Intégration de données provenant de réseaux de capteurs multisondes pour évaluer les scénarios de risque dans les systèmes hydrothermaux volcaniques

Michail Giannoulis

UNIVERSITY OF CLERMONT-FERRAND
DOCTORAL SCHOOL EDSPI
DOCTORAL SCHOOL OF ENGINEERING SCIENCES

PHD THESIS

to obtain the title of

PhD of Science

of the University of Clermont-Ferrand
Speciality: Computer Science

Defended by

MICHAIL GIANNOULIS

Deep Learning analysis of multiprobe sensor networks to assess risk scenarios at volcanic hydrothermal systems

Prepared at LIMOS
Defended on December 15, 2023

The members of Jury:

Benjamin Van Wyk de Vries Professor, University of Clermont Auvergne	Président
Gaetana Ganci Researcher, Istituto Nazionale di Geofisica e Vulcanologia	Rapporteur
Sylvie Vergniolle Professor, Institut de Physique du Globe de Paris	Rapporteur
Violaine Antoine Assistant Professor, University of Clermont Auvergne	Examinateur
Haridimos Kondylakis Associate Professor, University of Crete	Examinateur
Vincent Barra Professor, University of Clermont Auvergne (LIMOS)	Directeur de thèse
Andrew Harris Professor, University of Clermont Auvergne (LMV)	Co-directeur de thèse

Acknowledgements

Before we probe into the scientific purpose of this thesis, I would like to acknowledge and express gratitude to the people who participated from different roles in this three years journey.

I want to express my deepest gratitude to Professor Vincent Barra, my thesis supervisor, for his patience and guidance throughout this thesis as well as for the trust he showed to me, giving me the opportunity to continue develop both on personal and cognitive level. He was always available, despite the conditions of confinement that prevailed in the early years of the pandemic, while I have benefited greatly from his solid knowledge over artificial intelligence and his research experience.

I extend my appreciation to Professor Andrew Harris, my thesis co-supervisor, for his guidance and introducing me to the fascinating field of Volcanology and hydrothermal systems. His solid knowledge on this domain, was crucial for me to comprehend the parameters and challenges that needed to be addressed. He, as member of LMV, also met me to other PhD students and members from his lab, such as Sophie, with who we had productive discussions around this topic.

Many thanks to my reviewers Researcher Gaetana Ganci and Professor Sylvie Vergniolle, and also my examiners Professor Benjamin Van Wyk de Vries, Associate Professor Haridimos Kondylakis, and Assistant Professor Violaine Antoine, for their guidance and helpful suggestions to improve the quality of the thesis.

Last but not least, I want to thank two philosophers for keeping me on track. The English writer Alan Wilson Watts and the Cretan, Greek, writer Nikos Kazantzakis, using one of his quotes; Do not condescend to ask: Shall we conquer? Shall we be conquered? Fight on!

Abstract

In diverse application domains, the unsupervised exploration of temporal-based anomalies in multivariate time series presents an ongoing challenge. This exploration, when applied to sensor systems such as surface, spacecraft, or satellite, can assess hazardous situations or discover unexplored phenomena. The objective of this thesis is to develop a data-driven Deep Learning model capable of sending alerts concerning the timing, location, and hazard severity of environmental crises at hydrothermal systems. Indeed, the understanding of the modeling characteristics to facilitate the detection and interpretation of these phenomena remains limited.

To address these characteristics, this thesis examines literature limitations and subsequently proposes a new solution, namely DITAN. DITAN is an unsupervised and domain-agnostic framework developed specifically to tackle the challenge of detecting and interpreting temporal-based anomalies. It employs an encoder-decoder architecture with attention mechanisms and a dynamic structure with hyper-parameter optimization to learn normal behavior as regular context-horizon patterns. The model predicts normality and identifies critical regions associated with high prediction offsets for detecting anomalies. The detected anomalies are interpreted both numerically, by examining their root causes and similarities, and physically using temporal IF-THEN rules and the reasoning power of a knowledge system.

The effectiveness of DITAN is assessed on seven real-world datasets contaminated by different type of temporal-based anomalies with varying durations. DITAN's detection capabilities are evaluated against the ground truth and compared to eleven deep model-based approaches from the addressed literature. Additionally, DITAN's numerical interpretation of detected anomalies is empirically verified. The results show that DITAN outperforms the existing literature in terms of precision, at the cost of a lower recall rate mainly due to sub-capturing the actual anomalies. To ensure optimal results, it is crucial to carefully define the context and horizon sizes.

DITAN is then applied to assess risk scenarios at volcanic hydrothermal systems, detecting anomalies caused by external factors on the surface of a stable hydrothermal system. The results showcase a timeline of fourteen physical event occurrences, each of them assessed by a risk level and verified across actual sensor values. Specifically, ten occurrences involve two meteorological events, while four occurrences are attributed to three surface external driver events that indicate temperature decrease in different zones. Then such externally-driven anomalies can be cleaned and removed from periods when the internal drivers become variable and, hence, the hydrothermal system becomes

unstable.

Since DITAN is domain-agnostic framework due to its hyper-parameter optimization capabilities, its structure can be adapted to be applied on various problems involving predictable multivariate time series.

This PhD was funded by the Agence National de la Recherche (ANR, Program CES 04 Innovations scientifiques et technologiques pour accompagner la transition écologique) through project DIRE (Data-Integration, Risk and the Environment; Project No: ANR-19-CE04-0014-01). ANR-DIRE is led by three CNRS laboratories of Clermont-Ferrand (LMV, LIMOS, LPC) and aims at defining time varying thermal trends at hydrothermal systems in unrest, with a focus on Vulcano (Aeolian Islands, Italy).

Keywords— Anomaly Detection, Volcanic Hydrothermal Systems, Multivariate Time Series, Artificial Intelligence, Deep Learning, Knowledge Systems

Résumé

Dans de nombreux domaines d'application, l'exploration non supervisée d'anomalies temporelles dans des séries chronologiques multivariées constitue un défi à de nombreux titres. Cette exploration, lorsqu'elle est appliquée à des systèmes de capteurs tels que des capteurs de surface, les engins spatiaux ou les satellites, peut évaluer des situations dangereuses ou découvrir des phénomènes inexplorés. L'objectif de cette thèse est de développer un modèle de Deep Learning piloté par les données, capable d'émettre des alertes concernant l'instant, l'emplacement et la gravité des crises environnementales dans les systèmes hydrothermaux. La compréhension des caractéristiques de modélisation pour faciliter la détection et l'interprétation de ces phénomènes reste en effet à ce jour limitée.

Pour répondre à ces problématiques, cette thèse examine les limites de la littérature et propose ensuite une nouvelle solution, appelée DITAN. DITAN est un cadre non supervisé et agnostique développé spécifiquement pour relever le défi de la détection et de l'interprétation d'anomalies temporelles. Il utilise une architecture encodeur-décodeur avec des mécanismes d'attention et une structure dynamique avec optimisation des hyperparamètres pour apprendre le comportement normal comme des motifs réguliers de contexte-horizon. Le modèle prédit la normalité et identifie les régions critiques associées à des défauts de prédiction élevés pour détecter les anomalies. Ces dernières sont interprétées à la fois numériquement, en examinant leurs causes profondes et leurs similitudes, et physiquement, en utilisant des règles SI-ALORS temporelles et la capacité de raisonnement d'un système de connaissances.

L'efficacité de DITAN est évaluée sur sept ensembles de données réelles contaminées par différents types d'anomalies temporelles de durées variables. Les capacités de détection de DITAN sont évaluées par rapport à la vérité terrain et comparées à onze approches basées sur des modèles profonds issues de la littérature. En outre, l'interprétation numérique des anomalies détectées par DITAN est vérifiée empiriquement. Les résultats montrent que DITAN surpasse la littérature existante en termes de précision, au prix d'un taux de rappel plus faible, principalement dû à la sous-capture des anomalies réelles. Pour obtenir des résultats optimaux, il est essentiel de définir avec soin le contexte et la taille de l'horizon.

DITAN est ensuite appliqué pour évaluer les scénarios de risque dans les systèmes hydrothermaux volcaniques, détectant les anomalies causées par des facteurs externes à la surface d'un système hydrothermal stable. Les résultats présentent une chronologie de quatorze événements physiques, chacun d'entre eux étant évalué par

un niveau de risque et vérifié par rapport aux valeurs réelles des capteurs. Plus précisément, dix occurrences impliquent deux événements météorologiques, tandis que quatre occurrences sont attribuées à trois événements externes de surface qui indiquent une baisse de température dans différentes zones. Ces anomalies d'origine externe peuvent alors être nettoyées et supprimées des périodes où les facteurs internes deviennent variables et où, par conséquent, le système hydrothermal devient instable.

DITAN étant un cadre agnostique en raison de ses capacités d'optimisation des hyperparamètres, sa structure peut être adaptée pour être appliquée à divers problèmes impliquant des séries temporelles multivariées.

Cette thèse a été financée par l'Agence Nationale de la Recherche (ANR, Programme CES 04 Innovations scientifiques et technologiques pour accompagner la transition écologique) à travers le projet DIRE (Data-Integration, Risk and the Environment ; Project No : ANR-19-CE04-0014-01). ANR-DIRE est dirigé par trois laboratoires CNRS de Clermont-Ferrand (LMV, LIMOS, LPC) et vise à définir les tendances thermiques variables dans le temps des systèmes hydrothermaux en agitation, avec un focus sur Vulcano (Iles Eoliennes, Italie).

Keywords— Détection d'Anomalies, Système Volcanique Hydrothermal, Série Temporelle Multivariée, Intelligence Arificielle, Deep Learning, Système Expert

Contents

1 Introduction **9**

1.1 Temporal-based Anomaly . . . 10

1.1.1 Feature Resolution . . . 11

1.1.2 Temporal Resolution . . . 11

1.1.3 Severity Score . . . 12

1.2 Exploring Anomalies on Hydrothermal Systems: Requirements . . . 12

1.2.1 Data Characteristics . . . 13

1.2.2 Deep Model-based Anomaly Definition . . . 13

1.2.3 Generic Learning Feature Representations of Normality . . . 14

1.2.4 Predictability Modeling . . . 14

1.3 The DITAN Framework . . . 15

1.4 Contributions . . . 16

2 Gap Analysis: A Review of the Problem **18**

2.1 Autoencoders . . . 19

2.1.1 Anomaly Score . . . 21

2.1.2 Anomaly Detection . . . 22

2.1.3 Anomaly Interpretation . . . 22

2.2 GANs . . . 23

2.2.1 Anomaly Score . . . 25

2.2.2 Anomaly Detection . . . 26

2.2.3 Anomaly Interpretation . . . 27

2.3 Self-Supervised Classification . . . 27

2.4 Predictability Modeling . . . 28

2.4.1 Anomaly Score . . . 28

2.4.2 Anomaly Detection . . . 29

2.5 The Literature Gaps . . . 30
2.5.1 Temporal Support . . . 30
2.5.2 Anomaly Support . . . 31
2.6 Perspectives from the Gap Analysis . . . 32

3 Towards Anomaly Detection **34**
3.1 Pre-processing Strategy . . . 34
3.1.1 Fix Missing Values . . . 35
3.1.2 Data Partitioning . . . 35
3.1.3 Feature Decomposition . . . 35
3.1.4 Features Scaling . . . 36
3.1.5 Forecasting Protocol . . . 36
3.2 Modeling Normality . . . 36
3.2.1 Memorization Components . . . 37
3.2.2 Generalization Components . . . 40
3.3 Dynamic Threshold with built-in Pruning . . . 42
3.3.1 Critical Peaks . . . 43
3.3.2 Critical Regions . . . 43
3.3.3 Robust Pruning . . . 43
3.3.4 Severity Score . . . 44
3.4 Hyper-parameter Configurations . . . 44
3.5 DITAN Detection System: improvements to the Beta version . . . 47
3.5.1 Critical Regions: using non-centralized peak . . . 47
3.5.2 Bandwidth Selection: using Silverman's rule . . . 47

4 Towards Anomaly Interpretation **49**
4.1 Numerical Interpretation . . . 49
4.1.1 Root cause in Feature space . . . 49
4.1.2 Similarities in Unit space . . . 49
4.2 Physical Interpretation . . . 50
4.2.1 Knowledge Management . . . 51
4.2.2 Inference Engine . . . 52
4.2.3 The DITAN knowledge System . . . 54

5 Experimental Evaluation **55**

5.1 Dataset Profiling . . . 55

5.1.1 NASA Spacecraft Telemetry Channels . . . 55

5.1.2 MIT-BIH Supraventricular Arrhythmia . . . 56

5.2 Baseline Model . . . 56

5.3 Hyper-parameters and Model size . . . 57

5.4 Predictability Modeling . . . 58

5.5 Detecting Anomalies . . . 59

5.5.1 Effectiveness of DITAN across channels . . . 60

5.5.2 Effectiveness of DITAN versus other methods . . . 62

5.5.3 Effect of Thresholding on Recall . . . 63

5.6 Numerical Interpretation . . . 64

5.7 Exploring Methodological Improvements . . . 65

6 Application to a Hydro-thermal system **67**

6.1 Problem Description . . . 67

6.1.1 Vulcano and its Thermal History since 2000 . . . 68

6.1.2 The Research Question to be addressed by DITAN . . . 70

6.1.3 The Sensor Network used by DITAN . . . 71

6.2 Domain Knowledge . . . 73

6.3 Vulcano in Year 2020 . . . 74

6.3.1 Forecasting Scenario . . . 75

6.3.2 Pre-processing . . . 76

6.4 Modeling Normality . . . 77

6.5 Detecting Anomalies . . . 78

6.6 Physical Anomalies . . . 81

6.7 Timeline of External-Drivers . . . 86

7 Conclusions and Future Work **88**

7.1 Insights . . . 88

7.2 Limitations . . . 89

7.3 Future Research . . . 89

List of Figures

1.1 The Problem description of an active hydrothermal system . . . 10
1.2 Temporal-based anomalies on the anomalous exploration space (AES). Normal values are color-coded in white, and anomalies in red . . . 11
1.3 Data organization and guidance to detect surface-related anomalies . . . 12
1.4 The overview of the DITAN framework . . . 15

2.1 Architecture of an AE . . . 19
2.2 Architecture of a GAN . . . 24

3.1 A graphical illustration of the pre-processing steps . . . 34
3.2 A graphical illustration of the data partitioning, where each horizontal line represent a different partition, 6 in total . . . 35
3.3 A graphical illustration of the composite Encoder-Decoder with attention using context of size 3 and horizon 1. Solid lines represent information flow while dotted lines regularization on the features (horizontal) and temporal (vertical) axes . . . 37
3.4 A LSTM cell illustrating the gating mechanism . . . 38
3.5 Different batch size configurations per epoch, over a partition of four context-horizon mappings . . . 41
3.6 The dynamic thresholding process . . . 42
3.7 The *bandwidth selection* module implemented using dual-annealing . . . 44
3.8 Hyper-parameters optimization using the *value range* in which bounded. The *type* refers to its use in the DITAN, and the *tuned* refers to the method responsible to select a value from the *value range*. . . . 45

4.1 Overview of proposed knowledge system to characterize physical anomalies . . . 51
4.2 Create / Update a Rule, Web Graphical Environment . . . 51

5.1 MAE and standard deviations for the 6 channels . . . 58
5.2 The discovered and missed actual point anomalies in channel P-4 . . . 61
5.3 The FN and FP across channels per method . . . 63

5.4 The clustered contextual anomalies of channel T-13 65

6.1 The Figure 0.1 from [81]. Sketch of the main sources of thermal emission that can be detected by a satellite or airborne sensor, modified Figure 1 in [82] reproduced by permission of American Geophysical Union. In normal conditions ground (T_{ground}) and air temperature (T_{air}) are approximately equal, so that $\Delta T = T_{ground} - T_{air} \approx 0$. Over a subsurface heat supply, such as a magmatic intrusion above which natural convection in a porous, or fractured, medium carries heat to the surface, ΔT becomes positive. Over a high temperature surface heat source, such as an active lava, ΔT becomes strongly positive. Given data collected at the correct wavelengths and spatial resolution, both anomalies can be detected by a satellite infrared sensor. The schematic also shows the main sources of heat loss from an active lava body, as are typically calculated using infrared data. These being radiation (M_{rad}), convection (M_{conv}) and conduction (M_{cond}). 68

6.2 Figure 1 from [84]: (a) Upper, middle and lower zones of the Vulcano Fossa Fumarole field viewed from the south. (b) Thermal image mosaic of the same area, with the fumarole field limit as defined by the zones of discoloration and anomalous surface temperatures marked with black and white lines, respectively. Distance from rim to crater floor is ≈ 120 meters transects across the border of the heat chimney, that measure temperature below the surface, show clearly the limit of the thermal anomaly. . 69

6.3 Figure 1 from [85]: (a) Orthophotomosaic of Vulcano Fossa showing the zone of active fumaroles (delimited by a black dash line). White dashed line is the boundary of the diffuse heated zone; yellow line represents thermal transect (A–B) where measurements to define ΔT are made; black dots represent individual fumaroles. (b) Location of Vulcano Island within the Aeolian archipelago, north of Sicily (background of Google Earth image ©2018 Digital Globe 69

6.4 Figure 3 from [87]: Temperature at 15 centimeters depth on Profile A–B for 2019 to 2022. The large black arrow marks the hot zone offset into the cold zone after September 2021 70

6.5 A descriptive illustration of the thermal system at Vulcano and its parameters 70

6.6 The Hot (Grey), Cold (Red) and Weather (WS) stations installed on Vulcano's La Fossa Crater and used by DITAN 72

6.7 Preview of rule management module of DITAN 74

6.8 Down-sampled sensor time series for Vulcano in the year 2020 75

6.9 Box-plot of the pre-processed sensors in Vulcano 2020 77

6.10 The convergence plot of DITAN across configurations using Bayes Optimizer 77

6.11 The prediction errors of DITAN across the 7 sensors 79

6.12 A sub-part of the raw and smoothed error sequences on *pressure* sensor 79

6.13 The root causes diagram from our platform, showing critical regions across sensors 80

6.14 The root-causes diagram for pressure, rain and wind speed sensors 82
6.15 The classified anomalous groups and critical regions on the actual values for pressure, rain and wind speed sensors 82
6.16 The timeline of the detected meteorological events (R1, R2) 83
6.17 The root-causes diagram for pressure, wind speed and (red, grey) surface temperature sensors 84
6.18 The classified anomalous groups and critical regions on the actual values for surface temperature sensors 84
6.19 The timeline of the detected surface external driver events (R3, R5, R6) 85
6.20 The root-causes diagram of all temperature sensors 87

List of Tables

1.1 The detection techniques for temporal-based anomalies . . . 13
1.2 Model-based deep anomaly detection frameworks . . . 14
1.3 The model perspectives of the generic learning feature representation of normality framework . . . 15
1.4 Pros and cons of the core technical choices of DITAN . . . 16

2.1 Recent literature instantiations on AE . . . 19
2.2 Recent literature instantiations on GAN . . . 23
2.3 The fundamental temporal-based characteristics: Prediction-based Protocol (PredP), Implicit Attention (ImpAtt), Explicit Attention (ExpAtt), and Dynamic Structure (DynS) . . . 30
2.4 The fundamental detection and interpretation characteristics: Reconstruction Error (RecE), Regression Error (RegE), Dynamic Threshold (DynT), Anomaly Pruning (AnoP), Anomaly Numerical Interpretation (AnoNI), Anomaly Physical Interpretation (AnoPI) . . . 31

3.1 A summary of the DITAN hyper-parameters . . . 45

5.1 Synopsis of the chosen six (channels) multivariate time series . . . 56
5.2 The optimized hyper-parameter values over datasets . . . 57
5.3 The percentage difference of MAE between normal and abnormal records per model . . . 58
5.4 The percentage change of MAE from [20] to DITAN over normal and abnormal records per model . . . 59
5.5 Confusion Matrix of the records across channels; True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN), as well as the False Positive Ratio (FPR), Precision, Recall and F0.5-score . . . 60
5.6 The intersection over union (IoU) per anomaly, given critical regions of DITAN . . . 62
5.7 DITAN with 11 state-of-the-art methods on the MBA dataset, using the evaluation metrics Precision (P), Recall (R), Area under the ROC curve (AUC) and F1 score . . . 63
5.8 The details and quality assessment of unit space clustering the anomalous records per channel . . . 64
5.9 The unit space similarities verified in data space per cluster across channels . . . 65

5.10 The Intersection Over Union (IoU) from Table 5.6 and the new IoU scores 66

6.1 The sensor network used by DITAN 72
6.2 The data gaps in the measurements within the chosen period 72
6.3 The knowledge defined by Experts in the form of temporal rules 73
6.4 Sensors description of Vulcano in the year 2020 75
6.5 Pre-processed sensors description of Vulcano in the year 2020 76
6.6 The hyper-parameters of the optimal DITAN model 78
6.7 DITAN's detection results per sensor 80
6.8 Rules executed using the Inference Engine 81
6.9 The detected meteorological and surface external driver events 86
6.10 The number of critical regions not associated with the executed physical events 86

Chapter 1

Introduction

Surfaces at active hydrothermal systems are heated by an enhanced heat flux due to the condensation of water at depth of a few hundred meters below the surface, as depicted in Figure 1.1. Changes in internal factors, especially the heat flux component supplied by the magma body, changes the degree of heating, and hence the thermal anomaly recorded at the surface. For example, an injection of new magma can result in increased heat fluxes (e.g. [88], [89] and [90]) and, thus, an increase in the magnitude of the thermal anomaly. Alternatively, changes in the permeability can alter the efficiency of heat transfer. For example, fracturing of self-sealed zones will increase permeability, and hence heat flow and heating of the surface [86]. However, such low magnitude, "geothermal" anomalies [81] can also been modulated by at-surface external to the magmatic and hydrothermal systems, such as rain, wind, and solar heating. These multi-probe data are recorded in an orderly fashion and correlated in time constituting a *multivariate time series*[1], in which each time step represents a vector of sensor values referred to as a *record*. The aim here is to identify anomalies created by external factors using deep learning.

[1]Note that the techniques discussed in this thesis only consider regularly sampled time series with the same temporal granularity in all dimensions

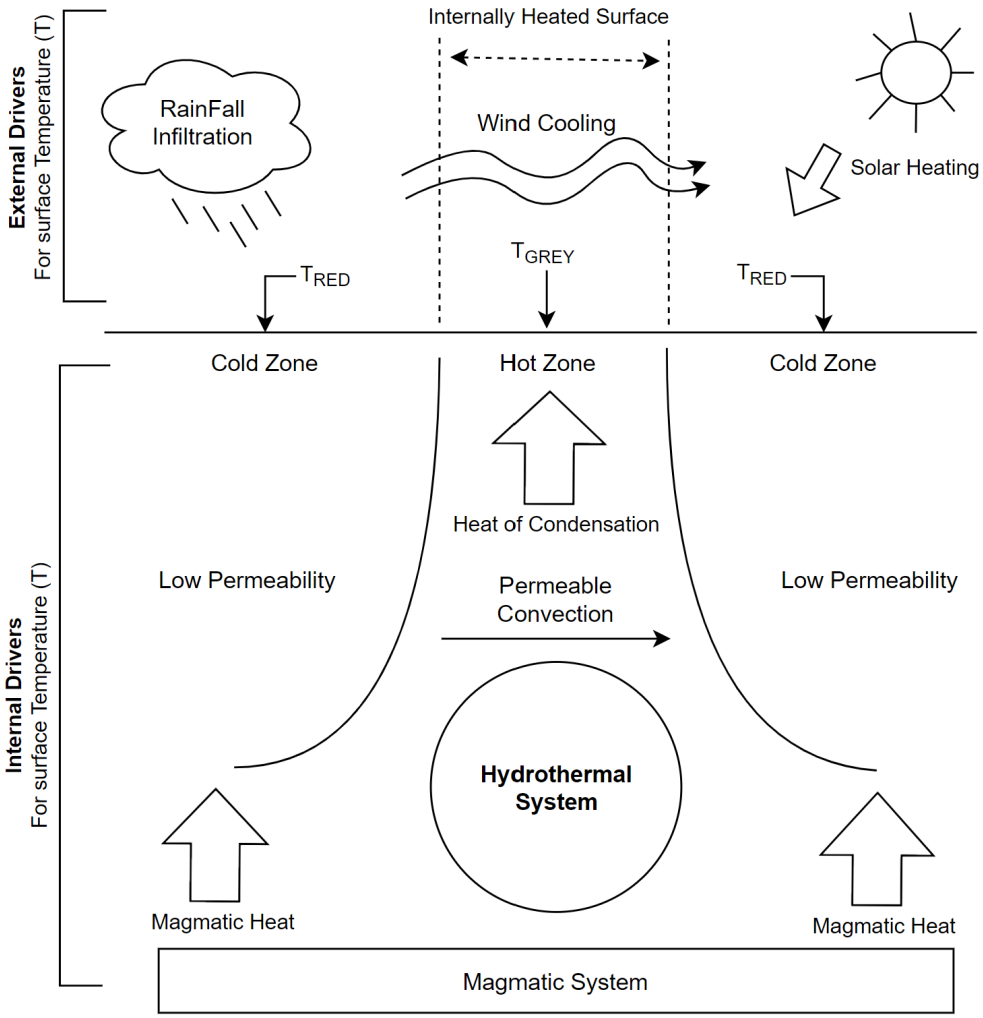


Figure 1.1: The Problem description of an active hydrothermal system

1.1 Temporal-based Anomaly

A *temporal-based anomaly* or simply *anomaly*, is one or more consecutive records (e.g. 3 time-steps) in which sensor values (e.g. pressure and wind speed) deviate from their expected or "normal" behavior. To examine the possible states of an anomaly with respect to its number of records and sensors, this thesis introduces the concept of the *anomaly exploratory space* (AES). The AES is a three-dimensional space where the depth and width represent the feature and temporal resolution respectively, while height corresponds to the severity (anomaly) score. In the AES, an anomaly is depicted as a three-dimensional object at four possible states, demonstrated in Figure 1.2.

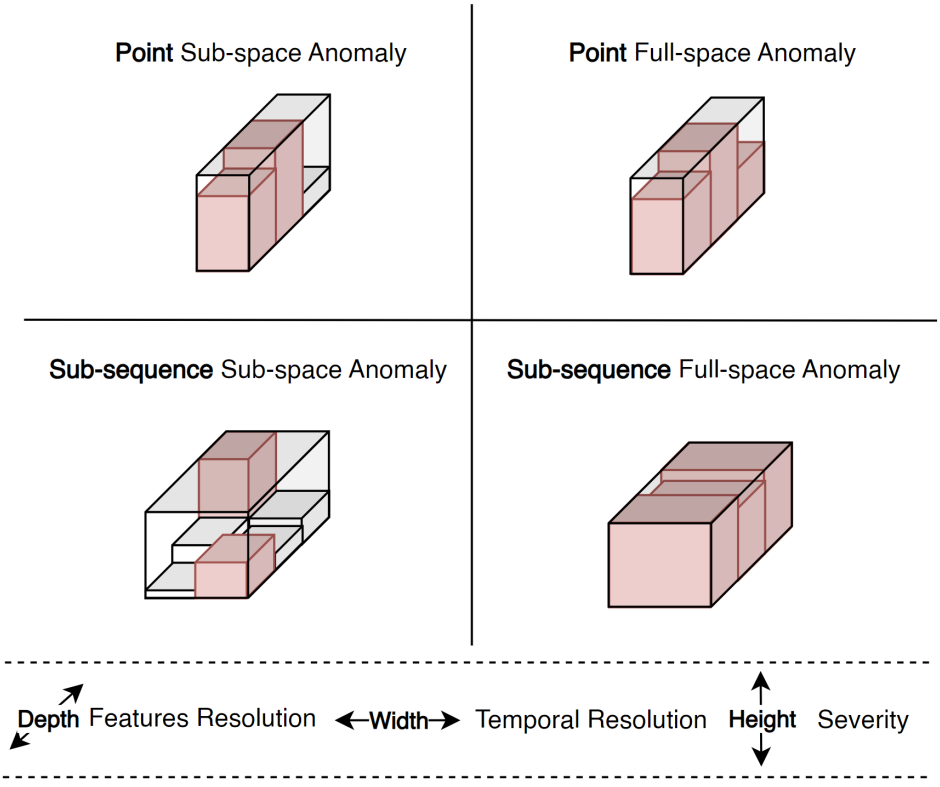


Figure 1.2: Temporal-based anomalies on the anomalous exploration space (AES). Normal values are color-coded in white, and anomalies in red

1.1.1 Feature Resolution

The feature resolution of an anomaly, refers to the number of sensor values within it. An anomaly can either be contaminated by a subset or by the full-set of its sensor values. The former case is called a *subspace anomaly* with respect to those sensors, while the latter is called *fullspace anomaly*. In Figure 1.2, the fullspace anomaly is depicted in the top-right and bottom-right panels, while the subspace anomaly is represented in the top-left and bottom-left panels.

1.1.2 Temporal Resolution

The temporal resolution of an anomaly, refers to the number of records within it. A *point* anomaly is a sub/full space anomaly occurring in an individual record. It is referred to as an individual, because the previous record and the next record is not contaminated. Instead a *subsequence* anomaly comprises two or more consecutive sub/full space anomalous records, not necessarily contaminated by joint sensors. In Figure 1.2, the point anomaly is illustrated in the top-left and top-right panels, and the subsequence anomaly is represented in the bottom-left and bottom-right panels.

1.1.3 Severity Score

The severity score of an anomaly, refers to the intensity of its contamination. To assess the severity of an anomaly, each sensor value is given a score as a result of the difference between the actual sensor value and the predicted (or estimated) value. To identify the turning point from normal to abnormal, it is necessarily to establish a threshold that uses both the temporal and feature resolutions. In Figure 1.2, anomalous (red) sensor values exhibit varying severity, however their minimum magnitude is always higher than the maximum of normal (white) values. The severity score is also an indicator of whether it is a *local* or *global* anomaly. Local anomalies typically receive lower scores due to their limited impact on the time series as a whole, being primarily confined to a specific region in data. Global anomalies, conversely, tend to receive higher scores, indicating their widespread impact on the time series and significance across multiple data regions.

1.2 Exploring Anomalies on Hydrothermal Systems: Requirements

Anomalies due to external drivers can be detected and classified as it is given in Figure 1.3. If carried during a period when external drivers are variable, but internal drivers are stable, all anomalies will be characteristic of a “stable” hydrothermal system whose surface temperature anomalies are only driven by atmospheric effects and meteorological events (the atmospheric system). Once this is defined, then such externally-driven anomalies can be identified, cleaned and removed from periods when the internal drivers become variable and, hence, the hydrothermal system becomes unstable.

