_1 \circ f_2 \circ \cdots \circ f_k \circ \cdots \circ f_K(\boldsymbol{u}), \quad \text{where} \quad \boldsymbol{u} \sim p_U(\boldsymbol{u}). \tag{3.5}$$

Applying the change of variables rule we can express the log-likelihood of $p_X(\boldsymbol{x})$ in terms of $p_U(\boldsymbol{u})$ and $f_1, f_2, \ldots, f_K$, simply by replacing (3.4) in (3.2) to have

$$\log p_K(\boldsymbol{z}_K) = \log p_0(\boldsymbol{z}_0) + \sum_{k=1}^{K} \log \left| \det \frac{\partial \boldsymbol{z}_{k-1}}{\partial \boldsymbol{z}_k} \right|, \tag{3.6}$$

where $\boldsymbol{x} \triangleq \boldsymbol{z}_K \sim p_K(\boldsymbol{z}_K)$, $\boldsymbol{u} \triangleq \boldsymbol{z}_0 \sim p_0(\boldsymbol{z}_0)$ and $\boldsymbol{z}_k = f_k(\boldsymbol{z}_{k-1}), \forall k = 1, 2, \ldots, K$. It is important to notice that, depending on the simplicity of the determinant Jacobian computing, we may want to express the above expression in terms of $\frac{\partial \boldsymbol{z}_k}{\partial \boldsymbol{z}_{k-1}}$ instead of $\frac{\partial \boldsymbol{z}_{k-1}}{\partial \boldsymbol{z}_k}$, since, applying the inverse function theorem (HÖRMANDER, 1990) and the determinant of the inverse matrix (AXLER, 2014) we find that

$$\log p_K(\boldsymbol{z}_K) = \log p_0(\boldsymbol{z}_0) - \sum_{k=1}^{K} \log \left| \det \frac{\partial \boldsymbol{z}_k}{\partial \boldsymbol{z}_{k-1}} \right|. \tag{3.7}$$

The term “normalization” denotes the process whereby the inverse flow $f_K^{-1}, \ldots, f_1^{-1}$ takes samples from $p_K(\boldsymbol{z}_K)$ and converts them into samples of a previously established density $p_0(\boldsymbol{z}_0)$. In Figure 5, we illustrate a one-dimensional NF.

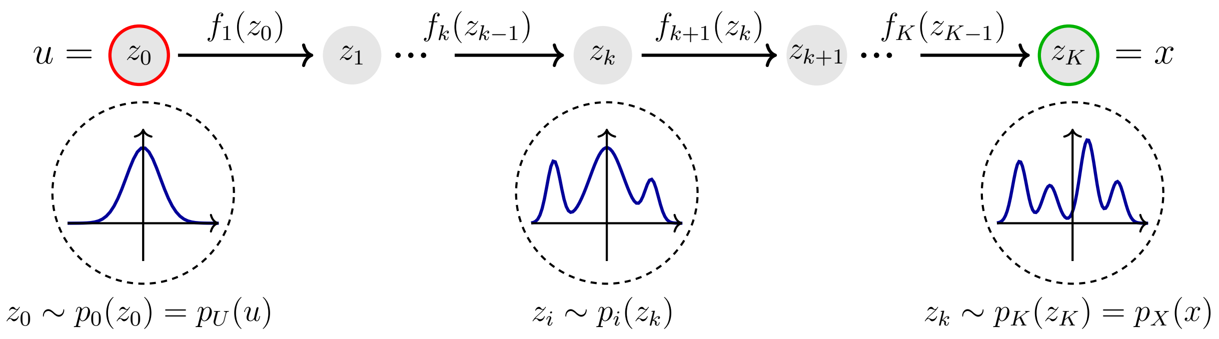


Figure 5 – Illustration of a normalizing flow model, transforming a simple distribution $p_0(z_0)$ to a complex one $p_k(z_k)$ step by step.

3.2 Transformations and determinants

A flow-based model can be used for two operations: sampling by using the Equation 3.5, and density evaluating by using the Equation 3.6 or 3.7. As mentioned in Kobyzev *et al.* (2021), an NF needs to satisfy some conditions in order to be used in practice: (i) be invertible for sampling and likelihood computation; (ii) be sufficiently expressive to map a basis distribution into the distribution of interest; (iii) be computationally efficient, both to forward and backward mapping and also in terms of Jacobian determinant computation.

In consequence of the conditions mentioned above, numerous advancements in NF involve the creation of transformations that possess manageable Jacobian determinants while still offering remarkably adaptable mappings through repeated composition (HENTER *et al.*, 2020). In the next subsections, we will present the main types of flow, their properties, advantages, and disadvantages. Besides that, we will discuss the use of normalizing flows in the context of manifold learning in order to facilitate the computation of anomaly scores.

3.2.1 Element-wise flows

The simplest transformation we can use to create an NF is element-to-element bijections in the form

$$f(\boldsymbol{u}) = [h(u_1), h(u_2), \ldots, h(u_D)]^\top \tag{3.8}$$

where $h : \mathbb{R} \to \mathbb{R}$ is a scalar-valued bijection, usually a smooth non-linearity, with valid derivative and inverse h^{-1}. As the Jacobian of this transformation is a diagonal matrix of the derivatives of the function h, the determinant can be computed by multiplying its diagonal elements,

$$\det \mathbf{J}_f(\boldsymbol{u}) = \det \operatorname{diag}\{h(u_1), h(u_2), \ldots, h(u_D)\}, \tag{3.9}$$

$$= \prod_{d=1}^{D} h'(u_d). \tag{3.10}$$

This transformation can be generalized by using different bijective functions for each dimension. The main disadvantage of this type of bijection is that it cannot express any correlation between dimensions (KOBYZEV *et al.*, 2021).

3.2.2 Linear flows

This type of mapping is created using an affine transformation in the form:

$$f(\boldsymbol{u}) = \mathbf{A}\boldsymbol{u} + \boldsymbol{b}, \tag{3.11}$$

where $\mathbf{A} \in \mathbb{R}^D \times \mathbb{R}^D$ and $\boldsymbol{b} \in \mathbb{R}^D$ are free parameters. The determinant of the Jacobian of this type of transformation is simply $\det \mathbf{A}$ that exhibits a computational complexity approaching $\mathcal{O}(D^3)$, which is not a problem for a small D. For a large D we can restrict the form of $\mathbf{A}$, avoiding expensive calculations of Jacobian determinants. Unfortunately, this also reduces the flow's expressiveness.

An important propriety that we can use to construct linear flows that reduces its cost is that the determinant of triangular matrices, as in the element-wise case, is computed by multiplying its diagonal (AXLER, 2014). Thus, we can simply force $\mathbf{A}$ to be diagonal and replace in Equation 3.11, in this case:

$$\det \mathbf{J}_f(\boldsymbol{u}) = \det \mathbf{A} \quad \Leftrightarrow \quad \det \mathbf{J}_f(\boldsymbol{u}) = \prod_{d=1}^{D} a_{d,d}, \tag{3.12}$$

where $a_{d,d}$ for $d = 1, 2, \dots, D$ is the diagonal element of the matrix $\mathbf{A}$.

As alternative to use a triangular matrix in linear mappings by considering $\mathbf{A}$ as a matrix of permutations. In this case, $\boldsymbol{b}$ becomes a vector of zeros and $\mathbf{A}$ becomes a binary matrix featuring precisely one entry of 1 in every row and column, with 0s elsewhere. The determinant of a permutation matrix is equal to 1. As shown by (DINH *et al.*, 2017), this type of mapping will be useful in combination with another type of transformation.

Another possibility is to use an orthogonal matrix, which extends the concept of permutation because the collection of permutation matrices is encompassed within the set of orthogonal matrices (HOOGEBOOM *et al.*, 2019). The determinant of this type of matrix is also equals to 1. However, as presented by Shepard *et al.* (2015), the effective parametrization of this matrix type can be challenging.

Proposed by Kingma e Dhariwal (2018) as an alternative of limiting the form of $\mathbf{A}$, we can create a linear transformation by using the LU factorization, where $\mathbf{A} = \mathbf{PL}(\mathbf{U} + \operatorname{diag}\{\boldsymbol{v}\})$, such that $\mathbf{P}$ corresponds to a permutation matrix, $\mathbf{L}$ is a lower triangular matrix with ones on its diagonal, $\mathbf{U}$ is an upper triangular matrix with zeros on its diagonal, and $\boldsymbol{v}$ denotes a vector. The Jacobian-determinant of this mapping is the product elements of $\boldsymbol{v}$.

3.2.3 Autoregressive flows

Originally proposed by Kingma *et al.* (2016), autoregressive flows stand out as one of the most popular categories of flows, similar to nonlinear versions of multiplication by a triangular matrix, as described in Kobyzev *et al.* (2021).

Let $\boldsymbol{u}, \boldsymbol{x} \in \mathbb{R}^D$ two random variables related by a function $f : \mathbb{R}^D \to \mathbb{R}^D$, such that is a function which outputs each entry of $\boldsymbol{x} = f(\boldsymbol{u})$ conditioned on the previous entries of the input, formally

$$x_d = \tau(u_d; \boldsymbol{h}_d), \tag{3.13}$$

$$\text{such that } \boldsymbol{h}_d = c_d(\boldsymbol{u}_{<d}), \tag{3.14}$$

where for each $d = 2, \ldots, D$, $\tau : \mathbb{R} \to \mathbb{R}$ is called *transformer* and $c_d : \mathbb{R}^{d-1} \to \mathbb{R}$ is called d-th *conditioner*, and h_1 is a constant. The transformer delineates the flow's behavior concerning u_d, transforming it to yield x_d as the output as described by Papamakarios *et al.* (2021).

It is easy to see that since each output x_d depends only on $\boldsymbol{u}_{<d}$, the Jacobian matrix of the autoregressive transform $f(\boldsymbol{u})$ is a lower triangular matrix, and its determinant is just a product of its diagonal entries:

$$\det \mathbf{J}_f(\boldsymbol{u}) = \prod_{d=1}^{D} \det \mathbf{J}_\tau(u_d; \boldsymbol{h}_d), \tag{3.15}$$

and, it is also easy to check that the above construction can be invertible simply by using

$$u_d = \tau^{-1}(x_d; \boldsymbol{h}_d), \tag{3.16}$$

$$\text{such that } \boldsymbol{h}_d = c_d(\boldsymbol{u}_{<d}). \tag{3.17}$$

To implement an autoregressive flow, we just need to implement a transformer τ and a conditioner c_d (PAPAMAKARIOS *et al.*, 2021). The transformer, characterized by $\boldsymbol{h}_d$, represents a strictly monotonic function of u_d while the conditioner c_d can be implemented as any function with input $\boldsymbol{u}_{<d}$ and output $\boldsymbol{h}_d$.

3.3 Practical considerations of implementing flows

Regardless of the type of mappings, the usual training criterion for a flow-based generative model $p_X(\boldsymbol{x}; \boldsymbol{\theta})$, with model's parameters $\boldsymbol{\theta} = \{\boldsymbol{\phi}, \boldsymbol{\psi}\}$, where ϕ are the parameters of f and ψ are the parameters of $p_U(\boldsymbol{u})$, is simply the forward the Kullback-Leibler Divergence (KL) divergence between the target distribution $p^\star(\boldsymbol{x})$ and the flow-based model $p(\boldsymbol{x}; \boldsymbol{\theta})$. This can be written as follows:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}) &= D_{\mathrm{KL}}\left[p^{\star}_X(\boldsymbol{x}) \| p_X(\boldsymbol{x};\boldsymbol{\theta})\right] \\ &= -\mathbb{E}_{p^{\star}_X(\boldsymbol{x})}\left[\log p_X(\boldsymbol{x};\boldsymbol{\theta})\right] + \text{const.} \\ &= -\mathbb{E}_{p^{\star}_X(\boldsymbol{x})}\left[\log p_U(f^{-1}(\boldsymbol{x};\boldsymbol{\phi});\boldsymbol{\psi}) + \log\left|\det \mathbf{J}_{f^{-1}}(\boldsymbol{x};\boldsymbol{\phi})\right|\right] + \text{const.}\end{aligned} \tag{3.18}$$

where, for a given set of training samples $\mathcal{X} = \{\boldsymbol{x}_n \sim p^{\star}(\boldsymbol{x})\}_{n=1}^{N}$, estimate the expectation over $p^{\star}(\boldsymbol{x})$ by Monte Carlo is equivalent to fitting the model to the samples in $\mathcal{X}$ by maximum likelihood estimation. This approximation can be written as follows:

$$\mathcal{L}(\boldsymbol{\theta}) \approx -\frac{1}{N}\sum_{n=1}^{N} \log p_U(f^{-1}(\boldsymbol{x}_n;\boldsymbol{\phi});\boldsymbol{\psi}) + \log\left|\det \mathbf{J}_{f^{-1}}(\boldsymbol{x}_n;\boldsymbol{\phi})\right| + \text{const.} \tag{3.19}$$

In practice, we often optimize the $\boldsymbol{\theta}$ parameters iteratively with gradient-based stochastic methods. Consequently, we have the same problems found in any deep method trained similarly. In order to maintain stable gradients throughout the flow, Kingma e Dhariwal (2018) proposed a mapping, called *activation normalization* that can be inserted between consecutive sub-transformations and treated simply as another member of the composition. This mapping act like a *batch normalization* for a deep neural networks (IOFFE; SZEGEDY, 2015).

Another important contribution, created in order to solve a practical problem, the cost of working with a high number of dimensions, is known as *multi-scale architecture*. Introduced by Dinh *et al.* (2017), acting as a skip-connection, when going from $\boldsymbol{x} = \boldsymbol{z}_K$ to $\boldsymbol{u} = \boldsymbol{z}_0$, some sub-dimensions of $\boldsymbol{z}_k$ are fixed and no additional transformations are applied. This is usually done by halving the dimensions after a random permutation (KOBYZEV *et al.*, 2021).

Over the years, several types of transformers have been proposed to approach the most different applications like affine transformers (DINH *et al.*, 2015; DINH *et al.*, 2017), combination-based transformers (HUANG *et al.*, 2018; CAO *et al.*, 2019), integration-based transformers (WEHENKEL; LOUPPE, 2019; JAINI *et al.*, 2019) and spline-based (MÜLLER *et al.*, 2019; DOLATABADI *et al.*, 2020) transformers. Below, we present the most known flow architectures, the Real NVP (DINH *et al.*, 2017) that uses an affine transform and a coupling layer, and the MAF (PAPAMAKARIOS *et al.*, 2017), which uses a masked conditioner and an affine transformer. We also present an overview of the methods for constructing flows based on finite compositions that can be seen in Figure 6.

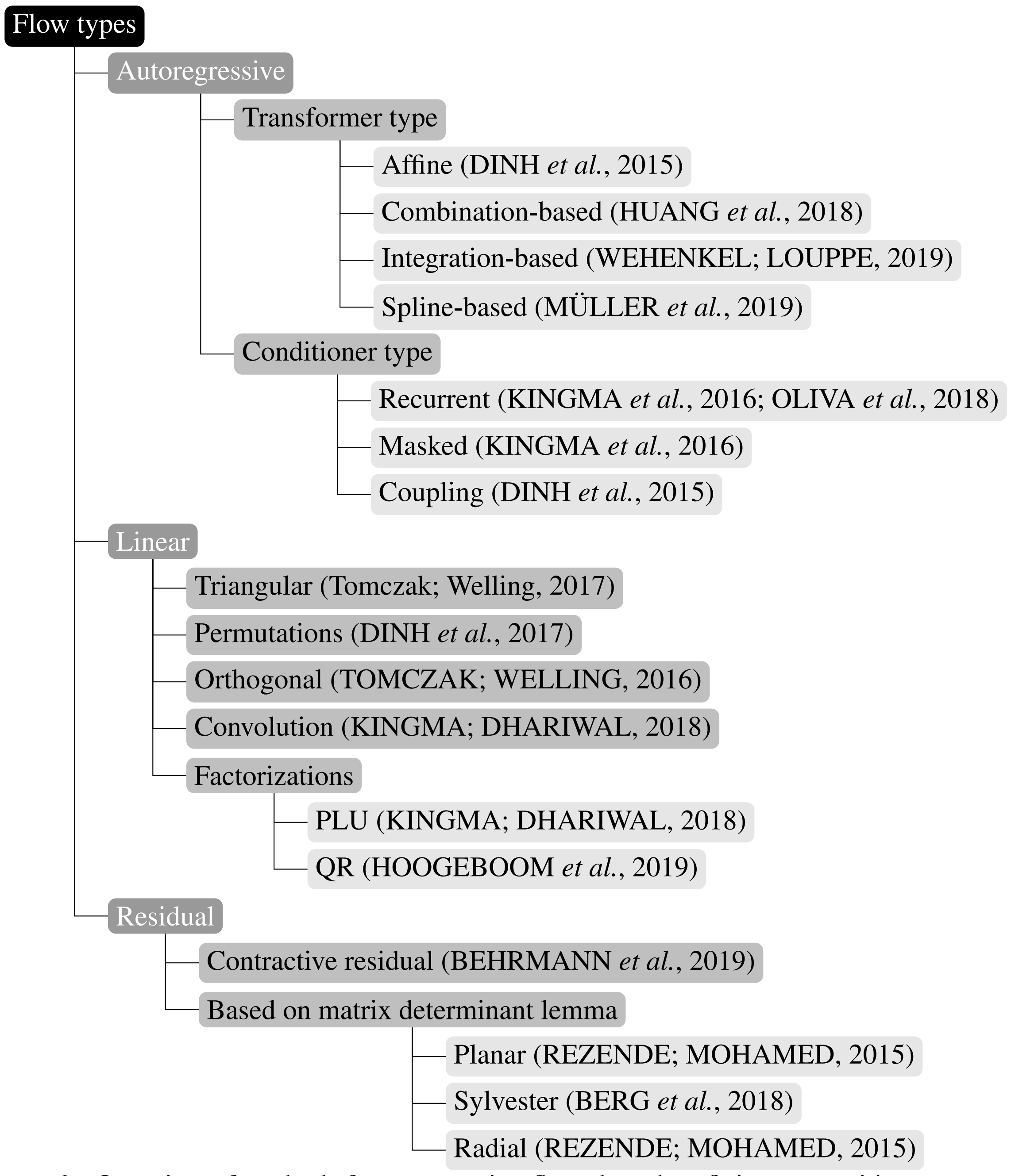

Flow types
Autoregressive
Transformer type
Affine (DINH *et al.*, 2015)
Combination-based (HUANG *et al.*, 2018)
Integration-based (WEHENKEL; LOUPPE, 2019)
Spline-based (MÜLLER *et al.*, 2019)
Conditioner type
Recurrent (KINGMA *et al.*, 2016; OLIVA *et al.*, 2018)
Masked (KINGMA *et al.*, 2016)
Coupling (DINH *et al.*, 2015)
Linear
Triangular (Tomczak; Welling, 2017)
Permutations (DINH *et al.*, 2017)
Orthogonal (TOMCZAK; WELLING, 2016)
Convolution (KINGMA; DHARIWAL, 2018)
Factorizations
PLU (KINGMA; DHARIWAL, 2018)
QR (HOOGEBOOM *et al.*, 2019)
Residual
Contractive residual (BEHRMANN *et al.*, 2019)
Based on matrix determinant lemma
Planar (REZENDE; MOHAMED, 2015)
Sylvester (BERG *et al.*, 2018)
Radial (REZENDE; MOHAMED, 2015)

Figure 6 – Overview of methods for constructing flows based on finite compositions.

3.4 Real NVP

Real-valued Non-Volume Preserving (RealNVP) is a type of NF that uses a bijection called *coupling layer* that transforms only some input dimensions via functions that depend on the untransformed dimensions (DINH *et al.*, 2017). If $1 : d$ denotes the sequential indexes of the d untransformed dimensions, the components of the layer output $\boldsymbol{x}$ are given by:

$$\boldsymbol{x}_{1:d} = \boldsymbol{u}_{1:d}, \tag{3.20}$$

$$\boldsymbol{x}_{d+1:D} = \boldsymbol{u}_{d+1:D} \odot \exp\left(\sigma(\boldsymbol{u}_{1:d})\right) + \mu(\boldsymbol{u}_{1:d}), \tag{3.21}$$

where $\sigma, \mu : \mathbb{R}^d \to \mathbb{R}^{D-d}$ respectively represent scale and translation functions parametrized by neural networks, and $\odot$ is the element-wise product operator. The elements in each flow are permuted to different orders, allowing all of the inputs to have a chance to be altered. The Jacobian matrix of these transformations can be calculated using the following:

$$\mathbf{J}_f(\boldsymbol{u}) = \begin{bmatrix} \mathbf{I}_d & \mathbf{0}_{d\times(D-d)} \\ \frac{\partial \boldsymbol{x}_{d+1:D}}{\partial \boldsymbol{u}_{1:d}} & \operatorname{diag}\left(\exp\left(\sigma(\boldsymbol{u}_{1:d})\right)\right) \end{bmatrix}, \tag{3.22}$$

where $\mathbf{I}_d \in \mathbb{R}^{d\times d}$ is the d-order identity matrix, $\mathbf{0}_{d\times(D-d)} \in \mathbb{R}^{d\times(D-d)}$ is a zero matrix and $\operatorname{diag}\left(\exp\left(\sigma(\boldsymbol{u}_{1:d})\right)\right) \in \mathbb{R}^{(D-d)\times(D-d)}$ is a diagonal matrix whose elements are equal to the vector $\exp\left(\sigma(\boldsymbol{u}_{1:d})\right)$. The Jacobian matrix in Eq. (3.22) is triangular, thus, its determinant is a simple product of the diagonal terms:

$$\det \mathbf{J}_f(\boldsymbol{u}) = \prod_{j=1}^{D-d} \exp\left(\sigma(\boldsymbol{x}_{1:d})\right)_j \tag{3.23}$$

$$= \exp\left(\sum_{j=1}^{D-d} \sigma(\boldsymbol{u}_{1:d})_j\right). \tag{3.24}$$

Since the computation of the Jacobian determinant of the mentioned transformations does not involve calculating the inverse of the functions $\sigma(\cdot)$ and $\mu(\cdot)$, such functions can be arbitrarily complex, usually a deep neural network (DINH *et al.*, 2017). All the model parameters (i.e., the networks' weights) are jointly optimized via minimization of the Equation (3.19) via stochastic gradient descent methods.

