FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

GAN-based Anomaly Detection in Solar Wind Profiles

Ivo Saavedra

Mestrado em Engenharia Informática e Computação

Supervisor: André Restivo

Co-Supervisor: Filipa Barros

May 2, 2023

GAN-based Anomaly Detection in Solar Wind Profiles

Ivo Saavedra

Mestrado em Engenharia Informática e Computação

May 2, 2023

Resumo

A Ciência do Clima Espacial é um campo de pesquisa vital que visa compreender as condições na superfície do Sol, que podem afetar negativamente a vida na Terra. Apesar de ser um campo bem desenvolvido, as condições que levam a essas fenómenos nefastos ainda não são totalmente compreendidas. Esta limitação é principalmente atribuída à dificuldade em obter dados de alta qualidade da superfície do Sol.

De forma a contornar este problema, alguns modelos de simulação tentam extrapolar as condições no Sol analisando dados de outras medições. Um exemplo disso é o MULTI-VP que usa magnetogramas de várias fontes (por exemplo, Wilcox Solar Observatory) e determina a estrutura do campo magnético de fundo. No entanto, essas simulações demoram muito tempo a convergir e requerem estimativas iniciais de especialistas, feitas manualmente. Recentemente, uma abordagem baseada em Machine Learning foi projetada para atenuar esses problemas. Esta removeu a necessidade de estimativas iniciais, prevendo automaticamente as condições iniciais da simulação. Além disso, provou que pode haver uma redução significativa no tempo de execução do simulador. Apesar disso, os modelos de previsão ainda não são robustos o suficiente para serem usados em aplicações do mundo real. Como em muitos outros problemas de Machine Learning, acreditamos que a presença de outliers no conjunto de dados de treinamento tenha prejudicado a sua performance.

Podemos classificar os outliers como dados fora da distribuição, anomalias, novidades e desvios. Estes podem ser causados por falhas de sistema, ruído de sensor, erros humanos e operações maliciosas. Mais recentemente, GANs têm provado ser métodos eficazes de deteção de anomalias de maneiras não supervisionadas.

Com isto, esta dissertação propõe uma abordagem baseada em GANs para deteção de outliers em perfis de vento solar que depois serão usados para treinar um modelo de previsão. O impacto do novo método será avaliado em relação aos resultados obtidos através de um modelo de previsão treinado com dados não normalizados.

Vários métodos de deteção de anomalias baseados em GANs foram estudados para determinar qual se adequava melhor ao problema. Com as lições aprendidas neste passo, propusemos uma metodologia que incorpora GANs para deteção de amostras de perfis de vento solar anómalos nos dados usados para treinar modelos para estimativa de condições iniciais. Um método de treino foi projectado, juntamente com um mecanismo de detecção de anomalias. Os critérios de avaliação da metodologia proposta também foram definidos.

Abstract

Space Weather Science is a vital field of research that aims to understand the conditions on the Sun's surface, which can negatively impact life on Earth. Despite being a well-developed field, the conditions that lead to these nefarious phenomena are still not fully understood. This limitation is mainly attributed to the difficulty in acquiring high-quality data from the Sun's surface.

To circumvent this issue, some simulation models try to extrapolate the conditions on the Sun by analyzing data from other measurements. An example of this is MULTI-VP which uses magnetograms from various sources (e.g., Wilcox Solar Observatory) and determines the structure of the background magnetic field. However, these simulations take a long time to converge and require initial expert estimations, which are done by hand. Recently, a machine learning approach has been designed to attenuate these issues. It removed the need for initial estimates by automatically predicting the starting conditions of the simulation. In addition, it has proved that there can be a significant reduction in the execution time of the simulator. Despite this, the prediction models are still not robust enough to be used in real-world applications. Like many other machine learning problems, we believe the presence of outliers in the training dataset hampers its predictive performance.

We can classify outliers as out-of-distribution data, anomalies, novelties, and deviations. These can be caused by system failures, sensor noise, human errors, and malicious operations. More recently, GANs have proved to be effective unsupervised anomaly detection methods.

Thus, this dissertation aims to apply GANs to detect anomalies in solar wind profiles to measure their influence on the already developed prediction models. This will be done by taking advantage of the discriminative and generative abilities of GANs. In addition, we will also evaluate the impacts of the improved prediction models, trained with normalized datasets, on the performance of the simulation.

Several GAN-based anomaly detection methods were studied to determine which suited the problem. With the insights from this step, we proposed a methodology incorporating GANs for detecting anomalous solar wind profile samples in the data used to train models for initial condition estimation. A training method was presented along with an anomaly detection mechanism. Additionally, the evaluation criteria for the proposed methodology were also defined.

Keywords: Space Weather, Machine Learning, Anomaly Detection, Generative Adversarial Networks, Solar Wind

Contents

List of Figures

List of Tables

Abbreviations

AE	Autoencoder
ANN	Artificial Neural Network
AUC	Area Under the Curve
CME	Coronal Mass Ejection
CNN	Convolutional Neural Network
D	Discriminator
DNN	Deep Neural Network
E	Encoder
FP	False Positive
FN	False Negative
GAN	Generative Adversarial Network
G	Generator
LSTM	Long Short-Term Memory
ML	Machine Learning
MHD	Magnetohydrodynamics
NN	Neural Network
RNN	Recurrent Neural Network
SLR	Systematic Learning Review
SVM	Support Vector Machine
TP	True Positive
TN	True Negative
VAE	Variational Autoencoder
KL	Kullback-Leibler
KNN	K-Nearest Neighbors

Chapter 1

Introduction

The Sun releases a constant stream of particles and magnetic fields called *solar wind*. This stream consisting of high-velocity charged particles (e.g. protons and electrons) can reach planetary surfaces unless thwarted by an atmosphere, magnetic field, or both. In Earth's case, the magnetosphere and the atmosphere, to a smaller extent, block out most of the radiation emitted by the Sun. However, other more extreme events like solar flares and CMEs (Coronal Mass Ejections) can provoke negative effects on the Earth's surface and upper atmosphere.