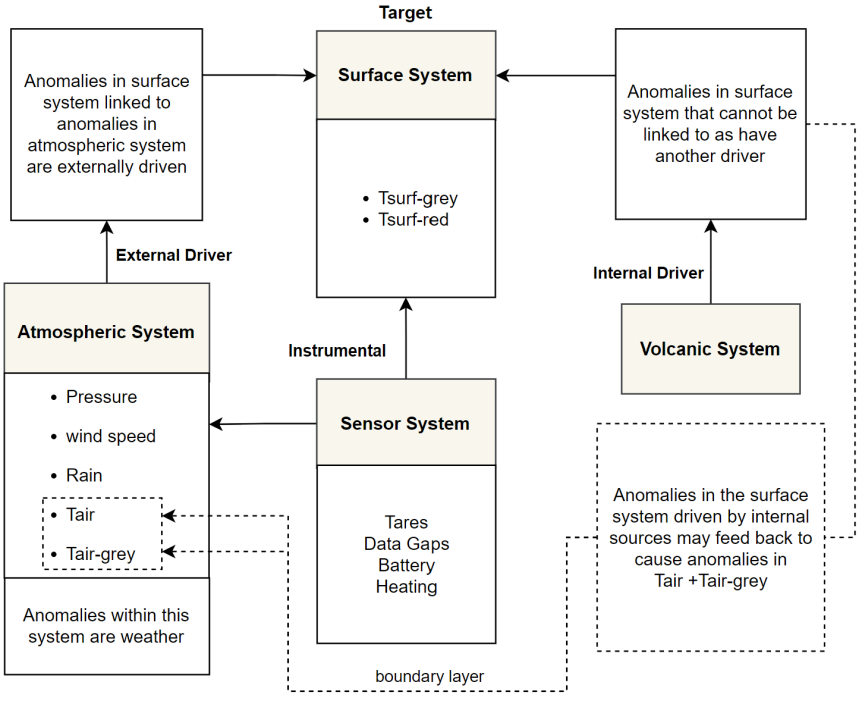


Figure 1.3: Data organization and guidance to detect surface-related anomalies

1.2.1 Data Characteristics

To provide time series data capable of addressing the problem, a sensor network was installed at the active hydrothermal system of the Fossa Crater located in Vulcano, Aeolian Islands, Italy. This network provided a multivariate time series comprising 85000 records, each record characterized by 7 sensors; two sensors from the surface system and five sensors from the atmospheric system (Figure 1.3). A record is sampled every 5 minutes over a one-year period in 2020, during which the internal system remained stable (Figure 1.1). Therefore to detect anomalies created by external factors on the surface of Vulcano, it is crucial to use a deep learning detection technique that supports the system's characteristics; (a) anomaly within correlated multivariate factors (i.e., surface and atmospheric systems), (b) anomaly of varying temporal resolution (e.g., 1 hour or 3 days), and (c) learn the expected (regular) behavior of the system in an unsupervised environment without relying on prior knowledge.

1.2.2 Deep Model-based Anomaly Definition

In an unsupervised environment where labels are not available, determining a threshold becomes crucial in converting the severity score into binary classification (normal or anomalous). Consequently, the definition of an anomaly is closely tied to the underlying assumptions of the detection technique employed. In a recent survey [8], seven anomaly detection techniques were reviewed. These detection techniques are summarized in Table 1.1 and linked to the feature and temporal resolutions defined in Sections 1.1.1 and 1.1.2.

Detection Techniques	Feature Resolution	Temporal Resolution
Model-based (Prediction and Estimation)	Multi/Univariate	Point / Sub sequence
Density-based	Multi/Univariate	Point
Histogram-based	Multi/Univariate	Point
Dissimilarity-based	Multi/univariate	Point / Sub sequence
Discord-based	Univariate	Sub sequence
Frequency-based	Univariate	Sub sequence
Information-Theory	Univariate	Sub sequence

Table 1.1: The detection techniques for temporal-based anomalies

Of the detection techniques collated in Table 1.1, model-based and dissimilarity-based techniques are able to support all the mentioned anomalous states. However, model-based techniques force the model to learn, instead of only comparing with the underlying characteristics of normality. This leads to more interpretable anomalies with flexible definitions which can be achieved by adjusting the model parameters. There are two types of model-based techniques. The *deep* model-based techniques that learn representations using neural networks, and the *traditional* model-based techniques which rely on explicit mathematical models and often require manual feature engineering. The benefits of model-based techniques are also stressed in [1], who concluded that both *deep* and *traditional* model-based techniques are a similarly good choice, with no significant difference in anomaly detection across the

datasets used to assess the models. However, [1] found that *deep* model-based techniques were shown a better choice when local anomalies.

1.2.3 Generic Learning Feature Representations of Normality

In their survey on *deep* model-based techniques, [40] categorized methods into three main frameworks, as given in Table 1.2. Each of these frameworks exhibit a different relation of feature learning and anomaly scoring. In the first framework, a fully disjoint relation is implemented, where an independent anomaly scoring method is applied on the features extracted by a deep model-based method. However, this approach often yields sub-optimal anomaly scores. In contrast, the second framework utilizes deep modeling to jointly learn feature representations and anomaly scores, establishing a *fully joint* relation. However, this approach requires prior knowledge of anomalies, which is not be feasible since data are not labeled. The third framework addresses these limitations by introducing a coupled relation that incorporates some form of dependency between modeling and anomaly scoring. Particularly, the *Learning Feature Representations of Normality* is the only framework aiming to learn meaningful feature representations of normality. It accomplishes this in two ways: through a *measure-dependent* approach limited to a shallow anomaly measurement (e.g. distance-based), or through a *generic* approach that captures various forms of normality as regularities. This, third, framework results into more interpretable anomaly scores, since less frequent patterns in data are expected to result in less frequent feature representations and thus higher anomaly scores. However, the learned feature representations can be biased by infrequent regularities and the presence of extreme values (outliers) in the training data.

Framework	**Categories**	**Learning / Scoring Relation**
Deep Learning for Feature Extraction	-	Fully Disjoint
End-to-end Anomaly Score Learning	-	Fully Joint
Learning Feature Representations of Normality	Generic/Measure-dependent	Coupled

Table 1.2: Model-based deep anomaly detection frameworks

1.2.4 Predictability Modeling

Hence, the choice of modeling is crucial for both detection and interpretation tasks. The four deep model perspectives by [40], that comprise the generic learning feature representations of normality framework, are summarized in Table 1.3, corresponding to four different definitions of anomaly. For the data set and problem considered here, I use the perspective of *Predictability Modeling* (PM) to learn (system's) normal behavior and subsequently define (externally-driven) anomalies. This is because *PM* encompasses the integral temporal properties of a time series to form the basis of my approach: DITAN.

Model Perspective	Anomaly Definition
Auto-encoders (AE)	Irregular records cannot be reconstructed from compressed space
Generative Adversarial Networks (GAN)	Irregular records cannot be generated from latent feature space
Predictability Modeling (PM)	Irregular records are not temporally predictable
Self-supervised Classification (SSC)	Irregular records are inconsistent within classifiers

Table 1.3: The model perspectives of the generic learning feature representation of normality framework

1.3 The DITAN Framework

In this thesis, I develop DITAN[2], a framework, to detect and interpret temporal-based anomalies, built upon three assumptions: (1) time series data are predictable, (2) normality is identical to regularity, and (3) irregular records are temporally less predictable than regular records. DITAN is developed on Python programming language, with the overview of its modules illustrated in Figure 1.4. The pros and cons of the techniques used to implement the modules of DITAN are summarized in Table 1.4, and detailed across Chapters 3 and 4.

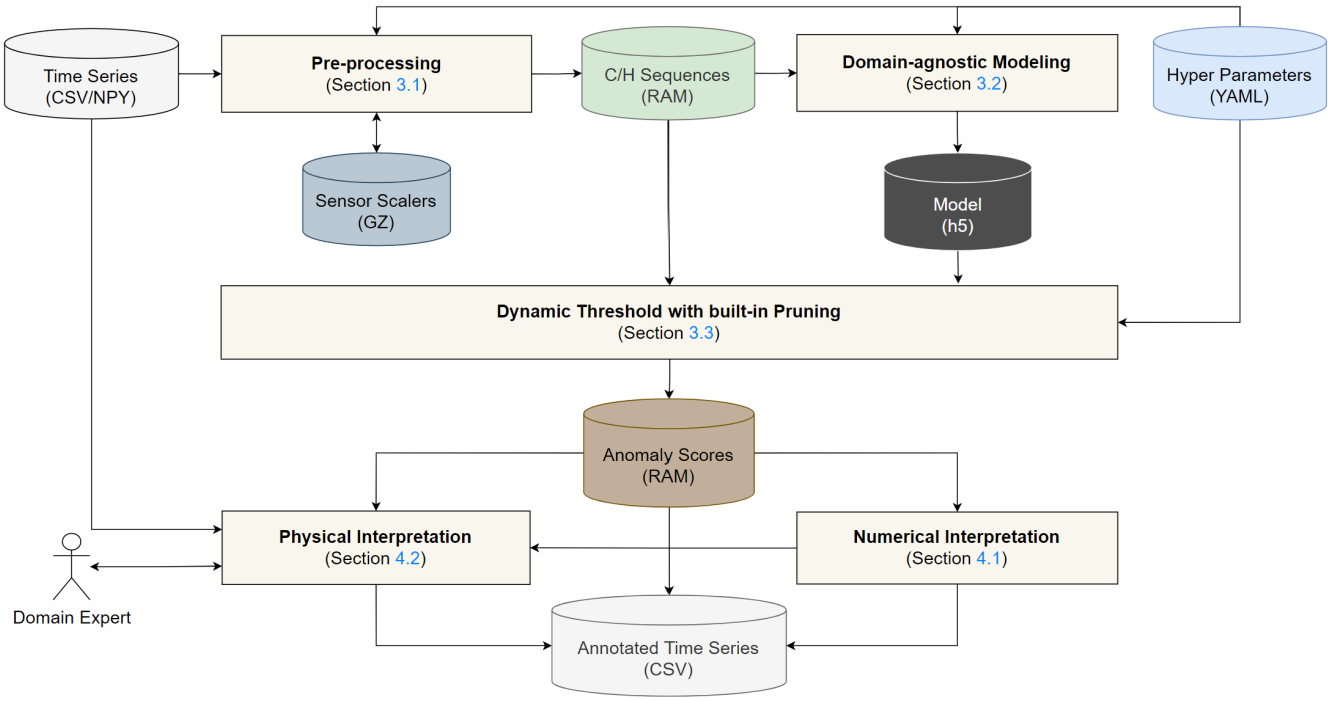


Figure 1.4: The overview of the DITAN framework

[2]https://github.com/migiannoul/DITAN

Section	Technique	Advantage	Disadvantage
3.2.1	Implicit and Explicit Attention	support temporal information	extra hyper/model parameters
3.2.2	Regularization	mitigate overfitting	extra Epochs
3.4	Hyper-parameters Optimization	domain-agnostic modeling	extra executions
3.3.1	Critical Peaks	anomalies of scalable severity	prone to miss-alignment
3.3.2	Critical Regions	anomalies of scalable duration	prone to miss-alignment
3.3.3	Built-in Pruning	reduce miss-alignment	bandwidth selection
4.1.1	Root Cause	the contributors to an anomaly	-
4.1.2	Internal Similarity	group detected anomalies	clusters selection
4.2.2	Knowledge System	infer physical anomalies	knowledge maintenance

Table 1.4: Pros and cons of the core technical choices of DITAN

In the training phase, I use both implicit and explicit attention to better memorize temporal signatures across the time series, at the cost of additional hyper/model parameters. I regularize these parameters to control overfitting during training, which leads to the need for more parses over the training sequence (epochs). The model parameters and number of epochs are then optimized, at the cost of time-prone iterations. Moreover, in the detection phase, I explore varying length temporal-based anomalies using critical peaks/regions, in exchange of possible misalignment across the vertical (frequency) and horizontal (temporal) axes. I force a correction on misalignments via a pruning methodology, which is sensitive to their spread (bandwidth). However, a moderate magnitude peak may still be overshadowed (in the frequency space) by a tail from larger peaks.

In the interpretation phase, detected anomalies are interpreted both numerically and physically. Numerical interpretation involves utilizing a low-cost root cause formula to assess the contribution of sensors to the detected anomalies within the data space, while their similarities are estimated within the model space using a clustering methodology that is sensitive to the defined number of total clusters. On the other hand, the detected anomalies are interpreted into physical events with risk implications, by developing a knowledge system. In this regard, and within the domain of artificial intelligence, risk is defined automatically using the severity scores. However, the maintenance of its knowledge is the task of domain experts who need to assess, interpret and classify the output.

1.4 Contributions

Having identified the approach most suitable to the data set and objective in hand (section 1.2), the remainder of this thesis is organized as follows: First, in Chapter 2, I discuss related work and knowledge gaps in the domain of *generic learning feature representations of normality*. My work attempts to address these limitations by describing the formal foundations for anomaly detection (Chapter 3), and carrying out an interpretation of temporal-based anomalies (Chapter 4). Next, I evaluate the effectiveness of the DITAN approach (Chapter 5), and in Chapter 6 I apply the approach to the real-world scenario (see Section 1.2) where anomalies created by external factors are

present in a time series data set for a stable hydrothermal system, but unlabeled. Finally, I assess directions for future research (Chapter 7). Across these chapters, The major contributions are:

- I identify gaps in current literature on detecting and interpreting temporal-based anomalies (Section 2.5);
- I propose a pre-processing approach tailored to predictable multivariate time series (Section 3.1);
- I introduce a domain-agnostic neural network architecture to predict normality (Section 3.2);
- I explore dynamic thresholding on error sequences as an anomaly detection method (Section 3.3);
- I investigate anomaly interpretation in both feature and model space (Section 4.1);
- I investigate classifying anomalies into different physical event groupings using domain-specific knowledge (Section 4.2);

The DITAN approach shows leading predictability power across different multivariate channels (Section 5.4), and demonstrates improved precision in anomaly detection (Section 5.5). In addition, the numerical interpretations of the DITAN approach are validated (Section 5.6). As a result, the application to the problem of anomaly detection in time series data for a volcanic hydrothermal system, and the physical interpretation of these anomalies, provides useful insights to the impact of chronologically ordered meteorological events on the surface temperature recorded for a hydrothermal system over a period of one year (Section 6.7).

The Chapters 2, 3, 5 and Section 4.1, have been published in [80] with title "*DITAN: A deep-learning domain agnostic framework for detection and interpretation of temporally-based multivariate ANomalies*", on the Elsevier journal of "*Pattern Recognition*", with DOI: "*https://doi.org/10.1016/j.patcog.2023.109814*" by the authors: *Michail Giannoulis* (myself), *Andrew Harris* (co-supervisor) and *Vincent Barra* (supervisor), which sets up and validates the DITAN framework using data sets well-known to artificial intelligence community. The rest of this thesis, Chapter 6 and Section 4.2 will be used for a paper that achieves the objective of this work at a hydrothermal system, that is to detect and define the cause of anomalies in the data sets collected for Vulcano in the year 2020.

Chapter 2

Gap Analysis: A Review of the Problem

In this chapter, I assess literature gaps by conducting a comprehensive analysis of recent work on *Generic Learning Feature Representation of Normality*. I specifically focus on the four model perspectives outlined in Table 1.3, also referred to as *meta-networks* equipped with one or more basic neural networks, such as feed-forward, convolutional, or recurrent networks.

For all model perspectives, the objective is to learn the feature representations in the form of general regularities, within a bulk of mostly normal records. Since a model is composed of artificial neural networks, regularities are expressed through weights Θ, W. During the learning process, these weights are learnt by minimizing a loss function of the form

$$\Theta^*, W^* = \arg\min_{\Theta,W} \sum_{c \in C} l(\psi_W(\phi_\Theta(c)), h)$$

Here, a context *c* is a sequence of one or more consecutive multivariate records, representing the input data. A function ϕ_Θ, parametrized by weight Θ, maps a context $c \in C$ from original space onto a latent representation, $z \in Z$. A surrogate learning task ψ_W, parametrized by weights *W*, operates on latent space and is dedicated to enforce the learning of underlying regularities, mapping latent representation *z* to a horizon $\hat{h} \in H$ back into original space. A horizon $\hat{h}$ is a sequence of one or more consecutive multivariate records, representing the generated output data. Next, a loss function *l* relative to the underlying modeling approach compares $\hat{h}$ to the corresponding actual data *h*. Where, higher weights are expected for regular (normal) patterns.

The purpose of modeling is to forecast normality, by utilizing the trained weights W^*, Θ^* along with mapping functions to construct any horizon $\hat{h}$ given a context c, i.e $\hat{h} = \psi_{W^*}(\phi_{\Theta^*}(c))$. The relation of the records between $\hat{h}$ and *c* is controlled by the forecasting protocol. In an *estimation-based* protocol (*AE*, *GAN* and *SSC*, Table 1.3), records in $\hat{h}$ correspond one by one to c, indicating their estimated normal behavior. In the *prediction-based* protocol (*PM*), records in $\hat{h}$ correspond to subsequent records of c, indicating the predicted normal behavior of a

future sequence. The goodness of a horizon $\hat{h}$ is assessed using a scoring function *f*, i.e. $score(h) = f(c, \phi_{\Theta^*}, \psi_{W^*})$.

2.1 Autoencoders

The *Auto Encoder* (AE) approach aims to learn some low-dimensional feature representation space from which the given records can be well reconstructed, assuming that normal records can be better reconstructed from compressed space than anomalous records. In the literature there is a large family of reconstruction meta-networks with plenty of recent applications (Table 2.1).

Meta Network	**Type**	**Basic Network**	**References**
AE	Multilayer	MLP	[2]
	Convolutional	CNN	[9]
	Recurrent	LSTM	[18, 32, 33, 43]
		BiLSTM	[55]
	Conv & Recurrent	CNN, ConvLSTM	[53]
		CNN, LSTM	[51]
Ensemble AE	Recurrent	LSTM	[24]
Denoising AE	Recurrent	MLP & RNN	[30]
		BiLSTM	[35]
Variational AE	Multilayer	MLP	[28]
	Convolutional	CNN	[36]
	Recurrent	GRU	[27, 44]
		BiLSTM	[42, 41, 52]
		CNN-GRU	[54]

Table 2.1: Recent literature instantiations on AE

An *AE* is composed of two identical and symmetrical networks: an encoder *E* and a decoder *D*, trained with a collaborative objective (Figure 2.1).

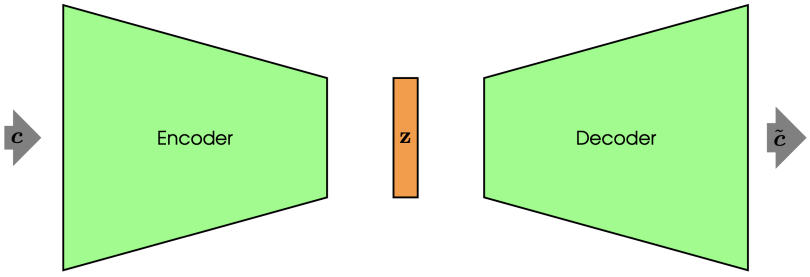


Figure 2.1: Architecture of an AE

E learns to map the input data *c* onto a low-dimensional feature representation *z*, while *D* attempts to find $\hat{h} \approx h = c$ from *z*. Hence, the objective of *E* and *D* is to identically reconstruct the input data with $D(E(c))$, by minimizing the reconstruction error (*RE*) loss function. Both networks are composed by one or more basic networks.

A *standard* AE architecture was implemented in [2, 9, 33, 18, 43, 32, 55, 53, 51, 24] to reconstruct the records of a multivariate time series. From these, [2, 9] used only feed forward (MLP, CNN, Table 2.1) layers. However, conventional (feed-forward) neural networks make the assumption that data is independent in time, which does not hold for sequential data [41] , such as time series. Therefore, recurrent networks are often used. Specifically, [18, 24, 32, 33, 43, 55] authors used only recurrent (LSTM, BiLSTM, GRU, Table 2.1) layers to capture the temporal aspect of the data. Because in some cases features are all-to-all correlated, as in a spatial domain (e.g. image pixels), [51, 53] used both feed forward (CNN) and recurrent (LSTM) layers to capture both spatial and temporal patterns. In particular, [53] both encoder and decoder consisting of CNN layers connected through an additive [3] attention based *ConvLSTM* layer, to adaptively select relevant hidden states across different time steps. On the other hand, [51] used two networks connected in sequence through a window feature sequence layer. CNN layers are first employed to extract spatial feature maps from data. Then, these features are reconstructed using LSTM layers in both the encoder and decoder, capturing potential temporal relevance.
An ensemble of standard AE architectures is proposed in [24], where each AE is trained independently and encourages sparsity in the LSTM layers by randomly removing some connections. An interesting modification in the learning process is proposed by [2, 9], who replaced the standard learning process with an adversarial approach to increase robustness in detecting small anomalies.

Denoising AE (DAE) learn representations that are sensitive to small variations, by forcing the hidden layers to retrieve more robust features and preventing the model from simply learning the identity. [35] used a denoising AE with bidirectional LSTM (BiLSTM) layers to learn robust temporal patterns in both positive and negative directions of the time axis. In addition, [30] proposed a DAE using simple feed forward (MLP) layers to capture feature representations, and then passed these representations to a simple RNN layer to address short-term temporal relations. Due to the difficulty of having a single global objective function for the whole structure, [30] proposed a layer-wise training procedure in which they first trained the MLP layers, then the RNN layers before finally performing a fine tuning to update the parameters of the entire model.

In [27, 28, 36, 42, 41, 44, 52, 54] a *Variational* AE (VAE) based approach was suggested, introducing a regularization into the representation space by encoding records using a prior parametrized distribution over the latent space. Basically, [27, 42, 41, 44, 52] used only recurrent (GRU and BiLSTM) layers, from which [41] employed a self-attention mechanism to improve the encoding-decoding process, with [27, 52] presenting some interesting customizations. In [27], the learning process was modified in a way that encouraged both smooth mean and variance transitions over time, to result a variational smoothness regularizer. [52] extended the VAE architecture to improve modeling capabilities of normal data, accompanied by a loss function which takes into account normal data characteristics. BiLSTM layers were used to construct a re-encoder layer after a creation of VAE network, enabling the extraction of more data features including both original and latent space. A constraint network was then stacked to limit the model ability to reconstruct abnormal data. A VAE network with only feed forward (CNN) units was also

used by [36].

Although feed forward layers are not ideal for capturing temporal information, the VAE network of [36] captured multiple local temporal dependencies by applying a series of convolutions (encoder) and deconvolutions (decoder) with different filter sizes over a windowed time series. Another interesting approach is found in [54], which used convolutional GRU layers to explicitly and jointly capture temporal and spatial dependencies. Finally, [28] trained an encoder and a decoder using dense (MLP) layers in a self-adversarial manner. In the training process, the reconstructed records from decoder are handed back to the encoder to produce the fake latent space. The encoder is also used to judge the "realness" of the reconstruction. The objective of the encoder is to minimize the divergence between the latent space and prior, and to maximize the divergence between the "fake" latent space and prior space. The difference in these models is that VAE instantiations are stochastic generative models that can give calibrated probabilities, while AE instantiations are deterministic discriminative models that do not have a probabilistic foundation.

2.1.1 Anomaly Score

The simplicity of reconstruction forces the vast majority of models [9, 18, 28, 30, 32, 35, 36, 43, 44, 51, 54] to assess an anomaly score using its reconstruction error over the original feature space, i.e $score(h) = ||h - \hat{h}||$. In [53], the anomaly score was defined as the number of poorly reconstructed pairwise correlations, using a predefined threshold. Instead, in [24] the score was defined as the median of its *N* reconstruction errors over an ensemble architecture. In [33], Maximum Likelihood Estimation (MLE) was applied to estimate parameters mean μ and standard deviation Σ of a Normal distribution. Likewise [55] proposed a division relation, considering both forward $\hat{h}$ and backward $\hat{c}$ reconstruction terms, due to their bi-directional architecture:

$$score(h) = \frac{((||h - \hat{h}||_2 + ||c - \hat{c}||_2)/2 - \mu)^2}{2\sigma^2} \tag{2.1}$$

In [33], parameters were combined in a multiplicative way,

$$score(h) = (||h - \hat{h}|| - \mu)^\top \Sigma^{-1} (||h - \hat{h}|| - \mu) \tag{2.2}$$

Other studies [27, 41] make use of the advantages of their variational architecture to assess an anomaly score based on the reconstruction probability using the Sequential Monte Carlo of *L* iterations:

$$score(h) = -\frac{1}{L} \sum_{l=1}^{L} log(p(h|\mu_h^{(l)}, \sigma_h^{(l)})) \tag{2.3}$$

In [42], an anomaly score is assessed using only its (variational) latent representation using a binary (K-means,

Spectral and Hierarchical) clustering on μ_z, as well as the computation of the median Wasserstein distance in both μ_z and Σ_z. In [52], an anomaly score was computed as the normalized addition between the original and latent reconstruction terms, constrained by complementary weights *b*:

$$score(h) = norm(b||h - \hat{h}||_1 + (1 - b)||z - \hat{z}||_1) \tag{2.4}$$

2.1.2 Anomaly Detection

Detecting anomalies by thresholding anomaly scores is a very different issue. Several AE models use a trivial approach to determine a threshold; classifying records into normal or anomalous. A static threshold was deployed by [30, 52] using a fixed reconstruction error value, and in [28, 43] using a fixed quantile over the reconstruction errors. In [43] the quantile was updated over time during the testing phase of the anomaly score, maximizing accuracy. A hyperparameter β can also be introduced into the threshold computation, shifting the median of reconstruction errors [35], or the maximum of the poorly reconstructed errors, in validation data sets [53]. [36] defined a threshold value as the addition between the mean and two standard deviations of the reconstruction errors in a training sequence. The Extreme Value Theory (EVT) was used in [44] to determine a threshold over reconstruction errors. The advantage of EVT is that it makes no assumption on data distribution when finding extreme values. Similarly, [32] was based on the central limit theorem over reconstruction losses to construct a threshold. Basically, the notion of the quantile can be extrapolated using EVT, when there is only normal data.

Thresholds can also be computed using performance measures. [33] selected the threshold maximizing the precision (P) - recall (R) relation over the likelihood values, and [18, 27, 54] chose to maximize the F1-score from a range of thresholds determined using the Area Under the Curve (AUC) of the PR curve. Some methods do not search for a specific threshold at all. In particular, [9] kept raw scores, sorting them in descending order. In [42, 41] a set of thresholds was proposed according to the AUC of the ROC curve. Similarly, [24] used both AUC-PR and AUC-ROC to include all possible thresholds. As an alternate to thresholding, anomalies in [55] (and also [51]) were detected using a Gaussian Segmentation Model (called a *softmax classifier*).

2.1.3 Anomaly Interpretation

Apart from scoring and detecting anomalies, only three of the aforementioned instantiations [9, 44, 53] investigated the anomaly characteristics. [9, 53] reported the root cause of each anomalous record using the top-k sub scores of its features. Instead, [44] used feature reconstruction probabilities in ascending order, under assumption that the most anomalous features can provide sufficient clues to understand and troubleshoot the detected anomaly. Note that [53] additionally reported the severity level as a function of the duration (length) of subsequence anomalies.

2.2 GANs

Generative Adversarial Networks (GAN) aim to learn a generative model capturing the normality of the given records, assuming that normal records can be better generated from the latent space than anomalous records. Table 2.2 gives an overview of some models that have used GANs. A standard GAN is composed of two networks trained simultaneously using adversarial objectives (Figure 2.2). A generator *G* network takes, as input, a noise vector *z* randomly selected from a latent space *Z* to generate a fixed size subsequence of synthetic records $\hat{h} \approx h = c$. These records are supposed to be realistic, capturing the actual *c* data distributions. Instead, a discriminator *D* network takes as input either actual *c* or synthetic *h* records to estimate a prediction score. The objective of *G* is to fool *D* that its synthetic records are real, while the objective of *D* is to correctly distinguish between synthetic and actual data.

Meta Network	**Type**	**Basic Network**	**References**
GAN	Multilayer and Convolutional	MLP& CNN	[45]
	Recurrent	LSTM	[26, 4]
AE-GAN	Convolutional	CNN	[56]
	Convolutional and Recurrent	BiLSTM, CNN	[14]
	Multilayer	MLP	[49]
VAE-GAN	Recurrent	LSTM	[39]
AE-E-GAN	Convolutional and Recurrent	CNN, ConvLSTM	[23]
	Convolutional	CNN	[21]

Table 2.2: Recent literature instantiations on GAN

Standard GAN architectures are proposed in [4, 26, 45] to identically generate the records of a multivariate time series. Univariate models are instantiated in [45], one per sensor, using feed-forward layers in both generator (MLP) and discriminator (CNN). A recent study by [25] suggests that recurrent (LSTM) layers are more suitable for the learning procedure of generative networks over complex time series data, due to their memory blocks. As a result, [4, 26] used a multivariate model using only LSTM layers for both generator and discriminator, and proposed a shallow discriminator and a medium depth generator, to capture anomalies when there is a small number of available records [4]. If standard GANs architectures learn to generate data from a latent space using the generator, they do not learn the inverse mapping G^{-1} back to the latent space.

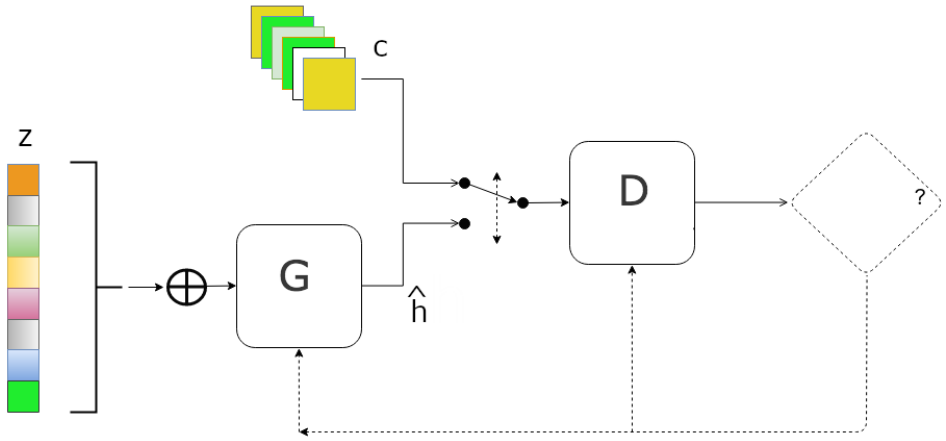


Figure 2.2: Architecture of a GAN

Several studies [14, 39, 49, 56] have introduced an encoder *E* that learns G^{-1}. Hence, the generator takes on the role of a decoder, formulating a new network *G* composed of an encoder G_E and decoder G_D, interacting to minimize a *reconstruction* (apparent) loss, i.e., a distance measure between actual records *h* and reconstructed (generated) records $\hat{h}$ in the original space. G_E and G_D are trained simultaneously or in a post-hoc manner to maintain the training complexity. In [14, 49, 56] the generator is a *standard* AE network, used to simultaneously map from the latent space to data space $G_D(z)$ and vice versa $G_E(c)$. In [39], the generator is a *variational* AE, applied to additionally learn a distribution in the latent space *Z*. A more advanced architecture is proposed in [21, 23], who introduced an additional encoder *E'* after a *standard* AE generator, to enforce similar inputs to lie close to each other in both original space and latent space. This variation introduces a new loss term for G, the *latent* loss, representing the distance between *z* and $\hat{z}$, where $\hat{z}$ is the encoded bottleneck representation of $\hat{h}$. Finally, in [21], a binary cross entropy term over the prediction scores of the discriminator on *h* in the loss function of G was introduced.

The one and only task of the discriminator is to distinguish between actual *h* and generated $\hat{h}$ records. If [21, 49, 56] used MLP, [14, 23, 39] retained temporal information. In [39] LSTM layers are used to implicitly learn temporality, while in [14] BiLSTM (respectively CNN) layers are defined for G (resp. D), with the aim of capturing local temporal features. In [23], an explicit attention layer is introduced through the use of a smoothed attention mechanism over an additional ConvLSTM layer along CNN layers, to jointly capture the spatial patterns and temporality.