3.5 MAF

We can decompose any joint density $p(\boldsymbol{x})$ of high-dimensional data into a product of one-dimensional conditionals using the chain rule of probabilities:

$$p(\boldsymbol{x}) = \prod_{d=1}^{D} p(x_d|x_1, x_2, \cdots, x_{d-1}) \;\Leftrightarrow\; p(\boldsymbol{x}) = \prod_{d=1}^{D} p(x_d|\boldsymbol{x}_{<d}). \tag{3.25}$$

Proposed by Papamakarios *et al.* (2017), the masked autoregressive flow uses the above autoregressive constraint to model the probability density whose conditionals are parameterized as single Gaussians. Thus, the d-th conditional probability is given by

$$p(x_d|\boldsymbol{x}_{<d}) = \mathcal{N}\left(x_d|\mu_d(\boldsymbol{x}_{<d}), (\exp\left(\alpha_d(\boldsymbol{x}_{<d})\right))^2\right), \tag{3.26}$$

where $\mu_d, \alpha_d : \mathbb{R}^{d-1} \mapsto \mathbb{R}$ are two unconstrained scalar functions that compute the mean and log-standard deviation of the d-th conditional given all previous variables. The bijective mapping of MAF generates each y_d conditioned on the past dimensions $\boldsymbol{u}_{1:d-1}$,

$$x_d = u_d \exp\left(\alpha_d(\boldsymbol{u}_{<d})\right) + \mu_d(\boldsymbol{u}_{<d}). \tag{3.27}$$

As a consequence of the autoregressive nature of this transformation, the dimension d of the resulting variable $\boldsymbol{x}$ depends only on the $1 : d$ dimensions of the input variable $\boldsymbol{u}$. Thus, the Jacobian matrix of this transformation is triangular (KINGMA *et al.*, 2016), and its determinant is equal to the product of its diagonal terms:

$$\det \mathbf{J}_f(\boldsymbol{u}) = \prod_{d=1}^{D} \exp\left(\alpha_d(\boldsymbol{u}_{<d})\right) \;\Leftrightarrow\; \det \mathbf{J}_f(\boldsymbol{u}) = \exp\left(\sum_{d=1}^{D} \alpha_d(\boldsymbol{u}_{<d})\right). \tag{3.28}$$

As in the RealNVP, the functions $\mu_d(\cdot)$ and $\alpha_d(\cdot)$ can be arbitrarily complex. In the MAF model, these functions are implemented by an efficient feedforward network called Masked Autoencoder for Distribution Estimation (MADE) that takes $\boldsymbol{x}$ as input and outputs the means and log-standard deviations for all dimensions in a single network pass (GERMAIN *et al.*, 2015).

The nature of MAF transformations allows more flexible generalizations when compared to the RealNVP model. As we can see, if for the first $j \leq d$ dimensions, we fix $\mu_j = \alpha_j = 0$ and apply the MAF transformations into the other $j > d$ dimensions, the MAF structure becomes equivalent to the RealNVP. Besides, we can see the coupling layer of the RealNVP as a special case of the MAF transformation (PAPAMAKARIOS *et al.*, 2017).

3.6 Conclusion

In this chapter we have defined how to build NF-based models starting from the change of variables rule. We also describe the main transformations and discuss about the determinant computation, present some practical considerations of implementing flows, the most known flow architectures, and shows how some authors approach to use NF models in the manifold learning problem. In the next chapter we will use the described class of models so far to approach the anomaly detection and diagnosis in multivariate time series data.

4 ANOMALY DETECTION IN TIME SERIES DATABASE

“The only reason for time is so that everything doesn’t happen at once.”

(Albert Einstein)

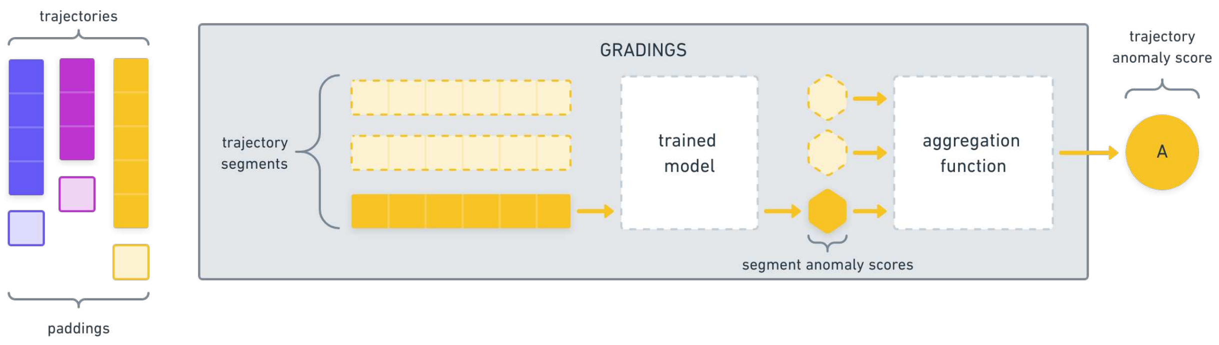


Figure 7 – An overview of the GRADINGS procedure for anomaly detection in trajectory data. The process involves the creation of segmenting trajectories (with padding at the end), calculating segment anomaly scores using a trained model, and aggregating these scores into a single trajectory anomaly score using an aggregation function.

This chapter is adapted from the work “Anomaly Detection in Trajectory Data with Normalizing Flows” (DIAS *et al.*, 2020). Aggre**g**ated **a**nomaly **d**etection with normaliz**ing** flow**s** (GRADINGS) is a framework for anomaly detection in time series database, applied to trajectory data, which is based on estimating the density for each segment of the trajectories, and then, aggregate the segments’ likelihoods into a single anomaly score. Such a strategy enables the handling of possibly large sequences of different lengths. We evaluate our methodology using real-world trajectory data and compare it with more traditional anomaly detection techniques. The promising results obtained in the performed computational experiments indicate the feasibility of the GRADINGS, especially the variant that considers autoregressive normalizing flows. In Figure 7 we present an overview of the GRADINGS procedure.

The widespread availability of spatial data collection devices, from specialized remote sensors to common GPS-enabled smartphones, has led to the development of numerous location-based applications. Regardless of whether the object being tracked is a vehicle, an animal, or a person, trajectory data generally consists of a series of ordered points representing the object’s movement (ZHENG, 2015). As described in the above chapters, an anomaly is a data point that significantly differs from the rest of the data set (AGGARWAL, 2013; TENG, 2010). In this context, detecting anomalies is crucial for monitoring spatial events and identifying unusual behaviors, an equivalent task to solve Problem 1.

Meng *et al.* (2019) outline a conventional taxonomy for anomaly detection, which includes methods based on classification, clustering, distance, density, and statistics. We focus on the latter, utilizing a model-based approach primarily within a probabilistic density estimation framework to interpret the available data. Anomalies are identified by assessing the model's fit to new data points. This unsupervised learning method does not require labeled data. However, probabilistic density estimation methods for trajectory anomaly detection often assume simple distributions like a multivariate Gaussian (HAZEL, 2000). Even with more flexible models like Gaussian Mixture Model (GMM), determining the number of components remains challenging (BASHARAT *et al.*, 2008; LI *et al.*, 2016).

A trajectory dataset is a set $\mathcal{T} = \{\mathbf{X}_n\}_{n=1}^{N}$ of multivariate time series (Definition 1), such that each trajectory consists of a sequence of GPS points (i.e., latitude, longitude, and timestamp) generated by a moving object on a monitoring system. Due to the nature of this application, data can be high-dimensional. Additionally, individual data examples can vary in length, making direct comparisons challenging. Our approach addresses both issues by segmenting the trajectories into fixed-size segments. We then use a flow-based generative model to estimate the probability density of these segments. Segments from normal trajectories are expected to show higher likelihoods compared to those from anomalous ones. During testing, an anomaly score for a new trajectory is calculated by aggregating its segment scores.

In contrast to contemporary approaches by Yamaguchi *et al.* (2019), Iwata e Yamanaka (2019), we evaluate flow-based anomaly detection models in the context of trajectory data, which is inherently sequential. Hence, we use in our evaluations the masked autoregressive flows in order to directly model the conditional distributions of the input variables (PAPAMAKARIOS *et al.*, 2017). Notably, to our knowledge, this work represents the first evaluation of NF-based models for anomaly detection in trajectory data. We assess the GRADINGS approach using real-world trajectories from the Microsoft GeoLife dataset. The experimental results demonstrate the practicality of our solution. Our framework, particularly the variant utilizing the MAF model, outperforms standard techniques such as the Gaussian mixture models and the Local Outlier Factor (LOF) method in anomaly detection.

4.1 Related work

In this section, we describe two of the most well-known classical techniques used in anomaly detection problems: the Local Outlier Factor and the Gaussian Mixture Model. The

LOF algorithm is an unsupervised distance-based anomaly detection method that calculates an anomaly score by comparing the local density of a sample to that of its neighbors. The local density is inversely related to the average distance from the sample to its surrounding points. This comparison helps identify outliers, with lower local densities indicating higher likelihoods of being anomalies, as described in Breunig *et al.* (2000).

Let $\mathcal{X}$ be a set of data points. The set of K-nearest neighbors of $\boldsymbol{x} \in \mathcal{X}$ is denoted by $\mathcal{N}_{\boldsymbol{x}}^{k}$ and defined as $\mathcal{N}_{\boldsymbol{x}}^{K} \triangleq k\mathrm{NN}\left(K, \boldsymbol{x}, \mathcal{X} \setminus \{\boldsymbol{x}\}\right)$, where $k\mathrm{NN}\left(\cdot,\cdot,\cdot\right)$ is the result of a K-nearest neighbor query (ROUSSOPOULOS *et al.*, 1995; PAPADIAS; TAO, 2009). Then, we define the K-distance neighborhood $\mathrm{KD}(\boldsymbol{x})$ of a sample $\boldsymbol{x} \in \mathcal{X}$ as $\mathrm{KD}(\boldsymbol{x}) \triangleq \max_{\boldsymbol{u} \in \mathcal{N}_{\boldsymbol{x}}^{K}} \|\boldsymbol{x} - \boldsymbol{u}\|$, where $\|\cdot\|$ is the Euclidean distance. We use the above to define the reachability distance $\mathrm{RD}(\boldsymbol{x}, \boldsymbol{u})$ of $\boldsymbol{x}$ with respect to another sample $\boldsymbol{u} \in \mathcal{X}$ as $\mathrm{RD}(\boldsymbol{x}, \boldsymbol{u}) \triangleq \max\left\{\mathrm{KD}(\boldsymbol{x}), \|\boldsymbol{x} - \boldsymbol{u}\|\right\}$. The local reachability density of $\boldsymbol{x}$ with respect to $\boldsymbol{u}$ is then denoted by $\mathrm{LRD}(\boldsymbol{x}, \boldsymbol{u})$ and defined as

$$\mathrm{LRD}(\boldsymbol{x}) \triangleq \frac{K}{\sum_{\boldsymbol{u} \in \mathcal{N}_{\boldsymbol{x}}^{K}} \mathrm{RD}(\boldsymbol{x}, \boldsymbol{u})}.$$

The LOF anomaly score can be finally formalized as the average ratio of local reachability densities with respect to $\boldsymbol{x}$ and its K-neighborhood:

$$A(\boldsymbol{x}) = \frac{1}{K} \sum_{\boldsymbol{u} \in \mathcal{N}_{\boldsymbol{x}}^{K}} \frac{\mathrm{LRD}(\boldsymbol{x})}{\mathrm{LRD}(\boldsymbol{u})}. \tag{4.1}$$

Note that the above score measures the local density deviation of a given data point with respect to its neighbors.

Differently from LOF, gaussian mixture models uses a linear combination of Gaussian density functions to approximate an unknown probability distribution and uses the negative log-likelihood of each testing data as an anomaly score, i.e., $A(\boldsymbol{x}) = -\ln p(\boldsymbol{x})$. The parameters of each component are usually adjusted using the expectation maximization (EM) algorithm (DEMPSTER *et al.*, 1977).

Consider a data set $\mathcal{D} = \{\boldsymbol{x}_n\}_{n=1}^{N}$, where $\boldsymbol{x}_n \in \mathbb{R}^D$. We assume that the points from $\mathcal{D}$ are generated in an i.i.d. fashion from an underlying density $p(\boldsymbol{x})$. Furthermore, suppose that $p(\boldsymbol{x})$ is defined as a finite mixture model with K components:

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{4.2}$$

where $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ is a multivariate Gaussian density with mean vector $\boldsymbol{\mu}_k$ and covariance matrix $\boldsymbol{\Sigma}_k$; $\{\pi_k\}_{k=1}^{K}$ are the mixture weights, which are restricted to be non-negative and sum up to 1, i.e., $\sum_{k=1}^{K} \pi_k = 1$. The mixture weights represent the probability that a randomly selected data point $\boldsymbol{x}$ was generated by the component k. After optimizing the GMM parameters via the EM algorithm, Eq. (4.2) can be directly applied as an anomaly score for new data points.

4.2 Method

We can compute an anomaly score for a sequential data type sample either directly or by first computing scores for local subsections and then aggregating them. Segment-based techniques usually perform better when compared to direct detection methods (GUPTA *et al.*, 2014). Furthermore, they enable the handling of large sequences of different lengths. As follows, we detail our proposal, named GRADINGS, which consists of three main steps.

In the first step, GRADINGS transforms the set of trajectories into a set of trajectory subsequences. Here, a trajectory is defined as:

Definition 7 (Trajectory) *A trajectory $\mathbf{T}_m \in \mathcal{T}$ with size L_m is defined as a finite ordered sequence*

$$\mathbf{T}_m \triangleq \left(\boldsymbol{q}_1^{(m)}, \boldsymbol{q}_2^{(m)}, \cdots, \boldsymbol{q}_l^{(m)}, \cdots, \boldsymbol{q}_{L_m}^{(m)}\right), \tag{4.3}$$

where $\boldsymbol{q}_l^{(m)} = \left(q_{l,1}^{(m)}, q_{l,2}^{(m)}, q_{l,3}^{(m)}\right)$ is a location point, *such that $q_{l,1}^{(m)}, q_{l,2}^{(m)}, q_{l,3}^{(m)}$ are respectively the l-th latitude, l-th longitude, and l-th timestamp of the trajectory $\mathbf{T}_m$. Furthermore, we have $q_{l_0,3} < q_{l_1,3}$, for all $l_0 < l_1$, which ensures a temporal ordered sequence of points.*

Given a set of trajectories $\mathcal{T} = \{\mathbf{X}_m\}_{m=1}^{M}$, the transformed set can be defined by

$$\mathcal{X} = \bigcup_{n=1}^{N} \left\{\boldsymbol{x}_n = \delta\left(\mathbf{S}_i^{(m)}\right)\Big|_{i=1}^{L_m-W+1}\right\}, \tag{4.4}$$

where $\boldsymbol{x}_n \in \mathbb{R}^D$, $1 \leqslant n \leqslant N = \sum_{n=1}^{N}(L_m - W)$, and $\delta : \mathbb{R}^{W\times 3} \to \mathbb{R}^D$ is a function that flattens a segment into a D-dimensional row vector, where $D = 3W$, i.e.,

$$\delta\left(\mathbf{S}_i^{(m)}\right) = \left(q_{i,1}^{(m)}, q_{i,2}^{(m)}, q_{i,3}^{(m)}, \cdots, q_{i+W,1}^{(m)}, q_{i+W,2}^{(m)}, q_{i+W,3}^{(m)}\right). \tag{4.5}$$

The second step consists in estimating the distribution $p(\cdot)$ from the available trajectory segments. This step is performed by using one of the NF generative models described in Chapter 3. At this point, the GRADINGS is able to compute the anomaly degree for any trajectory segment, denoted by $\alpha\left(\mathbf{S}_i^{(m)}\right)$:

$$\alpha\left(\mathbf{S}_i^{(m)}\right) = -\ln p\left(\delta\left(\mathbf{S}_i^{(m)}\right)\right). \tag{4.6}$$

In the last step, we aggregate the anomaly scores of the segments to compute a single anomaly score for the trajectory. More specifically, given a trajectory $\mathbf{T}_m$, its anomaly score, denoted by $A\left(\mathbf{T}_m\right)$, can be computed using an aggregation function φ that combines the anomaly degree of each segment $\mathbf{S}_i^{(n)}$ in the trajectory $\mathbf{T}_m$, i.e.,

$$A\left(\mathbf{T}_m\right) = \varphi\left(\left\{\alpha\left(\mathbf{S}_i^{(m)}\right)\right\}_{i=1}^{L_m-W+1}\right). \tag{4.7}$$

Possible choices for the aggregation function φ include the median or the average. An overview of the GRADINS procedure is presented in Figure 7.

4.3 Empirical evaluation

To assess the performance of the proposed methodology, we conduct experiments comparing GRADINGS when using either Real NVP or MAF estimators against standard LOF and GMM anomaly detectors with real-world data. We train all the models on the normal data and then apply them to unseen normal samples as well as abnormal data samples. The normal data have been partitioned into two folds, the first one with 80% of the data for the training, and the other 20% is grouped with the abnormal data to compute the evaluation metrics. We report results for individual segment scores and full trajectory scores.

4.3.1 Data set description

We utilize version 1.3 of the Microsoft GeoLife dataset, which consists of real trajectory data collected from 182 users over five years (April 2007 to August 2012). This dataset includes $17,621$ trajectories, with 91.5% of them recorded at intervals of 1 to 5 seconds. For 73 users, transportation modes such as driving, taking a bus, biking, and walking are labeled. Each trajectory represents a complete trip from the departure to the arrival location (ZHENG *et al.*, 2009; ZHENG *et al.*, 2008; ZHENG *et al.*, 2010).

In our experimental setup, we utilize a subset of data from Beijing, China, consisting of 126 car trajectories and 365 bus trajectories. We define two scenarios: CAR $\times$ BUS and BUS $\times$ CAR. In the CAR $\times$ BUS scenario, car trajectories are treated as in-distribution (normal) data, while bus trajectories are treated as out-of-distribution (anomalies). In the BUS $\times$ CAR scenario, the roles are reversed. For each scenario, we use segments of 10, 20, and 30 location points, resulting in six datasets. These datasets contain $230,632$ car trajectory segments and $850,082$ bus trajectory segments.

The timestamp information of the trajectory data is firstly converted to the hour of the week (e.g. Tuesday, 12:30, is equal to 36.5 if we consider Monday as the start of the week) and then encoded into two variables using $\left(\sin\left(2\pi\frac{hour}{168}\right), \cos\left(2\pi\frac{hour}{168}\right)\right)$. This encoding ensures that similar periodic times are close in the input space, even in different weeks (e.g., Sunday, 23:59 is close to Monday, 00:00).

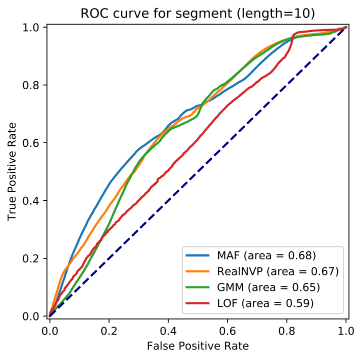
4.3.2 Results and discussion

For the MAF and RealNVP models, we employ 10 flows of neural networks as bijective functions, using the MADE structure for the MAF model. Each network comprises two hidden layers, each with 32 neurons. Both models were trained for 300 epochs. Hyper-parameter tuning for the GMM model was performed using grid search with 5-fold cross-validation on the training data. The K value for the LOF algorithm was determined using the heuristic from (BREUNIG *et al.*, 2000). The ROC curves and corresponding Area Under the Curve (AUC) values are shown in Figs. 8 and 9. Additionally, Table 1 presents the false positive rate when fixing a true positive rate at 80%, referred to as the FPR80 metric.

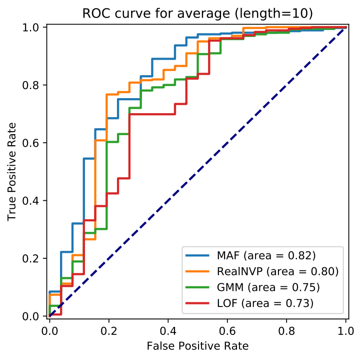
In all evaluated scenario-variant pairs, the NF-based solutions outperformed others regarding AUC. In most of them, the GRADINGS framework with the MAF model showed the best performance. Regarding FPR80, the MAF model achieved superior results in 16 out of 18 evaluations, with RealNVP slightly outperforming in the remaining cases. It is crucial to note that employing a segment aggregation strategy significantly improved AUC performance across all experiments. Specifically, models using the median as the aggregation function attained the best results for both AUC and FPR80.

Regarding segment length, when the median was used as the aggregation function, performance was inversely proportional to segment size. On the other hand, with the average as the aggregation function, performance declined as segment size increased. We hypothesize that the average score's sensitivity to outliers might cause larger segments to include more outliers. The results for individual segments did not exhibit any specific behavior concerning segment length.

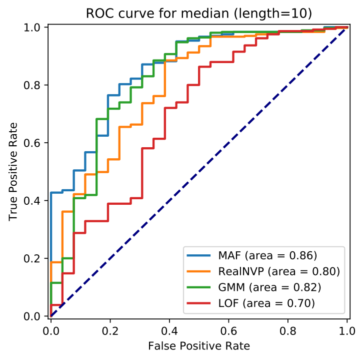
In summary, the results highlight the importance of the two main components of the GRADINGS framework: (*i*) the NF-based density estimation and (*ii*) the aggregation of individual segment scores into a single trajectory anomaly score. Additionally, our findings indicate that the combination of the autoregressive MAF model, the median aggregation function,



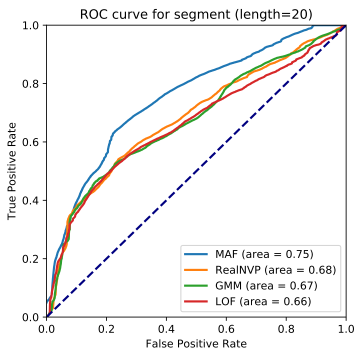
(a)



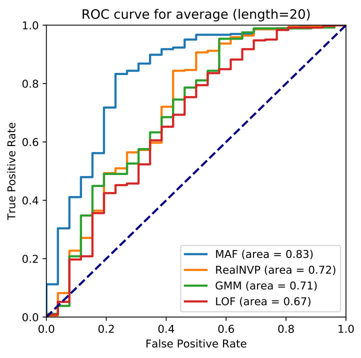
(b)



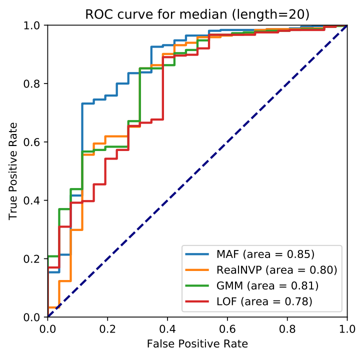
(c)



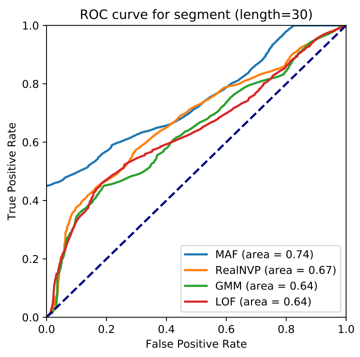
(d)



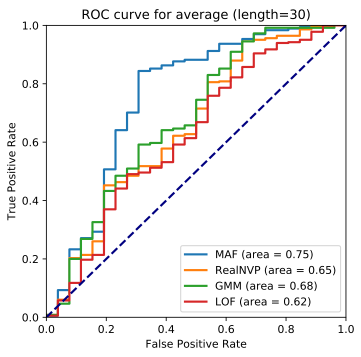
(e)



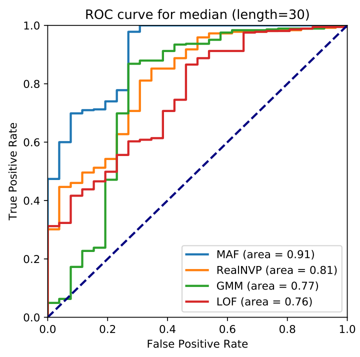
(f)



(g)

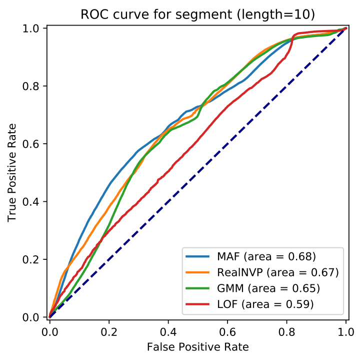


(h)

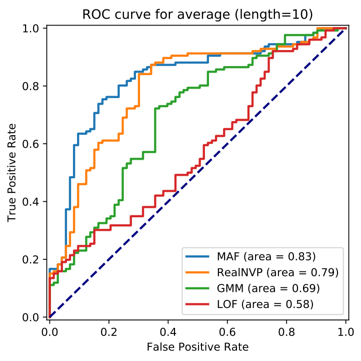


(i)

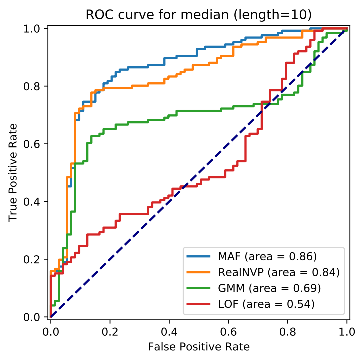
Figure 8 – Anomaly detection results for CAR $\times$ BUS scenario. ROC curves and respective AUC values for segments (left column) and for trajectories, using average (middle column), and median (right column). The rows represent the segment lengths – 10 (a, b, c), 20 (d, e, f), and 30 (g, h, i). The dashed line indicates a completely random detector.