The three main ways these events can affect the Earth are *radio blackouts* which mostly affect satellites and consequently geolocation systems and communication systems; *solar radiation storms* that can endanger astronauts and spacecraft orbiting the Earth; and probably the worst of all, *geomagnetic storms* which are known to have caused major disturbances in the past. The first significant example of this is the Carrington Event (1859) and, more recently, a geomagnetic storm that affected Quebec's power grid (1989)[1].

Space Weather Science is a field that aims to prevent the consequences of such events; however, the factors that result in their formation are still not fully understood. Some simulation models have been designed to try and fill this gap [1, 2], but they require initial expert guesses. Recently, an ML (Machine Learning) model has been developed [3] to make those initial predictions based on the input data. Like other ML problems, the quality of the predictions is very dependent on the quality of the training data.

1.1 Motivation

In 1859, Carrington recorded the first and largest solar flare in history, which is now commonly referred to as the Carrington event. This phenomenon was so extreme that it caused geomagnetic storms in unexpected latitudes and provoked fires on telegraph wires. He was also able to correlate the event with a geomagnetic storm that occurred several hours later. His work is considered to be the beginning of the science field of Space Weather [4].

[1] List of solar storms: https://en.wikipedia.org/wiki/List_of_solar_storms

The field of *Space Weather Science* emerged with the aim of understanding the formation of the phenomena that could affect Earth, in order to evaluate their effects and to create early warning systems. Despite this being a well-developed field of science, the correlations between the Sun's structures and these phenomena are not yet fully formulated and are mostly speculative. For instance, it is still unknown why the atmosphere of the Sun is considerately hotter than its surface. The main leading theory is that the magnetic field transports energy deep from the convection zone through the surface and up to the atmosphere. It is also posited that the magnetic fields on the surface sometimes collide with each other, provoking large explosions and therefore causing the atmosphere to heat even further. Another enigma is the acceleration of the solar wind (up to millions of miles) out of the corona. Some correlation between the magnetic field and solar wind acceleration has been found, however, the effect still remains a mystery.

The answer to questions like these can contribute to a greater understanding of the underlying processes of the Sun that influence solar weather. Consequently, it would become easier to predict future events that could impact the Earth, satellites, and space stations orbiting it. Thus far, these answers have been delayed by the technological limitations on measuring solar events. In 2018, NASA's Parker Solar [2] probe was launched on a mission to orbit the Sun's, in order to understand the acceleration of solar wind at the corona. More recently, the PUNCH[3] mission was launched to try and shed some light on the formation of the solar wind on the Sun's surface.

MHD (magnetohydrodynamic) simulators like MULTI-VP [1] and ENLIL [2], were developed in order to try and extrapolate coronal conditions from limited observations of solar events from probes and observatories. The execution of these simulations relies on initial estimations, usually performed by hand after an analysis of the data (a very time-consuming task). Additionally, it has been posited that good initial estimations have the potential to reduce the simulation's execution time significantly. The process of making the initial predictions as well as the extensive execution time of the simulations make it difficult to create early warning systems that can prevent the effects of solar events on Earth.

1.2 Problem Definition

The large volume of newly acquired data has made it increasingly difficult for a speedy analysis of all the available data. The sheer amount of data is becoming increasingly hard for researchers to process, especially on data that is connected to near-real-time utilization [5]. Machine learning has become one of the principal methods of evaluating the data efficiently for problems associated with space weather prediction. However, most deep learning models are very susceptible to large variations in the data that can severely decrease the performance of these models. The anomalies can originate from the instrument and detector noise, statistical noise from the small

[2]Parker Solar Probe: Humanity's First Visit to a Star https://www.nasa.gov/content/goddard/parker-solar-probe-humanity-s-first-visit-to-a-star

[3]NASA Selects Missions to Study Our Sun, Its Effects on Space Weather https://www.nasa.gov/press-release/nasa-selects-missions-to-study-our-sun-its-effects-on-space-weather

flux of photons, and external noise may include instrumentation jitter, stray starlight, and cosmic ray background [5].

Recently, a NN [3] was developed to perform initial estimations for solar wind profiles that would later be fed to MULTI-VP [1]. This reduced the time needed to generate the initial estimations required by the simulation, which were previously done by hand. However, the model suffered from a common problem in ML which is the existence of anomalous/outlier training data that hinders its prediction quality.

The problem that is addressed in this thesis is to try and improve the quality of the training data, which is then to be used for predicting initial conditions for solar wind formation.

1.3 Goals

The objective of this thesis is to take advantage of GANs' generative and discriminative abilities to detect anomalous input data in solar wind profiles. The anomalous input data will then be excluded before being passed to a NN model [3] used to generate initial predictions for solar wind formation. With this, we intend to improve the prediction of the NN model and, as a consequence, reduce the computation time that MULTI-VP [1] takes to generate feasible solutions.

The developed architecture should be able to generate new solar wind profiles similar to the real distribution and to correctly detect samples that fall out of that distribution.

1.4 Document Structure

This first chapter served to contextualize the problem that is being solved in this dissertation. The rest of the document is organized in the following manner:

- Chapter 2, explains the background needed to understand the current problems in the area of space weather science. Some concepts related to neural networks and GANs are also introduced to provide a basis for methods discussed in the remainder of the document.
- Chapter 3, provides an analysis o fthe current state-of-the-art methods for anomaly detection in tabular datasets.
- Chapter 4, goes into more depth on the problem this thesis is trying to solve and the approach that will be taken.
- Chapter 7 briefly describes the goals achieved during this dissertation.