A fundamental problem of GAN architectures, namely *mode collapse*, is that the generator tends to learn only a small fraction of data variability, such that it cannot perfectly converge to the actual distribution. This is mainly because the generator is reluctant to produce records that capture other modes in data beyond the ones which fool the discriminator. To overcome this limitation, [14, 23] applied Wasserstein loss as the adversarial loss. In this way, the generator is forced to not only focus on a subset of distribution and thus to theoretically converge to the actual

distribution.

The purpose of GAN is to reconstruct the input time series. However, an adversarial loss alone cannot guarantee mapping an individual record to a desired latent space, from which the record reconstruction can then be mapped. To reduce the search space of the mapping function, [14] adapted a cycle consistency loss to time series reconstruction, introducing two critics C_h and C_z to the discriminator, where, C_h is an indicator of how real the actual or generated records are, and C_z measures the "goodness" of the mapping into a latent space. Interestingly, they concluded that adding backward consistency loss did not notably improve the performance.

2.2.1 Anomaly Score

The architecture preferences also affect the anomaly scoring methodology. In *standard* GAN architectures, *G* is fed with a random vector *z* to generate a horizon $\hat{h}$. The horizon is acquired through a series of λ back propagation steps to update the parameters of *G* until $\hat{h}$ is close to the input context. Thus an anomaly score can only be determined after λ iterations.

In [45], an anomaly score was assessed for each sensor *s* as the prediction score of discriminator for its generated horizon:

$$score_s(h) = D_s(G_s(z_\lambda)) \tag{2.5}$$

In [4, 26], both generator and discrimination losses were considered. The former term measures the reconstruction error in the original space, while the latter measures the error in a rich feature space of the last intermediate layer *f* in the discriminator:

$$score(h) = (1-\alpha)||h - G(z_\lambda)||_1 + \alpha||f(h) - f(G(z_\lambda))||_1 \tag{2.6}$$

The addition of an encoder G_E in the generator of a GAN architecture is useful also in the anomaly scoring process. Such an architecture does not require λ back propagation steps. Instead, G_E is handed directly to the context *c* to estimate a latent representation $G_E(c)$, which is then fed to G_D to generate a horizon $\hat{h} = G_D(G_E(c))$. For example, [49] used both terms proposed in Eq. 2.6 and introduced scalars n_h, n_f, κ, where, n_h is the number of sensor values of the input record *h*, n_f is the number of neurons of the *f* layer, and κ is a coefficient to adjust the weights of the reconstruction and feature losses:

$$score(h) = \frac{\kappa}{n_h}||h - G_D(G_E(c))||_2 + \frac{1}{n_f}||f(h) - f(G_D(G_E(c)))||_2 \tag{2.7}$$

[56] only retained the first error term of Eq. 2.7 to define their loss function.

A relatively low prediction score is expected for input records which do not conform to the distribution of normal data. For this direction, reconstruction error along with the prediction score of discriminator for the actual data were

used in [39]:

$$score(h) = (1 - \alpha)||h - G_D(G_E(c))|| - \alpha D(h) \tag{2.8}$$

In [14], a reconstruction error was used along with critic C_h of their discriminator as an indicator of how real the actual input record is. Due to the contractive definition of the anomaly score, they standardized the scores into z-scores *Z*. Two scoring functions were used, where for *h* this is:

$$score(h) = \alpha Z(||h - G_D(G_E(h))||) \odot Z(C_h(h)) \tag{2.9}$$

The addition of an encoder *E'* after the generator of a GAN architecture is useful also in the anomaly scoring process. Such an architecture introduces the error between the latent representation *z* of *h* and the reconstructed $\hat{z}$ of $\hat{h}$. For example [21] defined a score taking into account the reconstruction errors in both original (h, $\hat{h}$) and latent (z, $\hat{z}$) spaces:

$$score(h) = \alpha||h - G_D(G_E(h))|| + (1 - \alpha)||G_E(h) - E'(G_D(G_E(h)))||_2 \tag{2.10}$$

In [23], the anomaly score was defined as the number of poorly reconstructed pairwise correlations, using a predefined threshold, considering the reconstruction in both original and latent space, to be less sensitive to severe anomalies.

2.2.2 Anomaly Detection

Detecting anomalous records is usually related to thresholding of anomaly scores. A dynamic thresholding methodology was proposed by [45], where a sensor value was considered anomalous if its anomaly score was higher than the prediction score of the discriminator for its actual sensor value. Static thresholds have also been proposed, either as predefined [4] values extracted using a fixed quantile over the reconstruction errors in training sequences [26], or using a locally adaptive [14], or even optimized [23, 39], approach. In [14] noticed that the sliding windows may produce false positives and, based on [20], applied a pruning methodology to mitigate them. In this approach each window is first assessed by its maximum anomaly score, and then a decreasing percent for all descending scores is computed, re-classifying as normal each window that does not exceed a certain threshold.

Alternatively, some methods used no threshold at all. Anomaly scores in [49, 56] were reported as heat maps, where sensor values with hot colors, attracted expert attention to the anomalous portions. In [21], the raw anomaly scores were simply reported, in which the closer to zero the anomaly score was, then the more "normal" the record was considered.

2.2.3 Anomaly Interpretation

Root causes of anomalies were examined in [23], to automatically extract the number of root causes using an elbow method on the anomaly scores distribution, as opposed to [53] which fixed the number of root causes as three. With this approach, [23], aim to find the point where the amount of errors become very small and close to each other.

2.3 Self-Supervised Classification

A few estimation-based models have been proposed using *Self Supervised Classification* (*SSC*). This approach learns representations of normality by building classification models in a self-supervised manner, and identifies records that are inconsistent with these models as anomalies. Traditionally, shallow methods have been introduced based on cross-feature analysis [19] and feature models [47], in which each model evaluates a sensor value of a record with respect to the rest of its sensor values. Hence, the consistency of a record is measured either as the average prediction results [19] or as the majority voting of binary decisions [47]. Recently, deep methods [16, 48] have focused on capturing spatial information in images and on using transformation-based feature augmentation to build different models. Formally, a context c is augmented by T different transformations, parsed through a function ϕ to result in a latent representation for each transformation $\{z_{(1)}, .., z_{(T)}\}$. The latent representations are then fed to a multi-class classifier ψ to result the corresponding horizons $\{\hat{h}_{(1)}, .., \hat{h}_{(T)}\}$. A standard cross entropy loss is then applied over $(\hat{h}_{(j)}, h_{(j)})$ pairs, where $h_{(j)}$ encodes the synthetic class for records augmented using the transformation operation $T_{(j)}$. In [16], the classification scores resulting from ψ were aggregated using a simple average associated with different $T_{(j)}$ to compute the anomaly score. In [48], three strategies were proposed to define the anomaly scores: average prediction probability, maximum prediction probability and negative entropy across all prediction probabilities.

Although the both traditional and deep models have focused on image data, there are some slight variations applicable to audio signals [15] or to broad types of data [6, 22]. Different types of audio-inspired augmentations were applied in [15], together with application of convolutional layers and resulting softmax classification scores, to define anomaly scores. Instead of using geometric or audio-inspired transformations, [6] used feature-level transformations to map data into a finite number of subspaces, before learning a feature mapping that maximizes the difference between inter-class and intra-class separations. Fully connected layers and resulting softmax classification scores over subspace transformations were then used to assess anomaly scores. In [22], authors proposed the autoML pipeline was proposed, consisting of three key parts: auto representation learning, auto anomaly score calculation, and auto negative sample generation. Here, *auto* stands for simultaneously Bayesian optimization of hyper-parameters along with anomaly detection in the latent space using Gaussian Mixture Model to characterize the level of abnormality.

2.4 Predictability Modeling

There are many natural phenomena that require application of a prediction algorithm to answer important questions, such as what will be future population variations, what are the likely orbits of astronomical objects, or what is the probable the occurrence of seismic waves? The *Predictability Modeling* (PM) approach aims at learning feature representations by predicting output data, assuming that normal records are temporally more predictable than anomalous ones. To my knowledge, only a few such methods have been introduced [20, 37, 46].

Formally, a context *c* is given to an encoder *E* to result in a latent representation *z*, which is fed to a decoder *D* to result a constructed horizon $\hat{h}$, where $\hat{h} \approx h \neq c$. As a result, *PM* uses an *Encoder-Decoder* (ED) architecture to predict a single record, or a sequence of records, using a sequence-to-point (S2P) or a sequence-to-sequence (S2S) learning process, respectively. *PM* is a complementary approach to *AE*, since *AE* only captures the case of input data reconstruction in which $h = c$. An *ED* architecture was proposed by [37, 46] using convolutional layers with automatically derived hyper-parameters obtained via a grid-search run for data collected over a series of experiments [37] or over a dataset augmented with synthetic anomalies [46]. In [37], employed an S2P learning process to allow the model to learnt how to predict a current record using a sequence of previous (prior) records, by minimizing the regression loss. In [46], a S2S learning process was employed to learn to predict both current and previous records using only the previous records. In [46], a reconstruction task was introduced to decoder. This mapping choice enabled minimization of a composite loss function, which took into account the relative importance of reconstruction and regression errors. Essentially, only the first part of $\hat{h}$ was used in the reconstruction error, while the entire $\hat{h}$ was used in the regression error.

Compared with the superior performance of LSTM in capturing temporal information, I notice a literature gap in predictability modeling through employing *ED* using recurrent (e.g. LSTM, BiLSTM, GRU) layers. An exception is the stand-alone LSTM implementation with fixed hyper-parameters that have been recently proposed by [20], where the regression loss is minimized using an S2P approach. Beyond anomaly detection, there is recent literature focusing on *PM* which has used an *ED* architecture with recurrent layers and an explicit attention mechanism. In [11], a S2S learning process was developed for multi-step prediction, using BiLSTM layers to encode sequential input data in both directions, a temporal attention layer based on [3], and LSTM layers to decode the hidden representation. These methods have been used to tackle real-world problems, such as in a pandemic crises [7]. A common conclusion of these studies is that an attention mechanism can effectively improve model performance, since the prediction ability gradually degrades as the length of input data increases.

2.4.1 Anomaly Score

Once a prediction is made, it is necessary to assess the anomaly score. The actual horizon *h* is scored as the L2-norm [37] (or L1-norm [20]) discrepancy from its predicted horizon $\hat{h}$, by formulating the regression error. Since

h represents a single record, the scores of *K* consecutive records are aggregated to assess an anomaly score to a fixed-length subsequence. A statistical approach was then proposed by [20] to score a dynamic-length subsequence of records:

$$score(e_{seq}^{(i)}) = \frac{max(e_{seq}^{(i)}) - \epsilon}{\mu(e_s) + \sigma(e_s)} \tag{2.11}$$

where e_s is a one-dimensional vector of regression errors, smoothed using an Exponentially Weight Moving Average (EWMA). In addition, $\epsilon = argmax(\mu(e_s) + k\sigma(e_s))$ in which *k* is an ordered set of positive values representing the number of standard deviations over e_s. Also, e_{seq} is an arbitrary-length subsequence of consecutive smoothed scores from e_s which exceed the ϵ value. Finally, *i* refers to the sensor (feature) dimension in a multivariate environment.

A composite approach (Eq.2.12) was proposed by [46]. Where a reconstruction rec and regression reg error are defined as metrics for evaluating the degree of anomaly of an actual horizon *h*. Both metrics use Frobenius norm to measure the deviation between *h* and its predicted horizon $\hat{h}$. Here, reconstruction refers to the first *p* records, while regression addresses all the *p+q* records of *h*. Since *h* represents a sequence of records, the average of its K_{rec} and K_{reg} scores were respectively used to assess a reconstruction and regression score per record, where *K* indicates the number of horizons that the record of interest appears in:

$$score_{rec}(h) = \sum_{0 \leq i < p} ||h_k^i - \hat{h}_k^i||_F \quad \text{and} \quad score_{reg}(h) = \sum_{0 \leq i < p+q} ||h_k^i - \hat{h}_k^i||_F \tag{2.12}$$

2.4.2 Anomaly Detection

The detection of an anomalous point or sequence of points over the derived scores is usually a result of a threshold value. A fixed-value is determined in [37, 46], either empirically [37] or from the maximum regression error in the training sequence [46]. A dynamic threshold is determined in [20], to result in a set of anomalous sequences E_{seq} as obtained from the smoothed scores e_s. In addition, [20] noticed that the precision of detection heavily depends on the amount of data used to set a threshold. They thus proposed a pruning mechanism to mitigate False Positives (FP), by re-examining the abnormality of sequences in E_{seq}. To aid this, a vector e_{max} is constructed from the sorted E_{seq} and the maximum score in e_s, where E_{seq} is sorted in descending order based on the maximum error of each $e_{seq} \in E_{seq}$. Finally, a vector *d* is computed by applying a percent decrease on e_{max}. The anomalous sequences whose score in *d* exceeds a minimum decrease percentage *p* remain anomalous, and are reclassified as normal otherwise.

2.5 The Literature Gaps

From my literature review of Generic Learning Feature Representations of Normality, I can identify 23 works that support analysis some form of temporality. Based on the characteristics of each framework, I can place the literature into two groups, those that provide *temporal analysis support* (Section 2.5.1) and those that provide *anomaly detection support* (Section 2.5.2).

2.5.1 Temporal Support

The fundamental characteristics to support extraction of temporal information, are illustrated in Table 2.3 along with the corresponding references.

References	PredP	ImpAtt	ExpAtt	DynS
[4, 14, 18, 24, 26, 27, 32, 33, 35] [39, 42, 43, 44, 51, 52, 54, 55]	✗	✓	✗	✗
[41]	✗	✓	✓	✗
[23, 53]	✗	✗	✓	✗
[37, 46]	✓	✗	✗	✓
[20]	✓	✓	✗	✗
DITAN	✓	✓	✓	✓

Table 2.3: The fundamental temporal-based characteristics: Prediction-based Protocol (PredP), Implicit Attention (ImpAtt), Explicit Attention (ExpAtt), and Dynamic Structure (DynS)

Prediction-based Protocol (PredP)

Three (13%) studies support prediction-based protocol to cross-map current and future records, thus allowing definition of an advanced temporal relation in time. Since, the vast majority use the estimation-based protocol to self-map current records.

Implicit and Explicit Attention (ImpAtt, ExpAtt)

Nineteen (83%) studies utilize implicit attention in the form of GRU or LSTM layers. In this approach, LSTM or GRU recurrent layers are used to learn an inference relation type from context to horizon (i.e. *If* $\rightarrow$ *Then*), or an equivalence relation type (i.e. *If* $\Leftrightarrow$ *Then*). However, only three (13%) studies support explicit attention to subjectively weight the cells of each layer.

Dynamic Structure (DynS)

The hyper-parameter values are controlling the generalization of a model over unseen data and thus the forecasting precision. To achieve optimal results, the hyper-parameter values have to be tailored to the domain features,

resulting a model with a dynamic structure. However, only two (9%) studies optimized the hyper-parameter values automatically, since majority optimized them manually. However, is is worth noting that the studies which automated the process of hyper-parameters optimization, they have used a stochastic instead of a more sophisticated heuristic approach.

Sum-up

So, in terms of temporal support, the literature gaps are:

1. memorize the relations of past records to future records
2. explicitly attend to the memorization components
3. adapt memorization capabilities to the domain features

2.5.2 Anomaly Support

The fundamental characteristics of models designed to support anomaly detection and interpretation are given in Table 2.4, along with the relevant studies.

References	RecE	RegE	DynT	AnoP	AnoNI	AnoPI
[18, 24, 27, 32, 33, 39, 41, 51, 54, 55]	OFS	✗	✗	✗	✗	✗
[44]	OFS	✗	✗	✗	✓	✗
[14]	OFS	✗	✗	✓	✗	✗
[35, 43]	OFS	✗	✓	✗	✗	✗
[53]	OFS	✗	✓	✗	✓	✗
[4, 26, 52]	OFS/LFS	✗	✗	✗	✗	✗
[23]	OFS/LFS	✗	✗	✗	✓	✗
[42]	LFS	✗	✗	✗	✗	✗
[37]	✗	OFS	✗	✗	✗	✗
[20]	✗	OFS	✓	✓	✗	✗
[46]	OFS	OFS	✗	✗	✗	✗
DITAN	OFS	OFS	✓	✓	✓	✓

Table 2.4: The fundamental detection and interpretation characteristics: Reconstruction Error (RecE), Regression Error (RegE), Dynamic Threshold (DynT), Anomaly Pruning (AnoP), Anomaly Numerical Interpretation (AnoNI), Anomaly Physical Interpretation (AnoPI)

Reconstruction and Regression Errors (RecE, RegE)

Errors measure irregularity as function of distance between actual and predicted/estimated values in original (OFS) or latent (LFS) feature spaces. The errors associated with predicted values are referred to as *regression errors* whereas the errors related to estimated values are known as *reconstruction errors*. Three (13%) studies used regression errors to learn regularities, while twenty (90%) studies rely on reconstruction errors for this purpose. In addition, one (4%) study incorporates both construction and reconstruction errors to learn regularities.

Dynamic Threshold and Anomaly Pruning (DynT, AnoP)

Four (17%) studies collated in Table 2.4 derive a *dynamic threshold* using robust statistics. These studies provide a reliable analysis of the errors sequence, even if the error values are not drawn from a normal probability distribution. While, the need of a *pruning* methodology to re-align the derived thresholds is only addressed in two (9%) studies.

Anomaly Numerical and Physical Interpretation (AnoNI, ANoPI)

Three (13%) studies utilized *numerical* interpretation in the form of root-cause or similarity. However, none of these approaches provide a *physical* interpretation of the anomaly, so as to relate the meaning of the anomaly to a real-world situation.

Sum-up

So, in terms of Anomaly support, that literature gaps are:

1. learn regularities using both the regression and reconstruction errors
2. define the threshold using robust statistics
3. Interpret the detected anomalies both numerically and physically

2.6 Perspectives from the Gap Analysis

The gap analysis in the Table 2.3 and Table 2.4 reveals a gap in existing research, where different studies support different characteristics leaving a fragmented understanding (see ✗) of the overall picture. These studies primarily emphasize anomaly detection, while partially addressing numerical interpretation and totally lacking a level that allows interpretation of the underlying physical phenomena that derives the anomalies (Figure 1.3). Thus, the approach that I develop here, DITAN, aims at integrating and developing all of the model characteristics reviewed

here, while offering meaningful insights into the physical interpretation by applying a knowledge system and expert-based classification options. The aim, now, is to develop and test an appropriate model, with the aim of delivering a detection and classification of anomalies found by DITAN in the Vulcano data set (Section 1.2).

Chapter 3

Towards Anomaly Detection

The firs task in my work flow is the task; a task which is followed by interpretation (Chapter 4). Following the gap analysis of Chapter 2, I apply a forecasting scenario where both temporal and feature information are used to predict normality, and then to detect abnormality using a dynamic thresholding methodology.

3.1 Pre-processing Strategy

A given *time series* (TS) needs to undergo appropriate pre-processing before it can be analyzed by a deep model. That is, because data are in a raw format, several factors may disrupt the learning capabilities. The pre-processing strategy proposed here as part of DITAN is illustrated in Figure 3.1.

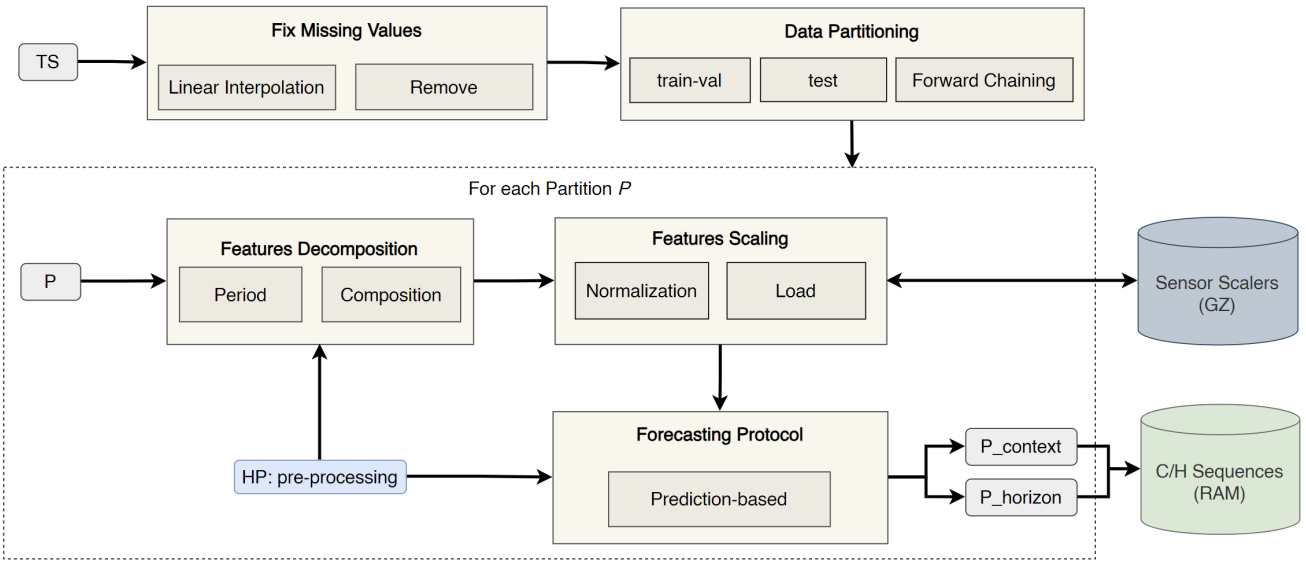


Figure 3.1: A graphical illustration of the pre-processing steps

3.1.1 Fix Missing Values

It is common to encounter missing sensor values across records in a time series. Managing missing values is an essential initial step in the pre-processing phase. For each sensor the consecutive missing values are linearly interpolated in a forward manner. However, this is done only if the gap size does not exceed the size of context (input) window. Alternatively, the record corresponding to the missing sensor values is removed. Although more advanced interpolation methods (e.g. polynomial interpolations) have been introduced, their degree and shape parameters are not easy to configure, and may induce a false sense of a real waveform that does not, in reality, exist.

3.1.2 Data Partitioning

In the training phase, the time series is referred to as *train-val* sequence. *Forward chaining* is then applied on the *train-val* sequence, to create four partitions *P* of *train* and *val* sequences. The size of *train* progressively increases to ensure similar statistical properties between the last *train* and *train-val* sequences. Also, the number of partitions must be sufficient to be a compromise between effectiveness of validation and computational time. In the testing phase, the time series referred to as *test* sequence, and is treated as a single partition denoted as *P*. The partitions are graphically illustrated in Figure 3.2.

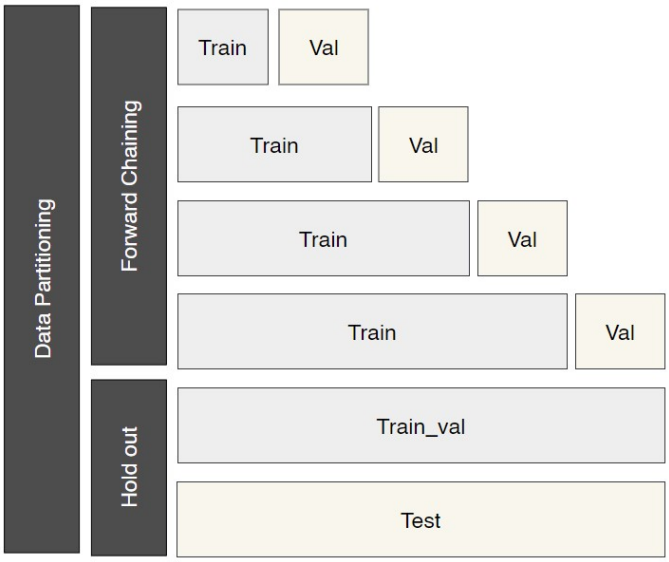


Figure 3.2: A graphical illustration of the data partitioning, where each horizontal line represent a different partition, 6 in total

3.1.3 Feature Decomposition

Within a partition *P*, each sensor is determined by three statistical properties: trend, seasonality and residuals. Short-term (high frequency) anomalies are typically present as residuals, while long-term (low frequency) anomalies

are commonly observed in the trend component. However, if decomposition is applied (see Table 3.1), estimations are required for the *composition type* and *period* of each sensor. The period of a sensor is estimated in three steps: (i) remove the mean signal (a.k.a DC component), (ii) calculate its auto-correlation, and (iii) select the largest valid peak from its second-order difference, where a peak value is valid when it enables at least two cycles. The composition type of a sensor is estimated in two steps: (i) compute the Simple Moving Average (SMA) using a window size equal to 5% of the total sensor values, and (ii) calculate the variances resulting from the subtraction and division between smoothed (SMA) and actual values. The composition type is assumed *multiplicative*, when a sensor exhibits only positive values and its variability of division is higher than that of the subtraction. Otherwise, the composition type is assumed to have an *additive* composition.

3.1.4 Features Scaling

Within a partition *P*, I apply *Min-Max* normalization in the range (-3, 3) to ensure that all sensors contribute equally using scaled values, while maintaining their correlations and preserving their original distributions. The chosen range is sufficient to increase tolerance against distribution shrinking, which may occur due to the presence of extreme values. For each sensor, a scaler is fitted and transformed in the training partitions (*train*, *training-val*), but are only transformed in the testing partitions (*val*, *test*). In the fitting process, the model learns the limit (min, max) values of a sensor, and then uses them in the transformation process. Note that, the fitted scalers of the *train-val* partition are stored for future use on a *test* partition.

3.1.5 Forecasting Protocol

Within each partition *P*, the *prediction-based* forecasting protocol (see Section 2.4) is used to introduce a temporal inference between preceding and succeeding records. This formulates a self-supervised environment of *context* and *horizon* mapping. Choosing the appropriate size of a context window is a challenging task. A large context size may overlook short-term patterns and lead to increased computation cost. On the other hand, a small context size may fail to capture long-term patterns adequately. Although the size of a *horizon* window is recommended to be smaller than the size of the *context* window, striking the right balance is domain-specific and thus configurable (see Table 3.1). Note that the sensors involved in the context window can differ, if needed by the application domain, from the sensors involved in the horizon window. The union of all the involved sensors is referred to as *active* sensors.

3.2 Modeling Normality

In Figure 3.3 I illustrate the model's Encoder-Decoder architecture. It is composed of Long Short-Term Memory (LSTM) layers along with a composite decoder and soft attention mechanisms (see Section 3.2.1), to capture short-

term and long-term normal (regular) patterns. The number of LSTM cells in the encoder (left) is equal to the context size, while the decoder (right) has a context sized number of LSTM cells to reconstruct, as well as a horizon sized number of LSTM cells to construct. Such a composite decoder forces the network to stay attentive to all time steps in the encoder, instead of only the last few steps [46]. Intuitively, this forces the model to understand the *question*, while finding the *answer* to be predicted. The LSTM cells of the encoder are regularized on both the vertical and the horizontal axes (see Section 3.2.2), to introduce generalization on the memorization process.

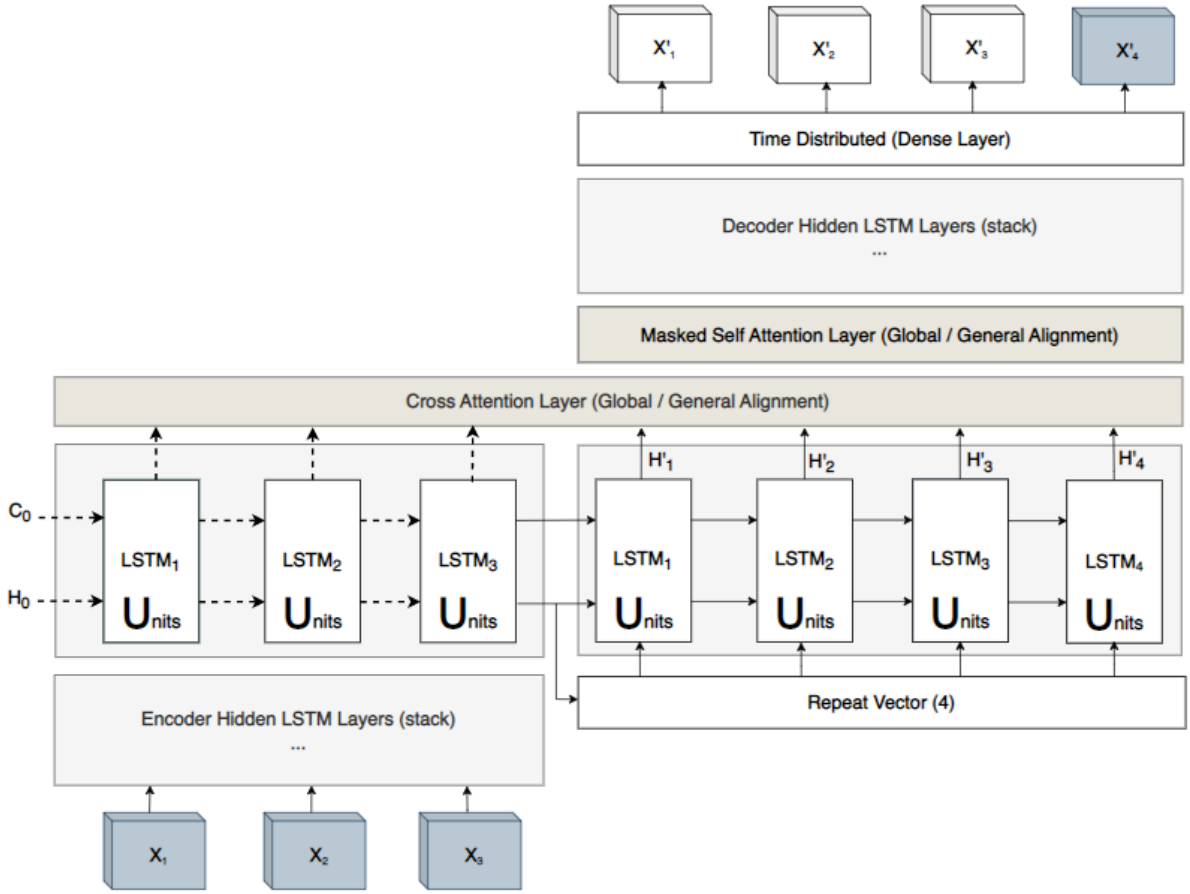


Figure 3.3: A graphical illustration of the composite Encoder-Decoder with attention using context of size 3 and horizon 1. Solid lines represent information flow while dotted lines regularization on the features (horizontal) and temporal (vertical) axes

3.2.1 Memorization Components

Short-term Memory

To memorize patterns with respect to time, the model uses long short-term memory *LSTM* cells [17]. An LSTM cell uses a gating mechanism that controls the memorizing process at each time step (Figure 3.4). Here, let f^t (respectively i^t; g^t, o^t) be the forget (resp. input, input modulation, output) gate, C^t the cell state and h_t the output state at time t. Given these definitions, a LSTM cell performs the following operations:

$$
\begin{aligned}
g^t &= \sigma\left({W_C}^\top [x_t, h_{t-1}] + b_C\right) \\
i^t &= \sigma\left({W_i}^\top [x_t, h_{t-1}] + b_i\right) \\
f^t &= \sigma\left({W_f}^\top [x_t, h_{t-1}] + b_f\right) \\
o^t &= \sigma\left({W_o}^\top [x_t, h_{t-1}] + b_o\right) \\
C^t &= g^t.i^t + C^{t-1}.f^t \\
h_t &= o^t tanh(C^t)
\end{aligned}
$$

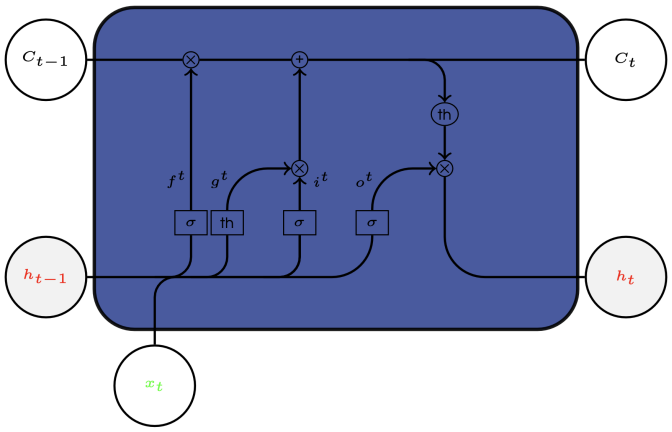


Figure 3.4: A LSTM cell illustrating the gating mechanism

The gates store information in an analog format, implementing an element-wise multiplication by a sigmoid, ranging between 0 and 1. The forget gate determines which information from prior steps is considered relevant and should be retained. The input gate decides what relevant information can be added from the current step, and the output gate finalizes the next hidden state. The hidden state is a *u* length vector, representing a prediction output, where *u* is a configurable (see Table 3.1) unit space into which a *d* dimension record is transformed. The cell state, also known as short-term memory, is thus a vector of length *u + d*.