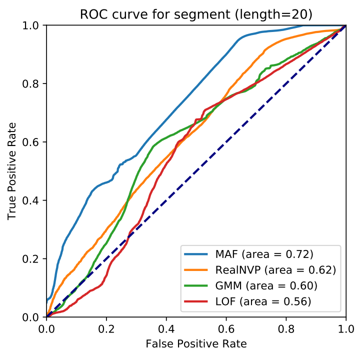
(a)



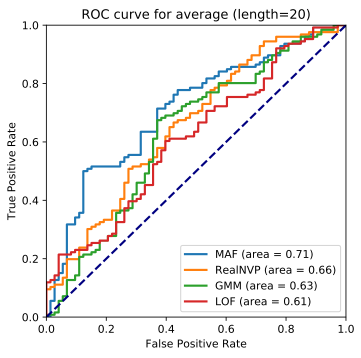
(b)



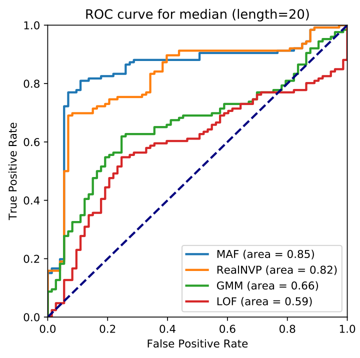
(c)



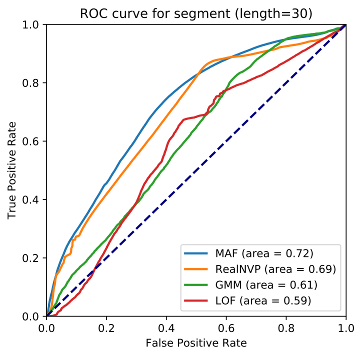
(d)



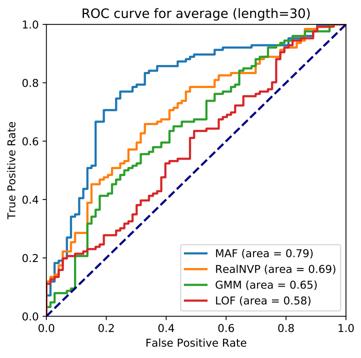
(e)



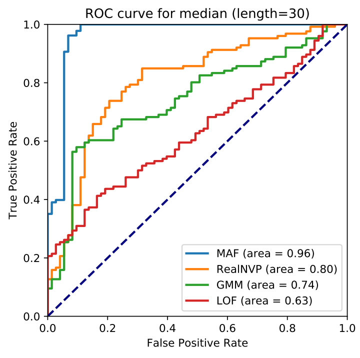
(f)



(g)



(h)



(i)

Figure 9 – Anomaly detection results for BUS $\times$ CAR scenario. ROC curves and respective AUC values for segments (left column) and for trajectories, using average (middle column), and median (right column). The rows represent the segment lengths – 10 (a, b, c), 20 (d, e, f), and 30 (g, h, i). The dashed line indicates a completely random detector.

Table 1 – False positive rates obtained when we fix a true positive rate of 80% (FPR80) for all experimental scenarios.

Scenario	*Variant*	*Length*	*Model*			
			MAF	RealNVP	GMM	LOF
CAR × BUS	segment	10	**0.423**	0.643	0.698	0.719
		20	**0.498**	0.640	0.653	0.688
		30	**0.608**	0.652	0.699	0.727
	average	10	0.342	**0.335**	0.376	0.465
		20	**0.272**	0.435	0.500	0.550
		30	**0.361**	0.577	0.556	0.622
	median	10	**0.245**	0.375	0.308	0.481
		20	**0.247**	0.335	0.353	0.419
		30	<u>**0.201**</u>	0.361	0.315	0.462
BUS × CAR	segment	10	0.603	**0.592**	0.597	0.684
		20	**0.510**	0.633	0.682	0.692
		30	**0.489**	0.517	0.631	0.689
	average	10	**0.252**	0.310	0.482	0.712
		20	**0.529**	0.601	0.635	0.704
		30	**0.311**	0.555	0.622	0.732
	median	10	**0.226**	0.330	0.761	0.771
		20	**0.190**	0.294	0.744	0.819
		30	<u>**0.055**</u>	0.328	0.564	0.747

and a longer segment length (e.g., 30 points) generally yields the best performance.

4.4 Conclusions

Detecting anomalies is a complex task with crucial practical applications. In the context of trajectory data, GPS measurements are typically abundant. However, the high dimensionality and absence of labeled data hinder the application of conventional techniques. In this chapter, we introduced GRADINGS, an unsupervised density estimation method incorporating flexible normalizing flows, specifically the RealNVP and MAF structures. GRADINGS consolidates the log-likelihood values of trajectory segments into a single, robust anomaly score, allowing for trajectories of different lengths. Real-world data experiments demonstrated promising performance compared to the LOF and GMM baselines, especially with the MAF-based variant.

5 ANOMALY DETECTION AND DIAGNOSIS WITHIN A GIVEN TIME SERIES

"Time present and time past are both perhaps present in time future,
and time future contained in time past."
(Thomas Stearns Eliot)

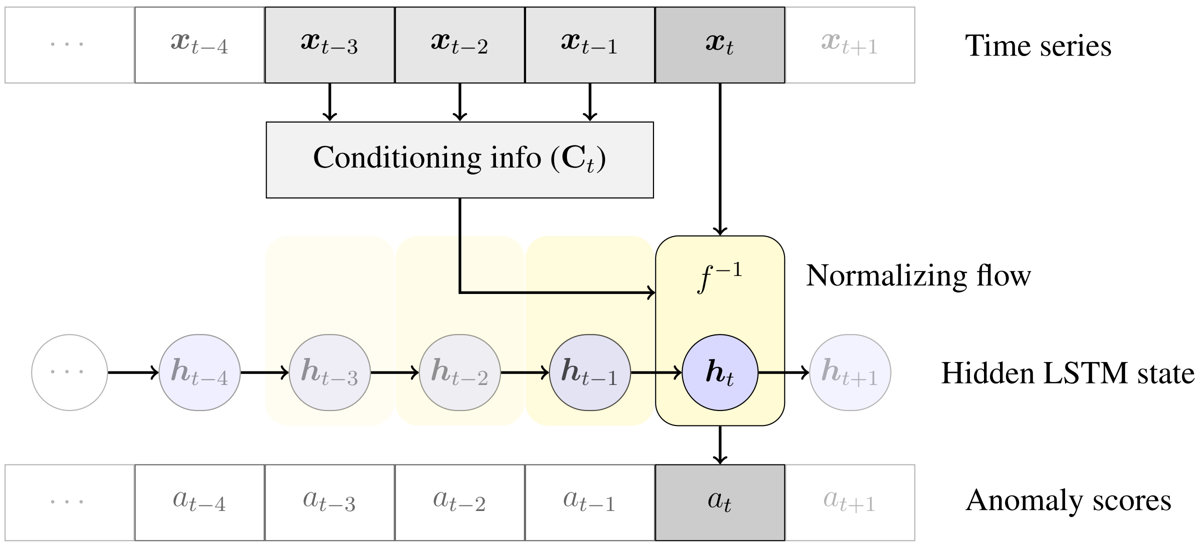


Figure 10 – Schematic of anomaly score computing. The inputs in a time step t are the time vector $\boldsymbol{x}_t$, the context window $\boldsymbol{C}_t$, and the RNN hidden state $\boldsymbol{h}_{t-1}$. The anomaly score is computed by the log conditional probability $a_t = \log p(\boldsymbol{x}_t \mid \mathrm{C}_t, \boldsymbol{h}_t)$, described in Equation (5.21).

In the previous chapter, we show how we can apply NFs to solve the problem of detecting anomalies in time series databases. In this chapter, we introduce a method called RANDOMS, to show how can we apply NFs and manifold learning in order to solve the anomaly detection and diagnosis problems (Problem 2 and Problem 3).

Based on ideas from Henter *et al.* (2020) and Kingma e Dhariwal (2018), this framework is composed of a series of flow steps, each comprising three sub-steps: activation normalization, a linear transformation, and an affine coupling layer. These components are collectively represented as a flow step. Moreover, as in the GRADINGS, the anomaly score is not the pure likelihood. In this case, we create a score based on the distance from the manifold and the log-likelihood of the data as criteria for AD. We also use this criteria for anomaly diagnosis. The following sections will provide an in-depth exploration of each component and elucidate the process of generating anomaly detection scores.

5.1 Related work

Time series data, with its inherent temporal dependencies and complex patterns, presents a challenging domain for AD (Blázquez-García *et al.*, 2022). Consequently, researchers have explored various techniques, from classical statistical methods to state-of-the-art deep learning approaches, in an effort to tackle this problem (RUFF *et al.*, 2021). In recent years, several advancements have been made in the pursuit of robust and effective methods for this task. This section presents related work that forms the basis of our research on multivariate time series AD using generative models and normalizing flows.

5.1.1 Deep learning methods for time series anomaly detection

Generative models have played an important role in advances in the field of AD by providing a powerful framework for modeling complex data distributions (RUFF *et al.*, 2021). In the domain of time series AD, the application of generative models has significantly enhanced our understanding of and ability to characterize the underlying structures within temporal data (Blázquez-García *et al.*, 2022).

The Deep Auto-encoding Gaussian Mixture Model (DAGMM) integrates a deep auto-encoder to produce a low-dimensional representation alongside a reconstruction error computed for each input data point. These elements are subsequently input into a Gaussian mixture model (GMM) in order to classify it into normal or anomaly. Additionally, DAGMM incorporates RNNs for temporal modeling (ZONG *et al.*, 2018). Another method, called Multi-scale Anomaly Detection using Generative Adversarial Networks (MAD-DAN), also employs a generative method (in this case, a GAN) with an LSTM-based generator to model the distribution of time series data. In addition to considering prediction errors, it incorporates the discriminator loss when calculating anomaly scores (LI *et al.*, 2019).

Following by the use of auto-encoders and LSTM nets, Zhang *et al.* (2019) proposes the Multi-Scale Convolutional Recurrent Encoder-Decoder (MSCRED). This approach involves the mapping of input data into a normalized two-dimensional image and traverses a convolutional encoder to capture inter-sensor correlations. Simultaneously, a convolutional LSTM captures temporal patterns, and a convolutional decoder then utilizes the feature maps encoding correlations and temporal information for input signature matrix reconstruction, then combining residual signature matrices for AD.

The Convolutional Auto-Encoding Memory Network (CAE-M) first processes time series data through a CNN and passes to a bidirectional LSTM to capture long-term temporal trends. The difference between such trends and the prediction of a given point is then used as an anomaly score (ZHANG *et al.*, 2023a). The Unsupervised Anomaly Detection for Multivariate Time Series (USAD) (AUDIBERT *et al.*, 2020) uses a combination of an encoder and two decoders to create a combined anomaly score. This method also uses adversarial training to reduce the training time.

The multivariate time-series anomaly detection via graph attention network model (MTAD-GAT) considers each univariate time series as an individual feature and employs a graph attention network to effectively model inter-feature and temporal correlations within the data (ZHAO *et al.*, 2020). The Graph deviation network (GND) (DENG; HOOI, 2021) models learns data mode relationships through a graph and employs attention-based forecasting and deviation scoring for anomaly detection. Also using an attention-based architecture, the Transformer-based Anomaly Detection (TranAD) (TULI *et al.*, 2022) model integrates an encoder-decoder structure for efficient model training. It also employs self-conditioning and adversarial training techniques to enhance error amplification and achieve training stability.

5.1.2 Normalizing flows for sequential modeling

Inspired by the Glow model in (KINGMA; DHARIWAL, 2018), Motion Glow (Moglow) is an NF model designed for probabilistic and controllable motion synthesis in animation and robotics (HENTER *et al.*, 2020). This model utilizes NF to learn the joint distribution of motion data and generate realistic motion sequences. Moglow also incorporates control inputs to facilitate fine-grained control over the generated motion. Additionally, Moglow employs a hierarchical architecture to model motion at multiple temporal scales, enabling the generation of complex and diverse motion sequences.

Similar to the Moglow model, the research by Rasul *et al.* (2021) integrates batch normalization, affine coupling, or masked autoregressive transformations with recurrent neural networks, which can alternatively be replaced by attention mechanisms. Employing this method for sample generation enables us to derive empirical quantiles for the model's predictive uncertainties. Consequently, we can establish a multivariate probabilistic time series forecasting approach via conditioned Normalizing Flows.

In the domain of continuous NF, recent research has introduced innovative approaches to modeling sequences. Mehrasa *et al.* (2019) proposed an intensity-free framework based on NF, which directly models point process distributions without relying on restrictive parametric forms, and Shchur *et al.* (2020) addressed its limitations by directly modeling the conditional distribution of inter-event times using NF. Additionally, Deng *et al.* (2020) leveraged NF to transform base distributions, resulting in a rich time series model with efficient computation of likelihoods and marginals. Their work offered a natural framework for handling irregular time series and demonstrated superior flexibility compared to other baseline methods.

5.1.3 Normalizing flow-based anomaly detection

Most recent advances in the application of anomaly detection using normalizing flows focus on detecting and locating anomalies in images. Rudolph *et al.* (2021) employed normalizing flows in their DifferNet approach, utilizing CNN-extracted features to estimate density for effective defect identification. Yu *et al.* (2021) introduced FastFlow, employing 2D normalizing flows as a probability distribution estimator, surpassing existing methods. Additionally, Gudovskiy *et al.* (2022) proposed CFLOW-AD, a model rooted in a conditional normalizing flow framework featuring a discriminatively pre-trained encoder and multi-scale generative decoders to estimate encoded feature likelihoods.

Time series, including metric anomaly detection, anomaly detection in time series databases, and point and collective anomaly detection, are also the focus of recent work on NF. Schmidt e Simic (2019) demonstrated the effectiveness of the direct use of NF in scoring unseen data samples relative to a model learned from normal data, particularly highlighting its suitability for metric anomaly detection. Dias *et al.* (2020) introduced GRADINGS, which uses NF to detect anomalies in high-dimensional and variable-length data, showing promising results in detecting multivariate time series anomalies. Furthermore, Su *et al.* (2019) proposed the Omnianomaly method, combining stochastic RNN and planar NF to reconstruct probabilities and score time intervals. These collective findings highlight the potential of NF as a robust tool for effectively identifying anomalies in time series data.

5.1.4 Normalizing flows and manifold learning

The manifold hypothesis conjectures that many real-world data sets with underlying structure, such as images or multivariate time series, despite appearing to have a large dimensionality D, actually belong to a low-dimensional unknown $d^{\star}$-dimensional sub-manifold $\mathcal{M}$

of $\mathbb{R}^D$, with $d^\star < D$ (BENGIO *et al.*, 2013). In recent years, the interest in studies involving the use of deep generative modeling applied to infer the low-dimensional representation of a manifold has grown (LOAIZA-GANEM *et al.*, 2024).

Unfortunately, as discussed in Feinman e Parthasarathy (2019), directly employing flow-based models for datasets that occupy only a subset $d^\star$ of the native environmental data space D may not be suitable due to the constraint of use of bijective invertible mappings. Some recent studies have proposed solutions for applying NF models to manifold learning. In this section, we will describe the main approaches. To enable an easy comparison, we will explain all models using two vectors of latent variables $u \in \mathcal{U}$ and $v \in \mathcal{V}$. Here, $\mathcal{U} = \mathbb{R}^{d^\star}$ represents the latent space that corresponds to the learned manifold $\mathcal{M}$, which are the coordinates of the manifold. $\mathcal{V} = \mathbb{R}^{D-d^\star}$ describes any remaining latent variables, indicating the directions "off the manifold".

Flow on a prescribed manifold (FOM): proposed by Gemici *et al.* (2016) and also used in Rezende *et al.* (2020), Bose *et al.* (2020), Kanwar *et al.* (2020), if the manifold is known, i.e. if the chart $g^\star : \mathbb{R}^D \to \mathbb{R}^{d^\star}$ that map the data from the ambient space $\mathbb{R}^D$ to the manifold $\mathcal{R}^{d^\star}$ is given, a flow in $\mathbb{R}^{d^\star}$ is sufficient to learn how to sample data from the manifold. The density $p_U(\boldsymbol{u})$ is modeled using a standard NF h in $d^\star$ dimensions. This NF maps $\boldsymbol{u}$ to another latent variable $\tilde{\boldsymbol{u}}$, as showed in Figure 11.

Pseudo invertible encoding (PIE): this approach considers that an NF-latent space can be disentangled into two spaces, on- and off-manifold, so that we have $p(\boldsymbol{z}) = p(\boldsymbol{u}) \times p(\boldsymbol{v})$. Proposed by Beitler *et al.* (2021), this model splits the latent space of an NF into two variables with different base distributions, modeling $\boldsymbol{u} \in \mathbb{R}^{d^\star}$ (on-manifold) as another NF, mapping to another latent variable $\tilde{\boldsymbol{u}}$. The remaining dimensions are mapped to a sharp distribution around 0, e.g., a Gaussian with a small variance $\sigma^2 \ll \mathrm{Var}[\boldsymbol{u}]$. According to Brehmer e Cranmer (2020), this approach has the same expressivity of a traditional flow (here expressed as an ambient flow).

SoftFlow (SF): in contrast with the above approach, this model adds complexity to the ambient space and estimates a conditional distribution of the perturbed input data rather than directly learning the data distribution (KIM *et al.*, 2020). More precisely, the input data goes through an inflation step, described by:

$$\tilde{\boldsymbol{x}}_n = \boldsymbol{x}_n + \varepsilon_n, \quad \text{such that} \quad \varepsilon_n \sim \mathcal{N}(0, \sigma_n^2 \boldsymbol{I}_D), \quad \text{and} \quad \sigma_n \sim \mathcal{U}(a, b),$$

where $\tilde{\boldsymbol{x}}_n$ is the inflate data of the input $\boldsymbol{x}_n$, ε_n is a Gaussian noise with random variance from a uniform distribution between a and b. After that, a conditional NF is trained over the modified data. To generate a new realistic sample $\boldsymbol{x}_{\text{sp}}$ by using the learned conditional NF and set σ_{sp} to a small value or even zero using $\boldsymbol{z}_{\text{sp}} \sim p_Z(\boldsymbol{z})$, where $\boldsymbol{x}_{\text{sp}} = f(\boldsymbol{z}_{\text{sp}}|\sigma_{\text{sp}})$.

Manifold-learning flows ($\mathcal{M}$-flow): using a very similar way to PIE, this approach also project the latent space to the on-manifold space ($\boldsymbol{u}$) and learn the distribution $p_{\boldsymbol{u}}$ using another NF, but in this case, the reconstruction of the input data $\boldsymbol{x}$ is made by a padding on $\boldsymbol{u}$, completing the rest dimensions by zeros (BREHMER; CRANMER, 2020). Moreover, according to the authors, the difference between this method and FOM lies in the fact that the g mapping can be learned instead of a prescribed, closed-format graph.

Denoising normalizing flows (DNF): as in SF, in this approach, we also find an inflation stage, but here we have to train two different flows f and h. As you can see in Figure 11, after inflating the data, in this case using a Gaussian with variance σ^2, this approach projects the latent variable $\boldsymbol{z}$ on $\boldsymbol{u} \in \mathbb{R}^{d^\star}$ and $\boldsymbol{v} \in \mathbb{R}^{D-d^\star}$, i.e., $(\boldsymbol{u}_n, \boldsymbol{v}_n) \leftarrow f^{-1}(\tilde{\boldsymbol{x}}_n)$, where $\tilde{\boldsymbol{x}}_n$ is an inflated input and f is a standard NF (HORVAT; PFISTER, 2021; HORVAT; PFISTER, 2023; RAZAVI *et al.*, 2025). To model the distribution of the manifold $p_{\boldsymbol{u}}$ another NF is used, mapping $\boldsymbol{u}$ to $\tilde{\boldsymbol{u}}$. This approach is trained by using a composed loss function that combines density learning with reconstruction error:

$$\mathcal{L}_{\text{DNF}} = \overbrace{\mathcal{L}_{\text{ENC}} + \mathcal{L}_{\text{DEC}}}^{\text{density learning}} + \overbrace{\mathcal{L}_{\text{RST}}}^{\text{denoising}}, \tag{5.1}$$

$$\mathcal{L}_{\text{DEC}} = \frac{1}{N}\sum_{n=1}^{N} \log p_{u'}(\boldsymbol{u}'_n) - \log\left|\det \boldsymbol{J}_{h_\phi}(\boldsymbol{u}'_n)\right|, \tag{5.2}$$

$$\mathcal{L}_{\text{ENC}} = \frac{1}{N}\sum_{n=1}^{N} \log(p_v(\boldsymbol{v}_n)) - \log\left|\det \boldsymbol{J}_{f_\psi}(\boldsymbol{u}_n, \boldsymbol{v}_n)\right|, \tag{5.3}$$

$$\mathcal{L}_{\text{RST}} = \frac{\lambda}{N}\sum_{n=1}^{N} \|\boldsymbol{x}_n - \hat{\boldsymbol{x}}_n\|^2. \tag{5.4}$$

Spread normalizing flows (SNF): this approach also uses an inflation step, but in this case, applied on the latent variable $\boldsymbol{z}$ (ZHANG *et al.*, 2023a). Moreover, the base distribution of the latent space is modified to be a generalized Gaussian distribution, such that $p_Z(\tilde{\boldsymbol{z}}) = \mathcal{N}(\mathbf{0}, \mathbf{A}\mathbf{A}^\top + \sigma_Z^2\mathbf{I})$, where σ_Z^2 is the variance of the noise added to the latent space and $\mathbf{A}$ is a $D \times D$ lower triangular matrix with $D(D+1)/2$ parameters, such that $\mathbf{A}\mathbf{A}^\top$ is constrained to be a positive semi-definite matrix. This modifications allows to project $\tilde{\boldsymbol{z}}$ into the manifold by using the non-zero eigenvalues of the matrix $\mathbf{A}\mathbf{A}^\top$ by using $\boldsymbol{u} = \text{proj}(\boldsymbol{z}) = \boldsymbol{z}\mathbf{E}_{1:d^\star}\big|_{\mathbf{A}\mathbf{A}^\top = \mathbf{E}\boldsymbol{\Lambda}\mathbf{E}^\top}$, where $\mathbf{E} = [\boldsymbol{e}_1, \cdots, \boldsymbol{e}_D]$ contains all the eigen-vectors and $\boldsymbol{\Lambda} = \mathrm{diag}\{\lambda_1, \cdots, \lambda_D\}$. The model is trained by using what the authors call "*spread KL divergence*", a modified version of the KL divergence, given by:

$$D_{\mathrm{KL}}(q(\tilde{\boldsymbol{z}}) \parallel p(\tilde{\boldsymbol{z}})) = \int q(\tilde{\boldsymbol{z}}) \log q(\tilde{\boldsymbol{z}}) d\tilde{\boldsymbol{z}} - \int q(\tilde{\boldsymbol{z}}) \log p(\tilde{\boldsymbol{z}}) d\tilde{\boldsymbol{z}}, \tag{5.5}$$

$$\approx - \int q(\tilde{\boldsymbol{z}}) \log p(\tilde{\boldsymbol{z}}) d\tilde{\boldsymbol{z}}, \tag{5.6}$$

$$\approx - \sum_{n=1}^{N} \left\{ \log p_Z(\tilde{\boldsymbol{z}}) - \log \left| \det \left(\frac{\partial f(\boldsymbol{z}_n)}{\partial \boldsymbol{z}_n} \right) \right| \right\}. \tag{5.7}$$

An overview of all of the described NF-based approaches to manifold learning is presented in Figure 11. For a more detailed description of the methods, please follow the references. In this work we are interested in a collateral effect of the manifold learning in flows, the error between the input data $\boldsymbol{x}$ and its reconstruction over the mapping, denoted by $\|\boldsymbol{x} - \boldsymbol{x}'\| = \|\boldsymbol{x} - f(f^{-1}(\boldsymbol{x}))\|$ for simplicity. This will help us to create a robust score for anomaly detection and diagnosis.