Chapter 2

Background

In this chapter, a basic introduction to solar weather and the main events associated with it will be presented. Additionally, a brief explanation of the Machine Learning (ML) terms that are needed in the context of this dissertation will be provided.

2.1 Space Weather

"Space weather refers to the dynamic, highly variable conditions in the geospace environment, including those on the Sun, in the interplanetary medium, and in the magnetosphere-ionosphere-thermosphere system. Adverse changes in the near-Earth space environment can diminish the performance and reliability of both spacecraft and ground-based systems." ([6])

2.1.1 Solar Phenomena

The increasing dependence on technologies vulnerable to solar weather conditions has made it increasingly important to detect incidents, like the Carrington event [4], that would significantly damage assets on Earth beforehand. In this section, a brief introduction to these phenomena will be provided.

Sunspots. These structures consist of dark central regions (umbra), which are colder than the rest of the Sun's surface, and more luminous external regions (penumbra). Sunspots are known to be regions with strong magnetic fields (1000 times stronger than in the surrounding normal surface). It is theorized that these magnetic fields interfere with the convection of the Sun, effectively cooling the regions where they appear. Sunspots often originate as groups concentrated in specific areas of the Sun, and their frequency and size vary with the 11-year solar cycle [7].

Coronal Mass Ejections (CME). These events are best described as mass ejections of plasma into space after solar eruptions. It is posited that their formation occurs mainly from magnetic reconnection, which occurs when magnetic field lines collide and realign into a new configuration (releasing large amounts of energy). After their formation, CMEs can expand through space to

great distances and collide with planetary atmospheres. The effects on Earth include geomagnetic storms, damage to electronics on orbiting satellites, endangerment of astronauts or planes, damage to electrical grids and disruption of radio communication. Due to their length (at least 0.25AU), CMEs can take over a day to pass Earth. While slower CMEs can take days before reaching Earth, the fastest ones arrive in approximately 15-18 hours. [8, 7].

Solar Flares. These events are often characterized as intense and temporary releases of energy that blast large amounts of charged particles into space. Flares are known to last only a few minutes and reach temperatures of 100 million K, much higher than the ones at the core of the Sun. Like CMEs, flare formations are associated with energy releases from magnetic reconnection and are primarily concentrated in the Sun's active regions. Flares are almost always associated with CMEs, but can also occur separately from them. Flares can be classified as A, B, C, M or X based on the X-ray flux measurements on Earth by the GOES spacecraft[1] [7, 8].

Solar Wind. This phenomenon results from plasma's constant expulsion and expansion into interplanetary space. More concretely, the solar wind consists of mostly protons, helium nuclei and electrons that move away from the Sun at supersonic speeds and carry the Sun's magnetic field with it. It streams away from the Sun at different velocities, which allows for it to be classified as fast (700 to 750 kms^{-1}) or slow (300 to 400 kms^{-1}). The latter usually occurs on the Sun's equatorial line, and the former is concentrated in open magnetic field regions of the Sun. The exact originating factors for the slow solar wind are still unknown; however, for fast solar wind, it is known that its origin is coronal holes. The solar wind has as properties (at 1 AU) a velocity of 400 kms^{-1}, a temperature of 1 million K, and a density of 5 particles cm^{-3} [7, 8].

2.1.2 Magnetohydrodynamic Simulation Models

The reasons for the acceleration of solar wind are mainly attributed to thermal heating; however, this does not explain the high speeds it reaches. The additional acceleration is often attributed to the magnetic field, but no physical model can currently explain the correlation. Similarly, the origins of the solar wind are mostly unknown, especially for slow solar wind compared to fast solar wind [8].

These difficulties are mostly attributed to the absence of sensitive, high-resolution coronal magnetic field measurements that do not allow for a full explanation of coronal physics. The limitations in this field make it more challenging to comprehend solar events like CMEs and the acceleration of the solar wind [9].

To try and fill these gaps in Space Weather Science, several magnetohydrodynamic (MHD) models have been developed to try and give an answer to these questions. These models compute numerically intensive problems based on MHD equations. For this, they require the definition of appropriate boundaries and initial state definitions. Due to their complexity, MHD models often

[1] GOES overview and history https://www.nasa.gov/content/goes-overview/index.html

focus on single events and introduce assumptions and simplifications for the surrounding phenomena. For this reason, the research community has developed relatively simple MHD models to describe complex processes in the past decades.

2.2 Neural Networks

Neural Networks (NN) or Artificial Neural Networks (ANNs) are one of the main ML models. Biological neural networks from animal brain structures inspire their design. NNs consist of a set of node layers, the input layer, one or more hidden layers and the output layer. Each node is loosely connected to other nodes, each with its associated weights and threshold values. The connections between the nodes are called edges, allowing for communication between nodes and also having their associated weights. Signals travel from the input layer through the hidden layers to the output layer. If a given node's output is higher than its associated threshold value, it is activated and sends data to the next layer.

2.2.1 Deep Neural Networks

Deep Neural Networks (DNNs) derive from the deep learning subfield of ML and are based on NNs. Their distinguishing factor from previous methods is representation learning (also known as feature learning) with multiple levels of abstraction. This technique allows models to discover the underlying data structures needed for feature detection and classification. These methods have been successfully applied in speech recognition, visual object recognition and detection, among others ([10]). The term "deep" comes from a large number of stacked layers that the model has compared to normal NNs. The most basic features are learned in the starting layers and the most complex at the bottom layers. DNNs usually have a feed-forward architecture where the data flows from the top layers to the output layer. In the end, the errors are back-propagated through the network to adjust the weights of the nodes in each layer.