Long-term Memory

LSTM cells can then be connected, creating a *LSTM layer*. The number of LSTM cells in a layer is equal to the number of input time steps. These LSTM cells are connected to each other, sequentially, within their hidden and cell states. The interconnected cell states represent the *long-term* memory, also known as *implicit attention*, while hidden states are connected to progressively construct predictions. A LSTM layer is inherently deep in time, because each time step can be seen as a change over time. LSTM layers can also be connected upwards, to construct a *stack* of configurable number of layers (see Table 3.1), also known as a LSTM *network*. Since LSTM layers operate

on sequential data, the additional layers recombine the learned representation from prior layers with respect to time. The goal is to create a more abstract feature representation using the same or different (units) dimensionality. LSTM layers are stacked in a way that each layer takes as input all the hidden states of its previous layer. However, stacking layers is not always an advantage over a single LSTM layer. It depends on the specificity of the problem and the relationships being modeled.

Explicit Attention

A large number of time steps may lead to fading memory [5]. To protect the architecture, I employ two Luong-based [31] attention layers, to explicitly control the weights of the units across time steps.

The first attention layer, in Figure 3.3, is a *cross* attention between encoder and decoder, which maps the important and relevant hidden states from the encoder to the hidden states in the decoder, and assigns higher weights to them. In this configuration, all the hidden states of the encoder's last layer are considered while calculating *attention weights* for each hidden state in the decoder.

The second attention layer, in Figure 3.3, is a *masked-self* attention over the hidden states in the decoder's first layer, and is used to examine their causal relation with respect to their temporal order. In this configuration, all the prior hidden states of the decoder are considered while calculating *attention weights* for each following time step.

Both attention layers use *global* attention with a general alignment scoring function. Attention weights define a probability distribution over the encoder (cross) or decoder (self-masked) states, to compose a context vector defined by:

$$
\begin{aligned}
c_t &= \sum_s a_{ts} h_s && \text{[Context vector]} \\
\forall s \; a_{ts} &= \frac{exp\left(score(h_t, h_s)\right)}{\sum_{s'} exp\left(score(h_t, h_{s'})\right)} && \text{[Attention weights]} \\
score(h_t, h_s) &= h_t^T W h_s && \text{[Luong's (general) multiplicative style]} \\
h_t^a &= tanh\left(W_c[c_t; h_t]\right) && \text{[attentional hidden state]}
\end{aligned}
$$

where h_s (and h_t) are the source (and target) hidden states of the encoder's last layer (cross) or decoder's first layer (self-masked), and W_c, W are trainable weights. An attentional hidden state h_t^a is then computed to concatenate the information of h_t and c_t. Note that the use of the exponential function in the computation of a_{ts} ensures positive attentions weights, while the softmax allows these weights to sum up to 1.

3.2.2 Generalization Components

The ability of a model to generalize is central to its success. In order to avoid overfitting, I propose an immune system to the architecture, by applying two regularization techniques over the feature and time dimensions, as well as a learning strategy.

Output Dropout

An ensemble of sub-networks is built, by applying *dropout* vertically on the output (hidden state) units of the LSTM cells. During the training process, output units are randomly and uniformly excluded from weight updating (set to zero) with a configurable (see Table 3.1) probability. The number of sub-networks corresponds to the number of (epochs) iterations over training data, since dropout refreshes on every epoch. However, dropping too many units may over regularize sub-networks. For this reason, dropout is applied only on the LSTM layer(s) in the encoder, regulating the generalization of the latent representation. For each epoch, the concept is to construct decoder LSTM cells using a different subset of the memorized units in encoder cells. Although there are horizontal dropout variations (e.g. recurrent dropout), I do not recommend dropping memory units arbitrary, because there is already a gating mechanism to maintain memory, and memory may be damaged through random temporal disconnections.

Elastic Regularization

The model imposes L1 and L2 norm constraints [57], horizontally on recurrent units within each LSTM layer in the encoder. This is known to have the effect of reducing overfitting and improving performance [29], without necessarily drooping units. During the training process, a regularization term is used in the form $\lambda(\|.\|_1 + \|.\|_2)$, where *strength* λ is configurable (see Table 3.1) and common to both L1 and L2 which are linearly combined. L1 forces the network to drop recurrent weights that do not significantly contribute to the predictive power. On the other hand, L2 forces the model to apply relatively small weights, while maintaining weights with strong pairwise correlation coefficients.

Learning Scheduler

The *learning rate* (*lr*) controls how quickly the model adapts to the problem, and therefore highly affects model stability. A model trained using small *lr* requires more epochs than a model with a high *lr*, since smaller changes in the weights require more epochs for convergence. However, when learning rate is too large, the model may converge too quickly to a sub-optimal solution and is prone to oscillations. The *lr* value, as well as the rate of change known as *scheduler*, are configurable (see Table 3.1). Here, the *scheduler* is responsible for either keeping the *lr* value constant or for decreasing *lr* linearly by 25% at every fourth epoch or to decrease *lr* exponentially by 10% at every epoch.

Learning Patience

A major challenge in training neural networks is setting the number of epochs. The objective is to train the neural network sufficiently that it learns the mapping, but not so much that the network overfits and not so little as to underfit. Here, I use the *early stopping* strategy to find an optimal epoch at which to stop the learning process and finalize the model weights. It uses a criterion, which gives a penalty as long as the difference δ of current *c* from previous *p* loss is less than 0.0003. A penalty is formally given when the condition $loss_c > loss_p - \delta$ is satisfied. The number of consecutive penalties that has to be given to end the training process, is configurable (see Table 3.1) and is referred to as *patience*. The performance of the model is monitored on the last 20% overlapping partition of the *train* sequence. This partition is called *internal validation* and the loss used for monitoring is called *internal* loss. The final model weights correspond to the last epoch, because updates are completed in a sequential manner.

Learning Period

The model weights in LSTM layers are updated using batch learning with backpropagation through time. The size of a batch is the amount of (information) context-horizon mappings used in a single update, as illustrated in Figure 3.5. The batch size affects the reset period of memory, since in our stateless LSTM layers, the *cell state* is cleared for every batch. Therefore, batch size impacts the stability as well as the duration of the learning process. The majority of studies in Chapter 2 have used 32 as a standard size, however an arbitrary value may produce an erratic definition of normality. It is known that small batch size slows down the learning process while increasing it produces a lack of change in data over epochs and higher memory cost. Here, the batch size is thus configurable (see Table 3.1), and batches maintain their position across epochs, not shuffled, to additionally learn dependencies between consecutive batches.

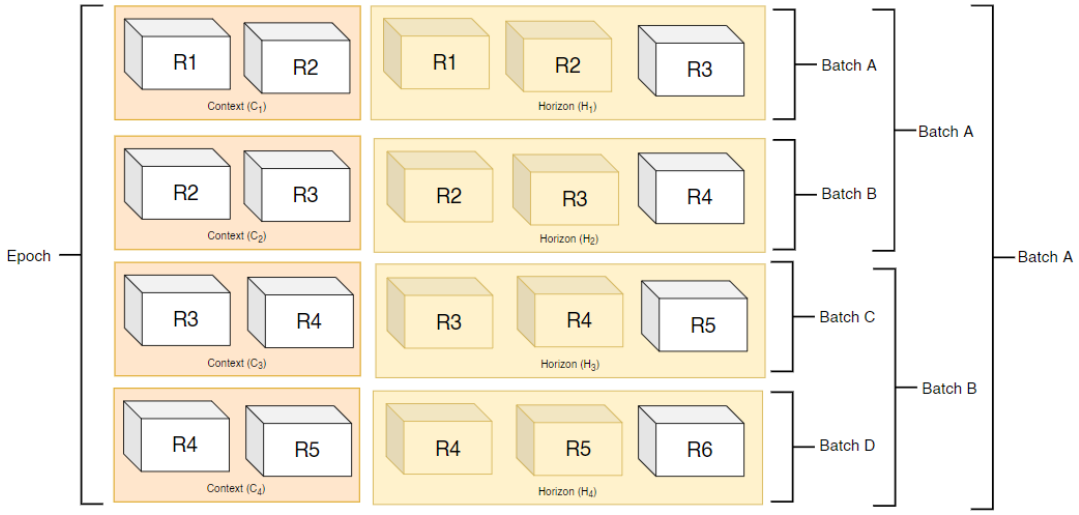


Figure 3.5: Different batch size configurations per epoch, over a partition of four context-horizon mappings

3.3 Dynamic Threshold with built-in Pruning

A model uses its latent representation of normality to predict the value *x* on each sensor *d* of a record *i*. The error e_d^i is then computed as the difference of what is predicted from what is actually observed. To compute the difference DITAN uses a configurable (see Table 3.1) loss function. Therefore e_d^i can either be defined as the squared or absolute error:

$$e_d^i = ||x_d^i - x_d^{i'}||_2 \quad or \quad e_d^i = |x_d^i - x_d^{i'}|, \quad \forall d \in D \tag{3.1}$$

The squared, as opposed to absolute, errors are non-linear weighted with respect to the error magnitude, and thus higher errors are magnified. The higher the error on a sensor value, the more abnormal the value is considered. Yet, a threshold is required to determine the turning point from normal to abnormal, where one threshold is required for each sensor.

In this approach, I start by identifying *critical peaks* within the testing error sequence E for each sensor *d*. Critical peaks are error values corresponding to infrequent local (or global) maxima values. Each critical peak is expanded into *critical region* using non-parametric statistical tests, considering the local error morphology. Finally, their boundaries are refined using non-parametric clustering. The error values belonging to critical regions are classified as anomalous, while the remaining error values are classified as normal. The entire process, is graphically illustrated in Figure 3.6.

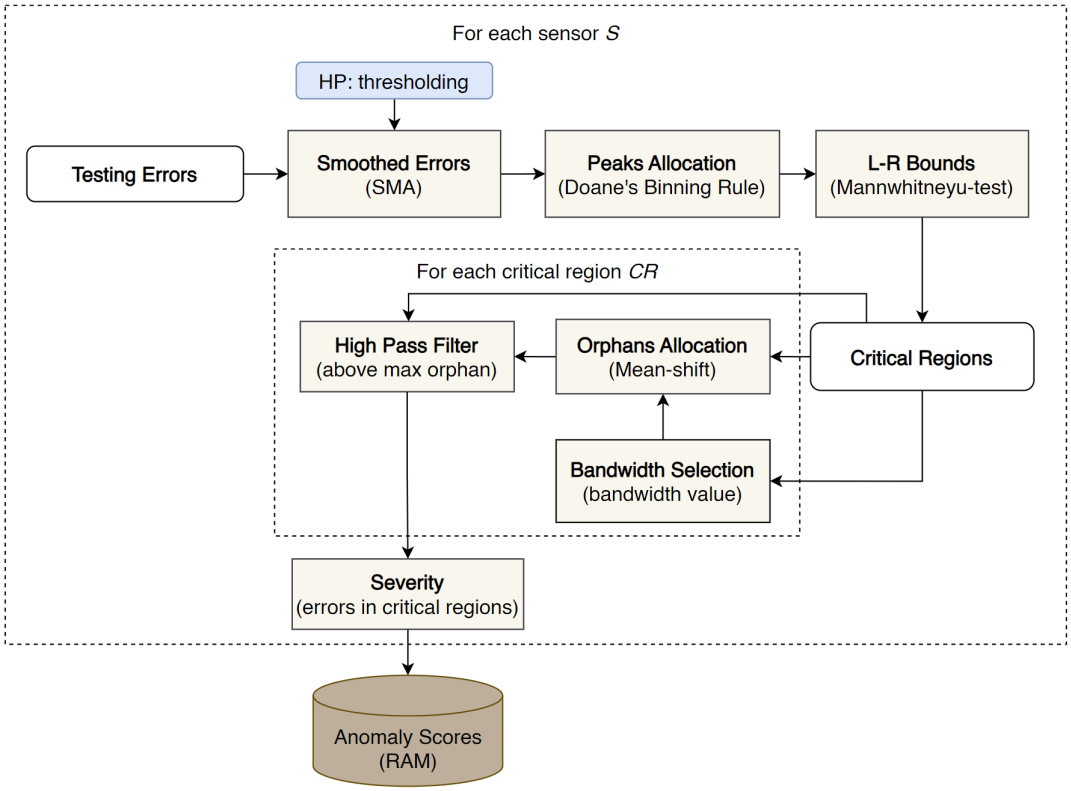


Figure 3.6: The dynamic thresholding process

3.3.1 Critical Peaks

I apply simple moving average (SMA) to the raw errors of an E_d, using Parzen windows of configurable *locality* size (see Table 3.1) to assess temporal weights of a desired locality, formulating Gaussian-like bumps. The *peaks* are defined as error values that exhibit a higher magnitude than both the preceding and succeeding errors. However, *critical* are only the peak values that reach a significantly high value. To identify critical peaks, I assume that higher error magnitudes tend to be more isolated (rarer) in the frequency space. Therefore, I use Doane's rule [68] to discretize all errors into an optimal number of bins b_{opt}: if n_e is the number of errors, then

$$b_{opt} = 1 + log_2(n_e) + log_2\left(1 + \frac{|m_3|}{\sigma_{m_3}}\right)$$

where m_3 is the estimated skewness of the error distribution and $\sigma_{m_3} = \sqrt{\frac{6(n_e-2)}{(n_e+1)(n_e+3)}}$. Then, starting from the bin with mode frequency, a pointer moves downwards across bins as long as the frequency of bin at position *k+1* or *k+2* is lower than in *k* and their frequency is at least $\sqrt[4]{|E_d|}$. The terminal bin is then used as the minimum peak height, above which *peaks* are considered *critical peaks*.

3.3.2 Critical Regions

Each critical peak is symmetrically expanded by using two consecutive values on each side of the peak, left and right, as long as the average error within the defined area is significantly greater than the overall average $mean(E_d)$. The significance is evaluated using a Mann and Whitney [34] test with a significance level 0.05. This approach highlights *critical regions* (cr) of dynamic duration. Note that the overlapping *cr* are merged by combining their respective error values using the union operation.

3.3.3 Robust Pruning

Following [14, 20], I further explore critical regions to refine their boundaries, by employing non-parametric clustering. Specifically, the mean-shift [66] algorithm is applied to each critical region, to identify rare error values in the form of *orphans*. These are indicative of gaps in the provided bandwidth, as they are located far from constructed clusters. The maximum among these orphans, is used to remove lower errors from both the left and right boundaries of any critical region. The removed error values are re-classified as normal.

Bandwidth Selection

Mean-shift is sensitive to the bandwidth of the underlying Gaussian kernel, and thus using a random value for bandwidth is not recommended. Thus, the *Bandwidth Selection* module is implemented using the dual-annealing methodology as illustrated in Figure 3.7. In this approach a bandwidth value is searched for in both local and global

space, where the use of local search and the number of global jumps are fixed hyper-parameters (see Section 3.4). Bandwidth can be seen as the kernel radius and thus candidate values are bounded by distances in the range $[min, max](nearest\text{-}dist(losses_{cr}))$. The objective is to select a bandwidth which results in compact (low standard deviation) and heavy (large size) clusters. This is formally achieved by minimizing the maximum standard error (SE) across clusters (C):

$$bandwidth_{cr}(C) = Argmin\{max(SE(c), \forall\ c \in C)\}_{cr} \tag{3.2}$$

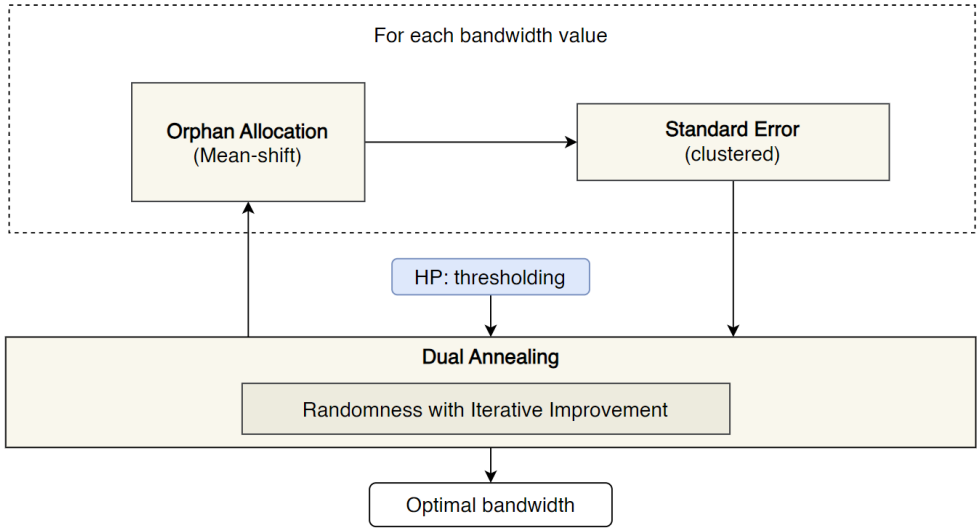


Figure 3.7: The *bandwidth selection* module implemented using dual-annealing

3.3.4 Severity Score

Anomaly refers to error values that belong to a critical region. These error values are used as *severity scores* for their corresponding sensor values, while the severity scores for the remaining sensor values are set to zero. A record is classified as anomalous if it contains at least one sensor with a positive severity score. On the other hand, records without a positive severity scores are considered normal. Hence, an arbitrary length of consecutive anomalous records is considered as a *subsequence* anomaly, while the individual anomalous records are *point* anomalies.

3.4 Hyper-parameter Configurations

The hyper-parameters (HP) of the DITAN are defined in Table 3.1. Five hyper-parameters are manually tuned as they are directly related to the application itself. On the other hand, nine hyper-parameters are automatically tuned due to their complex technical correlations. There are also five additional hyper-parameters, the values of which are fixed, not subjected to tuning.

Hyper-parameter	Value Range	Type	Tuned
Layers	[1, 3]	Compile	Automatically
Units	[32, 128]	Compile	Automatically
Units Decay	[0.5, 1.0]	Compile	Automatically
Dropout	[0.0, 0.3]	Compile	Automatically
Regularization Strength	{0.0; 0.0001; 0.001}	Compile	Automatically
Learning Rate	{0.001; 0.01}	Fit	Automatically
Learning Scheduler	{constant; step; exponential}	Fit	Automatically
Learning Patience	[2, 10]	Fit	Automatically
Batch Size	{32; 64; 128; 256}	Fit	Automatically
Loss Function	{MSE, MAE}	Compile/Evaluation	Manually
Stationarity	{residual; trend; no}	Pre-processing	Manually
Context Size	*N*	General	Manually
Horizon Size	*N*	General	Manually
Locality	{short; medium; long}	Thresholding	Manually

Table 3.1: A summary of the DITAN hyper-parameters

Automatically Optimized Hyper-parameters

To optimize the hyper-parameters automatically, a methodology based on *Bayesian Optimization* (BO) [65] is used, as illustrated in Figure 3.8. As opposed to a random (or grid) search, the Bayesian approach keeps track of the past evaluation results to form a surrogate function that maps hyper-parameters to probability values. In each run, it proposes a new configuration of hyper-parameters, that maximizes the expected improvement of the surrogate function. Then, the new configuration is handed to the objective function to retrieve an evaluation score. The surrogate function is updated using Bayes theorem, incorporating feedback from the objective function.

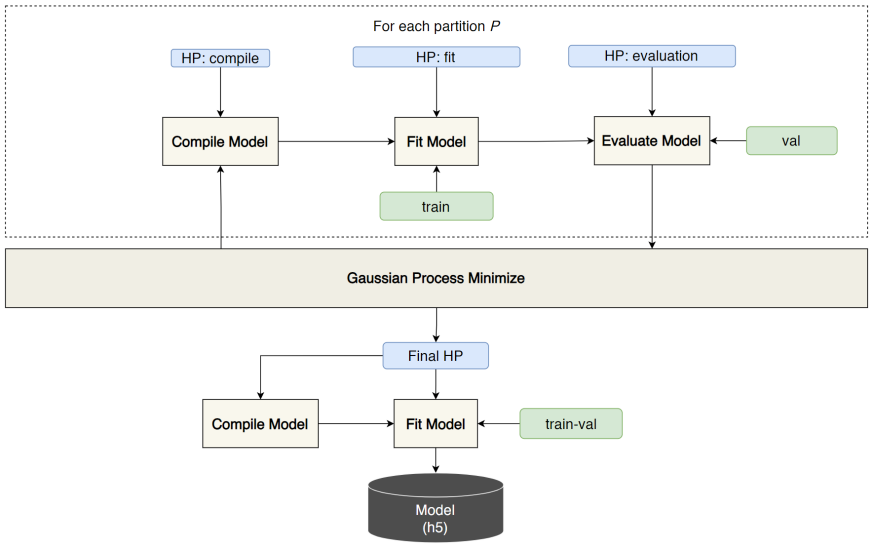


Figure 3.8: Hyper-parameters optimization using the *value range* in which bounded. The *type* refers to its use in the DITAN, and the *tuned* refers to the method responsible to select a value from the *value range*.

In each run, a new configuration *C* is compiled and executed partition times, fitted to the *train* sequence and evaluated by the *val* sequence of each partition *P*. In the evaluation it is considered only the error value of constructing the first record in the regression part of the horizon. Error values are computed using the same *loss function* (see Table 3.1) in both the evaluation and compile processes. Since the *train* sequence expands over partitions, the later the *train* partition the more available data there is to fit and thus the more reliable the evaluation. To this end, the configuration error C_E is computed as the weighted average of the average evaluation error E_{val} per partition *P*, weighted by the cardinal (size) of their corresponding *train* sequences:

$$C_E = AVG_P(E_{val} * |train|)$$

The optimal configuration, denoted as C_{opt}, is the configuration that exhibits the minimum configuration error:

$$C_{opt} = Argmin_C(C_E)$$

The optimal configuration is finally used to compile the model and fit on the entire *train-val* sequence. The trained model is saved for the testing phase.

Manually Selected Hyper-parameters

There are five application-based hyper-parameters that require manual selection. The *loss function* used to compute errors is the squared (MSE) or absolute (MAE) difference of the predicted from observed records (with MSE being the default). The selected option is used in both the compile and evaluation steps to keep consistency across different stages of the anomaly detection process. MSE is recommended when there is no significant number of extreme values in the training sequence, otherwise MAE is preferred to mitigate the influence of extreme values. Additionally, The size of the *context* is crucial and closely tied to domain knowledge since it controls the temporal resolution of the analysis. To determine the context size is important to consider factors such as measurement frequency, cycles, and temporal resolution of events. Similarly, the size of *horizon* is important, although value of 1 (default) typically works for most scenarios. Furthermore, *locality* pertains to the size of the Parzen windows employed in SMA to smooth the sequence of testing errors. There are three options: *short*, *medium* and *long* (with *long* being the default). The short option focuses on short-term variations in errors, while the medium and long options emphasize medium and long-term variations respectively. When horizon size is 1, the *medium* option aligns with the context size; otherwise, it aligns with the horizon size. The *long* option is twice the size, and *short* option half the size, of the *medium* option. Finally, *stationarity* determines whether the time series need to be transformed into a stationary form (default is *no*). This can be useful when looking for high frequency abnormalities.

Fixed-value Hyper-parameters

There are five hyper-parameters with fixed values. The first three hyper-parameters refer to the training phase, and the last two hyper-parameters to the dynamic threshold methodology. These are:

1. *Optimization runs* refer to the number of different configurations the Bayes optimizer examines. By default, this value is set to 20. However, it can be set to zero if an explicit hyper-parameters configuration is preferred
2. *Nadam* is used as the *model optimizer* to optimize the model's parameters
3. *tanh* is used as the *activation function* to introduce non-linearity across LSTM layers
4. *Global runs* (default is *40*) indicates the number of jumps to different global points needed to search for an optimal solution
5. *Local search* (default is *True*) indicates the need for a deeper consideration of local space or not.

3.5 DITAN Detection System: improvements to the Beta version

Two minor updates were made to the initial thresholding methodology. The first primarily emphasizes enhancement of the precision of expanding the boundaries of critical peaks, and the second aims to improve the efficiency of the refinement process while ensuring its effectiveness.

3.5.1 Critical Regions: using non-centralized peak

Instead of symmetric expansion for each critical peak (see Section 3.3.2), it is better to expand the left and right sides independently. In this approach critical regions in which one side is heavier than the other may occur, introducing scenarios where the most intense moments are closer to the start or end of an event, rather than closer to its center. To achieve this, for each critical peak the left or right side is expanded by 2 error values towards the left or right, as long as the average of its last (up to 64) consecutive errors remains significantly greater than the average of all errors in sequence. Note that the significance is evaluated in the same way using Mann and Whitney test with a significance level of 0.05.

3.5.2 Bandwidth Selection: using Silverman's rule

Instead of using the dual annealing methodology to implement the *bandwidth selection* module (see Section 3.3.3), I propose, instead, to use *Silverman's rule* (equation 3.3). This rule is applied directly to the vector of absolute distances between subsequent error values within a critical region, denoted as $dist_{cr}$, offering a more efficient

approach: it indeed only requires one execution instead of time-consuming iterations, and removes the hyper-parameters *Global Runs* and *Local Search* which are unnecessary if this approach is applied:

$$bandwidth_{cr} = 1.06 * Min(Std(dist_{cr}), \frac{IQR(dist_{cr})}{1.34}) * |dist_{cr}|^{-\frac{1}{5}} \tag{3.3}$$

Chapter 4

Towards Anomaly Interpretation

The DITAN framework deals with the interpretation of detected anomalies in two ways: numerically and physically. While DITAN's numerical-based interpretation based on [9, 23, 44, 53, 69], supporting physical interpretation involves using experts-knowledge as based on [62, 77] which consider the domain of knowledge systems.

4.1 Numerical Interpretation

The magnitude of scores across sensors within anomalous records can be used to provide sufficient information to allow an understanding and troubleshooting of anomalies. This involves both the analysis of anomalies in both feature (data) space and units (model) space.

4.1.1 Root cause in Feature space

Following [9, 23, 44, 53], the root-cause of an anomalous record *i* for any sensor *d* is computed by applying softmax to its score *s* values.

$$RootCause_d(s^i) = exp(s^i_d)/sum(exp(s^i)) \tag{4.1}$$

Note that, the root-cause values within a normal record are set to zero.

4.1.2 Similarities in Unit space

DITAN also characterizes similarities across anomalous records. Our analysis focuses on the unit space, where each anomaly is transformed into an internal representation. The internal representation *r* of an anomalous record *i* is estimated with

$$r^i = f_D(E(c^i))$$

where *f* is the last intermediate (LSTM) layer in decoder *D*, and *E* is the encoded representation of the context window *c*.

Anomalous records are then clustered using a Gaussian Mixture Model (GMM) across their internal representations $r \in R$. The probability *p* of an anomalous record *i* being generated by each component (cluster) C_j, is assessed using mixture weights ϕ_j, the cluster center μ_j and a tied covariance matrix Σ_R:

$$p(r_i) = \sum_{j=1}^{C} \phi_j \mathcal{N}(r_i|\mu_j, \Sigma_R)$$

The covariance matrix Σ_R is nevertheless not always invertible, e.g. when the number of units is higher than the number of anomalous records. Hence, DITAN uses a sparse inverse covariance estimation technique based on the Graphical Lasso [13] estimator.

The optimal number of components *C* is determined as a value in the range $[2, min(10, |R|)]$, that exhibits the maximum percentage difference across their Davies-Bouldin Index *DBI* [10] scores. The *DBI* evaluates the average distances between each cluster and the clusters most similar to it. These distances are computed using a Mahalanobis approach instead of a Euclidean approach, so as to consider possible correlations in *R* and allowing any cluster shape to be handled, for example elliptical rather than circular only. This is because in the Mahalanobis approach the assessment of distance between r_i and μ_j takes into account the inverse covariance Σ_R^{-1}. Hence, Mahalanobis approach is considered an efficient metric for a large number of records because distances in this approach do not require pairwise operations over records, only over centroids.

4.2 Physical Interpretation

Detected anomalies typically require further investigation to understand their underlying physical meaning (e.g. environmental factors, Figure 1.3). This examination calls for specialized knowledge to accurately identify the nature of these anomalies. Experts are recognized for their aptitude on providing domain-specific knowledge, but asking them to manually characterize all detected anomalies can be a time-consuming and intensive process. However, to automate this process knowledge system functionalities can be leveraged.

Figure 4.1 provides an overview of DITAN's knowledge system. Following [62, 77], all functionalities are accessible via a *web-based GUI*, enabling real-time interaction with experts to allow *knowledge management* and coupling with *inference engine* modules. To assist in meta-analysis of anomalies, the two numerical interpretations are used. The *root-causes* (Section 4.1.1) are used to assist experts in knowledge formation by graphically visualizing the critical regions that overlap on both the time and feature axes. *Similarities* (Section 4.1.2) is then used to graphically

preview the anomalies, by assigning the same color to all anomalies falling within the same cluster.

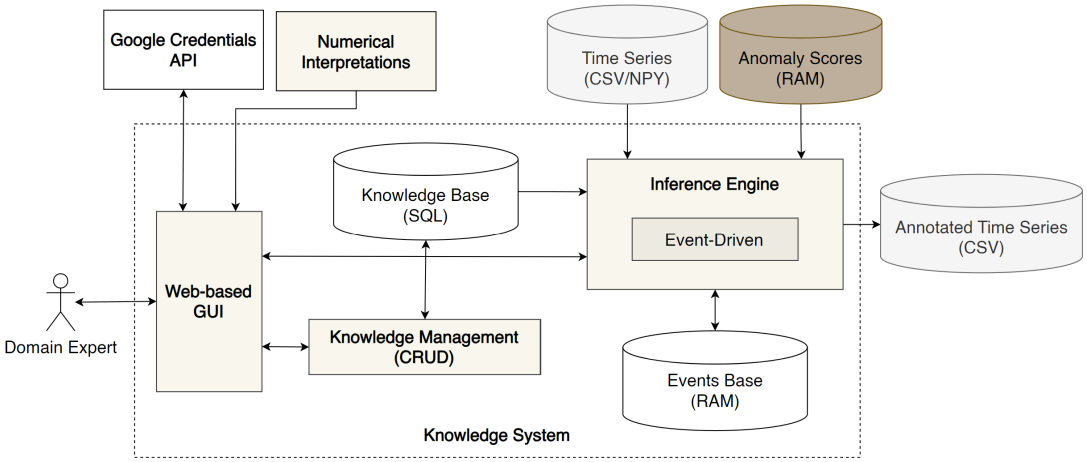


Figure 4.1: Overview of proposed knowledge system to characterize physical anomalies

4.2.1 Knowledge Management

The knowledge management module provides experts with functionalities to create, read, update and delete (CRUD) knowledge held in the knowledge base *KB* (see Appendix A). Experts have complete control over knowledge in the form of IF-THEN rules that incorporate temporal constraints, also known as *temporal* rules. The graphical environment to create (or update) a new (or existing) rule is illustrated in Figure 4.2. The left side corresponds to the *preconditions* of a rule and the right side to its *post-condition*. Additional information is provided, namely *description*, *license* and *version*, while the status (radio button) indicates when rules are executable or draft, with only executable rules being used. The scope of a rule in our system is to characterize:

"*The physical event responsible for the occurrence of a series of conditions*".