5.2 Method

RANDOMS is a novel method designed for anomaly detection and diagnosis. The method consists of a sequence of flow steps, each of them, composed of three main components: an Activation Normalization (Actnorm) layer, a linear transformation, and an affine coupling layer. The Actnorm layer normalizes the dimensions, ensuring that the data maintains a stable distribution throughout the flow. The linear transformation further adjusts the data, enhancing the flexibility of the model. The affine coupling layer, which is conditioned by an LSTM network, allows for complex, non-linear transformations by utilizing the sequential dependencies in the data.

During the training phase, the model learns the parameters of these flow steps, guided by a manifold learning technique. This phase maximizes the likelihood of the training data, ensuring that the normal data distribution is accurately represented in the latent space. For anomaly inference, RANDOMS computes an anomaly score based on a combination of the reconstruction error and the log-likelihood. The reconstruction error measures how well the model can reconstruct the input data from the latent space, while the log-likelihood assesses the probability of the data under the learned distribution. Points with high reconstruction error

AF:

$$\begin{tikzcd}
\boldsymbol{x} & (\boldsymbol{u},\boldsymbol{v}) \sim p_{\boldsymbol{uv}} \arrow[l, "f"']
\end{tikzcd}$$

FOM:

$$\begin{tikzcd}
\tilde{\boldsymbol{x}} & \boldsymbol{u} \arrow[l, dashed, "g^*"'] & \tilde{\boldsymbol{u}} \sim p_{\tilde{\boldsymbol{u}}} \arrow[l, "h"']
\end{tikzcd}$$

PIE:

$$\begin{tikzcd}
 & & \boldsymbol{v} \sim p_{\boldsymbol{v}} & \\
\boldsymbol{x} & (\boldsymbol{u},\boldsymbol{v}) \arrow[l, "f"'] \arrow[ur, dashed, "\text{proj}"] \arrow[dr, dashed, "\text{proj}"'] & & \\
 & & \boldsymbol{u} & \tilde{\boldsymbol{u}} \sim p_{\tilde{\boldsymbol{u}}} \arrow[l, "h"']
\end{tikzcd}$$

SF:

$$\begin{tikzcd}
\tilde{\boldsymbol{x}} & \tilde{\boldsymbol{z}} \sim p_{\boldsymbol{z}}(\boldsymbol{z}) \arrow[l, "f(\,\cdot\,|\sigma_n)"'] \arrow[dl, dashed, "g=f(\,\cdot\,|\sigma)_{\sigma\to 0}"] \\
\boldsymbol{x} \arrow[u, "\text{inflate}"'] &
\end{tikzcd}$$

$\mathcal{M}$-flow:

$$\begin{tikzcd}
\boldsymbol{x} & (\boldsymbol{u},\boldsymbol{v}) \arrow[l, "f"'] \arrow[r, dashed, shift left, "\text{proj}"] & \boldsymbol{u} \arrow[l, dashed, shift left, "\text{pad}"] \arrow[ll, dashed, bend left, "g=f\circ\text{pad}"] & \tilde{\boldsymbol{u}} \sim p_{\tilde{\boldsymbol{u}}} \arrow[l, "h"']
\end{tikzcd}$$

DNF:

$$\begin{tikzcd}
 & & \boldsymbol{v} \sim p_{\boldsymbol{v}} & \\
\tilde{\boldsymbol{x}} & (\boldsymbol{u},\boldsymbol{v}) \arrow[l, "f"'] \arrow[ur, dashed, "\text{proj}"] \arrow[r, dashed, shift left, "\text{proj}"] & \boldsymbol{u} \arrow[l, dashed, shift left, "\text{pad}"] \arrow[dll, dashed, bend left, "g=f\circ\text{pad}"] & \tilde{\boldsymbol{u}} \sim p_{\tilde{\boldsymbol{u}}} \arrow[l, "h"'] \\
\boldsymbol{x} \arrow[u, "\text{inflate}"'] & & &
\end{tikzcd}$$

SNF:

$$\begin{tikzcd}
\boldsymbol{x} \arrow[r, dashed, bend right, "g=f^{-1}\circ\text{inflate}"'] & \tilde{\boldsymbol{z}} \arrow[l, bend right, "f"'] \arrow[r, dashed, bend left, "\text{proj}"] & \boldsymbol{u} \arrow[l, dashed, bend left, "\text{sample}"]
\end{tikzcd}$$

Figure 11 – Schematic relationship between input data $\boldsymbol{x}$ and latent variables $\boldsymbol{u}$, $\boldsymbol{v}$, and $\tilde{u}$ in various manifold models. Solid arrows represent learnable transformations, illustrating how the input data is mapped to and from the latent variables. Dashed arrows denote prescribed transformations, which are predefined and not learned from the data. These mappings highlight the different ways models handle the latent space: $\boldsymbol{u}$ corresponds to the coordinates on the learned manifold $\mathcal{M}$, $\boldsymbol{v}$ parameterizes the remaining latent variables representing directions off the manifold, and $\tilde{\boldsymbol{u}}$ indicates the latent space of the manifold spaces.

and low log-likelihood are flagged as anomalies, facilitating both detection and diagnosis of anomalies.

Through this structured approach, RANDOMS provides a robust and efficient framework for identifying and understanding anomalies in complex datasets, leveraging the intricate transformations of normalizing flows and the powerful conditioning capabilities of LSTM networks. In this section, we will provide an in-depth exploration of each component of RANDOMS in order to elucidate the process of generating anomaly detection scores for detection and diagnosis problems.

5.2.1 Motivation

The data set configurations encountered in real-world AD problems display a wide range of diversity. These settings can contain the conventional unsupervised settings and extend to semi-supervised and supervised scenarios. For each of these learning configurations, the training data can be arranged in different ways, whether or not they contain anomalies. Using some assumptions about such sets will help create models that are better suited to the AD task.

The supervised setting is the case in which the complete training set is labeled, where the output set $\mathcal{Y} = \{\pm 1\}$, with $y = +1$ denoting normal instances and $y = -1$ denoting abnormal instances, respectively. This configuration is the rarest to occur since manually labeling anomalies is a very expensive task as it requires domain experts (LEUNG; LECKIE, 2005). On the other hand, a semi-supervised setting, where the training set is the union of a labeled and an unlabeled set, i.e., $\mathcal{D} = \{\boldsymbol{x}_n \in \mathcal{X}\}_{n=1}^{N} \cup \{(\tilde{\boldsymbol{x}}_m, \tilde{y}_m) \in \mathcal{X} \times \mathcal{Y}\}_{m=1}^{M}$, is a setting that may occur more frequently. In this case, we usually have $N \gg M$; that is, most of the data is unlabeled, and only a few labeled instances are available.

The most common setting, primarily due to the reasons explained above, is the unsupervised data setting, where the model is trained just using unlabeled data. In this context, we can make two distinct assumptions: (i) The first scenario assumes that the training set exclusively comprises normal data, meaning that the training set distribution is equivalent to the normal data distribution, denoted as $P(X) \equiv P^{+}(X)$ and $p(\boldsymbol{x}) \equiv p^{+}(\boldsymbol{X})$; or (ii) The second scenario allows for the presence of a small fraction of anomalous data within the training set. In this case, the majority of instances are drawn from the normal data distribution $P^{+}(X)$, while the remainder is sampled from an abnormal data distribution $P^{-}(X)$ (RUFF *et al.*, 2021).

Although some methods are designed based on the initial assumption, in practice, this assumption often falls short. The reason lies in the fact that real-world data generation processes are frequently susceptible to noise or contamination, as highlighted in earlier investigations (ANSCOMBE, 1960; GRUBBS, 1969). These studies initiated discussions on the origins of anomalies, delineating two primary mechanisms. In the first mechanism, anomalies manifest as rare or low-probability instances typically stemming from measurement errors or inherent data variability (HAWKINS, 1980). On the other hand, in the second mechanism, anomalies emerge from a distribution distinct from the norm, denoted as $P^{-}(X) \neq P^{+}(X)$. In this case, anomalies may result from errors in the execution of experiments or rare events. Notably, this mechanism aligns with the second assumption mentioned above.

Motivated by Leung e Leckie (2005) and Ruff *et al.* (2021), in this work, we use an unsupervised approach with the assumption that the data set arises from the integration of two distinct probability distributions. Furthermore, it is also understood that the data encompasses some level of inherent noise, which may not necessarily be qualified as an anomaly. These assumptions can be summarized as follows:

Assumption 1 *The data set predominantly comprises a significant number of normal elements, with anomalies being relatively rare occurrences.*

This implies that the majority of the data follows expected patterns or behaviors that align with the overall distribution of normal data points. The rarity of anomalies suggests that any deviations from this pattern are infrequent and thus can be considered outliers. This assumption is crucial for many anomaly detection techniques, which rely on the premise that anomalies are sparse compared to the normal data.

Assumption 2 *Anomalous data is generated by a distinct probability distribution.*

This means that the anomalies do not follow the same statistical properties as the normal data and are produced by different underlying processes. The distinct nature of their distribution allows for the identification and distinguishing of anomalies based on statistical methods. This assumption helps in the application of models that can separate the anomalies from the normal data by leveraging differences in their respective distributions.

Assumption 3 *There are rare instances that have arisen due to inherent noise in data.*

Such noise can be introduced by various factors, including measurement errors, data collection inconsistencies, or external environmental influences. While these noise-induced anomalies are not true reflections of underlying abnormal behavior, they can still impact the analysis and detection processes.

Therefore, considering the classical sense of noise, we can characterize the distribution $P(X)$ of the data in terms of the distribution anomalies $P^{-}(X)$, the distribution of normal instances $P^{+}(X)$, and the introduction of inherent randomness ε, following a normal distribution $\varepsilon \sim \mathcal{N}(\mu_\varepsilon, \sigma_\varepsilon)$. This implies that $P(X)$ takes on the form of $\boldsymbol{x} + \varepsilon$, where $\boldsymbol{x} \sim (1-\eta)P^{+}(X) + \eta P^{-}(X)$, such that $\eta \in (0,1)$ represents a contamination (or pollution) rate that approaches $\eta \approx 0$. Finally, as stated by Ruff *et al.* (2021), the assumptions about the noise distribution ε and the contamination rate η are essential for solving a particular AD problem. While these concepts have not been explicitly introduced for sequences, they are applied in a similar manner to time series data.

Combining the aspects of normalizing flows with manifold learning can be particularly well-suited to address the problem of AD using the above assumptions, since, for a small value of η we can consider that $P^{+}(X) \approx P^{+}(X)$ and $P(X) \approx P^{+}(X) + \mathcal{N}(\mu_\varepsilon, \sigma_\varepsilon)$. This means that, while the manifold learning part filters the noise ε in the corrupted data to the off-manifold part of the latent space, the flow can learn the expected law of normality $P^{+}(X)$ from the data. Besides that, we can create an anomaly score using a linear combination of negative log-likelihood and the reconstruction error over the inputs. This can also be used for anomaly diagnosis since we can calculate the error separately by dimensions. In the next sections, we will describe the flow steps and the anomaly inference approach.

5.2.2 *Flow step*

Consider a multivariate time series $\mathbf{X} = \{\boldsymbol{x}_t \in \mathbb{R}^D\}_{t=1}^{T}$, consisting of T vectors of D variables. For each time step t we also consider a context window $\mathbf{C}_t = \boldsymbol{x}_{t-\tau:t-1}$ that consists of a subsequence of length τ starting from time step $t-\tau$ to $t-1$. In order to maintain the window length of τ for each t we use replication padding, i.e., we append a constant vector $(x_t, x_t, \ldots, x_t)$ in each $\mathbf{C}_t$ where $t < \tau$. As aforementioned, our approach consists of K steps of three subsequent basic transformations, a *actnorm*, a *linear transformation*, and an *affine coupling layer*, as we can see in Figure 12[1]. For each of these transformations, called bijectors, we need to calculate the Jacobian log-determinant.

[1] In order to not generate very complex notations, we omitted the indices k (number of step flow) and t (time step) in the explanations below, leaving them explicit only when necessary for understanding.

Actnorm: Introduced by Kingma e Dhariwal (2018), the actnorm is a scale and bias layer with data-dependent initialization. This layer normalizes dimensions over a batch by an affine transformation with trainable scale and bias. Let $\boldsymbol{a}$ be the result of a vector $\boldsymbol{x}$ through an actnorm layer, we can write this transformation by

$$\boldsymbol{a} = \boldsymbol{s} \odot \boldsymbol{x} + \boldsymbol{t}, \tag{5.8}$$

where $\boldsymbol{s}, \boldsymbol{t} \in \mathbb{R}^D$ are the scale and shift vectors. The Jacobian log-determinant of this transformation is a simple sum of the log of scale elements:

$$\log \left| \det \frac{\partial \boldsymbol{a}}{\partial \boldsymbol{x}} \right| = \sum_{d=1}^{D} \log |s_d| . \tag{5.9}$$

Linear transformation: The outputs of the actnorm layer are subjected to a linear transformation given by

$$\boldsymbol{b} = \mathbf{W}\boldsymbol{a}, \tag{5.10}$$

where $\mathbf{W} \in \mathbb{R}^{D \times D}$ represents a weighted matrix, defined as $\mathbf{W} = \mathbf{PL}(\mathbf{U} + \mathrm{diag}\{\boldsymbol{v}\})$, such that $\mathbf{P}$ corresponds to a permutation matrix, $\mathbf{L}$ is a lower triangular matrix with ones on its diagonal, $\mathbf{U}$ is an upper triangular matrix with zeros on its diagonal, and $\boldsymbol{v}$ denotes a vector. The log-determinant of $\mathbf{W}$ can then be expressed as:

$$\log \left| \det \frac{\partial \boldsymbol{b}}{\partial \boldsymbol{a}} \right| = \sum_{d=1}^{D} \log |v_d| , \tag{5.11}$$

Affine Coupling Layer: The main idea of the original affine coupling layer (Figure 12, right) is to transform half of the input elements based on the values of the other half (DINH *et al.*, 2015; DINH *et al.*, 2017). If we use the superscript "lo" to denotes the sequential indexes of the $1 : d$ unchanged dimensions and "hi" to denotes the changed $d + 1 : D$ variables, the components of the layer output $\boldsymbol{z}$ are given by

$$\boldsymbol{z}^{\text{lo}} = \boldsymbol{b}^{\text{lo}}, \tag{5.12}$$

$$\boldsymbol{z}^{\text{hi}} = \left(\boldsymbol{b}^{\text{hi}} + \mu\left(\boldsymbol{b}^{\text{lo}}\right)\right) \odot \sigma\left(\boldsymbol{b}^{\text{lo}}\right), \tag{5.13}$$

where $\sigma, \mu : \mathbb{R}^d \to \mathbb{R}^{D-d}$ respectively represent scale and translation functions parametrized by neural networks, and $\odot$ is the element-wise product operator.

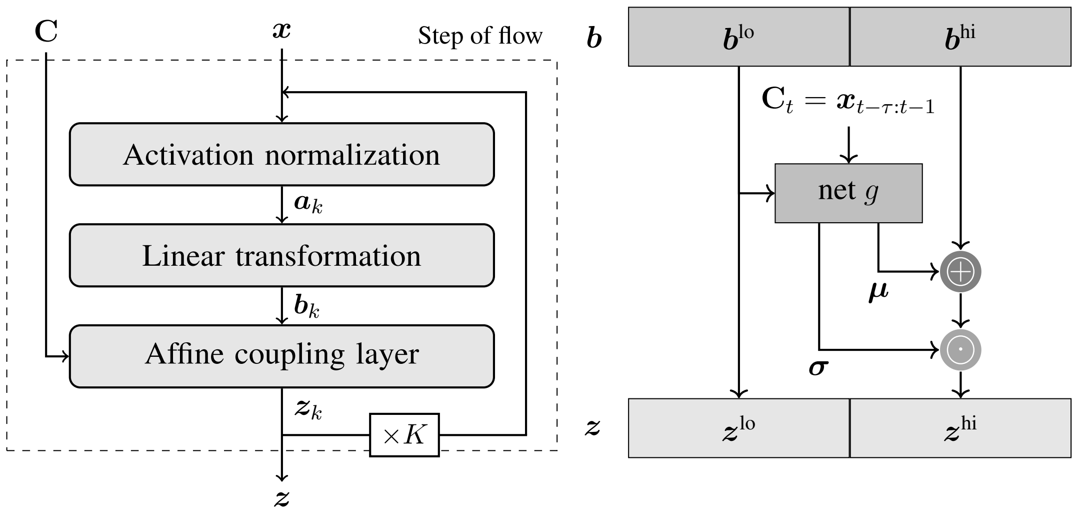


Figure 12 – Flow steps f^{-1} during inference. Detail of the affine coupling layer on right.

To enable explicit dependence on recent timesteps in the flow distribution, we feed the conditioning information (here $\boldsymbol{x}_{t-\tau:t-1}$, denoted by C_t) as additional inputs to the affine couplings and use a single RNN, as a Long Short-Term Memory (LSTM) or a Gated Recurrent Unit (GRU) (CHUNG *et al.*, 2014) to parameterize the scaling and bias terms of Equation (5.13), that can be rewritten as:

$$\boldsymbol{z}^{\text{hi}} = \left(\boldsymbol{b}^{\text{hi}} + \boldsymbol{\mu}\right) \odot \boldsymbol{\sigma}, \tag{5.14}$$

where, now, $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ represent the outcomes from a RNN (or GRU) denoted as $g(\cdot)$ that also has its next state $\boldsymbol{h}_t$ as an output. The following equation describes this computation:

$$[\boldsymbol{\mu},\ \boldsymbol{\sigma},\ \boldsymbol{h}_t] = g\left(\boldsymbol{b}^{\text{lo}},\ \mathrm{C}_t,\ \ \boldsymbol{h}_{t-1}\right). \tag{5.15}$$

The Jacobian matrix of the above-described transformation can be calculated using

$$\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{b}} = \operatorname{diag}(\boldsymbol{\sigma}), \tag{5.16}$$

where $\operatorname{diag}(\boldsymbol{\sigma}) \in \mathbb{R}^{D\times D}$ is a diagonal matrix whose elements are equal to the vector $\boldsymbol{\sigma}$, therefore, its determinant is a simple product of the diagonal terms:

$$\log\left|\det \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{b}}\right| = \sum_{d=1}^{D} \log|\sigma_d| \tag{5.17}$$

It is important to note that the elements in each flow are permuted to different orders by the previous linear transformation, allowing all of the inputs to have a chance to be altered.

5.2.3 Manifold learning

Inspiring by Brehmer e Cranmer (2020), Razavi *et al.* (2025), from the perspective of manifold learning, the latent space can be disentangled into two spaces, on-manifold ($\boldsymbol{u}$) and off-manifold ($\boldsymbol{v}$). The distribution of the transformed data is then the joint distribution of these two parts. If we consider the independence of these two subspaces, we have $p(\boldsymbol{z}) = p(\boldsymbol{u}) \times p(\boldsymbol{v})$. In order to modify the NF for manifold learning, it is necessary to penalize the off-manifold subspace so that the model is able to reconstruct the on-manifold subspace data (HORVAT; PFISTER, 2021). We can then add a constraint to the original optimization problem (3.19) for NF training:

$$\boldsymbol{\theta}^{\star} = \arg\max_{\boldsymbol{\theta}} \log p(\boldsymbol{x}) \tag{5.18}$$

$$\text{s.t. } \mathcal{C}(\boldsymbol{x}, \hat{\boldsymbol{x}}) \leq \delta \tag{5.19}$$

where $\boldsymbol{x} = (x_1, x_2, \ldots, x_D)$ is the input data, $\hat{\boldsymbol{x}} = f(\text{proj}(f^{-1}(\boldsymbol{x}))) = f(\boldsymbol{u}, \mathbf{0}_{D-d})$ is the corresponding reconstruction, $\mathcal{C}$ is a penalty function, and δ a penalization parameter. Using the Lagrange multipliers method, we can incorporate the constraint (5.19) into the corresponding optimization problem as

$$\mathcal{L}(\boldsymbol{\theta}; \lambda) = -\log p(\boldsymbol{x}) + \lambda \mathcal{C}(\boldsymbol{x}, \hat{\boldsymbol{x}}). \tag{5.20}$$

5.2.4 Anomaly inference

After passing the vector $\boldsymbol{x}_t$ through the flow f^{-1}, i.e., pass K times through the flow step, conditioned by the time window $\mathbf{C}_t$ and the hidden state $\boldsymbol{h}_{t-1}$ of a recurrent neural network, the logarithmic density of the flow can be obtained by:

$$\log p(\boldsymbol{x}_t \mid \mathbf{C}_t, \boldsymbol{h}_t) = \log p_{\mathcal{N}}(\boldsymbol{z}_t) + \sum_{k=1}^{K} \sum_{d=1}^{D} \left(\log |s_{k,d}| + \log |v_{k,d}| + \log |\sigma_{k,d}| \right), \tag{5.21}$$

where the subscript k indicates the step of the flow. We can now compute and define an anomaly score by using the calculated density and the reconstruction error as follow:

$$a_t \triangleq -\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_t \mid \mathbf{C}_t, \boldsymbol{h}_t) + \gamma \mathcal{C}(\boldsymbol{x}, \hat{\boldsymbol{x}}) \tag{5.22}$$

where γ is a parameter to weights a reconstruction error $\mathcal{C}(\boldsymbol{x}, \hat{\boldsymbol{x}})$, such that a high value of $\boldsymbol{a}_t$ indicates a more likely anomaly which suggests that abnormal behaviors could be traced to individual time steps. We also use this equation for anomaly diagnosis, just by replacing $\mathcal{C}(\boldsymbol{x}, \hat{\boldsymbol{x}})$ by a per-dimension reconstruction error.