2.2.2 Autoencoders

Autoencoders (AE) are a subtype of neural networks, and their main purpose is to compress existing data into meaningful representations, which are then decompressed back to the original input. Examples of applications include classification, clustering, anomaly detection (oftentimes adversarially trained), dimensionality reduction, and others. Its objective is to minimize the distance between the reconstructed and original samples. Therefore, the training goal is to minimize the loss function

$$L(\theta,\phi) = \mathbb{E}_{x\sim\mu_{ref}}[d(x, D_\theta(E_\phi(x))] \tag{2.1}$$

where θ, ϕ are the parameters of the encoder, E_θ and the decoder D_ϕ, respectively; x is the original sample, μ_{ref} is the reference probability distribution and d is the distance function that measures the reconstruction quality by comparing x with its reconstructed examples, $D_\theta(E_\phi(x)$.

Regularizations are often applied to objective functions in line with the application of the autoencoder or to avoid overfitting. Some methods purposely reduce the reconstruction ability of the autoencoder to produce more meaningful compressions and vice versa. Possible variations of AEs include *Sparse AEs*, which aims to reduce the dimensionality of the input data, *Denoising AEs*, mainly used to reduce noise in images and *Contracting AEs*, which aims to reduce the number of features that need to be learned by removing the unnecessary ones.

2.3 Generative Adversarial Networks

GANs were first introduced by [11] in their paper "Generative Adversarial Nets". Since then, many variations of GANs have surfaced and been applied to different areas like human face generation, image-to-image and text-to-image translation, and semantic generation, among others. The original GAN consisted of two models, a generator G and a discriminator D. The task of the first model was to capture the distribution of the data and generate new examples from that distribution. The function of the discriminator D is to distinguish actual samples from the fake data generated by G. The two components play an adversarial game in which G tries to fool D with increasingly realistic examples, and in turn, G tries to detect the fake samples from G. The authors proposed an analogy that would help the problem's dynamics:
The generative model can be thought of as analogous to a team of counterfeiters trying to produce fake currency and use it without detection, while the discriminative model is analogous to the police trying to detect the counterfeit currency. Competition in this game drives both teams to improve their methods until the counterfeits are indistinguishable from the genuine articles. ([11])

The problem is formulated as follows:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z))] \tag{2.2}$$

where x represents the data, z the latent space, $p_{data}(x)$ is the distribution of the data and $p_z(z)$ is the distribution of the latent points (usually Gaussian) that G uses to generate new samples. GANs can then be defined as a minimax game where D tries to maximise V, as it tries to recognise generated and real images better; and, on the other hand, G wants to minimise the function V because its goal is to fool G as many times as possible.

2.3.1 Challenges in the training phase

Earlier GAN architectures were very unstable and hard to train. Despite some proposed solutions to these issues ([12, 13]), GANs are still remarkably difficult to train. Following are some of the main problems experienced during this phase.

Mode collapse. Occurs when the GAN is incapable of reaching Nash equilibrium [2] and is a consequence of poor generalisation. It can occur when the generator G only creates samples from a subset of the data distribution or only learns part of the distribution. The leading causes for this issue can be attributed to a poor choice of the objective function ([14]). In other words, G focuses on a small subset of samples that consistently fool the discriminator D.

Vanishing gradients. The discriminator D does not provide enough information for G to update its gradients [15]. D can distinguish real samples from fake ones with high confidence, which in turn causes the loss function of G to decrease towards 0. As D gets better, the gradient of G progressively decreases until virtually none of the layers are updated, and G can't generate samples with new distributions. Some solutions for this problem include batch normalisation and clipping.

Evaluation metrics. Due to their wide range of applications, no global evaluation function can be applied to every GAN. The evaluation function varies greatly from the context of the problem in which the GAN was used. In some instances, like image generation, the principal evaluation criteria are still done qualitatively (the outputs are analysed by humans, who determine their quality). Evaluation functions are an essential part of machine learning and allow for the correct conclusions to be made [14].

2.4 Outliers

Outliers can be classified as data points significantly different from the rest of the data [16]. They can also be referred to as anomalies, out-of-distribution data, novelties, and deviations [16, 17].

Datasets often contain unusual characteristics that, in some cases, can be informative in determining the origins of outliers. They can be intentional when they result from nefarious actions(e.g., credit card fraud); and unintentional when they occur naturally (e.g., sensor anomalies, input errors). Following are some examples of outlier detection:

- **Credit-card fraud:** theft of credit card credentials can be detected by analysing the transaction history of the target.
- **Medical diagnosis:** anomalies in scans can indicate possible diseases.
- **Fault diagnosis:** detection of faults in critical components (e.g., space shuttles).
- **Intrusion detection:** detecting unauthorized access to computer networks.

[2] Two players, Alice and Bob, chose strategies A and B; Alice has no other strategy to maximise her goal better, and Bob has no different strategy other than B to maximise his goal in response to Alice's choice (https://en.wikipedia.org/wiki/Nash_equilibrium)

2.4.1 Outlier Detection Approaches

There are three main approaches for outlier detection based on unsupervised, supervised and semi-supervised methods. The objective of the first is to detect anomalies in the dataset with no prior knowledge of the data. This approach follows the same logic as clustering, in the sense that it defines one or clusters and then identifies every point outside the clusters as an outlier.

Supervised outlier detection aims at modelling the normality and abnormality of the data, and as any supervised learning problem, requires labelled data. The classification algorithms used for this purpose require balanced distributions of normal and anomalous data to be able to generalize better. However, this is not always possible in these types of problems, as outliers are almost always a minority class.

Semi-supervised detection is a compromise between the previous two approaches. The objective is to model only the normal distribution of the dataset and then use the model on the whole dataset. The new samples (outliers) that weren't observed during the training phase will detect by the model as anomalies. This approach requires a preprocessing of the dataset in order to create a dataset with only normal samples that can later be used in the training phase [18].