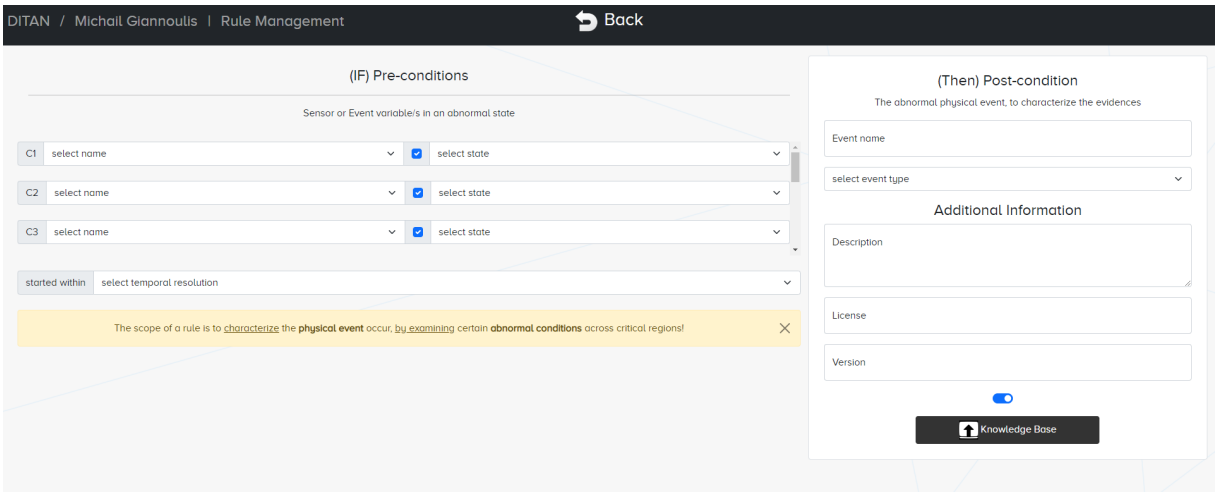


Figure 4.2: Create / Update a Rule, Web Graphical Environment

Conditions

Condition *C* specifies the name of a sensor *S* or physical event *E* that is experiencing an abnormal state. I provide a list of five abnormal states *increasing*, *decreasing*, *positive*, *tare* or *missing values*. Although an *S* condition can indicate any of these abnormal states, an *E* condition can only have a *positive* abnormal state, meaning that the event has occurred. Therefore, post-condition is always an *E* condition, while the preconditions consist of one or more *S*/*E* conditions.

Constraints

Following [62, 77], the preconditions of a rule are constrained using both logical and temporal operators. In particular, the conjunction (AND) logical operator is used to combine conditions, while the negation (NOT) logical operator is an option that can be used to the abnormal state of a condition. The conditions are temporally constrained such that the start time of the first condition and the start time of the last condition are at most 60 minutes, 24 hours or 7 days apart, introducing lag resolutions of *minutes*, *hours* or *days* across rules. These constraints ensure that all preconditions occur within a defined time interval. The post-condition of a rule exists from the start time of the first condition to the end time of the last condition.

4.2.2 Inference Engine

Following [62], the preconditions of rules guide the inference process by incorporating an event-driven protocol. In particular, experts are provided by an inference engine to automatically identify which of their executable rules apply to the critical regions of the DITAN model. DITAN's inference engine is given in Algorithm 1.

Inference Process

Algorithm 1 is handed the executable rules from the knowledge base (KB) and the anomalous sensor values from critical regions (CR). The objective is to report all post-conditions in the events base (EB), by identifying valid preconditions. This requires support by *rule-chains* in which the post-condition of a rule lies in the preconditions of another. Thus, in an outer-loop, *N* executable rules with *e* events in preconditions are selected from the knowledge base *KB*, with *e* being increased iteratively until *N* becomes zero. At each iteration, a function namely *FetchRules*, is responsible for selecting and ordering rules with respect to its references in *rule-chains*. Rules referenced the most frequently are executed first. Given a rule *R*, *RuleExecute* is responsible for validating the occurrences of its conditions across *CR* (sensor type) or *EB* (event type). The preconditions of a rule are valid when both temporal and logical constraints are satisfied. The occurrences of a post-condition can also overlap. In such a case, overlapping occurrences are merged into their union. It is important to note that merging is applied at the end of the algorithm, to preserve different event start times, which is useful in the chaining process. The final occurrences of *R* are stored

in *EB*.

Algorithm 1 Inference Engine

Require: KB, CR ▷ Knowledge Base (KB), Critical Regions (CR)
Ensure: EB ▷ Events Base (EB)
  $e \leftarrow 0$ ▷ number of events allowed in rule's precondition
  while $True$ **do**
    $rules \leftarrow FetchRules(KB, e)$ ▷ Fetch rules ordered by the number of events
    if $len(rules) == 0$ **then** ▷ Halt if there are no more rules
      $break$
    end if
    for $R \in rules$ **do**
      $occurences \leftarrow RuleExecute(CR, EB, R)$ ▷ find all occurrences in which the rule is validated
      if $occurences \neq []$ **then**
        $EB.update([R : occurences])$ ▷ Keep only the executed rules
      end if
    end for
    $e+ = 1$ ▷ increase number of events for the next iteration
  end while

Condition Validation

A condition occurs when its abnormal state is found to be valid for a certain duration. A condition can be validated multiple times across *CR* (for sensor type conditions) or *EB* (for event type conditions). Particularly, in a given sequence the validation of abnormal states is assessed as it follows,

- *increasing*: a subsequence of at least three consecutive values exhibiting only an increasing trajectory.
- *decreasing*: a subsequence of at least three consecutive values exhibiting only a decreasing trajectory.
- *positive*: a subsequence of consecutive values exhibiting only positive values.
- *tare*: the first order difference, at different steps, in the entire sequence result in the same maximum value.
- *missing value*: there is at least one missing time-step in the entire sequence.

Note that, when the negation operator is defined the corresponding abnormal state is valid for complementary sub-sequences.

Rule Validation

A rule is valid when its preconditions are satisfied. That means logical and temporal constraints are valid. Validation of the logical constraints (negation, conjunction) is straight-forward, because each of constraint is a well-defined operator. Instead, further analysis is required for handling temporal relations. Once all valid occurrences of the conditions have been examined, the next step is to efficiently analyze their temporal differences. To accomplish

this, DITAN constructs an upper triangular matrix containing all possible pairs of occurrences for all conditions. For instance, if there are four conditions (C1, C2, C3, C4) it will be six pair categories (C1-C2, C1-C3, C1-C4, C2-C3, C2-C4, C3-C4), and the upper triangular matrix allows consideration of each unique pair of conditions without redundancy. Next, DITAN constructs chains between the occurrences of different pairs so as to derive the ultimate temporal relations. A chain is deemed valid if it incorporates all the different pair categories. A pair is added to the chain if it does not violate the temporal constraint. Therefore, a rule linked to the occurrences of its valid chains (if multiple conditions apply) or its valid condition (if there is only a single condition).

Risk Factor

In the domain of Artificial Intelligence (AI), the risk of a (valid) rule is related to the intensity of its preconditions. This is usually quantified by experts using fuzzy logic [78] or certainty factors [79], such as in [62]. However, DITAN quantifies risk by using the severity scores (Section 3.3.4). The risk associated with an *S* condition is equal to the average of the severity scores within the partition (duration) *j* of a critical region:

$$RF_S = average(CR^j_{scores})$$

and the risk of an event *E* condition is equal to the average risk of its preconditions:

$$RF_E = average(RF_{S1}, ..., RF_{Ek})$$

As a result, the risk factor *RF* provides a general overview of the predictability offset across conditions within their validated durations. The maximum risk is selected when an *E* condition occurs multiple times within overlapping durations. Finally, to preserve a relative risk across valid rules, within a probabilistic-like range, max-normalization is applied.

4.2.3 The DITAN knowledge System

The proposed knowledge system of DITAN, exhibit some similarities with [62, 77]. These are (a) the use of web-based CRUD functionalities in knowledge formation, (b) temporality in rules is related to the duration over which conditions occur, (c) there is an event-driven protocol. However, there are also some crucial differences with previous approaches, these being (a) inference operates on explicit regions of the data that are considered critical by the model, referred to as *critical regions*, instead of operating everywhere and (b) risk factors are driven by the model instead of the experts belief.

Chapter 5

Experimental Evaluation

In this Chapter, I assess the effectiveness of DITAN in predicting normality, detecting and interpreting anomalies numerically. To do this, I use a total of seven datasets with varying anomaly types. Six datasets are chosen from the experiments held in [20], and one dataset is selected from [58]. This exercise allows a comparable environment to evaluate the DITAN's performance through comparison with other models, run on the same data. All runs are executed on the Python libraries given in Appendix B, using two Nvidia TESLA v100 graphics card of 32 GB each. The entire evaluation took approximately 8 hours to execute

5.1 Dataset Profiling

The datasets used to test DITAN are already pre-processed by [20] and [58]. As a result, these datasets do not have missing values and are already partitioned into *train-val*, *test* sequences. The time series are not decomposed into a stationarity data set, and have already been scaled using the min-max transformation. All of these steps could be part of DITAN's automatic pre-processing (Section 3.1) strategy. However, to be able to apply the hyper-parameter optimization, the *train-val* is additionally partitioned using *forward chaining*.

5.1.1 NASA Spacecraft Telemetry Channels

From the datasets[1] used in [20], I selected three time series from Soil Moisture Active Passive Satellite (SMAP) and three from Curiosity Rover on Mars (MSL). These datasets (time series) are given in Table 5.1. Both SMAP and MSL have been used due to the difference in the amount of telemetry (records) and feature (sensors) values they provide. The datasets are contaminated by real-world spacecraft anomalies of varying length, and annotated by experts as *point* or *contextual* anomalies. The *point P* anomalies are described as values that fall within low-density

[1] https://s3-us-west-2.amazonaws.com/telemanom/data.zip

regions of values, while *contextual C* anomalies do not, yet are anomalous with regard to local values.

Channel ID	Spacecraft	Features	Train-val	Test	Test Anomalies
P-4	SMAP	25	2609	7783	3x point
E-13	SMAP	25	2880	8640	3x contextual
T-1	SMAP	25	2875	8612	1x point, 1x contextual
D-14	MSL	55	3675	2625	2x point
T-13	MSL	55	1145	2430	2x contextual
C-1	MSL	55	2158	2264	1x point, 1x contextual

Table 5.1: Synopsis of the chosen six (channels) multivariate time series

Channels are selected based on their anomaly properties over the test sequence. Particularly, channels *P-4*, D-14, E-13 and T-13 exhibit the maximum number of *point* (P) anomalies or *contextual* (C) anomalies available in SMAP and MSL. While channels *T-1* of SMAP and *C-1* of MSL cover the joint contamination of both anomalous types. The number of records within the given anomalies per channel is as follows: channel P-4 has 130 records for P_1, 200 records for P_2, and 110 records for P_3; channel E-13 has 101 records for C_1, 40 records for C_2, and 120 records for C_3; channel T-1 has 1499 records for P_1 and 35 records for C_2; channel D-14 has 20 records for P_1 and 200 records for P_2; channel T-13 has 100 records for C_1 and 150 records for C_2; channel C-1 has 200 records for P_1 and 110 records for C_2.

5.1.2 MIT-BIH Supraventricular Arrhythmia

The MIT-BIH Supraventricular Arrhythmia Database (MBA) is a large-scale dataset of 2 features, popular for testing model performance in the [63, 64, 58] management community. In particular, the dataset[2] used by [58] is partitioned into *train-val*, *test* sequences of common sizes, each containing 7680 records. The *test* sequence contains 24 anomalies of two types: 18 *Ventricular ectopic beats* (V) and 6 *fusion beats* (F). These are subsequence anomalies of common length 40, where each anomaly is characterized by a peak value surrounded by an equal number of records on both the left and right sides of the peak.

5.2 Baseline Model

[20], also provide results of a model[3] implementation which instantiates the *predictability modeling* perspective. Its architecture is fixed and consists of a LSTM network using a stack of 2 LSTM layers and 2 input dropouts. The model uses horizon windows to predict normality, given context windows. The model was also used in the data explored by [58] . Here, I use the model of [20] as the baseline model, due to its straightforward design and so as to maintain relevant performance comparisons across all experiments.

[2]https://github.com/imperial-qore/TranAD/tree/main/data/MBA
[3]https://github.com/khundman/telemanom/tree/master/telemanom

5.3 Hyper-parameters and Model size

For the manual hyper-parameters, only *context size* was varied all others were set to default. In particular, for the six data channels of Table 5.1 context size was set to 25 instead of 250 [20], forcing DITAN to make predictions using a window of 10 times less records than that used by [20]. This is challenging because the average duration of labeled anomalies is in range of hundreds instead of dozens records. On the other hand, for the MIT-BIH Supraventricular Arrhythmia data set, context size was set to the same value 10 as used by [58], and 5 *epochs* were maintained, thereby maintaining the same difficulty.

	P-4	**E-13**	**T-1**	**D-14**	**T-13**	**C-1**	**MIT-BIH**
Layers	2	2	2	2	1	2	1
Units	94	94	128	60	34	32	35
Units Decay	0.528	0.528	0.5	0.9998	0.615	0.5	0.404
Dropout	0.185	0.185	0.0	0.0165	0.052	0.0	0.1049
Reg. Strength	0.0001	0.0001	0.0	0.001	0.0001	0.0001	0
Le. Rate	0.01	0.01	0.01	0.01	0.01	0.01	0.001
Le. Scheduler	step	step	step	exponential	step	step	constant
Le. Patience	5	5	10	7	8	9	5
Batch Size	32	32	32	32	64	64	32
Trainable Param/s	161,393	161,393	284,801	133,991	28,595	24,353	22,682
Run Found	3^{rd}	3^{rd}	12^{th}	16^{th}	6^{th}	12^{th}	13^{th}

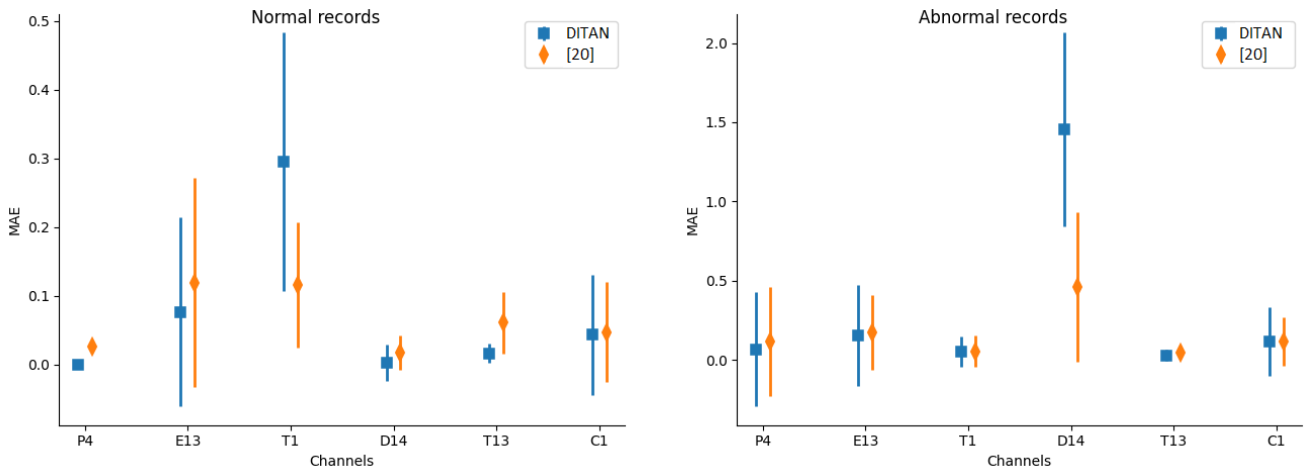
Table 5.2: The optimized hyper-parameter values over datasets

The optimized hyper-parameter values are given in Table 5.2. These are the result of 20 Bayes optimization runs, where each run evaluates the same configuration on 4 partitions, formulated by the forward chaining process. Hence, the total number of models trained (and validated) across the 7 series are $7\times20\times4$=*560*. The initial configuration for all series is that used in the baseline model of [20]. That is, *layers = 1 (per network), units = 80, units decay = 1.0, dropout = 0.3, reg. strength = 0.0, le. rate = 0.001, le. scheduler = constant, le. patience = 10, batch size = 64*. This guarantees that the extracted topology per series is at least as good as the initial one.

The size of each model (see *trainable parameters* in Table 5.2) is mainly affected by the total number of *units* across *layers*, and the *context size* hyper-parameters. The effect of lower context size is observed on MIT-BIH dataset, since it uses similar number of units as T-13, but has less parameters. The channels in Table 5.1 share the same context size, but their model parameters are correlated to the total number of units across layers. The model applied to channel T-1 has the greatest size, followed by those applied to channels P-4, E-13, D-14, T-13 and C-1.

5.4 Predictability Modeling

Predictability power refers to the ability of a model to predict the actual values when the actual values are labeled as normal, and to deviate from the actual values when labeled as abnormal. Since ground truth (labels) is provided by experts, I can separately evaluate the mean absolute error (MAE) over normal and anomalous records. Models are trained to predict normality, thus a good quality of predictions corresponds to low MAE over normal records in conjunction with high MAE over anomalous records. The objective of my ground-truthing exercise is to evaluate how well normality is learned by DITAN, compared with the models applied in [20]. I use the Mann and Whitney [34] non-parametric test to allow a statistical assessment of the agreement.



Normal records Abnormal records

Figure 5.1: MAE and standard deviations for the 6 channels

The resulted mean absolute errors (MAE) are given in Figure 5.1 for both the normal and abnormal records across the channels. The standard deviation of MAE on both normal and abnormal records is low, indicating a consistent mean, and similar between DITAN and [20]. The lowest standard deviation on MAE, is found for normal records in channel P-4, being 0.0001 for DITAN and 0.0017 for [20]. The maximum standard deviation on MAE, is found for anomalous records in channel D-14, being 0.609 for DITAN and 0.474 for [20]. DITAN generally produces lower mean absolute errors (MAE) on normal records, with a p-value of 0.85. On the other hand, [20] show higher MAE for abnormal records, but with a lower p-value of 0.62.

	P-4	**E-13**	**T-1**	**D-14**	**T-13**	**C-1**	AVG
DITAN	+200%	+68%	-140%	+199%	+60%	+93%	80%
[20]	+128%	+37%	-71%	+185%	-32%	+85%	55%

Table 5.3: The percentage difference of MAE between normal and abnormal records per model

The *percentage difference* of MAE between normal and abnormal records, given in Table 5.3, vary as a function

of overfitting for each model. The higher the percentage difference the better the distinction between normal and abnormal records. While zero suggests no distinction, and negative sign (-) implies a misconception of normality (i.e. normal MAE > abnormal MAE). DITAN maintains the highest level distinction across all channels, supported statistically by a p-value of 0.78. These results are an indicator that DITAN is more tolerant to overfitting when compared with the models used by [20]. In addition, for channels T-1 and T-13 models applied by [20] exhibit a negative result. This is a sign of normality misconception for these channels, while DITAN provides no misconception on T-13. On average, the percentage difference of MAE between normal and abnormal records on DITAN is 80%, compared with 55% for the models considered by [20].

	P-4	**E-13**	**T-1**	**D-14**	**T-13**	**C-1**
Normal	↓100%	↓36%	↑154%	↓83%	↓73%	↓8%
Abnormal	↓42%	↓11%	↓8%	↑216%	↓32%	↑1%
Total Change	+16%	+21%	-150%	+478%	+18%	+9%

Table 5.4: The percentage change of MAE from [20] to DITAN over normal and abnormal records per model

The *percentage change* of MAE over normal and abnormal records obtained by switching to DITAN modeling, given in Table 5.4, is to quantify the improvement in predictability power achieved by using DITAN. A large improvement on predicting normal records (i.e., decrease in MAE) is accompanied by a small cost on predicting abnormal records (i.e., decrease in MAE). For channels P-4, E-13 and T-13 the improvement on normal records is more than two times greater than the cost on abnormal records, indicating improvement of DITAN when compared with models applied by [20]. Similarly, for channels D-14 and C-1, results indicate a higher quality of predicting normal records, with also benefit (i.e., increase MAE) on predicting abnormal records. The only exception is channel T-1, because this channel contains an anomaly of length 1499 records, which is impossible to be captured by observing a context window with duration of 25 records. Finally, *Total Change* is computed for each channel as an indicator of the total benefit (+) and cost (-), in percentage terms using $A + B + (A * B)$, where *A* determines the percentage change of MAE over normal records and *B* over abnormal records. Note that *A* is positive only when a decrease is observed, while *B* is positive only when an increase is found. Therefore, by changing from [20] to DITAN modeling, improves the predictability power across all channels by 65%.

5.5 Detecting Anomalies

Next, I evaluate the detection performance of DITAN against the ground truth for NASA Spacecraft Telemetry Channels, and compare DITAN with other methods on the MIT-BIH Supraventricular Arrhythmia. Finally, I examine the effect of thresholding on recall.

5.5.1 Effectiveness of DITAN across channels

For each channel considered, I compute a confusion matrix, and assess precision, recall and provide F0.5 score (Table 5.5). I use *False Positive Ratio* (FPR) to examine the normal records mistakenly identified anomalous, while the correctly identified anomalous records are examined using both precision and recall. An overall performance, mainly affected by precision, is also obtained using the $F_{0.5}$ score.

Channel ID	TP	FP	TN	FN	FPR	Precision	Recall	$F_{0.5}$ Score
P-4	242	0	7340	201	0.0%	100%	54.6%	0.86
E-13	113	121	8255	151	1.4%	48.3%	42.8%	0.47
D-14	222	69	2334	0	2.9%	76.3%	100%	0.80
T-13	136	33	2145	116	1.5%	80%	54%	0.73
C-1	201	142	1810	111	7.3%	58.6%	64.4%	0.60

Table 5.5: Confusion Matrix of the records across channels; True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN), as well as the False Positive Ratio (FPR), Precision, Recall and F0.5-score

False Positive Rate (FPR)

The probability of a false alarm or False Positive Rate (FPR), is examined using $FPR = FP/(FP + TN)$ in which *False Positive* (FP) is the portion of normal records incorrectly identified as anomalous, and *True Negative* (TN) is the portion of correctly identified normal records. There is a low *FPR* probability across all channels, indicating an accurate prediction of normal records. In channel P-4, all critical regions are aligned to the actual anomalies (Figure 5.2), resulting to no FP. FP in channels D-14, E-13, T-13 and C-1, are the result of differences in alignment between critical regions and actual anomalies. While in channel E-13 an additional critical region is introduced. Among all the reported critical regions, only one critical region is considered false positive, all others are aligned with the 12 actual anomalies.

Precision

The portion of a true alarm when an anomalous record is reported is assessed using the *Precision* $P = TP/(TP + FP)$, with *True Positive* (TP) being the portion of correctly identified anomalous records. It is the fraction of records correctly detected as anomalous among all detected anomalies, whether false or true. I observe that the precision across channels (P-4 and D-14) contaminated by only point anomalies is slightly higher than in channels (E-13, T-13 and C-1) contaminated by at least one contextual anomaly. This is because contextual anomalies require more normal patterns to be memorized, and thus introduce noisier errors. These errors can be overshadowed by the larger errors associated with point anomalies. On average, the probability of reporting a true alarm (TP) instead of false alarm (FP) is greater than 70%.

Recall

The proportion of actual anomalies that are correctly identified is assessed using the *Recall* $R = TP/(TP + FN)$, with *False Negative* (FN) being the portion of missed anomalous records. Recall is the fraction of records correctly identified as anomalous among all discovered and missed anomalies. Performance is gain better for channels contaminated by point anomalies than those with contextual anomalies. In D-14, actual anomalies are all detected. While a subset of actual anomalies are considered FN in channels P-4 (Figure 5.2), E-13, T-13 and C-1 due to the pruning methodology. Also, in channels C-1 and E-13 an entire anomaly is masked due to noisy errors. From the 12 actual anomalies, only two are masked completely and thus failed to be discovered by DITAN.

Figure 5.2: The discovered and missed actual point anomalies in channel P-4

F0.5 score

The F-scores family is very useful for evaluating results in unbalanced data. Here I used the $F_{0.5}$ score, defined as $F_{0.5} = 1.25P.R/(0.25P + R)$. Unlike the F_1 score, which is the harmonic mean of precision (P) and recall (R), the $F_{0.5}$ score gives more weight to false positives (FP) than to false negatives (FN). A lower $F_{0.5}$ score is for channel C-1 due to a masked anomaly (FN) and for channel E-13 due to an additional critical region (FP) and a masked anomaly (FN). While a similar $F_{0.5}$ score is for channels P-4 and D-14 (containing only point anomalies) and T-13 (containing only contextual anomalies). On average, channels (P-4 and D-14) contaminated by point anomalies give a $F_{0.5}$ score 0.8, while the $F_{0.5}$ score of channels (E-13, T-13 and C-1) contaminated by contextual anomalies yield 0.6.

Intersection over Union (IoU)

I assess how well critical regions defined by DITAN are aligned to the 12 actual anomalies, using the intersection over union *IoU = Area of Overlap / Area of Union*. The IoU scores range from 0 to 1, where 0 indicates no overlap between the critical regions and actual anomalies, and 1 indicates perfect alignment where the critical regions completely overlap with the actual anomalies. The results are given in Table 5.6. On average, the IoU score (0.54) of point *P* anomalies is similarly well to the IoU score (0.55) of contextual *C* anomalies. This means that DITAN is

able to align critical regions with actual anomalies, independent of anomaly type. Also the majority of IoU scores are above 0.5, which is generally accepted as good result, while the scores below are mainly due to the presence of false negatives (FN), such as in channel P-4 (Figure 5.2).

Anomaly Type	Channel ID	Actual Range	Critical Regions	IoU
(P) Point	P-4: P1	950-1080	1008-1054	0.35
	P-4: P2	2150-2350	2161-2207, 2244-2343	0.72
	P-4: P3	4770-4880	4791-4838	0.43
	D-14: P1	1630-1650	1614-1674	0.33
	D-14: P2	1800-2000	1794-2023	0.87
(C) Contextual	E-13: C1	5309-5410	masked	-
	E-13: C2	5600-5640	5600-5658	0.69
	E-13: C3	6449-6569	6442-6520	0.56
	T-13: C1	690-790	657-767	0.58
	T-13: C2	1900-2050	1942-1999	0.38
Joint	C-1: P1	550-750	415-757	0.58
	C-1: C1	2100-2210	masked	-

Table 5.6: The intersection over union (IoU) per anomaly, given critical regions of DITAN

5.5.2 Effectiveness of DITAN versus other methods

I compare the output of DITAN with that of 11 other deep models, using the baseline data of MIT-BIH Supraventricular Arrhythmia (MBA). This allows a contribution to the ongoing experiment[4] initiated by [58]. The results of all methods are given in Table 5.7, using the same evaluation metrics as in their ongoing experiment. DITAN's critical regions align with the 24 actual anomalies of the MBA dataset. These critical regions are almost free of false positives (FP), since almost all (0.9910) the detected anomalies are actual anomalies. As a result, DITAN leads in terms of precision (P). However, the dynamic-length of critical regions produces false negatives (FN) in the form of misalignment with the fixed-length actual anomalies. As a result, DITAN exhibit to a relatively low recall (R) value (0.7785) and, consequently, relatively low AUC and F1 scores. The recall value suggests that more than 70% of the duration in each actual anomaly is overlapping with the corresponding critical region of DITAN.

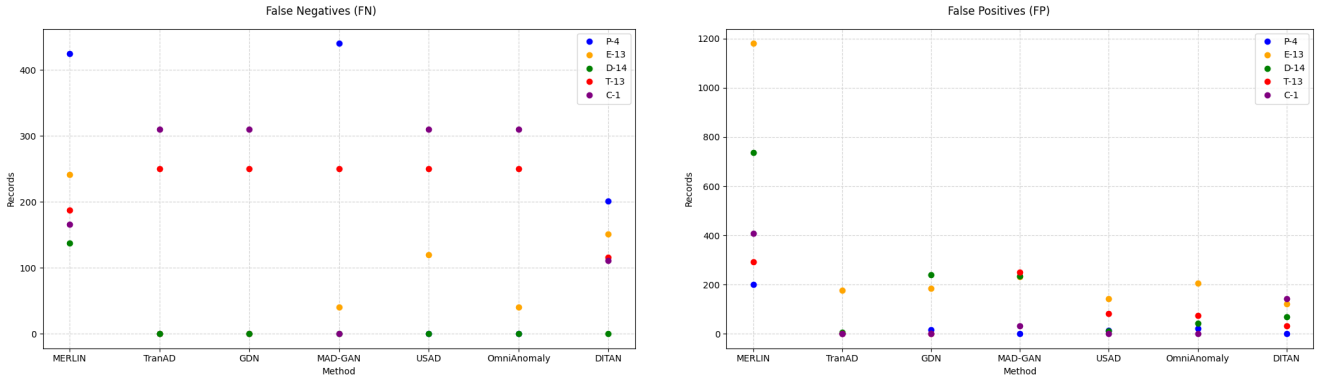
[4] https://github.com/imperial-qore/TranAD/tree/main

	P	R	AUC	F1
MERLIN [38]	0.9846	0.4913	0.7828	0.6555
LSTM-NDT [20]	0.9207	0.9718	0.9780	0.9456
DAGMM [59]	0.9475	0.9900	0.9858	0.9683
OmniAnomaly [44]	0.8561	1.0000	0.9570	0.9225
MSCRED [53]	0.9272	1.0000	0.9799	0.9623
MAD-GAN [26]	0.9396	1.0000	0.9836	0.9689
USAD [2]	0.8953	0.9989	0.9701	0.9443
MTAD-GAT [60]	0.9018	1.0000	0.9721	0.9484
CAE-M [61]	0.8442	0.9997	0.9661	0.9154
GDN [12]	0.8832	0.9892	0.9528	0.9332
TranAD [58]	0.9569	1.0000	0.9885	0.9780
DITAN	0.9910	0.7785	0.8878	0.8719

Table 5.7: DITAN with 11 state-of-the-art methods on the MBA dataset, using the evaluation metrics Precision (P), Recall (R), Area under the ROC curve (AUC) and F1 score

5.5.3 Effect of Thresholding on Recall

I compared DITAN with six other methods (MERLIN, TranAD, GDN, MAD-GAN, USAD, OmniAnomaly) using the data channels (P-4, E-13, D-14, T-13, C-1) provided in Table 5.1. The objective was to analyze the occurrence of false negatives (FN) and false positives (FP) for each model based on the threshold position in the error space. A conservative approach, which sets a threshold value closer to larger errors, is less likely to detect anomalies and more prone to false negatives (FN). Conversely, a loose approach, with a threshold value closer to lower errors, is more likely to detect anomalies and thus more prone to false positives (FP). The ideal threshold position seeks to minimize both FN and FP values. The results are given in Figure 5.3.