After obtaining the anomaly score for a given timestamp, we classify it as anomalous if its score exceeds a predefined threshold. As employed in previous studies (CHENG *et al.*, 2020; YANG *et al.*, 2024), we adopt Area Under the Curve Percentage Thresholder (AUCP) (REN *et al.*, 2019). This approach allows us to estimate the threshold without relying on strong assumptions or possessing detailed knowledge about the data distribution, ensuring a fair basis for comparison. In the next section, we present the results of applying the RANDOMS methodology to some data sets against several baselines.

5.3 Empirical Evaluation

The main goal of our experiments is to explore the significance of manifold learning in NFs for enhancing anomaly detection and diagnosis in time series data. We also are interested in verifying the importance of the step flow .components and the conditional architecture. To assess the performance of the proposed methodology, we conduct experiments first exploring the components of RANDOMS and then comparing it against state-of-the-art approaches for multivariate time-series anomaly detection. In the next subsections, we will describe the settings, data sets, and metrics and the baselines against which our methodology was evaluated.

5.3.1 Datasets

To facilitate a direct comparison of our approach with competing methods, we have carried out our experiments on commonly used real-world benchmark data sets for time series anomaly detection. It is important to note that, although we share the concerns expressed in the work of Wu e Keogh (2023) about the lack of reference datasets with adequate quality for the task of anomaly detection in time series, we will use such widely used benchmarks in order to allow direct comparison of our proposal with competing approaches.

The following datasets were used in this work: MIT-BIH Supra-ventricular Arrhythmia Database (MBA) (MOODY; MARK, 2001); Multi-Source Distributed System (MSDS) data set (NEDELKOSKI *et al.*, 2020); Mars Science Laboratory (MSL) data set (HUNDMAN *et al.*, 2018); Soil Moisture Active Passive (SMAP) data set (HUNDMAN *et al.*, 2018); Secure Water Treatment (SWaT) data set (GOH *et al.*, 2016); and Water Distribution (WADI) data set (AHMED *et al.*, 2017). Table 2 provides a summary of their characteristics, including the number of training samples (# *Train*), number of test samples (# *Test*), number of variables training samples (# *Variables*), and percentage of anomalies (Anomalies).

Table 2 – General description for data sets used in this work.

Data set	*# Train*	*# Test*	*# Variables*	*Anomalies* (%)
MBA	100000	100000	2	0.14
MSDS	146430	146430	10	5.37
MSL	58317	73729	55	10.72
SMAP	135183	427617	25	13.13
SMD	708405	708420	38	4.16
WADI	1048571	172801	123	5.99

To validate the accuracy of the results, we followed the guidance provided by (NAKAMURA *et al.*, 2020) to remove insignificant sequences from the MSL data set. For a detailed overview of the chosen data sets and specific sequences, please refer to (TULI *et al.*, 2022).

5.3.1.1 Data preprocessing

In order to give more robustness to the models, we performed a min-max normalization, parameterized using the minimum and maximum values of the respective training data set for all data sets. The calculation of this normalization for a given dimension can be written as follows:

$$x'_{t,d} = \frac{x_{t,d} - \min\left(\boldsymbol{x}^{(d)}\right)}{\max\left(\boldsymbol{x}^{(d)}\right) - \min\left(\boldsymbol{x}^{(d)}\right) + \epsilon}, \quad \forall d = 1, 2, \ldots, D, \tag{5.23}$$

where $\epsilon \approx 0$ is a small constant to vector prevent zero-division.

Since not all models intrinsically incorporate temporal context information, in order to ensure a fair comparison, we transform the original data set $\mathbf{X}$ into a set $\mathcal{W}$ of sliding windows with length equal to $\theta < T$, denoted by $\mathcal{W} = \left\{\mathbf{W}_t \in \mathbb{R}^{\tau \times D}\right\}_{t=1}^{T}$, such that $\mathbf{W}_t = \boldsymbol{x}_{t-\theta:t}$, and to maintain the window length of θ for each t we use replication padding, that is, we add a constant vector $(x_t, x_t, \ldots, x_t)$ in each $\mathbf{W}_t$ where $t < \theta$. It is important to note that, for our model, $\theta = \tau + 1$ since at each instant of time we have information about the current instant $\boldsymbol{x}_t$ and contextual information $\mathbf{C}_t = \boldsymbol{x}_{t-\tau:t-1}$.

5.3.2 Performance metrics

The performance evaluation of all models on anomaly detection (Problem 2) is carried out using precision (P), recall (R), and F$_\beta$-score (denoted as F-Score) to β values equals to 0.5, 1, and 2; such that

$$P = \frac{TP}{TP+FP}, \qquad R = \frac{TP}{TP+FN}, \text{ and } \qquad F_\beta = \left(1+\beta^2\right) \times \frac{P \times R}{\beta^2 \times P + R}$$

where TP, FP and FN are true positive, false positive and false negative rate, respectively. We also use the Matthews Correlation Coefficient (MCC), a metric used to evaluate the performance of binary classification models, particularly when the classes are imbalanced. This metric can be calculated by using:

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}.$$

In order to decouple the scorer from the threshold choice, we also use the area under the receiver operating characteristic curve (ROC/AUC) to evaluate the models without choosing a single threshold (LING *et al.*, 2003). Consider a trained model that computes an anomaly score A and a labeled set $\mathcal{D} = \mathcal{X}_N \cup \mathcal{X}_A$, such that $\mathcal{X}_N$ and $\mathcal{X}_A$ are sets of normal and abnormal samples, respectively. The AUC metric of such a model over the data set $\mathcal{D}$ is defined as

$$\mathrm{AUC}(A;\mathcal{D}) \triangleq \frac{1}{|\mathcal{X}_N||\mathcal{X}_A|} \sum_{\boldsymbol{x}\in\mathcal{X}_N} \sum_{\boldsymbol{z}\in\mathcal{X}_A} \mathbf{1}_{A(\boldsymbol{x})>A(\boldsymbol{z})}, \tag{5.24}$$

where $|\cdot|$ denotes the cardinality of a set, and $\mathbf{1}$ is the indicator function: it outputs 1 if the condition $(A(\boldsymbol{x}) > A(\boldsymbol{z}))$ is satisfied. All of the aforementioned metrics are computed after a point-adjustment (XU *et al.*, 2018). In this evaluation system, all instances in an anomalous segment are considered as true positives if a single alarm is raised in the entire segment, as shown in Figure 13.

To effectively evaluate the task of anomaly diagnosis (Problem 3), we employ two commonly used metrics to ensure the performance of all models. Based on the work of Zhao *et al.* (2020), we use the HitRate@P%, which measures the proportion of true anomalies identified among the top P% of detections, providing an indication of the model's precision in identifying the most relevant anomalous dimensions (ZHAO *et al.*, 2020; SU *et al.*, 2019). On the other hand, the Normalized Discounted Cumulative Gain (NDCG) considers the position of elements in the ranking of detections, applying a logarithm discount to lower-ranked positions, thus assessing the quality of the ranking produced by the model (JÄRVELIN; KEKÄLÄINEN, 2002).

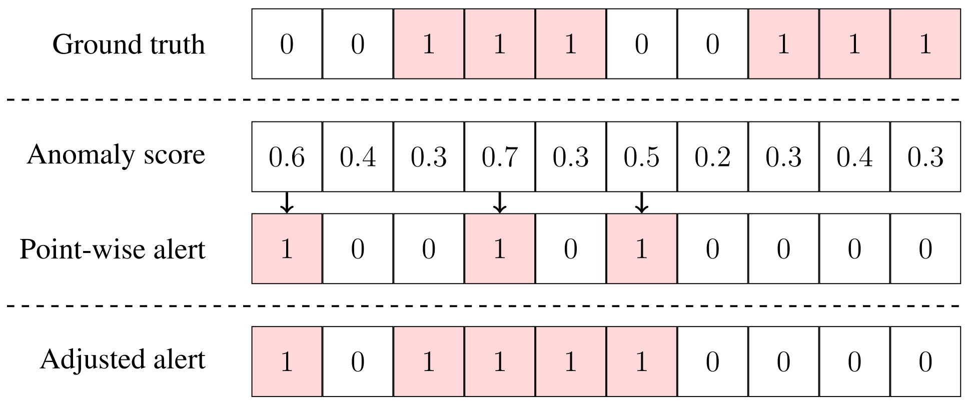


Figure 13 – This visual representation illustrates the point-adjustment strategy. The top row depicts the ground truth, consisting of ten consecutive points, with two anomaly segments highlighted in red. Moving down to the second row, we can observe the detector scores. The third row presents the point-by-point detector results using a threshold of 0.5. Finally, the fourth row displays the detector results following the adjustment process.

5.3.3 Experimental setup and Implementation details

We conducted our experiments using the AdanW optimizer (LOSHCHILOV; HUTTER, 2019), initiating with an initial learning rate of 0.01. We employed a step scheduler with a step size of 0.5. For all models, we employ a grid-search hold-out cross-validation conducted by using the training dataset split between training and validation based on the AUCP metric, where 80% of the training data was effectively used to train the model and the remaining 20% for hyperparameter combination validation. After choosing the best set of hyperparameters, the model was fitted with the complete training set. For the baselines, we use the standard values from the original papers in a set with values used in the research work of (TULI *et al.*, 2022). Our model was parametrized by using the following set of hyper-parameters:

Table 3 – Hyperparameters utilized for the RANDOMS model in the experiments.

Hyperparameter	*Value*
Number of flow-steps	$[5, 10, 20]$
Number of LSTM layers	2
Number of neurons per layer	$[32, 64, 128]$
Context window length (τ)	10
Penalization error (λ)	$[0.001, 1, 100]$

We implemented all models using Python, the PyTorch, and PyTorch Lightning libraries. To optimize computational performance, a Tesla GPU was used, allowing parallelization and distributed computing.

5.3.4 Results and discussion

5.3.4.1 Manifold techniques

The first part of our experiments provides a comparison of performance between the different manifold learning techniques. We have compared four of the presented techniques, namely: Ambient Flow (AF), that is the standard Euclidean NF, denoising normalizing flows (DNF), spread normalizing flows (SNF), and our approach (denoted by Our), presented in Section 5.2.3, that is based on the works of Brehmer e Cranmer (2020), Beitler *et al.* (2021). Table 4 presents the performance metrics of this experiment applied to the presented datasets.

dataset	*manifold*	P	R	F-Score			MCC	AUC
				$\beta = 1$	$\beta = 2$	$\beta = 0.5$		
MBA	AF	0.6289	0.9996	0.7722	0.8945	0.6793	0.8313	0.8490
	DNF	0.6420	0.9996	0.7820	0.8997	0.6915	0.8386	0.8573
	SNF	0.6407	0.9997	0.7810	0.8992	0.6903	0.8379	0.8565
	Our	**0.6516**	**0.9999**	**0.7891**	**0.9034**	**0.7004**	**0.8440**	**0.8632**
MSDS	AF	0.9798	0.9703	0.9750	0.9722	0.9779	0.9556	0.9591
	DNF	0.9799	**0.9708**	0.9753	**0.9726**	0.9781	0.9561	0.9595
	SNF	**0.9831**	0.9666	0.9748	0.9699	**0.9798**	0.9558	0.9617
	Our	0.9822	0.9695	**0.9758**	0.9720	0.9797	**0.9573**	**0.9619**
MSL	AF	0.3212	0.9995	0.4863	0.7029	0.3717	0.7311	0.8324
	DNF	0.3429	0.9995	0.5107	0.7229	0.3948	0.7443	0.8480
	SNF	0.4448	**0.9999**	0.6157	0.8002	0.5003	0.7986	0.9010
	Our	**0.4704**	0.9999	**0.6398**	**0.8162**	**0.5261**	**0.8108**	**0.9107**
SMAP	AF	0.1986	0.9997	0.3313	0.5533	0.2365	0.6741	0.8054
	DNF	0.2313	**0.9999**	0.3757	0.6007	0.2733	0.6982	0.8398
	SNF	0.2579	0.9995	0.4101	0.6348	0.3029	0.7159	0.8613
	Our	**0.2661**	0.9996	**0.4203**	**0.6445**	**0.3119**	**0.7210**	**0.8670**
SMD	AF	0.8077	**0.9998**	0.8936	0.9545	0.8400	0.9438	0.9877
	DNF	0.8609	0.9996	0.9252	0.9687	0.8855	0.9600	0.9917
	SNF	0.8060	0.9996	0.8926	0.9541	0.8385	0.9433	0.9876
	Our	**0.9673**	0.9993	**0.9830**	**0.9927**	**0.9736**	**0.9907**	**0.9979**
WADI	AF	0.7209	0.4953	0.5872	0.5284	0.6607	0.7915	0.7433
	DNF	**0.8994**	0.4048	0.5583	0.4548	**0.7228**	0.7965	0.7014
	SNF	0.6431	0.6431	0.6431	0.6431	0.6431	0.8135	0.8135
	Our	0.5099	**0.9214**	**0.6565**	**0.7934**	0.5600	**0.8340**	**0.9406**

Table 4 – Performance metrics for RANDOMS model with the different manifold learning techniques. Bold values indicate the highest performance within each dataset.

As we can see in Table 4, Our method demonstrates superior Precision, Recall, and F-scores with different beta values, MCC, and AUC on most datasets. For the MBA dataset, our manifold learning technique achieves the highest values across all metrics, indicating a robust combination of precision and recall. In the MSDS dataset, our method performs well, achieving the highest F-Score for $\beta = 1$ and maintaining competitive values of other metrics. Similarly, in the MSL and SMAP datasets, our manifold learning technique achieved better results in Recall and F-Scores.

The SMD dataset particularly highlights the effectiveness of our manifold learning technique, registering the highest scores in every evaluated metric. In the WADI dataset, Our technique also performs well, particularly in Recall, F-Scores, MCC, and AUC, although NDF leads in Precision. Overall, table results underscore the consistent superiority of our technique, making it the most reliable manifold learning technique across different datasets. This indicates that our technique provides a balanced and high-performance approach to the AD task, effectively handling diverse data characteristics and maintaining high classification reliability.

5.3.4.2 Conditional network type

One of the main characteristics of our model is the ability to effectively incorporate temporal information, which is crucial for analyzing time-dependent data. We believe that this is done through the conditioning part in our flow, which uses an RNN. In this specific case, we have employed an LSTM network known for its strength in capturing long-term dependencies in sequential data. In order to evaluate this hypothesis and determine the effectiveness of LSTM in our model, we conducted a series of experiments. These experiments were carried out using two types of neural networks in the flow conditioning layer, an LSTM and a Multi-Layer Perceptron (MLP) network. The performance metrics obtained from these experiments are summarized in Table 5. In our experiments, we use our manifold learning technique and optimize the remaining hyper-parameters.

The results support our hypothesis that RNNs, which are designed to handle sequential data, outperform MLPs across various metrics when incorporated as flow conditioners. For the MBA dataset, the RNN achieved superior results with a precision of 0.6516, recall of 0.9999, and the highest F-Scores across all β values (0.7891, 0.9034, and 0.7004), alongside better MCC (0.8440) and AUC (0.8632). The MSDS dataset also demonstrates the advantage of using RNNs, with RNN achieving a precision of 0.9822 and strong performance in other metrics, including

dataset	*network*	P	R	F-Score			MCC	AUC
				$\beta = 1$	$\beta = 2$	$\beta = 0.5$		
MBA	MLP	0.6407	0.9997	0.7810	0.8992	0.6903	0.8379	0.8565
	RNN	**0.6516**	**0.9999**	**0.7891**	**0.9034**	**0.7004**	**0.8440**	**0.8632**
MSDS	MLP	0.9803	**0.9711**	0.9757	**0.9729**	0.9785	0.9568	0.9602
	RNN	**0.9822**	0.9695	**0.9758**	0.9720	**0.9797**	**0.9573**	**0.9619**
MSL	MLP	0.3456	0.9996	0.5137	0.7253	0.3976	0.7459	0.8498
	RNN	**0.4704**	**0.9999**	**0.6398**	**0.8162**	**0.5261**	**0.8108**	**0.9107**
SMAP	MLP	0.1607	0.9995	0.2769	0.4891	0.1931	0.6412	0.7482
	RNN	**0.2661**	**0.9996**	**0.4203**	**0.6445**	**0.3119**	**0.7210**	**0.8670**
SMD	MLP	0.8866	**0.9998**	0.9399	0.9751	0.9072	0.9677	0.9934
	RNN	**0.9673**	0.9993	**0.9830**	**0.9927**	**0.9736**	**0.9907**	**0.9979**
WADI	MLP	**0.7470**	0.4953	0.5957	0.5311	**0.6781**	0.7972	0.7439
	RNN	0.5099	**0.9214**	**0.6565**	**0.7934**	0.5600	**0.8340**	**0.9406**

Table 5 – Performance metrics for different datasets using different networkds – Recurrent Neural Networks (RNN) and Multi Layer Perceptron (MLP) – in the conditioner of the coupling layer. Bold values indicate the highest performance within each dataset.

an AUC of 0.9619. For the MSL dataset, the RNN exhibited higher precision (0.4704), recall (0.9999), and F-Scores (0.6398, 0.8162, and 0.5261), with an MCC of 0.8108 and an AUC of 0.9107, outperforming the MLP.

The SMAP dataset results also favor the RNN, with significant improvements in precision (0.2661), F-Scores (0.4203, 0.6445, and 0.3119), MCC (0.7210), and AUC (0.8670) compared to the MLP. In the SMD dataset, the RNN achieved remarkable precision (0.9673) and recall (0.9993), F-Scores (0.9830, 0.9927, and 0.9736), MCC (0.9907), and AUC (0.9979). For the WADI dataset, the RNN showed a recall (0.9214) and surpassed the MLP in F-Scores (0.6565 and 0.7934), MCC (0.8340), and AUC (0.9406), although the MLP had higher precision (0.7470). These results demonstrate that RNNs are more effective in capturing and leveraging temporal information when used as conditioners in flow-based models, leading to improved performance across multiple datasets compared to MLPs.

5.3.4.3 Number of neurons per layer

To finish the tests related to the conditioner's neural network, we need to verify the role of the number of neurons in the model's performance. To do that, we set the manifold technique and the type of network, just modifying the number of neurons and optimizing the rest of the hyper-parameters. The results of this round of experiments are shown in Table 6. The table presents various configurations with three different numbers of neurons, ranging from smaller (32 neurons) networks to more complex architectures (128 neurons). This experiment aims to understand the trade-offs between network complexity and model performance, as well as to identify an optimal configuration that balances computational efficiency with performance.

dataset	*# neurons*	P	R	F-Score			MCC	AUC
				$\beta = 1$	$\beta = 2$	$\beta = 0.5$		
MBA	32	0.6420	0.9996	0.7820	0.8997	0.6915	0.8386	0.8573
	64	**0.6516**	**0.9999**	**0.7891**	**0.9034**	**0.7004**	**0.8440**	**0.8632**
	128	0.6289	0.9996	0.7722	0.8945	0.6793	0.8313	0.8490
MSDS	32	0.9798	**0.9703**	0.9750	**0.9722**	0.9779	0.9556	0.9591
	64	**0.9831**	0.9666	0.9748	0.9699	**0.9798**	0.9558	0.9617
	128	0.9822	0.9695	**0.9758**	0.9720	0.9797	**0.9573**	**0.9619**
MSL	32	0.4448	**0.9999**	0.6157	0.8002	0.5003	0.7986	0.9010
	64	0.3021	0.9997	0.4641	0.6840	0.3512	0.7188	0.8168
	128	**0.4704**	0.9999	**0.6398**	**0.8162**	**0.5261**	**0.8108**	**0.9107**
SMAP	32	0.2183	0.9995	0.3583	0.5826	0.2587	0.6890	0.8273
	64	0.2332	0.9996	0.3782	0.6032	0.2754	0.6995	0.8414
	128	**0.2661**	**0.9996**	**0.4203**	**0.6445**	**0.3119**	**0.7210**	**0.8670**
SMD	32	0.8539	**0.9993**	0.9209	0.9663	0.8795	0.9577	0.9908
	64	**0.9673**	**0.9993**	**0.9830**	**0.9927**	**0.9736**	**0.9907**	**0.9979**
	128	0.7238	**0.9993**	0.8395	0.9286	0.7660	0.9167	0.9799
WADI	32	0.6431	0.6431	0.6431	0.6431	0.6431	0.8135	0.8135
	64	**0.7459**	0.4847	0.5876	0.5212	**0.6733**	0.7937	0.7386
	128	0.5099	**0.9214**	**0.6565**	**0.7934**	0.5600	**0.8340**	**0.9406**

Table 6 – Performance metrics (Precision, Recall, F-Score with β values of 1, 2, and 0.5, MCC, and AUC) for different number of neurons (32, 64, 128) in the conditioner of the coupling layer. Bold values indicate the highest performance within each dataset.

For the MBA dataset, the configuration with 64 neurons performed best across all metrics, with the precision of 0.6516, recall of 0.9999, and F-scores of 0.7891, 0.9034, and 0.7004 for values β of 1, 2, and 0.5, respectively. The MCC and AUC were also higher at 0.8440 and 0.8632. Configurations with 32 and 128 neurons performed slightly worse, indicating that 64 neurons strike a balance between model complexity and performance for this dataset. In the MSDS dataset, the configuration with 32 neurons showed strong performance, particularly in precision (0.9798) and AUC (0.9591). However, the configuration with 128 neurons performed best overall, with the highest precision (0.9831) and F-scores across all β values, as well as the highest MCC (0.9573) and AUC (0.9619). This suggests that the model can effectively handle complex patterns with a higher number of neurons.

For the MSL dataset, the model with 128 neurons outperformed other configurations with a precision of 0.4704, recall of 0.9999, and the highest F-Scores (0.6398, 0.8162, and 0.5261) at $values\beta$. The MCC and AUC were also higher at 0.8108 and 0.9107. The 32-neuron configuration had close recall but lower precision and F-scores, suggesting that increasing the number of neurons increases the model's ability to capture complex patterns in the data. The 64-neuron configuration showed a notable decrease in performance compared to 32 and 128 neurons.

On the SMAP dataset, increasing the number of neurons progressively improved performance. The 128-neuron configuration achieved the best precision (0.2661), recall (0.9996), and F-Scores (0.4203, 0.6445, and 0.3119), along with the highest MCC (0.7210) and AUC (0.8670). This trend indicates that a small number of neurons are better suited to handle the complexity of the SMAP dataset. On the SMD dataset, the configuration with 64 neurons significantly outperformed the others, achieving the highest precision (0.9673), recall (0.9993), F-Scores (0.9830, 0.9927 and 0.9736), MCC (0.9907) and AUC (0.9979). Although the 32- and 128-neuron configurations also performed well, the 64-neuron model achieved the best balance between precision and recall for this dataset.

For the WADI dataset, performance varied significantly with different neuron numbers. The 128-neuron configuration achieved the highest recall (0.9214), F-Scores (0.6565 and 0.7934), MCC (0.8340), and AUC (0.9406). However, the 64-neuron model had the highest accuracy (0.7459) and a competitive MCC and AUC, indicating that although increasing neurons generally improves recall, it may not always improve accuracy. The 32-neuron configuration, with consistent metrics across all measurements (0.6431), performed noticeably worse. In general, the results demonstrate that the optimal number of neurons in the coupling layer conditioner varies by dataset, with intermediate to high number of neurons generally providing the best balance of performance metrics. This highlights the importance of adjusting model complexity to the specific characteristics of the data being analyzed.