2.4.2 Outlier Detection Techniques

Chapter 3

State of The Art

A Systematic Literature Review (SLR) was performed to understand the current trends of GANs applied to anomaly detection. This search method allows for the analysis of all the existing research on a defined question. To achieve this, a set of search questions is formulated (sec. 3.1) and grouped into the main search query (sec. 3.2). Next, the inclusion/exclusion criteria for the obtained results are identified (sec. 3.3).

3.1 Search Questions

Two search questions were defined to find out the most relevant articles that would more closely fit the needs of this work:
SQ1 *What are the current GAN-based architectures for anomaly detection in tabular data?* Most GANs are designed for problems with image datasets. However, the dataset used in this study consists only of tabular data. The intent is to include only GANs designed or adapted to anomaly detection in tabular datasets.
SQ2 Of the results from the previous question, which ones can generate new samples following the distribution of the dataset and detect out-of-distribution examples? The proposed goals for this thesis are to produce a model that can generate new solar wind profiles and another that can accurately distinguish out-of-distribution samples from normal ones. As so, the generator should be able to generate examples in the same distribution as the original dataset, while the discriminator must distinguish anomalous ones.

3.2 Search Query

The search questions defined in the previous sections were aggregated into a single search query. The query construction was incremental to narrow the search results to the desired questions. In the end, the search query was the following:

(gan* OR adversarial learning OR generative adversarial net*)
AND ((anomal* OR outlier? OR abnormal OR novel*) AND detect*)

AND NOT (imag* OR video* OR segment* OR photo*)

The first part consists of a mixture of terms associated with GANs and intendeds to only retrieve articles with one of those terms in the title, abstract and keywords. The second restricts the results to GANs for outlier detection in the same three fields as the previous one. Several synonyms for outlier were used to increase the number of relevant papers. The final term was only applied to the documents' keywords and was intended to exclude GANs applied to image datasets. All articles were retrieved from Scopus [1].

3.3 Inclusion and exclusion criteria

The query defined in the previous section yielded 1489 results, making the analysis of each one by hand prohibitive. A set of criteria was determined to reduce the number of documents that needed to be studied (see Table 3.1). Note that E3 only exists because the search question SQ2 failed in instances where the documents did not indicate the use of images in the keywords. These were later used with several steps to exclude non-relevant papers and narrow the state-of-the-art analysis iteratively.

Table 3.1: Inclusion and Exclusion criteria.

Criteria	ID	Description
Inclusion	I1	The document focuses on GANs for anomaly detection.
	I2	The results are clear, and the proposed goals are achieved.
	I3	The authors provide code or an extensive explanation of the architecture.
	I4	Provides a comparison of the developed GAN with other baseline models (not necessarily GANs).
Exclusion	E1	The article was cited less than 6 times. For earlier publications, the number of citations was reduced to half.
	E2	Surveys and reports on works carried out by other authors.
	E3	Does not use tabular data. Either it has one or more unwanted terms in the title (e.g. image "photo") or only performs tests on image datasets.
	E4	Was published before 2014.

An illustration of the processing pipeline can be seen in Fig. 3.1. 1489 results were retrieved from Scopus with the defined query. The first processing setup applies exclusion criteria *E1* to remove papers with little to no citations, which resulted in 168 documents. 61 were left after a preliminary title analysis with the criteria (I1; E2-E3). In the final step, a preliminary analysis of the remaining documents' abstracts and conclusions was undertaken to only select the most relevant to the defined search questions. The inclusion criteria for this step were I1 to I4. In addition, the documents that were surveys or reviews of multiple implementations and articles that did not deal with tabular data were excluded. In the next section, the resulting papers from this last step will be explained.

[1] Scopus: https://www.scopus.com/

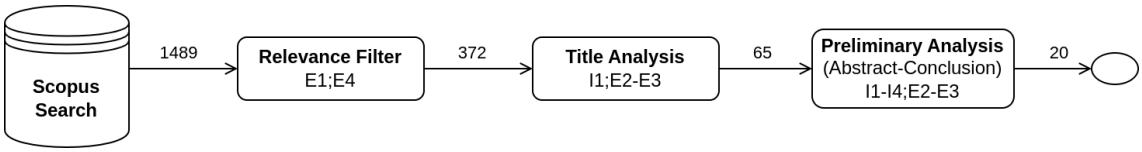

Figure 3.1: Systematic Literature Review Pipeline

3.4 Results

MAD-GAN [19] is an architecture intended for anomaly detection in multivariate data with spatio-temporal correlations. The generator and the discriminator are composed of Long-Short-Term Recurrent Neural Networks (LSTM-RNN). As is usual in other GANs, the generator creates fake samples from a vector of latent points and feeds them to the generator, whose goal is to distinguish generated from original samples. However, instead of just using the discriminator to detect abnormal samples in the testing phase, the authors propose employing the generator for the same task. The theory for this is that by generating correct samples, the generator can learn the normal distribution of the dataset.

During the test phase, the discriminator receives a sample from the test dataset and performs the same classification as in the previous stage. However, the generator will receive a version of the sample mapped to the latent space and be tasked with reconstructing the sample. Next, the reconstruction error is calculated by comparing the reconstructed sample with the original one. This error and the discriminator outputs are used to compute the *Discriminator and Reconstruction Anomaly Score* (DR-Score). A sample is considered abnormal if it has a DR-Score higher than a predefined value τ.

The developed architecture was compared with five baselines. These include PCA, K-Nearest Neighbours (KNN), Feature Bagging (FB), an Autoeconder (AE), and the *Efficient GAN* (EGAN) [20]. The tests were performed on three intrusion detection datasets. MAD-GAN was able to outperform the other baselines on almost all datasets consistently.