False Negatives (FN) False Positives (FP)

Figure 5.3: The FN and FP across channels per method

I find that methods MAD-GAN and MERLIN are unable to detect actual anomalies in channels P-4 and E-13 respectively, since the number of false negatives (FN) matches the number of actual anomalies. Similarly, the

methods TranAD, GDN, MAD-GAN, USAD and OmniAnomaly fail in channel T-13 and, except for MAD-GAN, in channel C-1. Moreover, in channel P-4, the methods DITAN, TranAD, and MAD-GAN exhibit to the lowest number of false positives (FP), but MAD-GAN maximizes the number of FN. In channel E-13, the method DITAN exhibits the lowest number of FP. The methods TranAD and GDN exhibit to no FP but also maximize the number of FN in channel T-13. The same case occur in channel C-1 for the methods TranAD, GDN, USAD and OmniAnomaly. This analysis shows that all methods exhibit a more conservative thresholding methodology, while in certain channels, the threshold value is set to an excessively high error value.

5.6 Numerical Interpretation

Finally, I empirically evaluate the similarity approach (Section 4.1.2) for interpreting the detected anomalous records of DITAN in the data channels provided in Table 5.1. The results of clustering the anomalous records in the unit space, are given in Table 5.8. In each channel, I compute the Davies-Bouldin index (DBI) score to assess the compactness and separation of the clusters. On average, the DBI score (0.23) for contextual anomalies (channels E-13, T-13, and C-1) is relatively higher than the DBI score (0.13) for point anomalies, because contextual anomalies only differ in certain conditions. However, the low DBI score across anomalous records, indicates that the clusters are well chosen.

Channel ID	No. of Anomalous Records	Unit Space	No. of Clusters	DBI
P-4	242	94	4	0.15
E-13	234	94	3	0.23
D-14	291	60	2	0.11
T-13	169	34	2	0.23
C-1	343	32	4	0.22

Table 5.8: The details and quality assessment of unit space clustering the anomalous records per channel

Given the clustering details provided in Table 5.8, I repeat the clustering of the anomalous records but now in the original feature space. The objective is to show how many of the anomalous records that are similar in the unit space also remain similar in the feature space. The results are given in Table 5.9. Overall, more than 85% of anomalous records maintain similarity also in their feature representation, indicating that the choice of clustering in the unit space is also intuitive in the feature space. For instance, the verified similarities of channel T-13 are presented in Figure 5.4. On the top panel, the two contextual anomalies are annotated by different colors (orange and green). These colors represent the two separate clusters (C0 and C1) that appear on the bottom panel.

Channel ID	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Overall
P-4	16/16	133/147	73/73	5/6	227/242 (94%)
E-13	32/52	128/144	32/38	-	192/234 (82%)
D-14	154/159	100/132	-	-	254/291 (87%)
T-13	111/111	58/58	-	-	169/169 (100%)
C-1	182/215	24/51	21/30	23/47	250/343 (73%)

Table 5.9: The unit space similarities verified in data space per cluster across channels

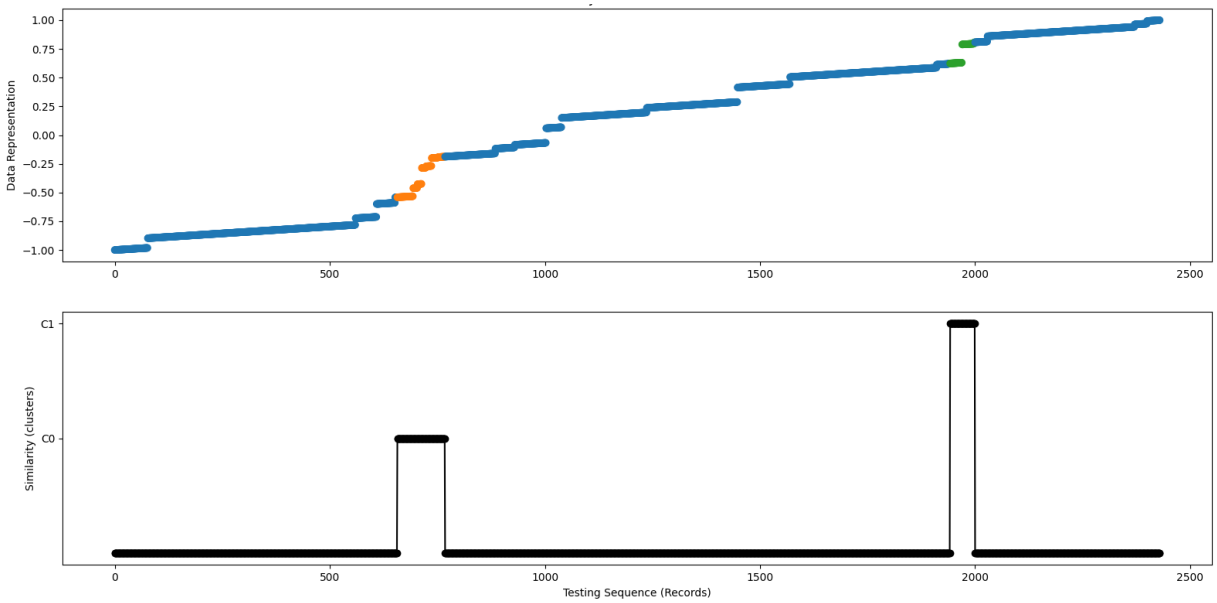


Figure 5.4: The clustered contextual anomalies of channel T-13

5.7 Exploring Methodological Improvements

In Section 3.5, I proposed two minor updates to improve the efficiency of the thresholding methodology of DITAN. I also want to see if there is a significant difference in the effectiveness. To this reason, I executed the updated version of DITAN on the same channels as in Table 5.6, to get the new intersection over union (IoU) scores. The comparison between the previous and new IoU scores are given in Table 5.10. On average, the new IoU score is increased (0.59) compared to the previous IoU score (0.55). That is because the boundaries of the new critical regions are aligned better to the actual range of the anomalies. However, there is insufficient evidence to conclude a significant difference.

Anomaly Type	Channel ID	IoU	IoU (new)
(P) Point	P-4: P1	0.35	0.55
	P-4: P2	0.72	0.58
	P-4: P3	0.43	0.57
	D-14: P1	0.33	0.53
	D-14: P2	0.87	0.90
(C) Contextual	E-13: C1	-	-
	E-13: C2	0.69	0.61
	E-13: C3	0.56	0.55
	T-13: C1	0.58	0.52
	T-13: C2	0.38	0.30
Joint	C-1: P1	0.58	0.83
	C-1: C1	-	-

Table 5.10: The Intersection Over Union (IoU) from Table 5.6 and the new IoU scores

Chapter 6

Application to a Hydro-thermal system

Having built and validated DITAN, I am now going to apply the approach to the real-world data for the scientific question in hand. That is, to examine the intensity, duration and type of anomalies caused on surface temperature by meteorological/atmospheric effects at a hydrothermal system, thereby defining the role of such external (to the volcanic system) drivers in modifying surface temperature at such systems. This will be done by focusing on Vulcano (Aeolian Islands, Italy) where a sensor network is capable of providing time series data to track surface temperature above an active hydrothermal system, co-located with a weather station.

6.1 Problem Description

A geothermally heated zone involves a heat exchange between the hydrothermal system and the surface, which buffers surface temperature [81], modulating diurnal and annual cycles. In Figure 6.1, the presence of the enhanced geothermal flux creates a thermal anomaly (ΔT) between heated (T_h) and non-heated zones (T_0): $\Delta T = T_h - T_0$. It will also create a positive temperature difference between the ground and air. However, interaction with the atmospheric system will also modulate the surface temperatures by, for example, solar heating, shading, and/or rain forced convection wind, (e.g., [83]). The atmospheric effect is an especially strong influence in situations where T_0 is particularly low (close to ambient) and thus where ΔT is very small, as is the case over hydrothermal systems. At near-ambient temperatures forced convection becomes the dominant heat loss, so that wind (through its influence on the convective heat transfer coefficient) can, for example, have a strong influence on surface warming and cooling patterns [83].

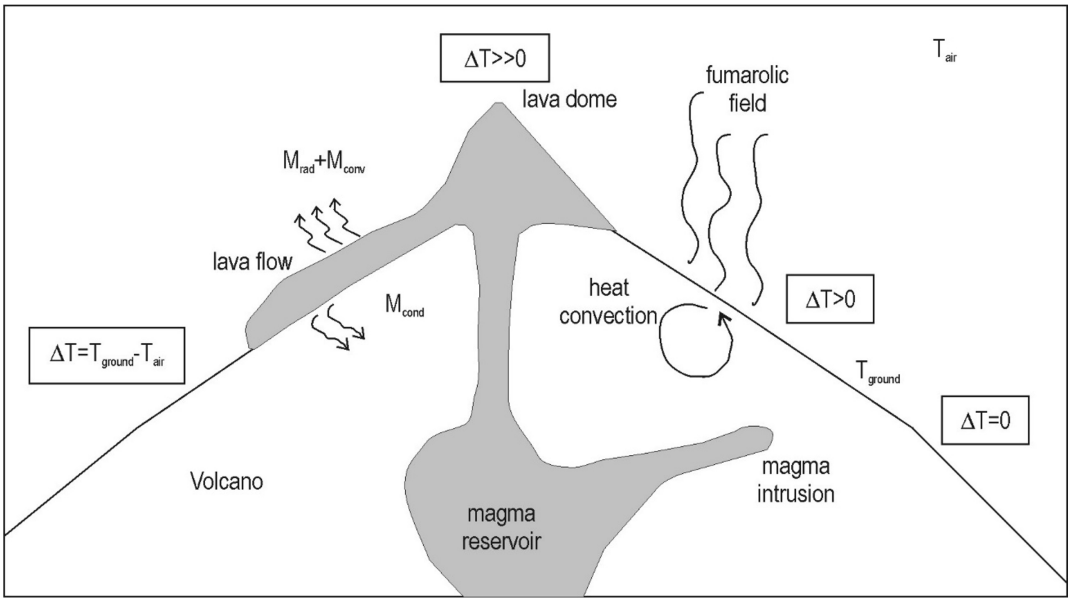


Figure 6.1: The Figure 0.1 from [81]. Sketch of the main sources of thermal emission that can be detected by a satellite or airborne sensor, modified Figure 1 in [82] reproduced by permission of American Geophysical Union. In normal conditions ground (T_{ground}) and air temperature (T_{air}) are approximately equal, so that $\Delta T = T_{ground} - T_{air} \approx 0$. Over a subsurface heat supply, such as a magmatic intrusion above which natural convection in a porous, or fractured, medium carries heat to the surface, ΔT becomes positive. Over a high temperature surface heat source, such as an active lava, ΔT becomes strongly positive. Given data collected at the correct wavelengths and spatial resolution, both anomalies can be detected by a satellite infrared sensor. The schematic also shows the main sources of heat loss from an active lava body, as are typically calculated using infrared data. These being radiation (M_{rad}), convection (M_{conv}) and conduction (M_{cond}).

6.1.1 Vulcano and its Thermal History since 2000

At Vulcano, heat ascends by permeable convection above a hydrothermal system that has its depth at around 1 kilometer below the fumarole field (280 meters asl), demonstrated in Figure 6.2. This causes a "heat chimney" at the top of which is a broad heated zone containing areas of fumarolic activity shown in Figure 6.3. While the heated zone is characterized by grey surfaces, non-heated zones are red, and the heated zone is typically marked by a very low (typically 1 to 10 degrees Celsius) thermal anomaly; with nighttime surface temperatures being higher in the grey (hot) zone than in the red (cold) zone. Fumarole temperatures, ΔT and, thus, also heat flux generally declined between 2000 and 2010, with one or two reversals in the trend due to increases in permeability associated with seismic activity and fracturing of the rock above the hydrothermal system [86]. Heat fluxes during the period 2010 to 2020 were particularly low and stable at around 4 to 12 megawatts [85], as opposed to up to 120 megawatts during times of unrest [87] (Figure 6.4). The period 2010–2020 has thus been defined as a *baseline* or background level for heat flux, against which change in the thermal state of the hydrothermal system can be assessed [87].

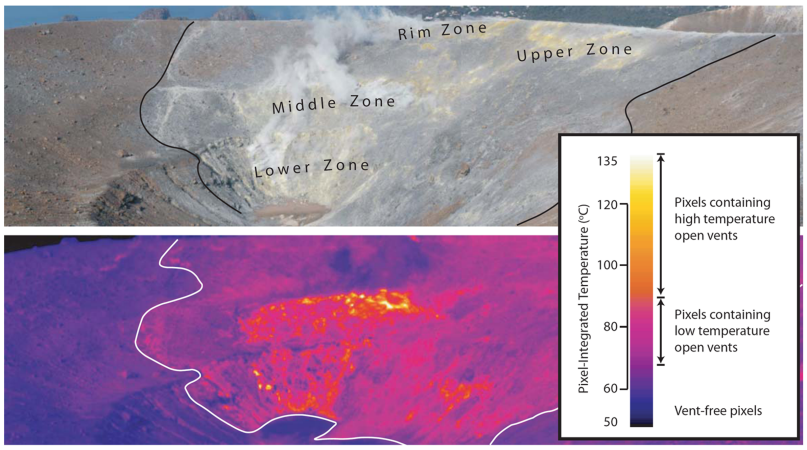


Figure 6.2: Figure 1 from [84]: (a) Upper, middle and lower zones of the Vulcano Fossa Fumarole field viewed from the south. (b) Thermal image mosaic of the same area, with the fumarole field limit as defined by the zones of discoloration and anomalous surface temperatures marked with black and white lines, respectively. Distance from rim to crater floor is $\approx$ 120 meters transects across the border of the heat chimney, that measure temperature below the surface, show clearly the limit of the thermal anomaly.

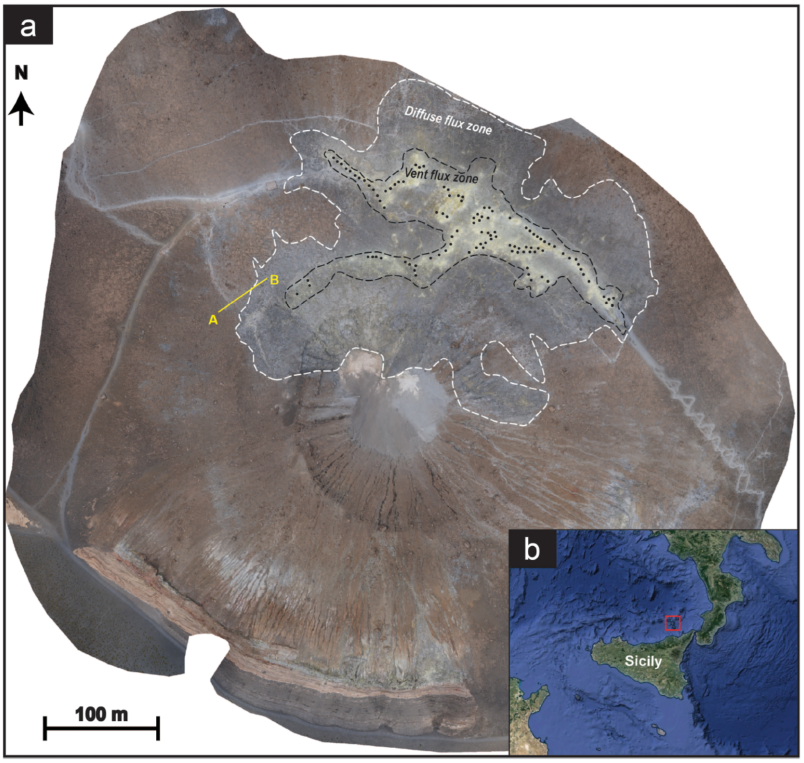


Figure 6.3: Figure 1 from [85]: (a) Orthophotomosaic of Vulcano Fossa showing the zone of active fumaroles (delimited by a black dash line). White dashed line is the boundary of the diffuse heated zone; yellow line represents thermal transect (A–B) where measurements to define ΔT are made; black dots represent individual fumaroles. (b) Location of Vulcano Island within the Aeolian archipelago, north of Sicily (background of Google Earth image ©2018 Digital Globe

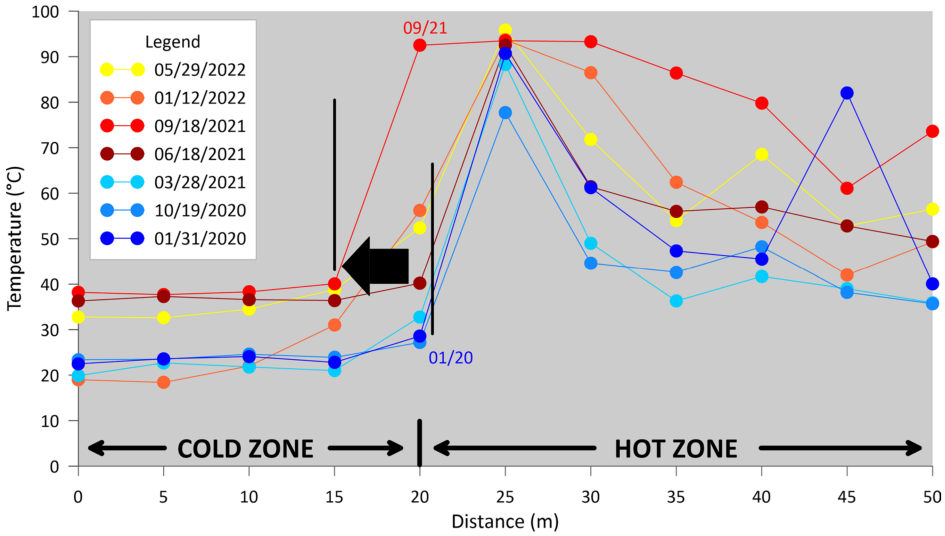


Figure 6.4: Figure 3 from [87]: Temperature at 15 centimeters depth on Profile A–B for 2019 to 2022. The large black arrow marks the hot zone offset into the cold zone after September 2021

6.1.2 The Research Question to be addressed by DITAN

The problem, in terms of system components and sensor values to collect to track the influence of these components is given in Figure 6.5. Vertically there are two elements to the system: subsurface where internal drivers modulate the surface temperature and the the atmosphere where external drivers modulate the surface temperature. The two elements are separated by the surface and its thermal boundary layers across which heat exchange between the two elements occurs. Horizontally, are two zones: one of which is heated *hot* and linked to enhanced heat flow from the subsurface hydrothermal system, and one of which is not heated and *cold*. The thermal state of the *hot* and *cold* zones are tracked by two sensors that monitor surface and air temperature, respectively. Parameters associated with the external weather conditions (rain, pressure and wind) are measured by a third sensor system.

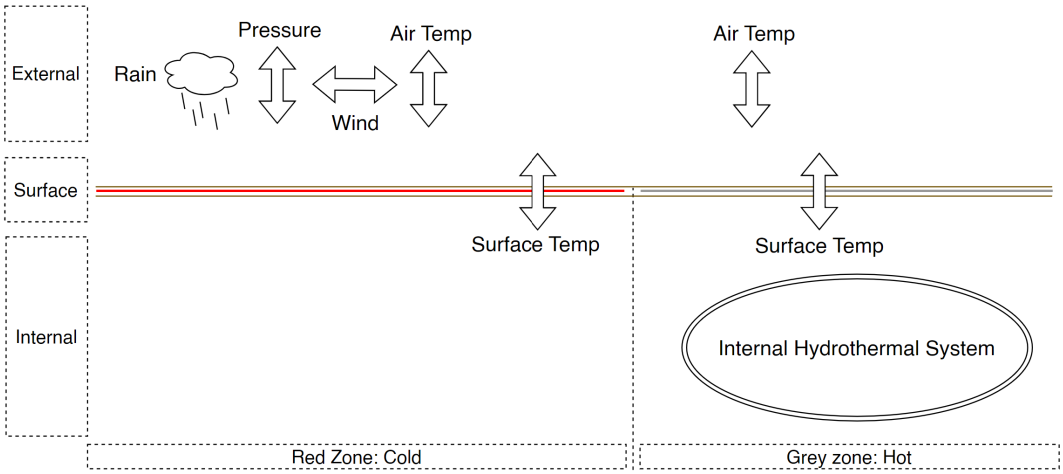


Figure 6.5: A descriptive illustration of the thermal system at Vulcano and its parameters

The focus here is on isolating, and defining external drivers to change in the thermal state. The measurements in the year 2020 were selected, when there were low levels of heat flux from the hydrothermal system and the internal element of the system is considered low and stable, and at a background/baseline state. External drivers, though remain highly variable and will be the main influences on the surface thermal state, although their influence my be buffered in the hot zone by the presence of the internal heat source. Therefore the objective is to detect anomalies across the seven sensors and then interpret them into physical events, by examining links using the domain knowledge capability of DITAN. Two or more anomalies are considered to be linked to each other when their start difference is within a defined time interval. A research question is then defined as:

> *what external factors drive changes in surface temperature for a surface above an active hydrothermal system ?*

The surface temperatures will exhibit recurring patterns as part of the diurnal cycle. I am asked to examine abnormal events, at which anomalies on surface temperature (hot, cold) sensors are *linked* to anomalies on the external weather (pressure, wind speed, rain) sensors. The type of these events is called *surface external driver*, since surface conditions are driven by external factors.

6.1.3 The Sensor Network used by DITAN

The sensor network used here was installed on the Fossa crater in January 2020, demonstrated in Figure 6.6. This network consists of two temperature stations separated by 50 meters, one inside the hot (grey) zone and one in cold (red) zone, and one weather station (ws) co-located with the cold (red) station, given in Table 6.1. The temperature stations are two thermocouples (Onset HOBO TMC1-HD) measuring surface temperature (T_s) and air temperature (T_a) at height 15 centimeters above the surface. These thermocouples are linked to an Onset HOBO U12-008 data logger with a sampling rate of one record (measurement) every five minutes. The weather station is an Onset HOBO H21-USB measuring the atmospheric pressure (S-BPB-CM50), air temperature (S-THC-M002), wind speed (S-WSB-M003), and rain fall using a tipping bucket rain gauge (S-RGF-M002). These sensors are installed at ground level, and the sampling rate is one record per minute. The chosen period of collecting measurements in the sensor network is from January 31, 2020, 12:00:00, to December 31, 2020, 23:00:00, with data gaps existing due to the logger capacity being reached before download, especially due to mobility restrictions during the pandemic period, as shown in Table 6.2.

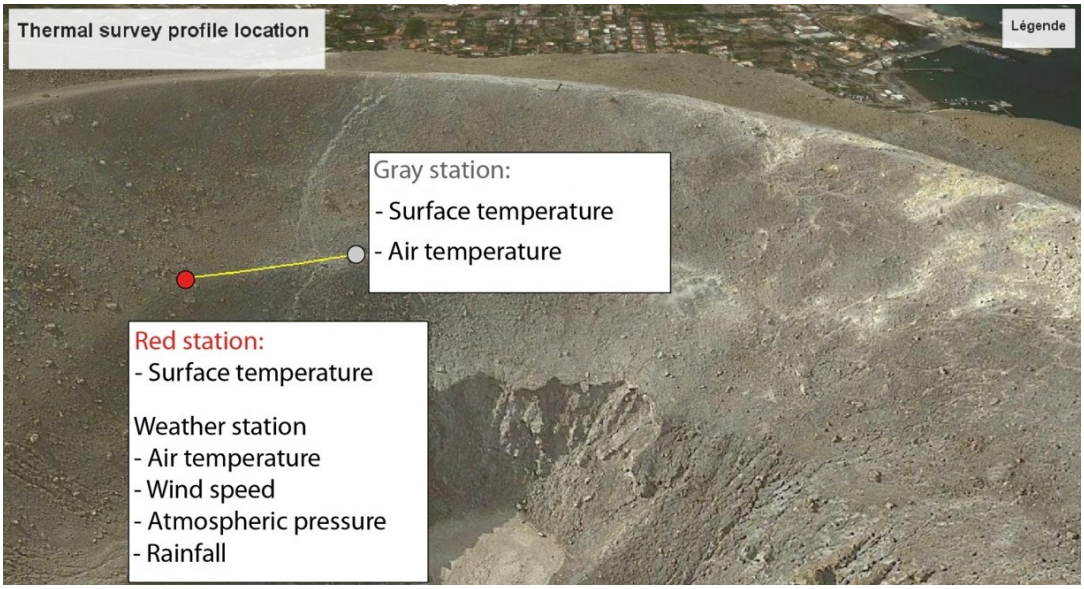


Figure 6.6: The Hot (Grey), Cold (Red) and Weather (WS) stations installed on Vulcano's La Fossa Crater and used by DITAN

Station	GPS (WGS-84)	Sensors Name	Sensors Unit
red (cold)	38.405367, 14.960742	Ts: surface temperature	degrees Celsius (°C)
grey (hot)	38.405222, 14.960564	Ts: surface temperature Ta: air temperature	degrees Celsius (°C) degrees Celsius (°C)
weather	38.405367, 14.960742	P: air pressure Ta: air temperature U: wind speed Rain: rainfall	millibar (mbar) degrees Celsius (°C) meters per second (m/s) millimeters (mm)

Table 6.1: The sensor network used by DITAN

Station	Data Gaps (mm/dd/yyyy)
red (cold)	05/31/2020 – 06/09/2020 09/04/2020 – 10/01/2020 10/20/2020 – 10/24/2020
grey (hot)	05/31/2020 – 06/09/2020 09/04/2020 – 10/01/2020 10/20/2020 – 10/24/2020
weather	05/28/2020 – 06/09/2020 09/01/2020 – 10/01/2020

Table 6.2: The data gaps in the measurements within the chosen period

6.2 Domain Knowledge

The graphical interface of DITAN (see Figure 4.2) was used to create and store knowledge within the knowledge base. Knowledge is expressed in the form of seven temporal rules *R* presented in Table 6.3. Rules were aligned to describe severe meteorological events (storm systems) and how such events may be expected to introduce a negative perturbation (decrease) on surface temperatures; which would then be followed by a recovery during, or after, the end of the event.

R1: if decrease on pressure sensor and increase on wind speed sensor started within hours, then low pressure system (meteorological) event.
R2: if positive on rain sensor and low pressure system event started within hours, then rainstorm (meteorological) event.
R3: if low pressure system event and decrease on red surface temperature and decrease on grey surface temperature started within hours, then decrease on surface temperatures (surface external driver) event.
R4: if rainstorm event and decrease on red surface temperature and decrease on grey surface temperature started within hours, then decrease on surface temperatures (surface external driver) event.
R5: if low pressure system event and decrease on red surface temperature started within hours, then decrease on red surface temperature (surface external driver) event.
R6: if low pressure system event and decrease on grey surface temperature started within hours, then decrease on grey surface temperature (surface external driver) event.
R7: if tare on air temperature, then rapid change on air temperature (instrumental) event.

Table 6.3: The knowledge defined by Experts in the form of temporal rules

This defines three event types: 1. Rules R3 to R6 characterize events considered external drivers on surface temperature change (Figure 6.5). 2. Rules R1 and R2 define a *meteorological* event, since only external conditions are linked to each other. 3. Rule R7 considers sensor failure, demonstrating an *instrumental* event.

The preview of rule R3 in DITAN is illustrated in Figure 6.7. The post-condition and part of the additional information (status, version) are located at the top of the card. The preconditions and other additional information (description, license, last update) are located below this as dynamic tabs. At the bottom of the card, two buttons allow update (left) or deletion (right) of the rule. The remaining rules (R1, R2, R4, R5, R6 and R7) are then visually depicted in the lower half of the screen.

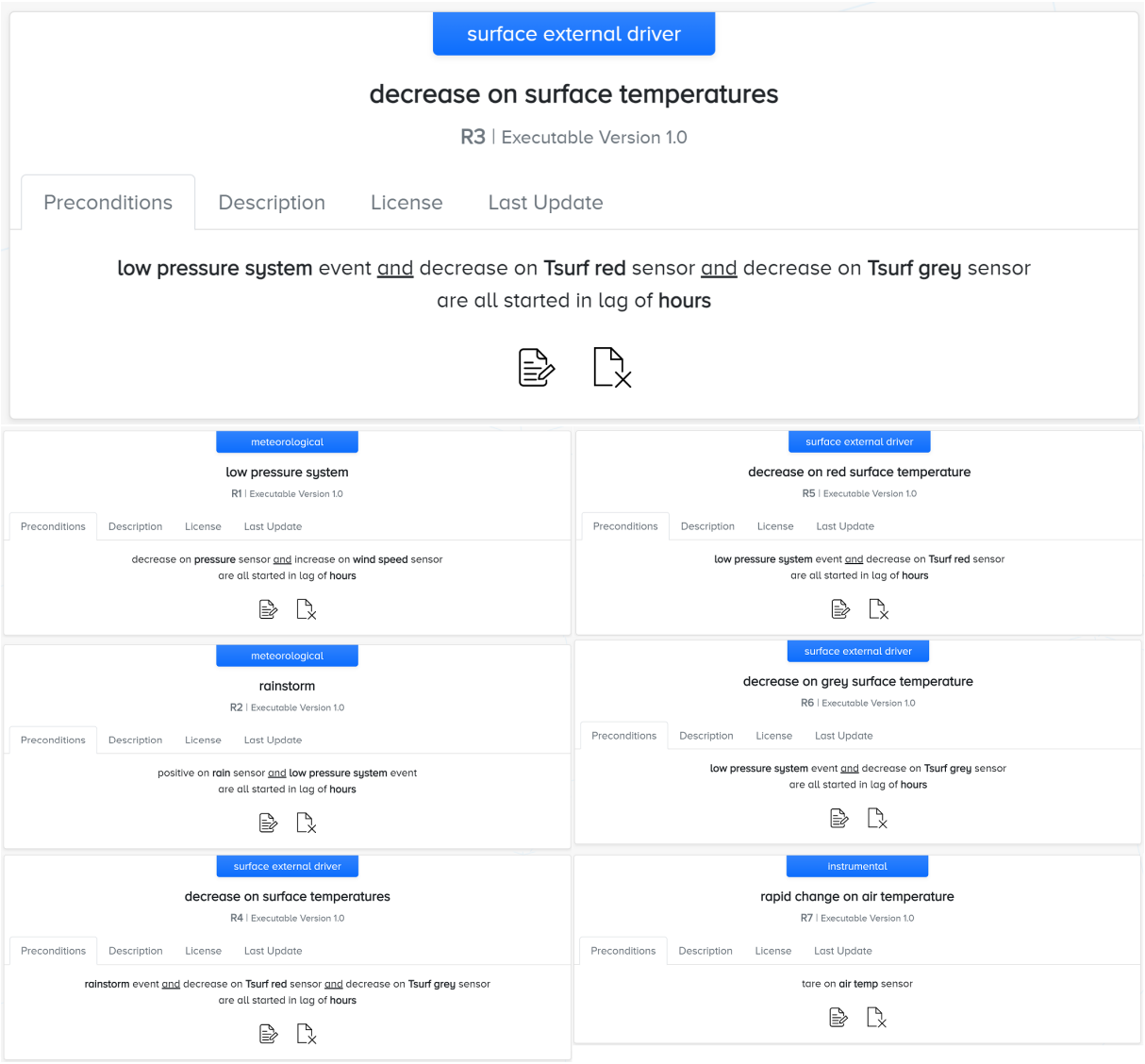


Figure 6.7: Preview of rule management module of DITAN

6.3 Vulcano in Year 2020

In total there are seven sensors in the network, for which the frequency of measurements varies. In particular, cold (red) and hot (grey) stations registers one record every five minutes while the weather station saves record per minute. Thus, all measurements from *31-01-2020 12:00:00* up to *31-12-2020 23:00:00*, were sub-sampled to every five minutes, and then down-sampled (averaged) to a common frequency of one record (time-step) per hour, where the down-sampled time series is given in Figure 6.8. This results in 6799 records (for around 283 days), where each record is a vector of seven sensor values. Thus, the temporal resolution of anomalies is expected to be of at least one hour. An overview of range of values recorded for data set is given in Table 6.4. In overall, three main data set characteristics. (1) sensors exhibit diurnal/annual cycles, (2) sensors do not have similar value ranges, (3) there are

three significant data gaps in June, September and October.