5.3.4.4 Number of flows

Finishing our analysis of our proposal, the following experiment examines the impact of varying the number of step flows on the performance of the method across different datasets. By evaluating key performance metrics, we aim to identify the optimal configuration for each dataset. This comparative assessment is essential for understanding how different flow configurations affect the model's ability to capture the complexity of the data. The results are detailed in Table 7, with bold values indicating the highest performance within each dataset.

dataset	*# flows*	P	R	F-Score			MCC	AUC
				$\beta = 1$	$\beta = 2$	$\beta = 0.5$		
MBA	5	**0.6516**	**0.9999**	**0.7891**	**0.9034**	**0.7004**	**0.8440**	**0.8632**
	10	0.6420	0.9996	0.7820	0.8997	0.6915	0.8386	0.8573
	20	0.6069	0.9998	0.7554	0.8853	0.6587	0.8185	0.8343
MSDS	5	0.9786	**0.9702**	0.9744	0.9719	0.9769	0.9545	0.9575
	10	**0.9831**	0.9666	0.9748	0.9699	**0.9798**	0.9558	0.9617
	20	0.9822	0.9695	**0.9758**	**0.9720**	0.9797	**0.9573**	**0.9619**
MSL	5	0.4448	**0.9999**	0.6157	0.8002	0.5003	0.7986	0.9010
	10	0.3189	0.9997	0.4836	0.7007	0.3692	0.7296	0.8306
	20	**0.4704**	0.9999	**0.6398**	**0.8162**	**0.5261**	**0.8108**	**0.9107**
SMAP	5	**0.2661**	0.9996	**0.4203**	**0.6445**	**0.3119**	**0.7210**	**0.8670**
	10	0.2595	0.9998	0.4121	0.6367	0.3047	0.7169	0.8624
	20	0.2313	**0.9999**	0.3757	0.6007	0.2733	0.6982	0.8398
SMD	5	0.7841	**0.9993**	0.8787	0.9473	0.8193	0.9362	0.9854
	10	**0.9673**	**0.9993**	**0.9830**	**0.9927**	**0.9736**	**0.9907**	**0.9979**
	20	0.8709	**0.9993**	0.9307	0.9706	0.8939	0.9628	0.9920
WADI	5	**0.6431**	0.6431	0.6431	0.6431	**0.6431**	0.8135	0.8135
	10	0.3522	**0.9214**	0.5096	0.6963	0.4018	0.7712	0.9222
	20	0.5099	**0.9214**	**0.6565**	**0.7934**	0.5600	**0.8340**	**0.9406**

Table 7 – Performance metrics for different number of step flows (5, 10, 20). Bold values indicate the highest performance within each dataset.

As you can see, for the MBA dataset, the configuration with 5 flows consistently outperformed others, achieving the highest precision (0.6516), recall (0.9999), F-Scores (0.7891, 0.9034, and 0.7004), MCC (0.8440), and AUC (0.8632). The performance decreases slightly as the number of flows increases to 10 and 20, suggesting that a lower number of flows might be more effective for this dataset. Similar trends were observed in the MSDS dataset, where the 10-flow configuration provided the best performance across multiple metrics, including precision (0.9831), F-Scores, and AUC (0.9617). The configuration with 20 flows, however, showed the best recall (0.9695) and comparable F-Scores, indicating that the MSDS dataset may benefit from slightly more complex models.

For the MSL dataset, the best performance was observed with 20 flows, achieving top scores in precision, F-Scores, MCC, and AUC, indicating that a higher number of flows better captures the temporal dependencies in this dataset. In the SMAP and SMD datasets, the performance varies with the number of flows. The SMAP dataset shows optimal performance with 5 flows, achieving the highest precision (0.2661) and F-Scores, while for the SMD dataset, the best performance is observed with 10 flows, with top precision (0.9673), F-Scores, MCC (0.9907), and AUC (0.9979). The WADI dataset presents a unique case where the performance peaks with 20 flows, achieving the highest recall (0.9214), F-Scores, MCC, and AUC (0.9406).

5.3.4.5 RANDOMS vs others

Our proposed method performance is also compared to *state-of-the-art* deep anomaly detection methods, which include DAGMM, Omnianomaly, USAD, and traditional flows RealNVP and MAF. We also compare our model with a non-deep model, the LSTM-NDT, which uses LSTMs to capture order dependence by predicting the next timestamp's data using previous outputs. The LSTM-NDT also introduces a Non-parametric Dynamic error Thresholding (NDT) strategy for anomaly labeling based on error sequence averages (HUNDMAN *et al.*, 2018).

The performance metrics presented in Table 8 and Table 9 show that our proposed method, RANDOMS, generally outperforms several state-of-the-art anomaly detection techniques on various datasets. Specifically, in Table 8, for the MBA dataset, RANDOMS achieves the highest levels of Precision, Recall, F-Scores (with β values – of 1, 2 and 0.5), MCC and AUC, except MAF, which stands out in Precision. In the MSDS dataset, RANDOMS also demonstrates strong performance, achieving the highest values in terms of the F-Score (with β values of 1 and 0.5), MCC, and AUC. However, LSTM-NDT slightly surpasses RANDOMS in the F-Score with a β value of 2, and DAGMM has a marginally higher Precision score. Similarly, on the MSL dataset, RANDOMS demonstrates superior performance across all metrics, significantly outperforming LSTM-NDT, USAD, and Omnianomaly.

As we can see in Table 9, for the SMAP dataset, RANDOMS achieves the highest MCC and AUC values, indicating robust anomaly detection capabilities. For the SMD dataset, while MAF and RANDOMS show comparable performance, RANDOMS stands out slightly in terms of F-Scores and MCC, showing its effectiveness. On the WADI dataset, despite RealNVP having higher accuracy, RANDOMS maintains the highest Recall, F-Scores, MCC, and AUC, highlighting its balanced performance across different evaluation metrics. These results show that RANDOMS consistently offers high recall rates and competitive precision.

dataset	*method*	P	R	F-Score			MCC	AUC
				$\beta = 1$	$\beta = 2$	$\beta = 0.5$		
MBA	DAGMM	0.3385	0.9998	0.5058	0.7190	0.3902	0.5000	0.5000
	LSTM-NDT	0.4017	0.9995	0.5731	0.7705	0.4563	0.6545	0.6188
	USAD	0.3390	0.9998	0.5064	0.7195	0.3907	0.5135	0.5011
	Omnianomaly	0.3521	0.9997	0.5208	0.7310	0.4045	0.5715	0.5290
	RealNVP	0.4244	0.9996	0.5958	0.7866	0.4796	0.6801	0.6529
	MAF	**0.9025**	0.9998	**0.9487**	**0.9788**	**0.9204**	**0.9617**	**0.9723**
	RANDOMS	0.6516	**0.9999**	0.7891	0.9034	0.7004	0.8440	0.8632
MSDS	DAGMM	0.9815	0.9644	0.9729	0.9678	0.9780	0.9524	0.9586
	LSTM-NDT	0.9799	0.9721	**0.9760**	0.9737	0.9783	0.9572	0.9600
	USAD	0.9765	0.9725	0.9745	0.9733	0.9757	0.9543	0.9558
	Omnianomaly	0.9657	0.9781	0.9718	**0.9756**	0.9681	0.9485	0.9437
	RealNVP	0.9784	0.9727	0.9755	0.9739	0.9772	0.9563	0.9584
	MAF	0.9702	0.9724	0.9713	0.9720	0.9706	0.9481	0.9473
	RANDOMS	**0.9822**	0.9695	0.9758	0.9720	**0.9797**	**0.9573**	**0.9619**
MSL	DAGMM	0.1895	0.9996	0.3186	0.5389	0.2261	0.6234	0.6607
	LSTM-NDT	0.2407	0.9997	0.3880	0.6131	0.2838	0.6734	0.7497
	USAD	0.1370	0.9995	0.2410	0.4425	0.1656	0.5042	0.5003
	Omnianomaly	0.1803	0.9997	0.3056	0.5238	0.2157	0.6121	0.6395
	RealNVP	0.2162	0.9996	0.3555	0.5797	0.2564	0.6515	0.7124
	MAF	0.4000	0.9998	0.5714	0.7692	0.4545	0.7760	0.8810
	RANDOMS	**0.4704**	**0.9999**	**0.6398**	**0.8162**	**0.5261**	**0.8108**	**0.9107**

Table 8 – Performance metrics for comparison of RANDOMS with baseline methods. Bold values indicate the highest performance within each dataset.

In order to verify the possible equivalence between the models, we also perform a Friedman statistical test with a Wilcoxon signed-rank post-hoc test for some metrics. This statistical comparison quantifies the consistency of the results obtained by a model when applied in several experiments according to their performance classifications (DEMSAR, 2006; BENAVOLI *et al.*, 2016). In Figure 14, we present a critical difference diagram that represents the result of the statistical test mentioned above for some metrics.

As we can see in Figure 14, RANDOMS consistently ranks at the top across all metrics – Precision, Recall, AUC, and MCC. This indicates that our method is highly reliable in both minimizing false positives and capturing most anomalies, demonstrating balanced sensitivity and specificity. Its top rankings in MCC further confirm the robustness and overall quality of RANDOMS in accurately identifying anomalies in time series data. This makes RANDOMS a competitive and effective method for practical applications.

dataset	*method*	P	R	F-Score			MCC	AUC
				$\beta = 1$	$\beta = 2$	$\beta = 0.5$		
SMAP	DAGMM	0.1406	0.9997	0.2466	0.4500	0.1698	0.6202	0.7054
	LSTM-NDT	0.2256	0.9996	0.3681	0.5929	0.2669	0.6942	0.8345
	USAD	**0.8404**	**0.9999**	**0.9133**	**0.9634**	**0.8682**	**0.9542**	**0.9908**
	Omnianomaly	0.0932	0.9999	0.1706	0.3396	0.1139	0.5381	0.5311
	RealNVP	0.1629	0.9997	0.2801	0.4931	0.1956	0.6433	0.7522
	MAF	0.0981	0.9997	0.1787	0.3523	0.1197	0.5528	0.5569
	RANDOMS	0.2661	0.9996	0.4203	0.6445	0.3119	0.7210	0.8670
SMD	DAGMM	0.5795	0.9974	0.7331	0.8717	0.6325	0.8655	0.9613
	LSTM-NDT	0.3780	0.9996	0.5486	0.7524	0.4317	0.7800	0.9150
	USAD	0.3729	0.9985	0.5430	0.7476	0.4263	0.7772	0.9125
	Omnianomaly	0.3644	0.9985	0.5340	0.7408	0.4175	0.7730	0.9093
	RealNVP	0.6450	0.9997	0.7842	0.9008	0.6943	0.8900	0.9716
	MAF	0.9556	**0.9997**	0.9773	0.9908	0.9641	0.9876	0.9976
	RANDOMS	**0.9673**	0.9993	**0.9830**	**0.9927**	**0.9736**	**0.9907**	**0.9979**
WADI	DAGMM	0.5333	0.2876	0.3737	0.3168	0.4555	0.6861	0.6381
	LSTM-NDT	0.5530	0.5486	0.5508	0.5495	0.5521	0.7652	0.7642
	USAD	0.2911	0.4554	0.3551	0.4092	0.3137	0.6640	0.7025
	Omnianomaly	0.4682	0.5486	0.5052	0.5304	0.4823	0.7413	0.7601
	RealNVP	**0.8974**	0.3262	0.4785	0.3738	**0.6647**	0.7653	0.6623
	MAF	0.7581	0.4048	0.5278	0.4464	0.6454	0.7701	0.6995
	RANDOMS	0.5099	**0.9214**	**0.6565**	**0.7934**	0.5600	**0.8340**	**0.9406**

Table 9 – Performance metrics (Precision, Recall, F-Score with β values of 1, 2, and 0.5, MCC, and AUC) for comparison of RANDOMS with baseline methods. Bold values indicate the highest performance within each dataset.

5.3.4.6 Anomaly diagnosis

The results, presented in Table 10[2], using the aforementioned metrics provides a comprehensive insight into the performance of the models. By analyzing the HitRate@P%, we can determine how effectively each model identifies the most critical anomalies, ensuring that key outliers are detected with high precision. Simultaneously, the NDCG scores offer a deeper understanding of the ranking quality, indicating how well the models prioritize the detected anomalies. This dual assessment framework not only highlights the strengths and weaknesses of each model but also facilitates a more nuanced comparison, guiding further improvements in anomaly detection methodologies.

The results highlight the superior performance of our approach, particularly in the

[2] It is important to note that the MAF and RealNVP models are omitted from the table, as they do not have the ability to directly generate scores by dimension, making them useless for this task.

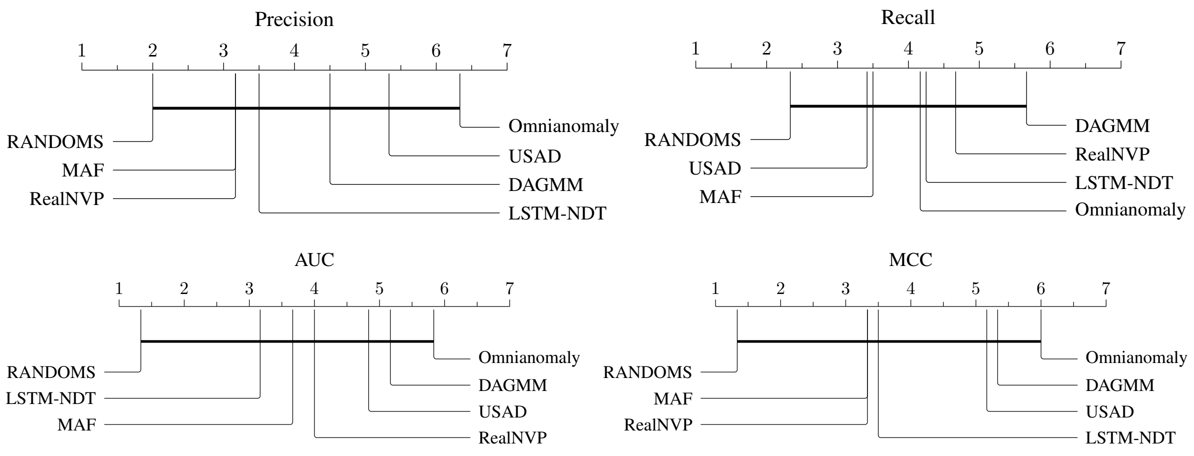


Figure 14 – Comparison of metrics for all methods against each other with the Friedman statistical test with a Wilcoxon signed-rank post-hoc test ($\alpha = 0.05$).

dataset	*method*	HitRate		NDCG	
		100%	150%	100%	150%
MSDS	DAGMM	0.0632	0.0683	0.0428	0.0666
	LSTM-NDT	0.3186	0.4133	0.3168	0.4771
	USAD	0.3062	0.3991	0.3069	0.4642
	Omnianomaly	0.3027	0.3973	0.3034	0.4635
	RANDOMS	**0.7078**	**0.8070**	**0.6559**	**0.8198**
SMD	DAGMM	0.3679	0.4475	0.3687	0.5014
	LSTM-NDT	0.3741	0.4549	0.3742	0.5090
	USAD	0.3691	0.4498	0.3691	0.5035
	Omnianomaly	0.3663	0.4459	0.3676	0.5002
	RANDOMS	**0.5780**	**0.6490**	**0.5375**	**0.6569**

Table 10 – Performance metrics (HitRate and NDCG) for comparison of RANDOMS with baseline methods. Bold values indicate the highest performance within each dataset.

MSDS dataset, where it achieved the highest HitRate and NDCG scores at both thresholds (100% and 150%). This demonstrates its effectiveness in accurately identifying and prioritizing anomalies. The consistency of RANDOMS across different metrics and datasets underscores its robustness compared to other methods. These findings emphasize the potential of RANDOMS as a reliable tool also for the problem of anomaly diagnosis, offering significant improvements over traditional and deep learning models such as DAGMM, LSTM-NDT, USAD, and Omnianomaly.

5.4 Conclusion

In this chapter, we address the problems of anomaly detection and diagnosis within a given time series by using NF-based models and manifold learning techniques called **r**obust **an**omaly **d**etection on **m**ultivariate time **s**eries (RANDOMS). The problems were solved by using an unsupervised learning approach based on some assumptions that fit well with the used models. The experimental results presented demonstrate the efficacy and robustness of our proposed method, RANDOMS, in anomaly detection and diagnosis across diverse datasets. Compared to several state-of-the-art deep learning approaches and traditional methods, RANDOMS consistently achieves higher or competitive performance metrics, including Precision, Recall, F-Score, MCC, and AUC. These metrics highlight the model's ability to accurately identify anomalies while maintaining a low false-positive rate, which is crucial for practical applications.

6 CONCLUSION

“There are unknown unknowns.”
(Donald Rumsfeld)

In this thesis, we address the challenge of automatic anomaly detection in time series data using machine learning techniques. This problem has accumulated significant attention from researchers across diverse fields, including defect analysis in industrial machines (SCHMIDT; SIMIC, 2019), cyber-intrusion detection (BUCZAK; GUVEN, 2016), video anomaly detection (KUMAR *et al.*, 2020), suspicious trajectory detection (DIAS *et al.*, 2020), and sensor networks (ZHANG *et al.*, 2023b). Our goal was to identify abnormal observations during the inference phase. To address this, we use Normalizing Flows and manifold learning methods in an unsupervised anomaly detection task, primarily relying on clustering assumptions proposed by Leung e Leckie (2005), Ruff *et al.* (2021).

According to the granularity, the problem of time series anomaly detection can be divided into three sub-problems. The first one (Problem 1), called *anomaly detection in time series databases*, where the training set $\mathcal{X}$ and test set $\mathcal{X}'$ are sets of times series, and the task is to detect if a time series of the test set is an abnormal instance. The second problem (Problem 2), *anomaly detection within a given time series* is related to solving the task of detecting which points, for a given test time series, are anomalous. The last problem (Problem 3) is closely related to the second one since, in this case, given an abnormal point, detect which of the dimensions contributed most to that point being considered an anomaly.

6.1 Main contributions

As the main contributions of this thesis, we introduce two new approaches to solve the problems mentioned above. The first one, called ag**gr**egated **a**nomaly **d**etection with normaliz**ing** flows (GRADINGS) is a framework for anomaly detection in time series database. Widely explored in trajectory data, this method combines segment anomaly scores into a single anomaly score using normalizing flows for density estimation, enabling the handling of possibly large sequences of different lengths. The second approach called **r**obust **an**omaly **d**etection on **m**ultivariate time **s**eries (RANDOMS), which uses NFs and manifold learning techniques to solve the anomaly detection and diagnosis problems.

In Chapter 4, we introduce GRADINS and discuss the benefits of NFs in time series anomaly detection. We use NF-based log-likelihood for segments and aggregation functions to create a single anomaly score for a time series. We also explore two classic anomaly detection methods, the Local Outlier Factor and Gaussian Mixture Models. Additionally, we present a series of experiments to evaluate the effectiveness of this technique in solving the problem. Our findings show that aggregation functions not sensitive to anomalies, such as the median, can outperform more sensitive functions, like the mean. We also found that larger segments can increase model performance for this type of application.

As our primary approach for solving anomaly detection and diagnosis problems, we introduce RANDOMS (Chapter 5). This Chapter reviews related works on sequential modeling and flow-based anomaly detection methods. The chapter provides a detailed explanation of our proposed method, starting with the motivation behind our approach, followed by an in-depth explanation of the flow step, manifold learning, and the anomaly inference process. In the empirical evaluation section, we highlight the impact of manifold learning, conditional network type, number of neurons per layer, and number of flows and compare our method with other state-of-the-art methods.

We also believe that the chapters on anomaly detection in time series (Chapter 2) and normalizing flows (Chapter 3) can be used as a solid foundation for future researchers of the field. These chapters provide comprehensive overviews of current methodologies, basic knowledge, and advances in these areas. However, the authors recognize the need for a cohesive review that can unify the diverse terminology and concepts across the various sub-fields related to rare event detection.

6.2 Future works

The results of the experiments for both proposals inspire further exploration of the use of normalizing flow-based methods for anomaly detection applications. Future works in anomaly detection in time series databases will examine different NF architectures as convolution-based flows, such as Glow (KINGMA; DHARIWAL, 2018), which can process multi-channel data or even models with more complex invertible transformations, as suggested in (HUANG *et al.*, 2018; OLIVA *et al.*, 2018), are promising future research.

Furthermore, the current research findings on anomaly detection and diagnosis within a given time series encourage exploring combining NF approaches with other techniques. In our future work, we aim to incorporate Extreme Value Theory (EVT) to select the detection threshold. This approach will be integrated into the training of model parameters, guided by the concept of latent outlier exposure (LOE) (QIU *et al.*, 2022). Finally, we also intend to enhance the latent space modeling for both approaches to facilitate the application of semi-supervised learning by using a Gaussian mixture as base distribution, based on the work of Izmailov *et al.* (2020). These exploratory investigations hold substantial potential for elevating the efficacy and adaptability of time series anomaly detection through the utilization of flow-based generative models.

REFERENCES

AGGARWAL, C. C. **Outlier Analysis**. Springer, 2013. ISBN 978-1-4614-6395-5. Disponível em: https://doi.org/10.1007/978-1-4614-6396-2. Acesso em: 15 Fev. 2023.

AGNELLI, J. P.; CADEIRAS, M.; TABAK, E. G.; TURNER, C. V.; Vanden-Eijnden, E. Clustering and Classification through Normalizing Flows in Feature Space. **Multiscale Model. Simul.**, v. 8, n. 5, p. 1784–1802, 2010.

AHMED, C. M.; PALLETI, V. R.; MATHUR, A. P. WADI: a water distribution testbed for research in the design of secure cyber physical systems. In: TSAKALIDES, P.; BEFERULL-LOZANO, B. (Ed.). **Proceedings of the 3rd International Workshop on Cyber-Physical Systems for Smart Water Networks, CySWATER@CPSWeek 2017, Pittsburgh, Pennsylvania, USA, April 21, 2017**. ACM, 2017. p. 25–28. Disponível em: https://doi.org/10.1145/3055366.3055375. Acesso em: 21 Mar. 2023.

AHMED, F.; COURVILLE, A. C. Detecting semantic anomalies. In: **The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020**. AAAI Press, 2020. p. 3154–3162. Disponível em: https://ojs.aaai.org/index.php/AAAI/article/view/5712. Acesso em: 19 Jul. 2023.

ANSCOMBE, F. J. Rejection of outliers. **Technometrics**, Taylor Francis, v. 2, n. 2, p. 123–146, 1960. Disponível em: https://www.tandfonline.com/doi/abs/10.1080/00401706.1960.10489888. Acesso em: 21 Jun. 2023.

AUDIBERT, J.; MICHIARDI, P.; GUYARD, F.; MARTI, S.; ZULUAGA, M. A. USAD: unsupervised anomaly detection on multivariate time series. In: GUPTA, R.; LIU, Y.; TANG, J.; PRAKASH, B. A. (Ed.). **KDD ’20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020**. ACM, 2020. p. 3395–3404. Disponível em: https://doi.org/10.1145/3394486.3403392. Acesso em: 28 Set. 2023.

AXLER, S. **Linear Algebra Done Right**. 3. ed. Cham, Switzerland: Springer International Publishing, 2014. (Undergraduate texts in mathematics).

BAO, Y.; TANG, Z.; LI, H.; ZHANG, Y. Computer vision and deep learning–based data anomaly detection method for structural health monitoring. **Structural Health Monitoring**, v. 18, n. 2, p. 401–421, 2019. Disponível em: https://doi.org/10.1177/1475921718757405. Acesso em: 17 Mai. 2023.