ALAD [21] is a reconstruction-based anomaly detection architecture that employs multiple bi-directional GANs. The proposed method, *Adversarily Learned Anomaly Detection* (ALAD), intends to use both the discriminator and the generator for the task. The ALAD architecture can be seen in Fig. 3.2.

An encoder network E maps data samples x into the latent space z during training. Several additional discriminators are used to achieve cycle consistency (to ensure that the reconstructed samples resemble the original ones) and to provide stability to the model. D_{xz} is an improvement from other similar solutions that solve the saddle-point problem by ensuring cycle consistency, which is not always the case when using encoders. Further entropy regularisation is imposed on both G and E by the discriminators D_{xx} and D_{zz}. The latter receives two pairs of latent points and must distinguish the real (z, z) from the synthesized one (z, $E(x)$); the former follows a similar logic but with examples extracted from the dataset.

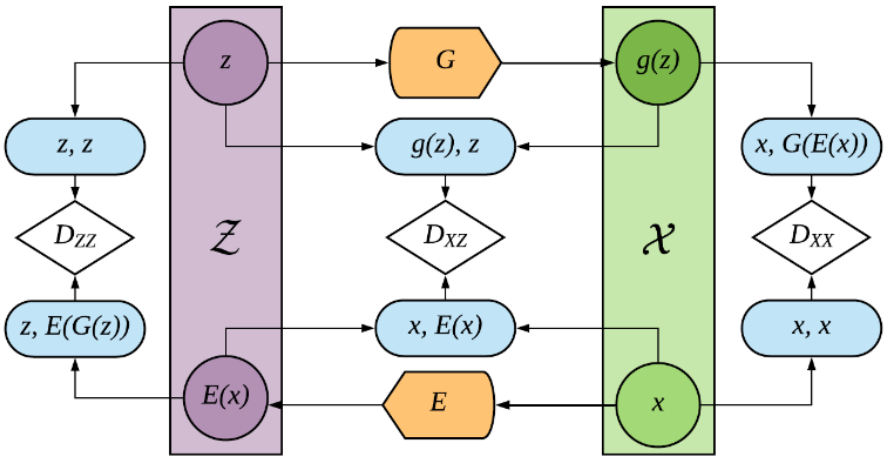

Figure 3.2: ALAD Architecture. D_{xx}, D_{xz} and D_{xx} are the discriminators (white), Z (purple) and X (green) represent the latent and data spaces, respectively; G and E (orange) are the generator and the encoder, respectively. Reprinted from [21].

The authors propose a new score function for anomaly detection that captures the confidence of D_{xx} when distinguishing real from synthesized pairs. This is because a poor-quality reconstruction would indicate that the generator did not learn how to reconstruct that sample and, consequently, should be considered an anomaly. Finally, the designed model was compared with five standard anomaly detection methods and another GAN for anomaly detection on two anomaly detection datasets. The developed model outperformed the baselines on one of the datasets but could not do so on the other. This was because this dataset had a small number of samples, and ALAD, like other GANs, requires large amounts of training data.

USAD [22] is an architecture based on adversarially trained autoencoders for anomaly detection in multivariate time series data, more concretely, logs from IT systems. The authors proposed solving the convergence and mode collapse problems experienced in other GANs. USAD is composed of one encoder E and two decoders $D1$ and $D2$, which in conjunction with the encoder, result in two autoencoders ($AE1$ and $AE2$). E takes data samples as windows W and encodes them to latent space vectors z. The function of the decoders is to reconstruct the samples in those windows.

The autoencoders are trained with normal samples to learn the data distribution during this phase. In the detection stage, both autoencoders are trained adversarially. $AE1$ reconstructs samples from the real dataset and $AE2$ must distinguish examples reconstructed by $AE1$ from the real data. The anomaly score is calculated based on the reconstruction errors obtained on both autoencoders. The proposed model was evaluated along with other unsupervised methods for anomaly detection on intrusion detection datasets. USAD outperformed the other baselines in terms of F1-Score.

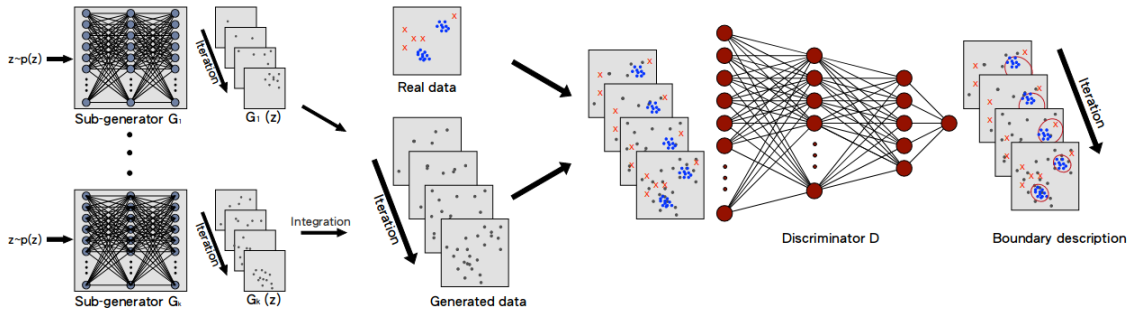

Figure 3.3: MO-GAAL Architecture. Each generator G_i to G_k (left) generates outliers for the respective cluster; The discriminator D (right) aims to draw increasingly smaller boundaries around the real data distribution. Taken from [23].