Figure 6.8: Down-sampled sensor time series for Vulcano in the year 2020

Station	Sensor	Unit	Min	Max	Mean	Standard Deviation
weather	pressure	mbar	967	1002	986	5.8
weather	air-temp	°C	4.6	48.3	19.8	8.6
weather	wind-speed	m/s	0	4.4	0.3	0.49
weather	rain	mm	0	0.5	0.002	0.019
grey (hot)	tsurf-grey	°C	6.7	45.2	23	7.7
grey (hot)	tair-grey	°C	6.9	45.2	22.5	7.6
red (cold)	tsurf-red	°C	5.9	45.3	20.7	8.4

Table 6.4: Sensors description of Vulcano in the year 2020

6.3.1 Forecasting Scenario

The selection of the appropriate size for observation context and forecast horizon relies on knowledge of the time scale of expected variations. Surface temperatures (see Figure 6.8) will exhibit diurnal cycles of 24 hours, while also following an annual cycle. In addition, following [74] major storm systems will develop over hours, so that parameters such as wind speed and pressure will evolve at a timescale of 6-24 hours during high intensity events, such as Medicanes. Thus, and following [75] the context window is set to 24 hours to allow forecast a horizon of 6 hours. This provides an appropriate temporal resolution to operate on.

6.3.2 Pre-processing

Results of pre-processing are given in Table 6.5. Of the 1253 missing time-steps (hours), only 73 are linearly interpolated. The remaining 1180 have been removed, because the formed gaps were too large to allow interpolation. Although decomposition is an option, I chose to decompose measurements solely into the residuals component. This decision was based on the understanding that external phenomena manifest themselves as short-term interruptions to *normality* causing perturbations to the diurnal cycle. In contrast, the internal driver primarily affects the long-term trend. Therefore, the values of each sensor are transformed into residuals, by explicitly estimating its decomposition type and period. Min-max normalization is then applied across all sensors to introduce a common scale, without biasing any correlations or underlying distributions. The resulting value range [-3, 3], allows to be spread as much as possible in range that is not too broad. This ensures that the presence of outlier (extreme) values are not excessively compressed and maintain their relative positions and magnitudes. The effectiveness of the pre-processing strategy is closely tied to its ability to preserve the actual correlations between sensors. The objective is to convert the data into a format suitable for analysis and input into DITAN, while still preserving the inherent relationships within the data.

sensor	decomposition	period	min	max	mean	std
pressure	additive	143	-3	3	-0.04	0.6
air-temp	additive	24	-3	3	-0.4	1.07
wind-speed	additive	22	-3	3	-0.9	0.44
rain	additive	78	-3	3	-2.6	0.2
tsurf-grey	additive	24	-3	3	-0.04	0.83
tair-grey	additive	24	-3	3	0.004	0.82
tsurf-red	additive	24	-3	3	-0.3	0.97

Table 6.5: Pre-processed sensors description of Vulcano in the year 2020

The key statistics of the pre-processed sensors are given in Figure 6.9. Despite the fact that decomposing to only residuals results in more values outside of the interquartile ranges, the sensors are able to preserve their correlations. Firstly, it can be visually validated that the four sensors (pressure, air-temp, wind-speed and rain) of the meteorological station, referred to as *external parameters*, exhibit the expected negative correlation. That is, as pressure falls so too does air temperature, but wind speed and rainfall will increase. Secondly, the surface and air temperatures for the hot zone exhibit a higher median than the surface temperature of the cold zone. That is because the cold zone is strongly influenced by decreasing trends in the external parameters, but the hot zone temperatures are buffered by the influence of the hydrothermal system, i.e., internal drivers. Thirdly, the slight difference between surface and air temperature in hot zone is because air temperature is a little less buffered by the internal drivers, and a little more modulated by the external drivers.

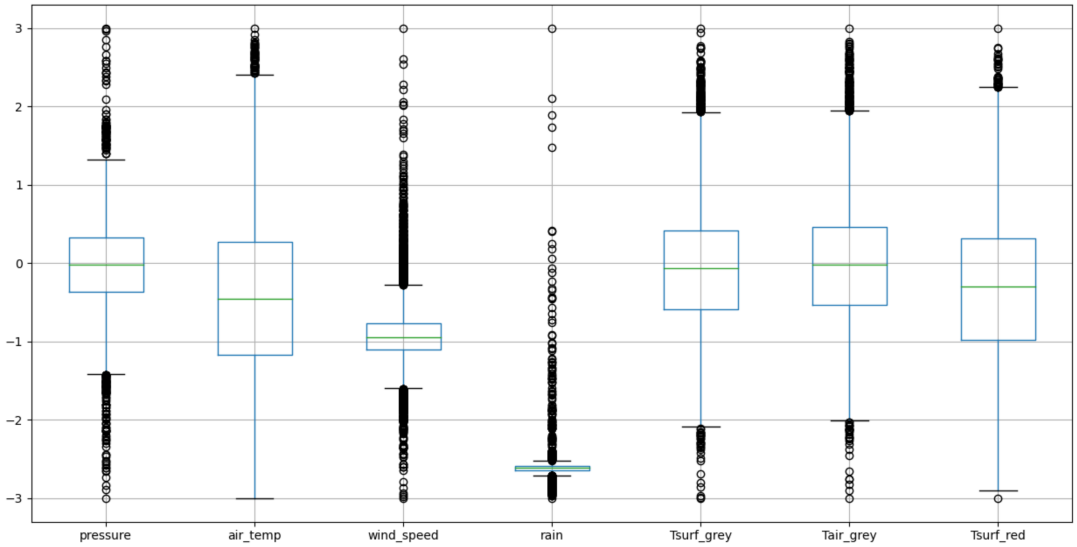


Figure 6.9: Box-plot of the pre-processed sensors in Vulcano 2020

6.4 Modeling Normality

The training phase of DITAN is conducted on data with 24 hour context size and 6 hour horizon size. The aim of learning is to reduce the differences between forecasted and actual values by identifying the suitable model parameters. To ensure equal importance in minimizing all differences, I employ mean absolute error *MAE* as the loss function. By using absolute differences to compute gradients of the loss function, DITAN can mitigate the influence of extreme events such as rainstorms, which would otherwise dominate as the main indicator of normality. Instead, it prioritizes the average understanding of underlying patterns, with patterns appearing more frequently having a greater influence on determining normality.

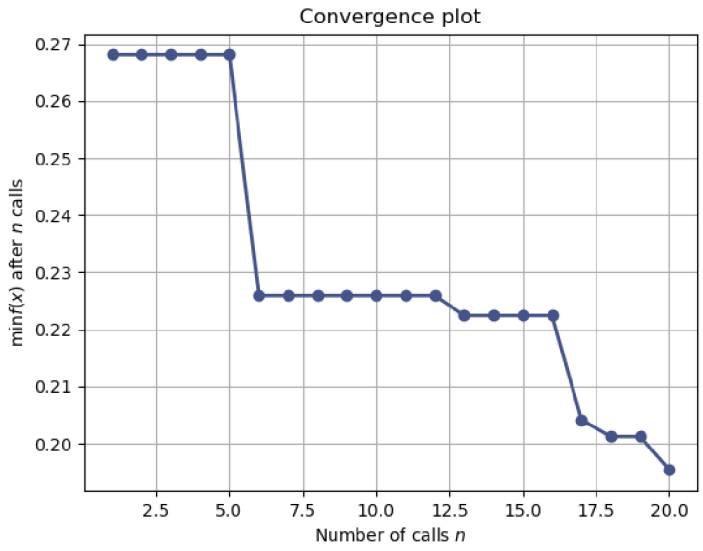


Figure 6.10: The convergence plot of DITAN across configurations using Bayes Optimizer

During the Bayes optimization process, a total of 20 different hyper-parameter configurations are examined, with their convergence history given in Figure 6.10. Each configuration is assessed using four expanding windows over the pre-processed data set, resulting into an examination of 4×20=80 models in total. The first (initial) configuration is the same as that used in Section 5.3. I observe that no improvement occurs until the 5^{th} configuration, with a significant improvement being observed on 6^{th} configuration with relatively small variations that occurring up to the 16^{th}. A gradual decrease is then observed between the 16^{th} and 20^{th} configurations, at which the objective function converges. By changing from the initial to the optimal configuration, the optimization error is decreased from 0.268 to 0.195, resulting in a 27% improvement on the objective function.

Hyper-parameters	**Values**
Layers	1
Units	32
Units Decay	0.9103629453531021
Dropout	0
Regularization Strength	0
Learning Rate	0.01
Learning Scheduler	step decay
Learning Patience	10
Batch Size	32

Table 6.6: The hyper-parameters of the optimal DITAN model

The hyper-parameters of the optimal (20^{th}) configuration are reported in Table 6.6. The resulting model consists of 19.815 parameters. These parameters are updated in batches of 32 consecutive patterns, where each record within these patterns is encoded using 32 units. In addition, use of a larger learning rate of 0.01 means that the convergence process becomes capable at exploring global maxima more effectively throughout all epochs. To maintain stability during training, a step decay factor is used, which gradually reduces the learning rate every 4 epochs. This approach helps to strike a balance between exploration and stability in the optimization process. The final parameters are selected at epoch 78, at which the patience of early stopping is exhausted, with the internal validation loss not improving by at least 0.0003 for 10 consecutive epochs after epoch 68.

6.5 Detecting Anomalies

In detection phase the trained model is used to predict normality across 6843 (records) hours, resulting in a corresponding number of errors per sensor, as illustrated in Figure 6.11. Each error is computed as the absolute difference between the predicted and observed value, according to the selected loss function. Each sequence is then smoothed using simple moving average *SMA* of 6-hours (medium-locality) Parzen windows.

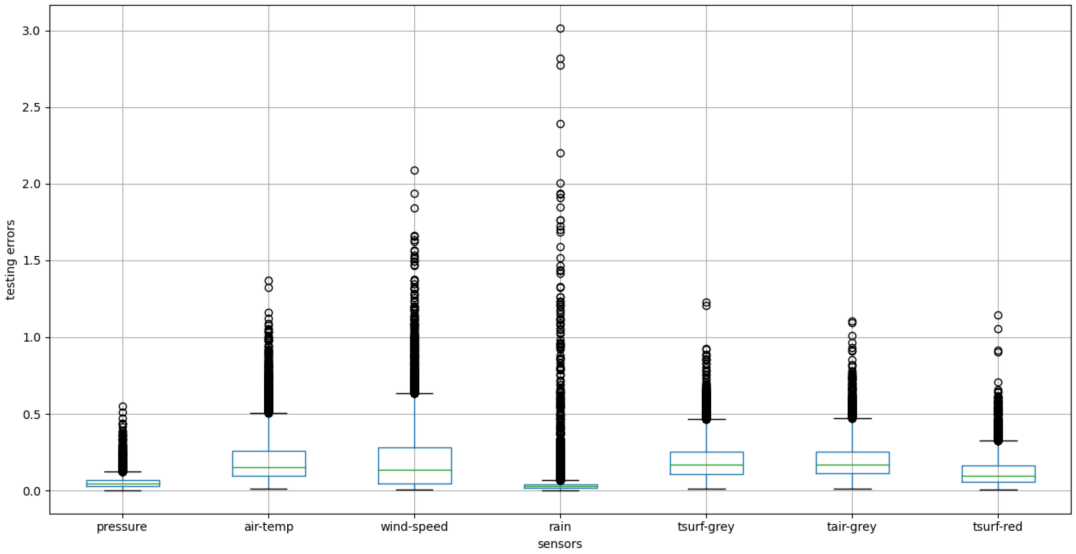


Figure 6.11: The prediction errors of DITAN across the 7 sensors

An important consideration when reducing temporal resolution is to maintain a balanced trade-off between smoothness and introduced lag. The "goodness" of the proposed window size is demonstrated in Figure 6.12, using a subset of errors from a randomly selected (pressure) sensor. I observe that the smoothed versions of the errors maintain a responsiveness to the raw errors. Also, the objective function of SMA is observed in action. In the given frame, the most intense raw error value is at position *47*, while in the smoothed errors the most intense value is at position *81*. This is because the magnitude of the raw error at position 47 itself was not high enough, in a relative sense, to overcome the highly varying raw errors in the locality of position 81. As a result, rarity refers to either high magnitude errors or errors occurring within high magnitude locality.

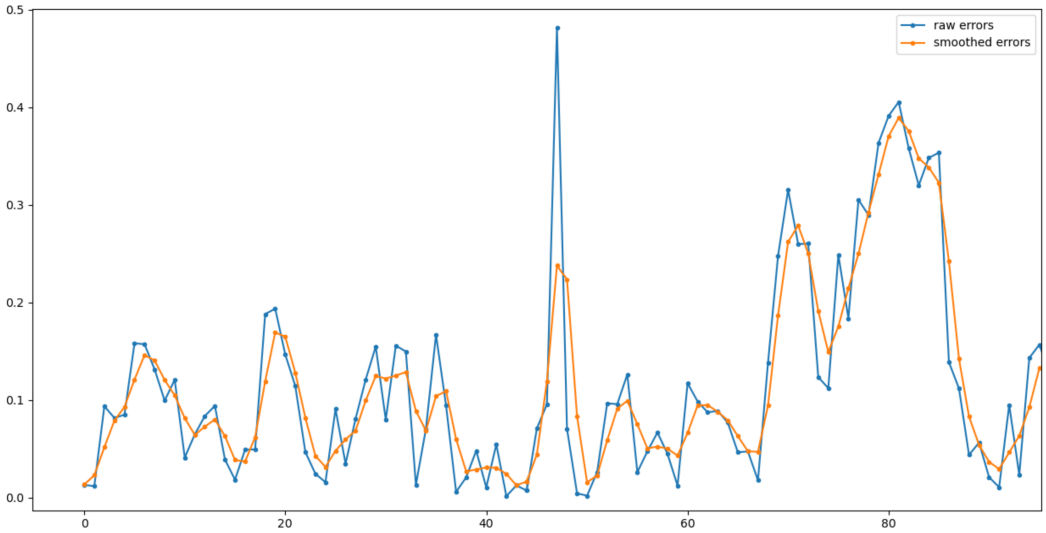


Figure 6.12: A sub-part of the raw and smoothed error sequences on *pressure* sensor

From the 6843 predicted records, 1737 are detected as anomalous and can be grouped into three similarity groups. This means around 72 days (or 25% of total records) are comprise detected, anomalous events. The number of anomalies and critical regions per sensor are given in Table 6.7. The most critical regions occur at the temperature sensors, since they are sensitive to short-term variations. In addition, the number of critical regions in the cold zone is slightly higher than in hot zone for air and surface temperatures. This is because hot zone is buffered by the internal (hydrothermal system) driver. Furthermore, the external parameters (wind-speed, rain and pressure) exhibit a high number of anomalies, indicating that the main source of anomalies on temperature sensors are due to external conditions.

Sensor Name	Critical Regions	Anomalous Values
pressure	11	462
air-temp	11	99
wind-speed	11	678
rain	6	537
tsurf-grey	13	298
tair-grey	10	363
tsurf-red	16	562

Table 6.7: DITAN's detection results per sensor

All 78 critical regions across all sensors are in the assessed root-cause diagram of Figure 6.13. Each color represents critical regions in a different sensor. The overlapping critical regions are assessed by root cause values of less than one, which is due to joint severity on the corresponding records. Only a few of critical regions are isolated, with the vast majority of critical regions being overlapping or closely spaced in time.

Figure 6.13: The root causes diagram from our platform, showing critical regions across sensors

6.6 Physical Anomalies

After detecting critical regions, the expert knowledge module (see Table 6.3) is used in the inference engine to identify physical events. From the inference perspective, rule chains play a crucial role in the analysis. The post-condition of R1 is in the precondition of R2, R3, R4 and R5, and the post-condition of R2 is in precondition of R4. Instead, R1 and R7 have no events in their preconditions. Therefore, all the possible inferences are: (a) R1, (b) R1→R2, (c) R1→R3, (d) R1→R2→R4, (e) R1→R5, (f) R1→R6, (g) R7. According to Algorithm 1 (Section 4.2.2) executions are divided into three iterations. In the first iteration, (a, g) are executed. In the second iteration, (b, c, e, f) are executed. Finally, in the third iteration, (d) is executed.

Rule ID	Event Type	Valid Executions
R1	meteorological	7
R2	meteorological	3
R3	surface external driver	1
R4	surface external driver	0
R5	surface external driver	2
R6	surface external driver	1
R7	Instrumental	9

Table 6.8: Rules executed using the Inference Engine

The number of valid executions per rule are given in Table 6.8. The two meteorological events (rules R1 and R2) are executed 7 and 3 times, respectively. This indicates large amount of external drivers influencing the system. Instead R4 does not occur and R3 is executed only once, indicating that external conditions were sufficiently strong to decrease the surface temperatures in both cold and hot zones. Given that R3 is equivalent to the conjunction of R5 and R6, both are executed once as well. However, R5 occurs twice, indicating that at another time, the impact of external conditions to the cold zone was significantly higher than for the hot zone. Finally, R7 is executed nine times indicating sudden changes in the air temperature due to a systematic failure on the sensor, such as automatic reset.

Meteorological Events at Vulcano 2020

The main meteorological events occurring at Vulcano in the year 2020 are characterized as *low pressure system* and *rainstorm* by rules R1 and R2 respectively. To gain a comprehensive understanding of the occurrences of these meteorological events, it is crucial to analyze the critical regions of DITAN within the pressure, rain and wind speed sensors.

The root-causes of the critical regions for pressure, rain and wind speed are given in Figure 6.14. I observe that the most critical regions for wind speed and pressure are closely correlated in terms of time, contributing to the

occurrence of R1. Also, the critical regions for rain are concentrated later in the year, in close proximity to critical regions for wind speed and pressure, suggesting a significant impact on the occurrence of R2 at this time.

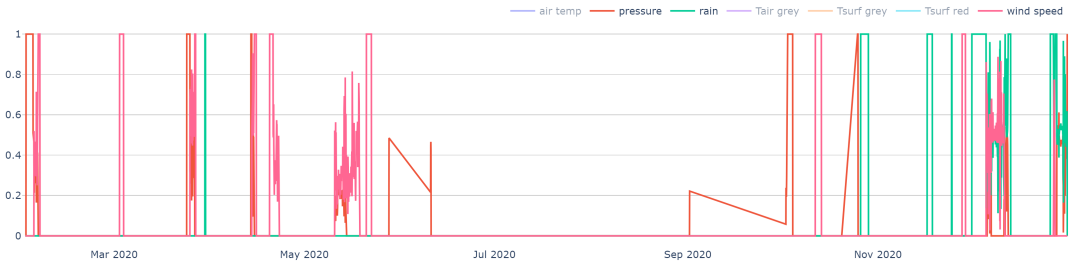


Figure 6.14: The root-causes diagram for pressure, rain and wind speed sensors

Critical regions for pressure, rain and wind speed are correlated with respect to the actual values, as depicted in Figure 6.15. Notably, the critical regions for the pressure sensor demonstrate a negative correlation with those of the wind speed sensor. This implies that when pressure exhibits an anomalous decrease, the anomalous values of the corresponding critical region for wind speed tends to increase. Conversely, the critical regions of the rain sensor coincide with periods of intense rain activity, indicating the identification of the most intense rain fall events.

Figure 6.15: The classified anomalous groups and critical regions on the actual values for pressure, rain and wind speed sensors

From all the critical regions, 9 are for pressure, 7 are for wind speed, and 2 are for rain. The timeline of their occurrence is given in Figure 6.16. The *low pressure system* (R1) physical event occurred during the winter and spring months of 2020, while summer and autumn were at relatively low (normal) levels. R1 was detected once in early-February and lasted 2 days, once in late-March (lasting 2 days), once in mid-April (lasting 1 day), once in early-May (lasting 5 days), and three times throughout December (lasting 4 days). The rain activity remained at relatively low (normal) levels in the spring, summer and autumn months of 2020, while in the winter the *rainstorm* (R2) physical event occurred three times in December (lasting 5 days) and coinciding with the three occurrences of the low pressure system as part of its preconditions.

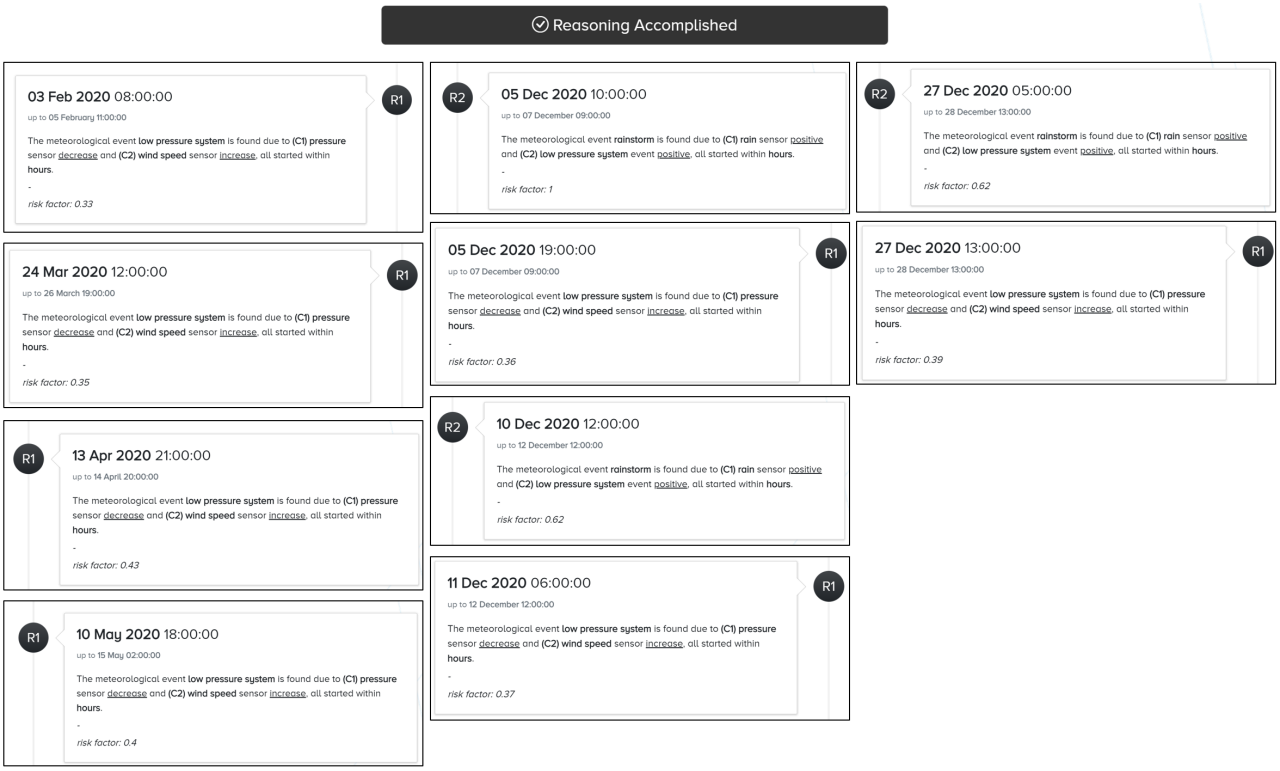


Figure 6.16: The timeline of the detected meteorological events (R1, R2)

Figure 6.11 presents the prediction errors across sensors, and based on these errors, the risk (Section 4.2.2) associated with the occurrences of meteorological events is also depicted in Figure 6.16. The risk of a *low-pressure system* varies from 0.33 to 0.43, whereas the risk of a *rainstorm* varies from 0.62 to the maximum risk of 1.0. That is because, especially during the early days of December, the prediction offset in rain activity was relatively higher than the prediction offset of wind speed and pressure activities.

Surface External Driver Events on Vulcano 2020

In 2020, the occurrence of the *low pressure system* meteorological event introduced anomalies involving decreases in surface temperature.

The critical regions of cold (red) and hot (grey) surface temperatures, as well as wind speed and pressure, are given as the root-causes diagram in Figure 6.17. Critical regions for surface temperature, wind speed and pressure are closely associated with each other, or overlap.

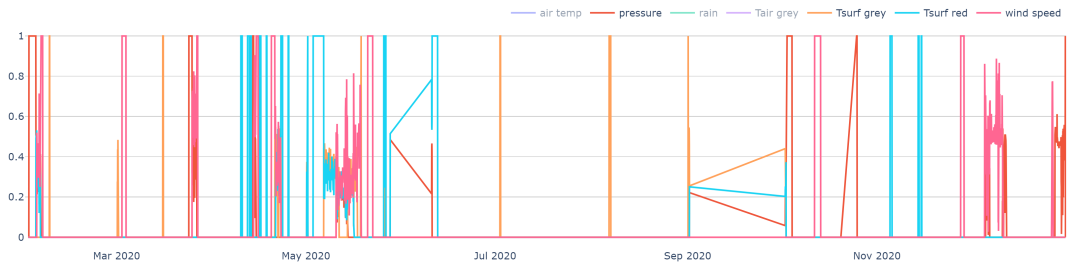


Figure 6.17: The root-causes diagram for pressure, wind speed and (red, grey) surface temperature sensors

The values of the surface temperatures on both cold and hot zones are positively correlated, as it is depicted in Figure 6.18. However, a partial correlation is observed between their anomalous values within critical regions, since the surface temperature of the hot zone is buffered to the external conditions by the internal driver, that its cooling is modulated by the effect of the hydrothermal system heat source. Thus critical regions are more likely to be associated with passage of a low pressure system for the cold zone than the hot zone.

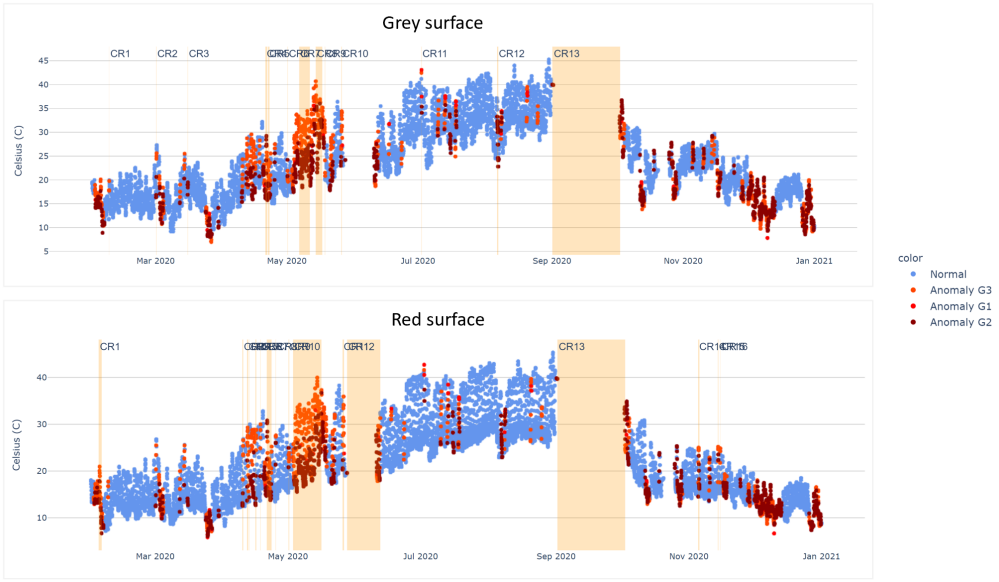


Figure 6.18: The classified anomalous groups and critical regions on the actual values for surface temperature sensors

Two critical regions were identified for surface temperature in both the hot zone and cold zone (Figure 6.19). Anomalies are confined to the winter and spring months, while summer and autumn are free of critical regions. An anomalous decrease in surface temperature in the cold zone (R5) is detected in early-February and lasted two days, although this did not effect the hot zone. An anomalous decrease in surface temperature was recorded for

both zones in mid-May, and lasted six days. During this period, the surface temperature at the hot zone began to decrease a few hours before the cold zone. This has implications for the effect of external drivers on the apparent thermal anomaly (ΔT).

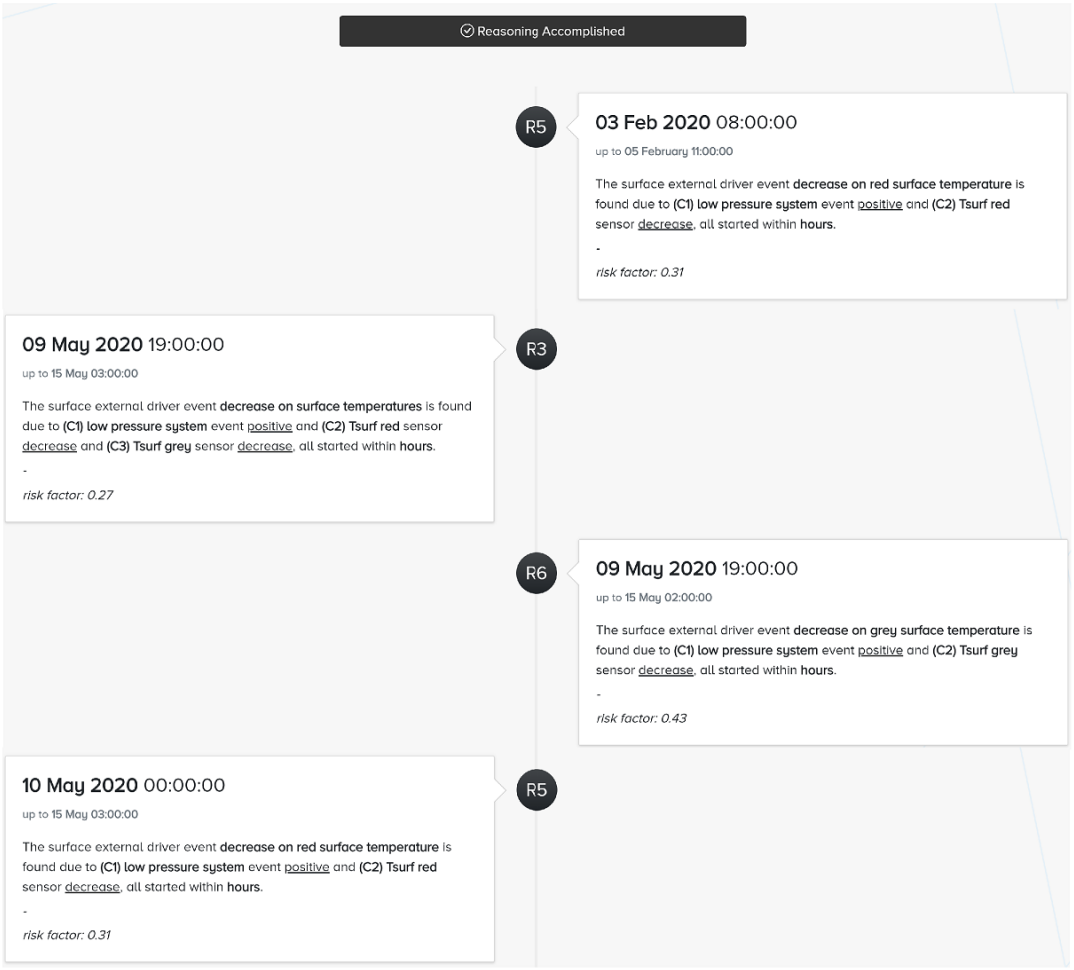


Figure 6.19: The timeline of the detected surface external driver events (R3, R5, R6)

Based on prediction errors presented in Figure 6.11, the risks associated with the occurrences of surface external driver events are also given in Figure 6.19. The risk on decrease of cold surface temperature (R5) is 0.31, while the risk on decrease of hot surface temperature (R6) is higher at 0.43. This observation suggests that the surface temperature changes in the hot zone pose a higher level of risk compared to the cold zone. In addition, the risk of decrease on both cold and hot zone surface temperatures (R3) is 0.27, which is lower because it considers all conditions from R5 and R6. This observation suggests that the temperature changes across the entire surface carry slightly less risk compared to isolated changes in either the cold zone or hot zone.