BASHARAT, A.; GRITAI, A.; SHAH, M. Learning object motion patterns for anomaly detection and improved object detection. In: **2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2008), 24-26 June 2008, Anchorage, Alaska, USA**. IEEE Computer Society, 2008. Disponível em: https://doi.org/10.1109/CVPR.2008.4587510. Acesso em: 13 Out. 2023.

BEHRMANN, J.; GRATHWOHL, W.; CHEN, R. T. Q.; DUVENAUD, D.; JACOBSEN, J. Invertible residual networks. In: CHAUDHURI, K.; SALAKHUTDINOV, R. (Ed.). **Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019,**

Long Beach, California, USA. PMLR, 2019. (Proceedings of Machine Learning Research, v. 97), p. 573–582. Disponível em: http://proceedings.mlr.press/v97/behrmann19a.html. Acesso em: 18 Fev. 2024.

BEITLER, J. J.; SOSNOVIK, I.; SMEULDERS, A. W. M. PIE: pseudo-invertible encoder. **CoRR**, abs/2111.00619, 2021. Disponível em: https://arxiv.org/abs/2111.00619. Acesso em: 15 Fev. 2024.

BENAVOLI, A.; CORANI, G.; MANGILI, F. Should we really use post-hoc tests based on mean-ranks? **The Journal of Machine Learning Research**, v. 17, n. 1, p. 152–161, 2016.

BENGIO, Y.; COURVILLE, A. C.; VINCENT, P. Representation learning: A review and new perspectives. **IEEE Trans. Pattern Anal. Mach. Intell.**, v. 35, n. 8, p. 1798–1828, 2013. Disponível em: https://doi.org/10.1109/TPAMI.2013.50. Acesso em: 9 Ago. 2023.

BERG, R. van den; HASENCLEVER, L.; TOMCZAK, J. M.; WELLING, M. Sylvester normalizing flows for variational inference. In: GLOBERSON, A.; SILVA, R. (Ed.). **Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence, UAI 2018, Monterey, California, USA, August 6-10, 2018**. AUAI Press, 2018. p. 393–402. Disponível em: http://auai.org/uai2018/proceedings/papers/156.pdf. Acesso em: 16 Mar. 2023.

BIANCHINI, M.; BELAHCEN, A.; SCARSELLI, F. A comparative study of inductive and transductive learning with feedforward neural networks. In: ADORNI, G.; CAGNONI, S.; GORI, M.; MARATEA, M. (Ed.). **AI*IA 2016: Advances in Artificial Intelligence - XVth International Conference of the Italian Association for Artificial Intelligence, Genova, Italy, November 29 - December 1, 2016, Proceedings**. Springer, 2016. (Lecture Notes in Computer Science, v. 10037), p. 283–293. Disponível em: https://doi.org/10.1007/978-3-319-49130-1_21. Acesso em: 26 Jun. 2023.

Blázquez-García, A.; CONDE, A.; MORI, U.; LOZANO, J. A. A Review on Outlier/Anomaly Detection in Time Series Data. **ACM Computing Surveys**, v. 54, n. 3, p. 1–33, abr. 2022. ISSN 0360-0300, 1557-7341.

BOSE, A. J.; SMOFSKY, A.; LIAO, R.; PANANGADEN, P.; HAMILTON, W. L. Latent variable modelling with hyperbolic normalizing flows. In: **Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event**. PMLR, 2020. (Proceedings of Machine Learning Research, v. 119), p. 1045–1055. Disponível em: http://proceedings.mlr.press/v119/bose20a.html. Acesso em: 15 Dec. 2023.

BREHMER, J.; CRANMER, K. Flows for simultaneous manifold learning and density estimation. In: LAROCHELLE, H.; RANZATO, M.; HADSELL, R.; BALCAN, M.; LIN, H. (Ed.). **Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual**. NeurIPS, 2020. Disponível em: https://proceedings.neurips.cc/paper/2020/hash/051928341be67dcba03f0e04104d9047-Abstract.html. Acesso em: 9 Fev. 2024.

BREUNIG, M. M.; KRIEGEL, H.; NG, R. T.; SANDER, J. LOF: identifying density-based local outliers. In: CHEN, W.; NAUGHTON, J. F.; BERNSTEIN, P. A. (Ed.). **Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, May 16-18, 2000, Dallas, Texas, USA**. ACM, 2000. p. 93–104. Disponível em: https://doi.org/10.1145/342009.335388. Acesso em: 01 Jun. 2023.

BUCZAK, A. L.; GUVEN, E. A survey of data mining and machine learning methods for cyber security intrusion detection. **IEEE Commun. Surv. Tutorials**, v. 18, n. 2, p. 1153–1176, 2016. Disponível em: https://doi.org/10.1109/COMST.2015.2494502. Acesso em: 28 Ago. 2023.

CAO, N. D.; AZIZ, W.; TITOV, I. Block neural autoregressive flow. In: GLOBERSON, A.; SILVA, R. (Ed.). **Proceedings of the Thirty-Fifth Conference on Uncertainty in Artificial Intelligence, UAI 2019, Tel Aviv, Israel, July 22-25, 2019**. AUAI Press, 2019. (Proceedings of Machine Learning Research, v. 115), p. 1263–1273. Disponível em: http://proceedings.mlr.press/v115/de-cao20a.html. Acesso em: 15 Dez. 2022.

CHEN, Z.; PENG, Z.; ZOU, X.; SUN, H. Deep learning based anomaly detection for muti-dimensional time series: A survey. In: LU, W.; ZHANG, Y.; WEN, W.; YAN, H.; LI, C. (Ed.). **Cyber Security - 18th China Annual Conference, CNCERT 2021, Beijing, China, July 20-21, 2021, Revised Selected Papers**. Springer, 2021. (Communications in Computer and Information Science, v. 1506), p. 71–92. Disponível em: https://doi.org/10.1007/978-981-16-9229-1_5.

CHENG, L.; WANG, Y.; LIU, X.; LI, B. Outlier detection ensemble with embedded feature selection. In: **The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020**. AAAI Press, 2020. p. 3503–3512. Disponível em: https://doi.org/10.1609/aaai.v34i04.5755. Acesso em: 04 Set. 2023.

CHO, M.; KIM, T.; KIM, W. J.; CHO, S.; LEE, S. Unsupervised video anomaly detection via normalizing flows with implicit latent features. **Pattern Recognit.**, v. 129, p. 108703, 2022. Disponível em: https://doi.org/10.1016/j.patcog.2022.108703. Acesso em: 25 Jul. 2023.

CHOI, K.; YI, J.; PARK, C.; YOON, S. Deep Learning for Anomaly Detection in Time-Series Data: Review, Analysis, and Guidelines. **IEEE Access**, v. 9, p. 120043–120065, 2021.

CHUNG, J.; GÜLÇEHRE, Ç.; CHO, K.; BENGIO, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. **CoRR**, abs/1412.3555, 2014. Disponível em: http://arxiv.org/abs/1412.3555. Acesso em: 13 Ago. 2023.

CUNNINGHAM, E.; COBB, A. D.; JHA, S. Principal component flows. In: CHAUDHURI, K.; JEGELKA, S.; SONG, L.; SZEPESVÁRI, C.; NIU, G.; SABATO, S. (Ed.). **International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA**. PMLR, 2022. (Proceedings of Machine Learning Research, v. 162), p. 4492–4519. Disponível em: https://proceedings.mlr.press/v162/cunningham22a.html.

CUNNINGHAM, E.; ZABOUNIDIS, R.; AGRAWAL, A.; FITERAU, I.; SHELDON, D. Normalizing flows across dimensions. **CoRR**, abs/2006.13070, 2020. Disponível em: https://arxiv.org/abs/2006.13070.

DARBAN, Z. Z.; WEBB, G. I.; PAN, S.; AGGARWAL, C.; SALEHI, M. Deep learning for time series anomaly detection: A survey. **ACM Comput. Surv.**, v. 57, n. 1, p. 15:1–15:42, 2025. Disponível em: https://doi.org/10.1145/3691338. Acesso em: 22 Dez. 2024.

DEMPSTER, A. P.; LAIRD, N. M.; RUBIN, D. B. Maximum likelihood from incomplete data via the em algorithm. **Journal of the Royal Statistical Society: Series B (Methodological)**, v. 39, n. 1, p. 1–22, 1977. Disponível em: https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.2517-6161.1977.tb01600.x. Acesso em: 25 Ago. 2023.

DEMSAR, J. Statistical comparisons of classifiers over multiple data sets. **J. Mach. Learn. Res.**, v. 7, p. 1–30, 2006. Disponível em: http://jmlr.org/papers/v7/demsar06a.html. Acesso em: 10 Jul. 2023.

DENG, A.; HOOI, B. Graph neural network-based anomaly detection in multivariate time series. In: **Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021**. AAAI Press, 2021. p. 4027–4035. Disponível em: https://doi.org/10.1609/aaai.v35i5.16523. Acesso em: 04 Set. 2023.

DENG, R.; CHANG, B.; BRUBAKER, M. A.; MORI, G.; LEHRMANN, A. M. Modeling continuous stochastic processes with dynamic normalizing flows. In: LAROCHELLE, H.; RANZATO, M.; HADSELL, R.; BALCAN, M.; LIN, H. (Ed.). **Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual**. NeurIPS, 2020. Disponível em: https://proceedings.neurips.cc/paper/2020/hash/58c54802a9fb9526cd0923353a34a7ae-Abstract.html. Acesso em: 19 Jan. 2023.

DIAS, M. L. D.; MATTOS, C. L. C.; SILVA, T. L. C. da; MACÊDO, J. A. F. de; SILVA, W. C. P. Anomaly detection in trajectory data with normalizing flows. In: **2020 International Joint Conference on Neural Networks, IJCNN 2020, Glasgow, United Kingdom, July 19-24, 2020**. IEEE, 2020. p. 1–8. Disponível em: https://doi.org/10.1109/IJCNN48605.2020.9206939. Acesso em: 06 Out. 2022 15:44:35 +0200.

DINH, L.; KRUEGER, D.; BENGIO, Y. NICE: non-linear independent components estimation. In: BENGIO, Y.; LECUN, Y. (Ed.). **3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings**. ICLR, 2015. Disponível em: http://arxiv.org/abs/1410.8516. Acesso em: 17 Jul. 2022.

DINH, L.; SOHL-DICKSTEIN, J.; BENGIO, S. Density estimation using real NVP. In: **5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings**. OpenReview.net, 2017. Disponível em: https://openreview.net/forum?id=HkpbnH9lx.

DOLATABADI, H. M.; ERFANI, S. M.; LECKIE, C. Invertible generative modeling using linear rational splines. In: CHIAPPA, S.; CALANDRA, R. (Ed.). **The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]**. PMLR, 2020. (Proceedings of Machine Learning Research, v. 108), p. 4236–4246. Disponível em: http://proceedings.mlr.press/v108/dolatabadi20a.html. Acesso em: 29 Jun 2023.

FANG, T.; LU, N.; NIU, G.; SUGIYAMA, M. Rethinking importance weighting for deep learning under distribution shift. In: LAROCHELLE, H.; RANZATO, M.; HADSELL, R.; BALCAN, M.; LIN, H. (Ed.). **Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual**. NeurIPS, 2020. Disponível em: https://proceedings.neurips.cc/paper/2020/hash/8b9e7ab295e87570551db122a04c6f7c-Abstract.html. Acesso em: 19 Jan. 2022.

FAWAZ, H. I.; FORESTIER, G.; WEBER, J.; IDOUMGHAR, L.; MULLER, P. Deep learning for time series classification: a review. **Data Min. Knowl. Discov.**, v. 33, n. 4, p. 917–963, 2019. Disponível em: https://doi.org/10.1007/s10618-019-00619-1. Acesso em: 19 Out. 2023.

Feinman, R.; Parthasarathy, N. A Linear Systems Theory of Normalizing Flows. **arXiv e-prints**, p. arXiv:1907.06496, jul. 2019.

FLOURIS, K.; KONUKOGLU, E. Canonical normalizing flows for manifold learning. In: OH, A.; NAUMANN, T.; GLOBERSON, A.; SAENKO, K.; HARDT, M.; LEVINE, S. (Ed.). **Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023**. NeurIPS, 2023. Disponível em: http://papers.nips.cc/paper_files/paper/2023/hash/572a6f16ec44f794fb3e0f8a310acbc6-Abstract-Conference.html. Acesso em: 01 Mar. 2024.

GARG, A.; ZHANG, W.; SAMARAN, J.; SAVITHA, R.; FOO, C.-S. An Evaluation of Anomaly Detection and Diagnosis in Multivariate Time Series. **IEEE Transactions on Neural Networks and Learning Systems**, v. 33, n. 6, p. 2508–2517, jun. 2022. ISSN 2162-2388.

GEMICI, M. C.; REZENDE, D. J.; MOHAMED, S. Normalizing flows on riemannian manifolds. **CoRR**, abs/1611.02304, 2016. Disponível em: http://arxiv.org/abs/1611.02304. Acesso em: 13 Ago. 2023.

GERMAIN, M.; GREGOR, K.; MURRAY, I.; LAROCHELLE, H. MADE: masked autoencoder for distribution estimation. In: BACH, F. R.; BLEI, D. M. (Ed.). **Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015**. JMLR.org, 2015. (JMLR Workshop and Conference Proceedings, v. 37), p. 881–889. Disponível em: http://proceedings.mlr.press/v37/germain15.html. Acesso em: 29 Mai. 2022.

GOH, J.; ADEPU, S.; JUNEJO, K. N.; MATHUR, A. A dataset to support research in the design of secure water treatment systems. In: HAVÂRNEANU, G. M.; SETOLA, R.; NASSOPOULOS, H.; WOLTHUSEN, S. D. (Ed.). **Critical Information Infrastructures Security - 11th International Conference, CRITIS 2016, Paris, France, October 10-12, 2016, Revised Selected Papers**. Springer, 2016. (Lecture Notes in Computer Science, v. 10242), p. 88–99. Disponível em: https://doi.org/10.1007/978-3-319-71368-7_8. Acesso em: 21 Mai. 2023.

GRUBBS, F. E. Procedures for detecting outlying observations in samples. **Technometrics**, Taylor Francis, v. 11, n. 1, p. 1–21, 1969. Disponível em: https://www.tandfonline.com/doi/abs/10.1080/00401706.1969.10490657. Acesso em: 18 Jan. 2023.

GUDOVSKIY, D. A.; ISHIZAKA, S.; KOZUKA, K. CFLOW-AD: real-time unsupervised anomaly detection with localization via conditional normalizing flows. In: **IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2022, Waikoloa, HI, USA, January 3-8, 2022**. IEEE, 2022. p. 1819–1828. Disponível em: https://doi.org/10.1109/WACV51458.2022.00188. Acesso em: 17 Fev. 2023.

GUPTA, M.; GAO, J.; AGGARWAL, C. C.; HAN, J. Outlier Detection for Temporal Data: A Survey. **IEEE Transactions on Knowledge and Data Engineering**, v. 26, n. 9, p. 2250–2267, set. 2014. ISSN 1558-2191.

HAWKINS, D. M. **Identification of Outliers**. Springer, 1980. (Monographs on Applied Probability and Statistics). ISBN 978-94-015-3996-8. Disponível em: https://doi.org/10.1007/978-94-015-3994-4. Acesso em: 25 Jul. 2022.

HAZEL, G. G. Multivariate gaussian MRF for multispectral scene segmentation and anomaly detection. **IEEE Trans. Geosci. Remote. Sens.**, v. 38, n. 3, p. 1199–1211, 2000. Disponível em: https://doi.org/10.1109/36.843012. Acesso em: 12 Mai. 2024.

HENTER, G. E.; ALEXANDERSON, S.; BESKOW, J. MoGlow: Probabilistic and controllable motion synthesis using normalising flows. **ACM Trans. Graph.**, v. 39, n. 6, p. 236:1–236:14, 2020.

HOOGEBOOM, E.; BERG, R. van den; WELLING, M. Emerging convolutions for generative normalizing flows. In: CHAUDHURI, K.; SALAKHUTDINOV, R. (Ed.). **Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA**. PMLR, 2019. (Proceedings of Machine Learning Research, v. 97), p. 2771–2780. Disponível em: http://proceedings.mlr.press/v97/hoogeboom19a.html. Acesso em: 11 Jun. 2023.

HÖRMANDER, L. **The analysis of linear partial differential operators I**. 2. ed. Berlin, Germany: Springer, 1990. (Grundlehren der mathematischen Wissenschaften).

HORVAT, C.; PFISTER, J. Denoising normalizing flow. In: RANZATO, M.; BEYGELZIMER, A.; DAUPHIN, Y. N.; LIANG, P.; VAUGHAN, J. W. (Ed.). **Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual**. NeurIPS, 2021. p. 9099–9111. Disponível em: https://proceedings.neurips.cc/paper/2021/hash/4c07fe24771249c343e70c32289c1192-Abstract.html. Acesso em: 03 Mai. 2024.

HORVAT, C.; PFISTER, J. Density estimation on low-dimensional manifolds: an inflation-deflation approach. **J. Mach. Learn. Res.**, v. 24, p. 61:1–61:37, 2023. Disponível em: http://jmlr.org/papers/v24/21-0235.html.

HUANG, C.; KRUEGER, D.; LACOSTE, A.; COURVILLE, A. C. Neural autoregressive flows. In: DY, J. G.; KRAUSE, A. (Ed.). **Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018**. PMLR, 2018. (Proceedings of Machine Learning Research, v. 80), p. 2083–2092. Disponível em: http://proceedings.mlr.press/v80/huang18d.html.

HUNDMAN, K.; CONSTANTINOU, V.; LAPORTE, C.; COLWELL, I.; SODERSTROM, T. Detecting Spacecraft Anomalies Using LSTMs and Nonparametric Dynamic Thresholding. In: **Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining**. London United Kingdom: ACM, 2018. p. 387–395. ISBN 978-1-4503-5552-0.

IOFFE, S.; SZEGEDY, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: BACH, F. R.; BLEI, D. M. (Ed.). **Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015**. JMLR.org, 2015. (JMLR Workshop and Conference Proceedings, v. 37), p. 448–456. Disponível em: http://proceedings.mlr.press/v37/ioffe15.html.

IWATA, T.; YAMANAKA, Y. Supervised anomaly detection based on deep autoregressive density estimators. **CoRR**, abs/1904.06034, 2019. Disponível em: http://arxiv.org/abs/1904.06034. Acesso em: 27 Abr. 2023.

IZMAILOV, P.; KIRICHENKO, P.; FINZI, M.; WILSON, A. G. Semi-supervised learning with normalizing flows. In: **Proceedings of the 37th International Conference**

on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event. PMLR, 2020. (Proceedings of Machine Learning Research, v. 119), p. 4615–4630. Disponível em: http://proceedings.mlr.press/v119/izmailov20a.html. Acesso em: 15 Dez. 2023.

JAINI, P.; SELBY, K. A.; YU, Y. Sum-of-squares polynomial flow. In: CHAUDHURI, K.; SALAKHUTDINOV, R. (Ed.). **Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA**. PMLR, 2019. (Proceedings of Machine Learning Research, v. 97), p. 3009–3018. Disponível em: http://proceedings.mlr.press/v97/jaini19a.html.

JÄRVELIN, K.; KEKÄLÄINEN, J. Cumulated gain-based evaluation of IR techniques. **ACM Trans. Inf. Syst.**, v. 20, n. 4, p. 422–446, 2002. Disponível em: http://doi.acm.org/10.1145/582415.582418. Acesso em: 09 Jun 2022.

KANWAR, G.; ALBERGO, M. S.; BOYDA, D.; CRANMER, K.; HACKETT, D. C.; RACANIÈRE, S.; REZENDE, D. J.; SHANAHAN, P. E. Equivariant flow-based sampling for lattice gauge theory. **CoRR**, abs/2003.06413, 2020. Disponível em: https://arxiv.org/abs/2003.06413. Acesso em: 14 Out. 2022.

KAPLAN, W. **Advanced calculus**. 5. ed. Upper Saddle River, NJ: Pearson, 2002.

KELLY, M. G.; HAND, D. J.; ADAMS, N. M. The impact of changing populations on classifier performance. In: FAYYAD, U. M.; CHAUDHURI, S.; MADIGAN, D. (Ed.). **Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA, August 15-18, 1999**. ACM, 1999. p. 367–371. Disponível em: https://doi.org/10.1145/312129.312285. Acesso em: 06 Nov 2023.

KIM, H.; LEE, H.; KANG, W. H.; LEE, J. Y.; KIM, N. S. Softflow: Probabilistic framework for normalizing flow on manifolds. In: LAROCHELLE, H.; RANZATO, M.; HADSELL, R.; BALCAN, M.; LIN, H. (Ed.). **Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual**. NeurIPS, 2020. Disponível em: https://proceedings.neurips.cc/paper/2020/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html. Acesso em: 08 Fev. 2024.

KINGMA, D. P.; DHARIWAL, P. Glow: Generative flow with invertible 1x1 convolutions. In: BENGIO, S.; WALLACH, H. M.; LAROCHELLE, H.; GRAUMAN, K.; CESA-BIANCHI, N.; GARNETT, R. (Ed.). **Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada**. NeurIPS, 2018. p. 10236–10245. Disponível em: https://proceedings.neurips.cc/paper/2018/hash/d139db6a236200b21cc7f752979132d0-Abstract.html. Acesso em: 03 Mar. 2023.

KINGMA, D. P.; SALIMANS, T.; JÓZEFOWICZ, R.; CHEN, X.; SUTSKEVER, I.; WELLING, M. Improving variational autoencoders with inverse autoregressive flow. In: LEE, D. D.; SUGIYAMA, M.; LUXBURG, U. von; GUYON, I.; GARNETT, R. (Ed.). **Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain**. NeurIPS, 2016. p. 4736–4744. Disponível em: https://proceedings.neurips.cc/paper/2016/hash/ddeebdeefdb7e7e7a697e1c3e3d8ef54-Abstract.html. Acesso em: 19 Mai. 2023.

KOBYZEV, I.; PRINCE, S. J. D.; BRUBAKER, M. A. Normalizing Flows: An Introduction and Review of Current Methods. **IEEE Trans. Pattern Anal. Mach. Intell.**, v. 43, n. 11, p. 3964–3979, 2021.

KUMAR, M.; BABAEIZADEH, M.; ERHAN, D.; FINN, C.; LEVINE, S.; DINH, L.; KINGMA, D. Videoflow: A conditional flow-based model for stochastic video generation. In: **8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020**. OpenReview.net, 2020. Disponível em: https://openreview.net/forum?id=rJgUfTEYvH. Acesso em: 07 Mai. 2023.

LEUNG, K.; LECKIE, C. Unsupervised anomaly detection in network intrusion detection using clusters. In: ESTIVILL-CASTRO, V. (Ed.). **Computer Science 2005, Twenty-Eighth Australasian Computer Science Conference (ACSC2005), Newcastle, NSW, Australia, January/February 2005**. Australian Computer Society, 2005. (CRPIT, v. 38), p. 333–342. Disponível em: http://crpit.scem.westernsydney.edu.au/abstracts/CRPITV38Leung.html. Acesso em: 09 Set. 2023.