MO-GAAL [23] as the goal of generating informative outliers to overcome the significant class imbalance and lack of correct labels in outlier detection datasets. The authors developed two proximity-based outlier detection methods that do not rely on previous knowledge of the dataset. The first one was given the name of *Single-objective Generative Adversarial Active Learning* (SO-GAAL). Like other GANs, it plays the same mini-max game between the generator G and the discriminator D. However, the objective for G is to produce outliers that occur inside or close to the real data. Similarly, the goal of D is to create a division boundary that separates the real data from potential outliers. G gradually learns the generating mechanism and synthesizes an increasing number of potential outliers, and D gets better at creating the divisions that enclose the real data. The point of generating outliers is to create a reasonable reference distribution for the real data.

Despite providing good results, the first model proved to be very unstable due to the problem of mode collapse. At some point, after a good amount of iterations, the precision of the model would greatly diminish. To solve this issue, another technique called *Multiple-objective Generative Active Learning* (MO-GAAL) was designed. A general workflow of the architecture can be seen in Fig. 3.3. The authors proposed generating outliers for specific real data subsets using a generator for each cluster in the dataset (which requires cluster identification). This solved the mode collapse problem on the first model and stabilized the performance.

Both architectures were tested on fourteen datasets and ten other baseline outlier detection methods. Despite other methods performing better in specific datasets, MO-GAAL proved to be more reliable even on non-cluster datasets.

IGAN-IDS [24] or *Imbalanced Generative Network* was designed to cope with class imbalance problems that other outlier methods for intrusion detection suffer from. IGAN, which can be seen in Fig. 3.4 (middle module) is composed of an imbalanced data filter, Generator G, and a Discriminator D. Each sample, $s = (x, y)$, is a vector containing the values and the class labels. The imbalanced filter takes only samples from the minority classes, denoted as

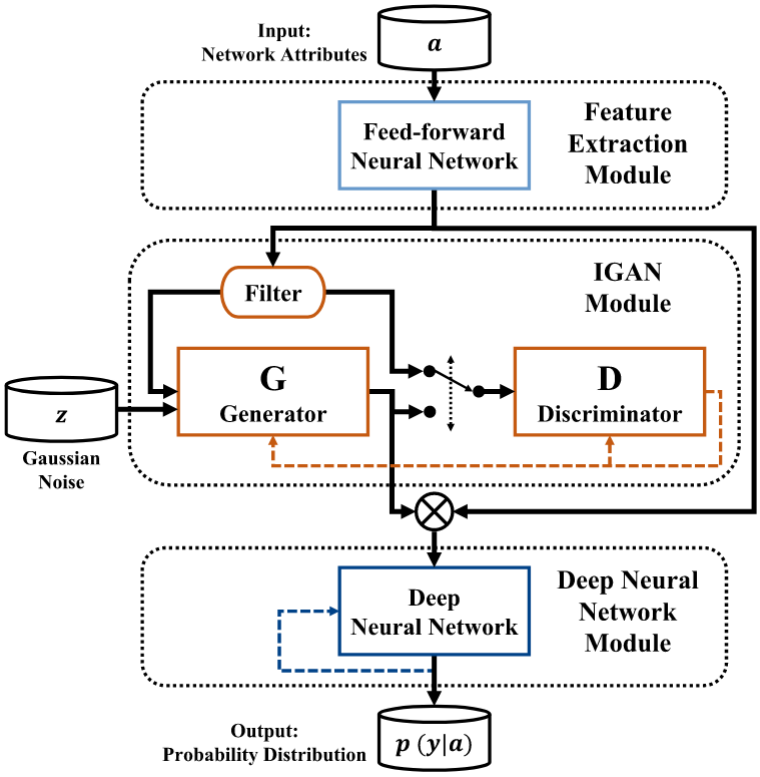

Figure 3.4: Full IGAN-IDS architecture. Taken from [24].

$s' = (x', y')$. It calculates the generating factor k for each class (ratio of samples that should be generated for each minority class). G receives a set of latent points z and the class label y' and outputs a vector with a generated value $G(z, y')$ from the class label that it received. This vector is then passed to the discriminator, which aims to distinguish synthesized feature vectors from the ones extracted from the dataset. In the training process, G and D are trained alternatively. First, D is fed only real samples while G is fixed, and in the next iteration, G is optimized, and D is fixed.

With the problem of class imbalance dealt with, the authors set out to perform intrusion detection with IGAN-IDS (Fig. 3.4). The feature extraction module (top) embeds discrete data into one-hot encoded vectors and discretizes continuous variables, which are encoded. All values are concatenated and fed to IGAN, which generates samples for the imbalanced classes. Finally, a DNN (bottom module) is used for outlier detection. In the training stage, it receives both synthesized and real data and, during the testing phase, calculates the distributed probabilities for each inclusion class of each sample. The proposed solution was tested with several other class-balancing techniques on three standard intrusion detection datasets. It outperformed every method in precision, accuracy, recall and AUC score.

Jiang et al. (2019) [25] propose a conditional GAN architecture to detect anomalies in univariate Industrial Time Series Data. The model is trained with only normal samples. The dimen-

sionality of the original data was reduced by employing a feature extractor. The generator consists of two encoders G_e and $G_{e'}$ and an intermediate decoder G_d. The first encoder maps real samples into the latent space z while the decoder G_d is tasked with reconstructing the encoded sample back to the real data space.

During training, the GAN is only tasked with reconstructing normal data samples so that both components learn the normal distribution of the dataset. Two loss functions are defined for the generator, the *Apparent loss* and the *Latent loss*. The first measures the distance between the original and synthesized samples, and the other measures the distance between the latent vectors z and z encoded by G_e and $G_{e'}$, respectively. The loss function of the discriminator compares the feature vector from the actual sample $f(x)$ with the synthesized one $f(G(x))$. The anomaly score is defined as the sum of the two losses.

In the testing phase, two rolling bearing datasets were used. The authors fine-tunned the models by adjusting the hyper-parameters of the network. A significant difference in anomaly score $A(x)$ was observed for faulty parts in the dataset, which proved the efficacy of the designed model. A comparison was also performed with the state-of-the-art BiGAN [26], which showed that the developed GAN was more reliable on datasets with differing sizes.