6.7 Timeline of External-Drivers

Aligned with the research question, the decrease in pressure and increase in wind speed, within a low-pressure system, drive to a decrease on surface temperature for both hot (grey) and cold (red) zones of the active hydrothermal system of Vulcano in the year 2020. The detected physical events have been checked as true positives, given in Table 6.9.

Rule ID	Physical Event	Average Risk	Verified
R1	low pressure system	0.38	✓
R2	rainstorm	0.75	✓
R3	decrease on surface temperatures	0.27	✓
R5	decrease on cold zone surface temperature	0.31	✓
R6	decrease on hot zone surface temperature	0.43	✓

Table 6.9: The detected meteorological and surface external driver events

In early-February, a low pressure system passed over Vulcano, persisting for two days. During this period, the cold zone experienced a notable decrease on surface temperature, indicating a response to the external driver in this zone, but not the hot zone. This drives the thermal anomaly upwards, but is a result of an external rather than an internal driver. A second low pressure system passed over Vulcano in late-March and lasted two days, was followed a one day-long period of low pressure system conditions in mid-April. In both of these cases, there was no impact on the surface temperatures in either cold or hot zones, meaning that thermal anomaly was unaffected. However, in early May, a low-pressure system persisted for approximately five days. It decreased the surface temperature in the hot zone and the cold zone, disrupting normality across the entire surface. Rainstorms were detected throughout December, and were associated heavy rainfall, strong winds and low pressure. However, surface temperatures retained their normality in both the cold and hot zones. This means that, even when at low, baseline levels, external drivers have a minimal role on influencing surface temperature in the hot and cold zones, and hence also the apparent thermal anomaly.

Sensor Name	Unused Critical Regions
pressure	2
wind speed	4
rain	4
tsurf-grey	11
tsurf-red	14

Table 6.10: The number of critical regions not associated with the executed physical events

Table 6.10 provides the number of the remaining critical regions, which are not associated with anomalous meteorological events. The presence of unused critical regions suggests that there might be additional events to

explore, prompting the creation of new rules. For instance, in Figure 6.20 one can discern that the critical regions on both air and surface temperatures are coupled over periods ranging from a few hours to several days. However, this may also imply the existence of false positives. To mitigate this, the model can be retrained on more sensor values or to minimize potential inaccuracies by estimating the significant missing gaps such the one started in early-September up to early-October.

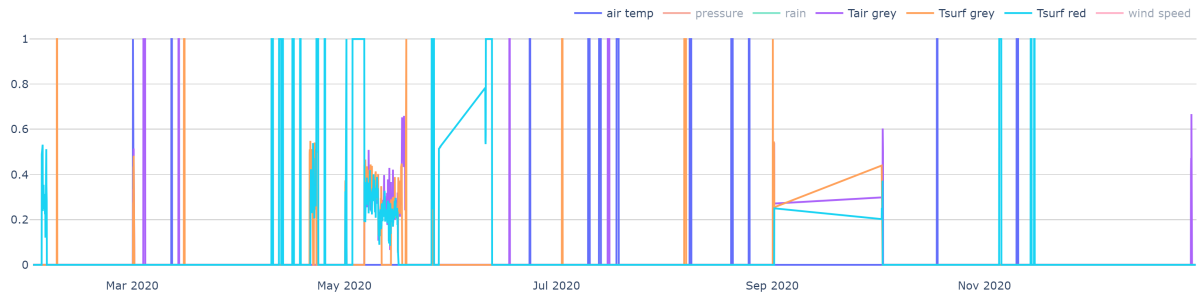


Figure 6.20: The root-causes diagram of all temperature sensors

Chapter 7

Conclusions and Future Work

In this thesis I proposed the DITAN, a domain agnostic framework for the detection and interpretation of temporal-based anomalies. First, I lightened the requirements of contamination in multivariate time series, demonstrating an anomalous exploratory space and addressing the limitations of the current literature on their anomalous and temporal support. I have provided qualitative evidence that no related work can support all the requirements, in contrast to DITAN, which successfully address all of them. Furthermore, I conducted a quantitative evaluation of DITAN in a comparative environment to assess its detection capabilities. Additionally, in a real-world application, I demonstrated the physical interpretation of the detected anomalies using expert domain knowledge. In unsupervised environment, where no labels are required, the method instantiates a neural network model based on Encoder-Decoder architecture with implicit/explicit attention and adjustable layers/units to capture normality as regular patterns over records. Anomalous records are then detected by introducing a dynamic thresholding methodology that reveals critical regions to their errors sequence. Using detected anomalies, root cause is examined on their data space and similarities are seen in their units space using a clustering method. In addition, by leveraging domain-specific knowledge using a knowledge (expert) system, we were able to characterize the detected anomalies into physical events.

7.1 Insights

DITAN is assessed on the well-known MSL and SMAP real-world datasets. From these, 6 multivariate channels are selected to cover different (point, contextual and joint) contamination types at a varying duration. To highlight the generic nature of the proposed framework, an optimizer is used to automatically configure the hyper-parameters of a model per channel. The proposed models are capable of predicting normality with high tolerance to overfitting, dominating against to the original ones [20] . Critical regions are well aligned to all anomalous events (IoU > 0.5), from which point anomalous records ($F_{0.5} \approx 0.8$) are captured slightly better than contextual anomalies ($F_{0.5} \approx 0.6$).

In total, from the 12 anomalous events only 2 are masked, while the rest are detected with more than 70% precision. Moreover, a good clustering (DBI < 0.19) is applied to the units space of the detected anomalous records, with more than 85% of them to be verified similar also in their actual data space. This provides support for the numerical interpretation choices we made. Thereafter, a comparative evaluation on MBA dataset against 11 other deep model models demonstrated that our approach achieved the highest precision (99.1%) on discovering anomalies. While a lower recall (77.9%) suggests that more than the 70% of the duration in each actual anomaly is captured. Finally, in a practical real-world application on Vulcano in the year 2020, during which the internal hydrothermal system was stable, DITAN physically interpreted 10 occurrences of two meteorological events and 4 occurrences of three surface external driver events. The occurrences of the physically interpreted events were verified with respect to the actual sensor values. Specifically, external factors were observed to drive a decrease on surface temperature in the following zones: hot zone at risk 0.43, cold zone at risk 0.31, and joint (both hot and cold) zones at risk 0.27.

7.2 Limitations

Although Table 1.4 outlines the advantages and disadvantages of DITAN's technical specifications, this section aims to highlight the limitations specifically related to the assumptions of DITAN. A major limitations of the unsupervised approach, is that a poor data quality can lead to misconception of the modeled normality. This requires a good understanding of how normality is present in the training sequence, so that abnormal patterns are not regular enough to be considered normal. Moreover, normality is sensitive to the context size which controls the temporal scalability. Particularly, anomalous events that last longer than context size, may be partially detected or in extreme case (saw in experiments) considered normal. In summary, DITAN is sensitive to data quality issues and the context size. Having a well-defined notion of normality in the training data and an optimal context size, are important to maximize DITAN's ability to model and detect/interpret anomalies. Besides that, DITAN's domain independence is associated with the capability of choosing a suitable hyperparameter configuration, which can be challenging given the significant number of hyper-parameters involved.

7.3 Future Research

I investigate to reduce the number of hyper-parameters in the existing configurations of DITAN to enhance the efficiency of the optimization process, while still ensuring effectiveness. For instance, the *units-decay* hyperparameter may be redundant since the *units* already exists. In addition, I plan to extend the DITAN by incorporating a module that predicts future anomalies. This involves training an additional model in supervised manner, using a series of records labeled as normal or abnormal by the existing detection module. However, methodological concerns are raised regarding the reliability of the labeling process and the sufficient number of anomalies to establish meaning-

ful anomalous patterns for predicting future occurrences. The overall project was to develop a tool for assessing internal and external drivers to the thermal surface state at a hydrothermal system, into the following objectives:

- consider period when internal and external drivers play a role
- consider two measurements that characterize the internal drivers (soil temperature and seismicity)
- define all categories of internal and external drivers
- characterize time scales and intensities of anomalies associated with each driver
- provide thresholds above and below which any given driver does and does not have an effect on ΔT
- Assess the ability to provide a forecasting tool to allow to understand whether a developing thermal anomaly is associated with an external or internal driver.

DITAN identified anomalies on the surface temperature of the hot and cold zones driven by external factors in the baseline year 2020 of Vulcano. To address the other objectives, I will apply DITAN in the unrest year 2021 of Vulcano, as illustrated in Figure 6.4. DITAN needs to be retrained using the measurements from 2020, including trend and seasonality, in addition to residuals. The retraining process will use similar context (24 records) and horizon (6 records) windows. The testing phase will exclusively use data from 2021. The aim of this analysis is to identify the difference in trend between the surface temperatures of 2020 and 2021 and, thus, to detect anomalies on the surface temperature of Vulcano in the year 2021 with respect to internal factors.

Appendix A

The knowledge base (KB) of DITAN, consists of relational tables built using the structured query language (SQL). The proposed KB schema, is coded as follows:

```
DROP DATABASE IF EXISTS ditankb;
CREATE DATABASE ditankb;
USE ditankb;
DROP TABLE IF EXISTS kb_conditions;
DROP TABLE IF EXISTS kb_rules;
DROP TABLE IF EXISTS kb_sensors;
--       RULE TABLE         --
CREATE TABLE kb_rules(
        r_id int NOT NULL,
        r_description text,
        r_executable ENUM('on', 'off') NOT NULL,
        r_license varchar(255) NOT NULL,
        r_version float NOT NULL,
        r_type ENUM('meteorological',
 'surface_external_driver',
 'surface_internal_driver',
 'instrumental') NOT NULL,
        r_temporal_resolution ENUM('',
 'minutes',
 'hours',
 'days',
 'weeks') NOT NULL,
        r_user_update varchar(255),
```

```
        r_update_time varchar(255),
        PRIMARY KEY (r_id)
);
--      CONDITIONS TABLE        --
CREATE TABLE kb_conditions(
        r_id int NOT NULL,
        c_id int NOT NULL,
        c_type ENUM('sensor', 'event') NOT NULL,
        c_name varchar(255) NOT NULL,
        c_state_operator ENUM('is', 'not') NOT NULL,
        c_state ENUM('increase',
 'decrease',
 'positive',
 'tare',
 'missing_values') NOT NULL,
        c_output ENUM('no', 'yes') NOT NULL,
        FOREIGN KEY (r_id) REFERENCES kb_rules(r_id),
        PRIMARY KEY (r_id, c_id)
);

--      AVAILABLE SENSORS       --
CREATE TABLE kb_sensors(
        s_name varchar(255) NOT NULL,
        s_unit varchar(255),
        s_info varchar(255),
        PRIMARY KEY (s_name)
);
INSERT INTO kb_sensors(s_name, s_unit, s_info)
VALUES
('pressure', 'Pascal (Pa)', ''),
('gust_speed', 'Miles per Hour (mph)', ''),
('air_temp', 'Celsius (C)', ''),
('humidity', 'Relative Humidity (RH)', ''),
```

```
('wind_speed', 'Miles per Hour (mph)', ''),
('rain', 'Centimeters (cm)', ''),
('Tsurf_grey', 'Celsius (C)', ''),
('Tair_grey', 'Celsius (C)', ''),
('Tsurf_red', 'Celsius (C)', ''),
('Tair_red', 'Celsius (C)', '');
```

Appendix B

The DITAN framework, is written in Python. It is compiled using Python version 3.8.0, while the libraries (packages) utilized are the following:

Package	Version
Flask	2.0.3
google-auth	1.35.0
google-auth-oauthlib	0.4.6
joblib	1.3.1
keras-nightly	2.5.0.dev2021032900
Keras-Preprocessing	1.1.2
kneed	0.6.0
matplotlib	3.2.1
mysql-connector-python	8.0.20
numpy	1.19.5
oauthlib	3.2.2
pandas	1.1.0
pip	19.2.3
plotly	4.12.0
protobuf	3.12.4
PyYAML	5.3.1
requests	2.31.0
requests-oauthlib	1.3.1
scikit-learn	0.23.2
scikit-optimize	0.8.0
scipy	1.5.1
statsmodels	0.12.1
tensorboard	2.5.0
tensorboard-data-server	0.6.1
tensorboard-plugin-wit	1.8.1
tensorflow	2.5.0
tensorflow-estimator	2.5.0

Bibliography

[1] J. Audibert et al. Do deep neural networks contribute to multivariate time series anomaly detection? *Pattern Recognition*, 132, December 2022.

[2] J. Audibert et al. USAD: unsupervised anomaly detection on multivariate time series. In R. Gupta et al. editors, *KDD '20: The 26th ACM Conf. on Knowledge Discovery and Data Mining, August 23-27*, : 3395–3404, 2020.

[3] D. Bahdanau et al. Neural machine translation by jointly learning to align and translate. In Y. Bengio et al., editors, *3rd ICLR'2015, San Diego, USA, May 7-9*, 2015.

[4] M. Abul Bashar and R. Nayak. Tanogan: Time series anomaly detection with generative adversarial networks. In *2020 IEEE Symposium Series on Computational Intelligence, 2020, Australia, December 1-4, 2020*: 1778–1785. IEEE, 2020.

[5] J. Bayer. *Learning Sequence Representations*. PhD thesis, Technical Univ. Munich, 2015.

[6] L. Bergman and Y. Hoshen. Classification-based anomaly detection for general data. In *8th Int. Conf. on Learning Representations, Addis Ababa, Ethiopia, April 26-30*, 2020.

[7] V.A. Bharadi and S.S. Alegavi. *Using Luong and Bahdanau Attention Mechanism on the Long Short Term Memory Networks - A COVID-19 Impact Prediction Case Study*: 1–25. 2020.

[8] A. Blázquez-Garcí et al. A review on outlier/anomaly detection in time series data. *ACM Comput. Surv.*, 54(3)1–56, 2021.

[9] X. Chen et al. DAEMON: unsupervised anomaly detection and interpretation for multivariate time series. In *37th IEEE Int. Conf. on Data Engineering, Chania, April 19-22*: 2225–2230, 2021.

[10] D. Davies and D. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(2):224–227, 1979.

[11] S. Du et al. Multivariate time series forecasting via attention-based encoder-decoder framework. *Neurocomputing*, 388:269–279, 2020.

[12] Deng, A. and Hooi, B. Graph Neural Network-Based Anomaly Detection in Multivariate Time Series. *35th AAAI Conf. On Artificial Intelligence, Feb. 2-9*. pp. 4027-4035, 2021

[13] J. Friedman et al. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 12 2007.

[14] A. Geiger et al. Tadgan: Time series anomaly detection using generative adversarial networks. In X. Wu, et al. editors, *IEEE Int. Conf. on Big Data, Big Data 2020, Atlanta, USA, December 10-13, 2020*: 33–43, 2020.

[15] R. Giri et al. Self-supervised classification for detecting anomalous sounds. In *Detection and Classification of Acoustic Scenes and Events Workshop 2020*, 2020.

[16] I. Golan and R. El-Yaniv. Deep anomaly detection using geometric transformations. In S. Bengio et al. editors, *Annual Conf. on Neural Information Processing Systems 2018 December 3-8, Montréal*, : 9781–9791, 2018.

[17] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[18] R. Hsieh et al. Unsupervised online anomaly detection on multivariate sensing time series data for smart manufacturing. In *12th IEEE Conf. on Service-Oriented Computing and Applications,Kaohsiung, November 18-21*: 90–97, 2019.

[19] Y. Huang et al. Cross-feature analysis for detecting ad-hoc routing anomalies. In *23rd Int. Conf. on Distributed Computing Systems 19-22 May, Providence, USA*, page 478, 2003

[20] K. Hundman et al. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In Y. Guo and F. Farooq, editors, *Proc. of the 24th Int. Conf. on Knowledge Discovery & Data Mining, 2018, London, August 19-23*: 387–395. ACM, 2018.

[21] W. Jiang et al. A gan-based anomaly detection approach for imbalanced industrial time series. *IEEE Access*, 7:143608–143619, 2019.

[22] Y. Jiao et al, D. TimeAutoAD: Autonomous Anomaly Detection With Self-Supervised Contrastive Loss for Multivariate Time Series. *IEEE Trans. Netw. Sci. Eng.*. **9**, 1604-1619, 2022

[23] F. Khoshnevisan and Z. Fan. RSM-GAN: A convolutional recurrent GAN for anomaly detection in contaminated seasonal multivariate time series. *CoRR*, abs/1911.07104, 2019.

[24] T. Kieu et al. Outlier detection for time series with recurrent autoencoder ensembles. In S. Kraus, editor, *Proc of the 28th Int. Joint Conf. on Artificial Intelligence, Macao, August 10-16*, : 2725–2732, 2019.

[25] C.-Ki Lee et al. Studies on the gan-based anomaly detection methods for the time series data. *IEEE Access*, 9:73201–73215, 2021.

[26] D. Li et al. MAD-GAN: multivariate anomaly detection for time series data with generative adversarial networks. In I. V. Tetko et al. editors, *Artificial Neural Networks and Machine Learning, Munich, Germany, September 17-19*, LNCS, 11730: 703–716. Springer, 2019.

[27] L. Li et al. Anomaly detection of time series with smoothness-inducing sequential variational autoencoder. *IEEE Trans. Neural Networks Learn. Syst.*, 32:1177–1191, 2021.

[28] Y. Liu et al. Self-adversarial variational autoencoder with spectral residual for time series anomaly detection. *Neurocomputing*, 458:349–363, 2021.

[29] Z. Liu et al. A regularized LSTM method for predicting remaining useful life of rolling bearings. *Int. J. Autom. Comput.*, 18(4):581–593, 2021.

[30] W. Lu et al. Unsupervised sequential outlier detection with deep architectures. *IEEE Trans. Image Process.*, 26(9):4321–4330, 2017.

[31] T. Luong et al. Effective approaches to attention-based neural machine translation. In L. Màrquez et al. editors, *Proc. of the 2015 Conf. on Empirical Methods in Natural Language Processing, Lisbon, September 17-21*: 1412–1421, 2015.

[32] S. Maleki et al. Unsupervised anomaly detection with LSTM autoencoders using statistical data-filtering. *Appl. Soft Comput.*, 108:107443, 2021.

[33] P. Malhotra et al. Long short term memory networks for anomaly detection in time series. In *23rd ESANN Conf 2015, Bruges, April 22-24*, 2015.

[34] H. B. Mann and D. R. Whitney. On a test of whether one of two random variables is stochastically larger than the other. *Annals of Mathematical Statistics*, 18:50–60, 1947.

[35] E. Marchi et al. Non-linear prediction with LSTM recurrent neural networks for acoustic novelty detection. In *Int. Joint Conf. on Neural Networks, Killarney, July 12-17*: 1–7, 2015.

[36] M. Memarzadeh et al. Unsupervised anomaly detection in flight data using convolutional variational auto-encoder. *Aerospace*, 7(8), 2020.

[37] M. Munir et al. Deepant: A deep learning approach for unsupervised anomaly detection in time series. *IEEE Access*, 7:1991–2005, 2019.

[38] T. Nakamura et al, MERLIN: Parameter-Free Discovery of Arbitrary Length Anomalies in Massive Time Series Archives. *IEEE Int. Conf. On Data Mining* . 1190-1195, 2020

[39] Z. Niu et al. Lstm-based VAE-GAN for time-series anomaly detection, *Sensors*, 20:3738, 2020.

[40] G. Pang et al. Deep learning for anomaly detection: A review. *ACM Comput. Surv.*, 54:1–38, 2021.

[41] J. Pereira and M. Silveira. Unsupervised anomaly detection in energy time series data using variational recurrent autoencoders with attention. In M. Arif Wani, et al, editors, *17th IEEE Int. Conf. on Machine Learning and Applications, Orlando, USA, December 17-20*: 1275–1282, 2018.

[42] J. Pereira and M. Silveira. Learning representations from healthcare time series data for unsupervised anomaly detection. In *IEEE Int. Conf. on Big Data and Smart Computing, BigComp 2019, Kyoto, Japan, February 27 - March 2*: 1–7, 2019.

[43] O. Provotar et al. Unsupervised anomaly detection in time series using lstm-based autoencoders. In *IEEE Int. Conf. on Adv Trends in Information Theory*: 513–517, 2019.

[44] Y. Su et al. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In A. Teredesai et al. editors, *Proc. 25th ACM SIGKDD Int. Conf. on Knowledge Discovery & Data Mining, Anchorage, USA, August 4-8*: 2828–2837, 2019.

[45] Y. Sun et al. Time series anomaly detection based on GAN. In M. A. Alsmirat and Y. Jararweh, editors, *6th Int. Conf. on Social Networks Analysis, Management and Security, Granada, Spain, October 22-25*: 375–382, 2019.

[46] Y. Tan et al. An encoder-decoder based approach for anomaly detection with application in additive manufacturing. In M. Arif Wani et al. editors, *18th IEEE Int. Conf. On Machine Learning And Applications, Boca Raton, FL, December 16-19*: 1008–1015, 2019.

[47] L. Tenenboim-Chekina et al. Ensemble of feature chains for anomaly detection. In Z. Zhou et al. editors, *Multiple Classifier Systems, 11th Int. Workshop, Nanjing, China, May 15-17, 2013*, Vol. 7872 of *LNCS*: 295–306. Springer, 2013.

[48] S. Wang et al. Effective end-to-end unsupervised outlier detection via inlier priority of discriminative network. In H. M. Wallach, et al. editors, *Annual Conf. on Neural Information Processing Systems 2019, December 8-14, Vancouver, Canada* : 5960–5973, 2019.

[49] P. Wu et al. Unsupervised anomaly detection for underwater gliders using generative adversarial networks. *Eng. Appl. Artif. Intell.*, 104:104379, 2021.

[50] R. Wu and E. J. Keogh. Current time series anomaly detection benchmarks are flawed and are creating the illusion of progress. In *Proc. of IEEE ICDE*, 2022.

[51] C. Yin et al. Anomaly detection based on convolutional recurrent autoencoder for iot time series. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*: 1–11, 2020.

[52] C. Zhang et al. Velc: A new variational autoencoder based model for time series anomaly detection. *CoRR*. abs/1907.01702, 2020

[53] C. Zhang et al. A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. In *The 9th AAAI Symposium on Educational Advances in Artificial Intelligence, USA, January 27 - February 1*: 1409–1416. AAAI Press, 2019.

[54] K. Zhang et al. Federated variational learning for anomaly detection in multivariate time series. In *IEEE Int. Performance, Computing, and Communications Conf.*: 1–9. IEEE Computer Society, oct 2021.

[55] J. Zhao et al. One-step predictive encoder - gaussian segment model for time series anomaly detection. In *Int. Joint Conf. on Neural Networks, Glasgow, United Kingdom, July 19-24, 2020*: 1–7 2020.

[56] B. Zhou et al. Beatgan: Anomalous rhythm detection using adversarially generated time series. In Sarit Kraus, editor, *Proc of the 28th Int. Joint Conf. on Artificial Intelligence, Macao, China, August 10-16*: 4433–4439, 2019.

[57] H. Zou and T. J. Hastie. Regularization and variable selection via the elastic net. *Journal of The Royal Statistical Society Series B-statistical Methodology*, 67:301–320, 2005.

[58] S. Tuli., G. Casale, & N. Jennings, TranAD: Deep Transformer Networks for Anomaly Detection in Multivariate Time Series Data. *Proc. VLDB Endow.*. **15**, 1201-1214, 2022

[59] B. Zong et al., Deep Autoencoding Gaussian Mixture Model for Unsupervised Anomaly Detection. In *6th Int. Conf. on Learning Representations, Vancouver, April 30-May 3, 2018*

[60] H. Zhao, Y. Wang., J. Duan, C. Huang, D. Cao., Y. Tong , B. Xu, J. Bai, J. Tong & Q. Zhang, Multivariate Time-series Anomaly Detection via Graph Attention Network. *CoRR*. abs/2009.02040, 2020

[61] Y. Zhang, Y. Chen, J. Wang, Z. & Pan, Unsupervised Deep Anomaly Detection for Multi-Sensor Time-Series Signals. *CoRR*. abs/2107.12626, 2021

[62] Giannoulis, M., Kondylakis, H. & Marakakis, E. Designing and implementing a collaborative health knowledge system. *Expert Syst. Appl.*. **126** pp. 277-294 (2019), https://doi.org/10.1016/j.eswa.2019.02.010

[63] Boniol, P., Paparrizos, J., Palpanas, T. & Franklin, M. SAND: Streaming Subsequence Anomaly Detection. *Proc. VLDB Endow.*. **14** pp. 1717-1729 (2021)

[64] Goldberger, A., Amaral, L., Glass, L., Hausdorff, J., Ivanov, P., Mark, R., Mietus, J., Moody, G., Peng, C. & Stanley, H. PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals.. *Circulation*. **101 23** pp. E215-20 (2000)

[65] Hebbal, A., Brevault, L., Balesdent, M., Talbi, E. & Melab, N. Bayesian optimization using deep Gaussian processes with applications to aerospace system design. *Optimiztion And Engineering*. **22** pp. 321-361 (2020)

[66] D.Comaniciu & Meer, P. Mean Shift: A Robust Approach Toward Feature Space Analysis.. *IEEE Trans. Pattern Anal. Mach. Intell.*. **24**, 603-619 (2002)

[67] Shipmon, D., Gurevitch, J., Piselli, P. & Edwards, S. Detection of anomalous drops with limited features and sparse examples in noisy highly periodic data. *CoRR*. **abs/1708.03665** (2017)

[68] Doane, D. Aesthetic Frequency Classifications. *American Stat.* . **30** pp. 181-183 (1976)

[69] Amarbayasgalan, T., Pham, V., Theera-Umpon, N. & Ryu, K. Unsupervised Anomaly Detection Approach for Time-Series in Multi-Domains Using Deep Reconstruction Error. *Symmetry*. **12**, 1251 (2020)

[70] Sauro, J. & Lewis, J. Correlations among prototypical usability metrics: evidence for the construct of usability. *Proceedings Of The 27th International Conference On Human Factors In Computing Systems, CHI 2009, Boston, MA, USA, April 4-9, 2009*. pp. 1609-1618 (2009), https://doi.org/10.1145/1518701.1518947

[71] Brooke, J. SUS—a quick and dirty usability scale. 1996. *Usability Eval Ind*. **189**, 4-7 (1996)

[72] Likert, R. A technique for the measurement of attitudes. *Archives Of Psychology*. (1932)

[73] ISO/IEC DIS 25023 Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) *Measurement of system and software product quality*. (2016)

[74] Sanders, F. The Meteorological "Bomb"; An Explosive Maritime Cyclone. *OCEANS 1984*. pp. 318-323 (1984)

[75] Haque, E., Tabassum, S. & Hossain, E. A Comparative Analysis of Deep Neural Networks for Hourly Temperature Forecasting. *IEEE Access*. **9** pp. 160646-160660 (2021), https://doi.org/10.1109/ACCESS.2021.3131533

[76] Segura-Delgado, A., Gacto, M., Alcalá, R. & Alcalá-Fdez, J. Temporal association rule mining: An overview considering the time variable as an integral or implied component. *WIREs Data Mining Knowl. Discov.*. **10** (2020), https://doi.org/10.1002/widm.1367

[77] Vardell, E. & Moore, M. Isabel, a clinical decision support system. *Medical Reference Services Quarterly*. **30**, 158-166 (2011)

[78] Leung, K. & Lam, W. Fuzzy concepts in expert systems. *Computer*. **21**, 43-56 (1988)

[79] Shortliffe, E. Books: Computer-Based Medical Consultations: MYCIN. *Journal Of Clinical Engineering*. **1**, 69 (1976)

[80] Giannoulis, M., Harris, A. & Barra, V. DITAN: A deep-learning domain agnostic framework for detection and interpretation of temporally-based multivariate ANomalies. *Pattern Recognition*. **143** pp. 109814 (2023)

[81] Harris, A. Thermal remote sensing of active volcanoes: a user's manual. (Cambridge university press,2013)

[82] Bonneville, A. & Gouze, P. Thermal survey of Mount Etna volcano from space. *Geophysical Research Letters*. **19**, 725-728 (1992)

[83] Keszthelyi, L., Harris, A. & Dehn, J. Observations of the effect of wind on the cooling of active lava flows. *Geophysical Research Letters*. **30** (2003)

[84] Harris, A., Lodato, L., Dehn, J. & Spampinato, L. Thermal characterization of the Vulcano fumarole field. *Bulletin Of Volcanology*. **71** pp. 441-458 (2009)

[85] Mannini, S., Harris, A., Jessop, D., Chevrel, M. & Ramsey, M. Combining ground-and ASTER-based thermal Measurements to Constrain fumarole field heat budgets: The case of Vulcano Fossa 2000–2019. *Geophysical Research Letters*. **46**, 11868-11877 (2019)

[86] Harris, A., Alparone, S., Bonforte, A., Dehn, J., Gambino, S., Lodato, L. & Spampinato, L. Vent temperature trends at the Vulcano Fossa fumarole field: the role of permeability. *Bulletin Of Volcanology*. **74** pp. 1293-1311 (2012)

[87] Pailot-Bonnétat, S., Rafflin, V., Harris, A., Diliberto, I., Ganci, G., Cappello, A., Boudoire, G., Bilotta, G., Grassa, F., Gattuso, A. & Others Anatomy of thermal unrest at a hydrothermal system: Case study of the 2021-2022 crisis at Vulcano. (2023)

[88] Carapezza, M., Nuccio, P. & Valenza, M. Genesis and evolution of the fumaroles of Vulcano (Aeolian Islands, Italy): a geochemical model. *Bulletin Volcanologique*. **44** pp. 547-563 (1981)

[89] Chiodini, G., Cioni, R., Marini, L. & Panichi, C. Origin of the fumarolic fluids of Vulcano Island, Italy and implications for volcanic surveillance. *Bulletin Of Volcanology*. **57** pp. 99-110 (1995)

[90] Nuccio, P., Paonita, A. & Sortino, F. Geochemical modeling of mixing between magmatic and hydrothermal gases: the case of Vulcano Island, Italy. *Earth And Planetary Science Letters*. **167**, 321-333 (1999)