LI, D.; CHEN, D.; JIN, B.; SHI, L.; GOH, J.; NG, S. MAD-GAN: multivariate anomaly detection for time series data with generative adversarial networks. In: TETKO, I. V.; KURKOVÁ, V.; KARPOV, P.; THEIS, F. J. (Ed.). **Artificial Neural Networks and Machine Learning - ICANN 2019: Text and Time Series - 28th International Conference on Artificial Neural Networks, Munich, Germany, September 17-19, 2019, Proceedings, Part IV**. Springer, 2019. (Lecture Notes in Computer Science, v. 11730), p. 703–716. Disponível em: https://doi.org/10.1007/978-3-030-30490-4_56. Acesso em: 08 Out. 2023.

LI, L.; HANSMAN, R. J.; PALACIOS, R.; WELSCH, R. Anomaly detection via a gaussian mixture model for flight operation and safety monitoring. **Transportation Research Part C: Emerging Technologies**, Elsevier, v. 64, p. 45–57, 2016.

LIAO, T. W. Clustering of time series data - a survey. **Pattern Recognit.**, v. 38, n. 11, p. 1857–1874, 2005. Disponível em: https://doi.org/10.1016/j.patcog.2005.01.025. Acesso em: 03 Jan. 2023.

LIM, B.; ZOHREN, S. Time series forecasting with deep learning: A survey. **CoRR**, abs/2004.13408, 2020. Disponível em: https://arxiv.org/abs/2004.13408. Acesso em: 17 Mar. 2023.

LING, C. X.; HUANG, J.; ZHANG, H. AUC: A better measure than accuracy in comparing learning algorithms. In: XIANG, Y.; CHAIB-DRAA, B. (Ed.). **Advances in Artificial Intelligence, 16th Conference of the Canadian Society for Computational Studies of Intelligence, AI 2003, Halifax, Canada, June 11-13, 2003, Proceedings**. Springer, 2003. (Lecture Notes in Computer Science, v. 2671), p. 329–341. Disponível em: https://doi.org/10.1007/3-540-44886-1_25. Acesso em: 28 Set. 2023.

LOAIZA-GANEM, G.; ROSS, B. L.; HOSSEINZADEH, R.; CATERINI, A. L.; CRESSWELL, J. C. Deep generative models through the lens of the manifold hypothesis: A survey and new connections. **CoRR**, abs/2404.02954, 2024. Disponível em: https://doi.org/10.48550/arXiv.2404.02954. Acesso em: 13 Mai. 2024.

LOSHCHILOV, I.; HUTTER, F. Decoupled weight decay regularization. In: **7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9,**

2019. OpenReview.net, 2019. Disponível em: https://openreview.net/forum?id=Bkg6RiCqY7. Acesso em: 25 Jul. 2023.

MEHRASA, N.; DENG, R.; AHMED, M. O.; CHANG, B.; HE, J.; DURAND, T.; BRUBAKER, M. A.; MORI, G. Point process flows. **CoRR**, abs/1910.08281, 2019. Disponível em: http://arxiv.org/abs/1910.08281. Acesso em: 09 Ago. 2023.

MENG, F.; YUAN, G.; LV, S.; WANG, Z.; XIA, S. An overview on trajectory outlier detection. **Artif. Intell. Rev.**, v. 52, n. 4, p. 2437–2456, 2019. Disponível em: https://doi.org/10.1007/s10462-018-9619-1. Acesso em: 18 Out 2023.

MOODY, G.; MARK, R. The impact of the MIT-BIH Arrhythmia Database. **IEEE Engineering in Medicine and Biology Magazine**, v. 20, n. 3, p. 45–50, maio 2001. ISSN 1937-4186.

MORENO-TORRES, J. G.; RAEDER, T.; ALAÍZ-RODRÍGUEZ, R.; CHAWLA, N. V.; HERRERA, F. A unifying view on dataset shift in classification. **Pattern Recognit.**, v. 45, n. 1, p. 521–530, 2012. Disponível em: https://doi.org/10.1016/j.patcog.2011.06.019. Acesso em: 14 Out. 2023.

MÜLLER, T.; MCWILLIAMS, B.; ROUSSELLE, F.; GROSS, M.; NOVÁK, J. Neural importance sampling. **ACM Trans. Graph.**, v. 38, n. 5, p. 145:1–145:19, 2019. Disponível em: https://doi.org/10.1145/3341156. Acesso em: 06 Ago. 2023.

NAKAMURA, T.; IMAMURA, M.; MERCER, R.; KEOGH, E. J. MERLIN: parameter-free discovery of arbitrary length anomalies in massive time series archives. In: PLANT, C.; WANG, H.; CUZZOCREA, A.; ZANIOLO, C.; WU, X. (Ed.). **20th IEEE International Conference on Data Mining, ICDM 2020, Sorrento, Italy, November 17-20, 2020**. IEEE, 2020. p. 1190–1195. Disponível em: https://doi.org/10.1109/ICDM50108.2020.00147. Acesso em: 17 Fev. 2023.

NEDELKOSKI, S.; BOGATINOVSKI, J.; MANDAPATI, A. K.; BECKER, S.; CARDOSO, J.; KAO, O. Multi-source distributed system data for ai-powered analytics. In: BROGI, A.; ZIMMERMANN, W.; KRITIKOS, K. (Ed.). **Service-Oriented and Cloud Computing - 8th IFIP WG 2.14 European Conference, ESOCC 2020, Heraklion, Crete, Greece, September 28-30, 2020, Proceedings**. Springer, 2020. (Lecture Notes in Computer Science, v. 12054), p. 161–176. Disponível em: https://doi.org/10.1007/978-3-030-44769-4_13. Acesso em: 15 Fev. 2023.

OLIVA, J. B.; DUBEY, A.; ZAHEER, M.; PÓCZOS, B.; SALAKHUTDINOV, R.; XING, E. P.; SCHNEIDER, J. Transformation autoregressive networks. In: DY, J. G.; KRAUSE, A. (Ed.). **Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018**. PMLR, 2018. (Proceedings of Machine Learning Research, v. 80), p. 3895–3904. Disponível em: http://proceedings.mlr.press/v80/oliva18a.html. Acesso em: 17 Nov. 2023.

PAPADIAS, D.; TAO, Y. Reverse nearest neighbor query. In: LIU, L.; ÖZSU, M. T. (Ed.). **Encyclopedia of Database Systems**. Springer US, 2009. p. 2434–2438. Disponível em: https://doi.org/10.1007/978-0-387-39940-9_318. Acesso em: 29 Set. 2022.

PAPAMAKARIOS, G.; MURRAY, I.; PAVLAKOU, T. Masked autoregressive flow for density estimation. In: GUYON, I.; LUXBURG, U. von; BENGIO, S.; WALLACH, H. M.; FERGUS, R.; VISHWANATHAN, S. V. N.; GARNETT, R. (Ed.). **Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information**

Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA. NeurIPS, 2017. p. 2338–2347. Disponível em: https://proceedings.neurips.cc/paper/2017/hash/6c1da886822c67822bcf3679d04369fa-Abstract.html. Acesso em: 21 Jan 2022.

PAPAMAKARIOS, G.; NALISNICK, E. T.; REZENDE, D. J.; MOHAMED, S.; LAKSHMINARAYANAN, B. Normalizing flows for probabilistic modeling and inference. **J. Mach. Learn. Res.**, v. 22, p. 57:1–57:64, 2021. Disponível em: http://jmlr.org/papers/v22/19-1028.html. Acesso em: 31 Jan. 2022.

PARZEN, E. An Approach to Time Series Analysis. **The Annals of Mathematical Statistics**, Institute of Mathematical Statistics, v. 32, n. 4, p. 951 – 989, 1961. Disponível em: https://doi.org/10.1214/aoms/1177704840. Acesso em: 21 Ago. 2023.

QIU, C.; LI, A.; KLOFT, M.; RUDOLPH, M.; MANDT, S. Latent outlier exposure for anomaly detection with contaminated data. In: CHAUDHURI, K.; JEGELKA, S.; SONG, L.; SZEPESVÁRI, C.; NIU, G.; SABATO, S. (Ed.). **International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA**. PMLR, 2022. (Proceedings of Machine Learning Research, v. 162), p. 18153–18167. Disponível em: https://proceedings.mlr.press/v162/qiu22b.html. Acesso em: 26 Mai. 2023.

RASUL, K.; SHEIKH, A.; SCHUSTER, I.; BERGMANN, U. M.; VOLLGRAF, R. Multivariate probabilistic time series forecasting via conditioned normalizing flows. In: **9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021**. OpenReview.net, 2021. Disponível em: https://openreview.net/forum?id=WiGQBFuVRv. Acesso em: 03 Mai. 2023.

RAZAVI, S. F.; MEHMANCHI, M. M.; HOSSEINI, R.; TAVASSOLIPOUR, M. Out-of-distribution detection using normalizing flows on the data manifold. **Appl. Intell.**, v. 55, n. 7, p. 658, 2025. Disponível em: https://doi.org/10.1007/s10489-025-06499-x. Acesso em: 12 Mai. 2024.

REN, K.; YANG, H.; ZHAO, Y.; CHEN, W.; XUE, M.; MIAO, H.; HUANG, S.; LIU, J. A robust AUC maximization framework with simultaneous outlier detection and feature selection for positive-unlabeled classification. **IEEE Trans. Neural Networks Learn. Syst.**, v. 30, n. 10, p. 3072–3083, 2019. Disponível em: https://doi.org/10.1109/TNNLS.2018.2870666. Acesso em: 27 Ago. 2023.

REZENDE, D. J.; MOHAMED, S. Variational inference with normalizing flows. In: BACH, F. R.; BLEI, D. M. (Ed.). **Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015**. JMLR.org, 2015. (JMLR Workshop and Conference Proceedings, v. 37), p. 1530–1538. Disponível em: http://proceedings.mlr.press/v37/rezende15.html. Acesso em: 29 Mai. 2021.

REZENDE, D. J.; PAPAMAKARIOS, G.; RACANIÈRE, S.; ALBERGO, M. S.; KANWAR, G.; SHANAHAN, P. E.; CRANMER, K. Normalizing flows on tori and spheres. In: **Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event**. PMLR, 2020. (Proceedings of Machine Learning Research, v. 119), p. 8083–8092. Disponível em: http://proceedings.mlr.press/v119/rezende20a.html. Acesso em: 15 Dec. 2021.

ROUSSOPOULOS, N.; KELLEY, S.; VINCENT, F. Nearest neighbor queries. In: CAREY, M. J.; SCHNEIDER, D. A. (Ed.). **Proceedings of the 1995 ACM SIGMOD International**

Conference on Management of Data, San Jose, California, USA, May 22-25, 1995. ACM Press, 1995. p. 71–79. Disponível em: https://doi.org/10.1145/223784.223794. Acesso em: 21 Mar. 2023.

RUDOLPH, M.; WANDT, B.; ROSENHAHN, B. Same same but differnet: Semi-supervised defect detection with normalizing flows. In: **IEEE Winter Conference on Applications of Computer Vision, WACV 2021, Waikoloa, HI, USA, January 3-8, 2021**. IEEE, 2021. p. 1906–1915. Disponível em: https://doi.org/10.1109/WACV48630.2021.00195. Acesso em: 21 Mar. 2023.

RUFF, L.; KAUFFMANN, J. R.; VANDERMEULEN, R. A.; MONTAVON, G.; SAMEK, W.; KLOFT, M.; DIETTERICH, T. G.; MÜLLER, K. A unifying review of deep and shallow anomaly detection. **Proc. IEEE**, v. 109, n. 5, p. 756–795, 2021. Disponível em: https://doi.org/10.1109/JPROC.2021.3052449. Acesso em: 27 Jul. 2023.

SALEHI, M.; MIRZAEI, H.; HENDRYCKS, D.; LI, Y.; ROHBAN, M. H.; SABOKROU, M. A unified survey on anomaly, novelty, open-set, and out of-distribution detection: Solutions and future challenges. **Trans. Mach. Learn. Res.**, v. 2022, 2022. Disponível em: https://openreview.net/forum?id=aRtjVZvbpK. Acesso em: 19 Mai. 2023.

SALLES, R.; BELLOZE, K. T.; PORTO, F.; GONZALEZ, P. H.; OGASAWARA, E. S. Nonstationary time series transformation methods: An experimental review. **Knowl. Based Syst.**, v. 164, p. 274–291, 2019. Disponível em: https://doi.org/10.1016/j.knosys.2018.10.041. Acesso em: 12 Fev. 2023.

SCHLIMMER, J. C.; GRANGER, R. H. Incremental learning from noisy data. **Mach. Learn.**, v. 1, n. 3, p. 317–354, 1986. Disponível em: https://doi.org/10.1023/A:1022810614389. Acesso em: 02 Mar. 2024.

SCHMIDT, M.; SIMIC, M. Normalizing flows for novelty detection in industrial time series data. **CoRR**, abs/1906.06904, 2019. Disponível em: http://arxiv.org/abs/1906.06904. Acesso em: 24 Jun. 2022.

SHCHUR, O.; BILOS, M.; GÜNNEMANN, S. Intensity-free learning of temporal point processes. In: **8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020**. OpenReview.net, 2020. Disponível em: https://openreview.net/forum?id=HygOjhEYDH. Acesso em: 07 Mai 2024.

SHEPARD, R.; BROZELL, S. R.; GIDOFALVI, G. The representation and parametrization of orthogonal matrices. **The Journal of Physical Chemistry A**, v. 119, n. 28, p. 7924–7939, 2015. PMID: 25946418. Disponível em: https://doi.org/10.1021/acs.jpca.5b02015. Acesso em: 21 Jun. 2023.

SHUMWAY, R. H.; STOFFER, D. S. **Time Series Analysis and Its Applications**. 1. ed. Springer, 2000. (Springer texts in statistics). ISBN 978-3-031-70584-7. Disponível em: https://doi.org/10.1007/978-94-015-3994-4. Acesso em: 21 Jan. 2023.

SU, Y.; ZHAO, Y.; NIU, C.; LIU, R.; SUN, W.; PEI, D. 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**. Anchorage AK USA: ACM, 2019. p. 2828–2837. ISBN 978-1-4503-6201-6.

TABAK, E. G.; Vanden-Eijnden, E. Density estimation by dual ascent of the log-likelihood. **Communications in Mathematical Sciences**, International Press of Boston, v. 8, n. 1, p. 217–233, mar. 2010. ISSN 1539-6746, 1945-0796.

TANG, Z.; CHEN, Z.; BAO, Y.; LI, H. Convolutional neural network-based data anomaly detection method using multiple information for structural health monitoring. **Structural Control and Health Monitoring**, v. 26, n. 1, p. e2296, 2019. E2296 STC-18-0112.R1. Disponível em: https://onlinelibrary.wiley.com/doi/abs/10.1002/stc.2296. Acesso em: 05 Jun. 2023.

TENG, M. Anomaly detection on time series. In: **2010 IEEE International Conference on Progress in Informatics and Computing**. IEEE, 2010. v. 1, p. 603–608. Disponível em: https://ieeexplore.ieee.org/document/5687485. Acesso em: 04 Abr. 2023.

TIAN, J.; HSU, Y.; SHEN, Y.; JIN, H.; KIRA, Z. Exploring covariate and concept shift for detection and calibration of out-of-distribution data. **CoRR**, abs/2110.15231, 2021. Disponível em: https://arxiv.org/abs/2110.15231. Acesso em: 02 Nov. 2023.

TOMCZAK, J. M.; WELLING, M. Improving variational auto-encoders using householder flow. **CoRR**, abs/1611.09630, 2016. Disponível em: http://arxiv.org/abs/1611.09630. Acesso em: 13 Ago. 2022.

Tomczak, J. M.; Welling, M. Improving Variational Auto-Encoders using convex combination linear Inverse Autoregressive Flow. **arXiv e-prints**, p. arXiv:1706.02326, jun. 2017.

TULI, S.; CASALE, G.; JENNINGS, N. R. Tranad: Deep transformer networks for anomaly detection in multivariate time series data. **Proc. VLDB Endow.**, v. 15, n. 6, p. 1201–1214, 2022. Disponível em: https://www.vldb.org/pvldb/vol15/p1201-tuli.pdf. Acesso em: 16 Mar. 2023.

WEHENKEL, A.; LOUPPE, G. Unconstrained monotonic neural networks. In: WALLACH, H. M.; LAROCHELLE, H.; BEYGELZIMER, A.; D'ALCHÉ-BUC, F.; FOX, E. B.; GARNETT, R. (Ed.). **Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada**. NeurIPS, 2019. v. 32, p. 1543–1553. Disponível em: https://proceedings.neurips.cc/paper/2019/hash/2a084e55c87b1ebcdaad1f62fdbbac8e-Abstract.html. Acesso em: 16 Mai. 2022.

WEI, W. W. Time Series Analysis. In: **The Oxford Handbook of Quantitative Methods in Psychology: Vol. 2: Statistical Analysis**. Oxford University Press, 2013. ISBN 9780199934898. Disponível em: https://doi.org/10.1093/oxfordhb/9780199934898.013.0022. Acesso em: 15 Ago. 2023.

WILES, O.; GOWAL, S.; STIMBERG, F.; REBUFFI, S.; KTENA, I.; DVIJOTHAM, K.; CEMGIL, A. T. A fine-grained analysis on distribution shift. In: **The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022**. OpenReview.net, 2022. Disponível em: https://openreview.net/forum?id=Dl4LetuLdyK. Acesso em: 20 Ago. 2023.

WU, R.; KEOGH, E. J. Current time series anomaly detection benchmarks are flawed and are creating the illusion of progress. **IEEE Trans. Knowl. Data Eng.**, v. 35, n. 3, p. 2421–2429, 2023. Disponível em: https://doi.org/10.1109/TKDE.2021.3112126. Acesso em: 28 Ago. 2023.

XU, H.; CHEN, W.; ZHAO, N.; LI, Z.; BU, J.; LI, Z.; LIU, Y.; ZHAO, Y.; PEI, D.; FENG, Y.; CHEN, J.; WANG, Z.; QIAO, H. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In: CHAMPIN, P.; GANDON, F.; LALMAS, M.; IPEIROTIS, P. G. (Ed.). **Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018**. ACM, 2018. p. 187–196. Disponível em: https://doi.org/10.1145/3178876.3185996. Acesso em: 16 Abr. 2023.

YAMAGUCHI, M.; KOIZUMI, Y.; HARADA, N. Adaflow: Domain-adaptive density estimator with application to anomaly detection and unpaired cross-domain translation. In: **IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019**. IEEE, 2019. p. 3647–3651. Disponível em: https://doi.org/10.1109/ICASSP.2019.8683072. Acesso em: 07 May 2024.

YANG, J.; ZHOU, K.; LI, Y.; LIU, Z. Generalized out-of-distribution detection: A survey. **CoRR**, abs/2110.11334, 2021. Disponível em: https://arxiv.org/abs/2110.11334. Acesso em: 18 Jun. 2023.

YANG, X.; ZHUANG, Y.; SHI, M.; CAO, X.; CHEN, D.; TANG, Y. Spiforest: An anomaly detecting algorithm using space partition constructed by probability density-based inverse sampling. **IEEE Trans. Neural Networks Learn. Syst.**, v. 35, n. 6, p. 8013–8025, 2024. Disponível em: https://doi.org/10.1109/TNNLS.2022.3223342. Acesso em: 17 Mai. 2023.

YU, J.; ZHENG, Y.; WANG, X.; LI, W.; WU, Y.; ZHAO, R.; WU, L. Fastflow: Unsupervised anomaly detection and localization via 2d normalizing flows. **CoRR**, abs/2111.07677, 2021. Disponível em: https://arxiv.org/abs/2111.07677. Acesso em: 10 Out. 2022.

ZHANG, C.; SONG, D.; CHEN, Y.; FENG, X.; LUMEZANU, C.; CHENG, W.; NI, J.; ZONG, B.; CHEN, H.; CHAWLA, N. V. A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. In: **The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019**. AAAI Press, 2019. p. 1409–1416. Disponível em: https://doi.org/10.1609/aaai.v33i01.33011409. Acesso em: 21 Jun. 2023.

ZHANG, Y.; CHEN, Y.; WANG, J.; PAN, Z. Unsupervised deep anomaly detection for multi-sensor time-series signals. **IEEE Trans. Knowl. Data Eng.**, v. 35, n. 2, p. 2118–2132, 2023. Disponível em: https://doi.org/10.1109/TKDE.2021.3102110. Acesso em: 25 Fev. 2023.

ZHANG, Y.; WANG, J.; CHEN, Y.; YU, H.; QIN, T. Adaptive memory networks with self-supervised learning for unsupervised anomaly detection. **IEEE Trans. Knowl. Data Eng.**, v. 35, n. 12, p. 12068–12080, 2023. Disponível em: https://doi.org/10.1109/TKDE.2021.3139916. Acesso em: 14 Jan. 2024.

ZHAO, H.; WANG, Y.; DUAN, J.; HUANG, C.; CAO, D.; TONG, Y.; XU, B.; BAI, J.; TONG, J.; ZHANG, Q. Multivariate time-series anomaly detection via graph attention network. In: PLANT, C.; WANG, H.; CUZZOCREA, A.; ZANIOLO, C.; WU, X. (Ed.). **20th IEEE International Conference on Data Mining, ICDM 2020, Sorrento, Italy, November 17-20, 2020**. IEEE, 2020. p. 841–850. Disponível em: https://doi.org/10.1109/ICDM50108.2020.00093. Acesso em: 17 Mar. 2023.

ZHENG, Y. Trajectory data mining: An overview. **ACM Trans. Intell. Syst. Technol.**, v. 6, n. 3, p. 29:1–29:41, 2015. Disponível em: https://doi.org/10.1145/2743025. Acesso em: 01 Mai. 2023.

ZHENG, Y.; LI, Q.; CHEN, Y.; XIE, X.; MA, W. Understanding mobility based on GPS data. In: YOUN, H. Y.; CHO, W. (Ed.). **UbiComp 2008: Ubiquitous Computing, 10th International Conference, UbiComp 2008, Seoul, Korea, September 21-24, 2008, Proceedings**. ACM, 2008. (ACM International Conference Proceeding Series, v. 344), p. 312–321. Disponível em: https://doi.org/10.1145/1409635.1409677. Acesso em: 07 Nov. 2023.

ZHENG, Y.; XIE, X.; MA, W. Geolife: A collaborative social networking service among user, location and trajectory. **IEEE Data Eng. Bull.**, v. 33, n. 2, p. 32–39, 2010. Disponível em: http://sites.computer.org/debull/A10june/geolife.pdf. Acesso em: 06 Jul. 2022.

ZHENG, Y.; ZHANG, L.; XIE, X.; MA, W. Mining interesting locations and travel sequences from GPS trajectories. In: QUEMADA, J.; LEÓN, G.; MAAREK, Y. S.; NEJDL, W. (Ed.). **Proceedings of the 18th International Conference on World Wide Web, WWW 2009, Madrid, Spain, April 20-24, 2009**. ACM, 2009. p. 791–800. Disponível em: https://doi.org/10.1145/1526709.1526816. Acesso em: 06 Nov. 2024.

ZHONG, Z.; FAN, Q.; ZHANG, J.; MA, M.; ZHANG, S.; SUN, Y.; LIN, Q.; ZHANG, Y.; PEI, D. A survey of time series anomaly detection methods in the aiops domain. **CoRR**, abs/2308.00393, 2023. Disponível em: https://doi.org/10.48550/arXiv.2308.00393. Acesso em: 02 Set. 2023.

ZONG, B.; SONG, Q.; MIN, M. R.; CHENG, W.; LUMEZANU, C.; CHO, D.; CHEN, H. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In: **6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings**. OpenReview.net, 2018. Disponível em: https://openreview.net/forum?id=BJJLHbb0-. Acesso em: 18 Dez. 2023.