TadGAN [27] aims to solve the problem of scalability and portability in state-of-the-art unsupervised methods for anomaly detection. Its goal is to detect anomalies in time series datasets. The authors used LSMT Recurrent Neural networks for both the generator and the critic. Two types of anomalies are identified single point (abnormal data point) and collective (sequence of abnormal data points) anomalies.

The proposed architecture is a reconstruction-based anomaly detector with a generator G which receives encoded samples in the form of latent points z and reconstructs them back to the original sample distribution; an encoder that takes samples in the normal distribution and encodes them into latent point vectors; and two critics, one to distinguish real data points from synthesized ones (C_x) and the other (C_z) to evaluate between the distribution of real z and the ones that were encoded by E ($E(x)$). To cope with the mode collapse problem, common in most GANs, the authors adopt the Wasserstein loss function (for critics) and a cycle consistency loss function (for the generator and encoder). For the reconstruction errors, the point-wise difference (distance between the real and synthesized point) and area difference (distance between "windows" of the same area in the real and synthesized data) were defined. Furthermore, the authors also chose a *Dynamic Time Warp* (DWT) measure, which, similarly to area difference, can identify minor differences over long periods but can also handle time shift issues.

Two methods of combining the critic outputs with the reconstruction errors were devised. For the first, a weighted sum of the reconstruction error and the critic output is done; in the other method, both values are directly multiplied. Several baselines are chosen to compare with the developed model in the testing phase. TadGAN and the baselines were tested on eleven datasets for anomaly detection (two of which were from NASA). The developed

network outperformed every other baseline on six of the eleven datasets (based on the F1-score). Despite this, the mean, standard deviation and average of the F1-score in all datasets showed that TadGAN was more reliable than the other methods. Finally, the authors defined ten iterations of TadGAN, each with a different anomaly score with either one reconstruction function, one critic output or a combination of the two. The worst result was with the anomaly function consisting only of the critic output, and the best was the one in which the critic score and DWT were multiplied.

adVAE [28] employs a Variational Autoencoder (VAE) for anomaly detection. The authors propose an encoder E that encodes real samples into random point vectors z, fed to a generator G that is then tasked with synthesizing examples close to the real distribution of the dataset. Competition is introduced in the form of a Gaussian transformer T that receives the encoded vectors from E and is tasked to generate latent vectors z_T with a similar distribution to z (outliers). G is tasked with generating as different as possible examples from both similar latent vectors. Finally, the examples from G are encoded again by E, and the resulting distributions are compared with the original ones. To make the equilibrium of these three models feasible, the authors freeze the gradients of E in the first training phase to only train the G and T. This way, T will generate abnormal latent variables close to the real distribution, and G will be able to distinguish them using reconstruction errors.

In the second phase, both G and T have their gradients frozen, and the encoder is trained to encode the inputs as close as possible to the prior distribution (only if the inputs are from the dataset and do not result from Gaussian variables generated by T). In the testing phase, the trained encoder and generator are used to detect anomalies by calculating the reconstruction error of the input samples. The solution was evaluated on tabular anomaly detection datasets and several other state-of-the-art outlier detection methods. Furthermore, several ablation models from adVAE were derived by removing the discriminative factors of either the generator or the encoder. The results showed that adVAE and its variations outperformed most baselines on the chosen datasets regarding precision and AUC score.

Blance et al. (2019) [29] propose using adversarially trained autoencoders as a way of improving the separation between background from the signal in synthesized high-energy collision events. The authors train an NN that can distinguish signal events from the background and intentionally smear the background data in distinct directions. With this, they prove that the classifier's performance is highly dependent on the smearing of the background samples.

An adversary is used to try and remove the dependence of the classifier on the smearing of samples. Both are forced into a zero-sum game in which the classifier must learn how to make predictions without using any information from smearing and tries to make it as hard as possible for the adversary to discriminate the background samples. The classifier receives samples from the dataset and sends its outputs to the adversary, which tries to determine the background class based on the outputs. The results showed that this method significantly reduced the dependence of the classifier on the smearing of background samples.

In addition, the authors propose another method in which they use an autoencoder alongside an adversary. The function of the autoencoder is to reconstruct only background samples, and the adversary is tasked with identifying them. As the autoencoder only learns the distribution of background events, it will not be able to reconstruct signal events as well (i.e. signal events will be considered outliers). The adversary takes as input the loss of the AE and tries to determine the background smear class. The results show that the method could remove the dependence between the autoencoder classification and the smear direction of the background samples. Despite this, this architecture proved less effective than the previous one.

FGPAA [30] is an adversarially trained autoencoder that aims to monitor the conditions of roll-bearings by analyzing the vibration signals. The model consists of four components, a discriminator D, a generative discriminator GD, an encoder E and a Low-dimension discriminator LD. E takes one signal at a time and encodes it into the low-dimension manifold (latent space) z. LD discriminates if the output of the encoder follows the same distribution as the latent space z. The latent points vector z is passed to the encoder that works as a generator by synthesizing high-dimension signals from the low manifold distribution. The generative discriminator, GD, tries to distinguish reconstructed signals from those originating from the dataset.

The anomaly score is calculated for each sample by comparing the distributions of the generated low-dimension manifold and the reconstructed sample with the distributions from the actual dataset. This score is then used during detection to identify signal data faults. The proposed solution was tested on three roll-bearing datasets alongside well-known ML anomaly detection methods. FGPAA outperformed all in terms of F1-score, but they displayed a higher execution time than the rest.

FGAN [31] is an architecture close to original GANs but with a modified loss function more suited for anomaly detection. The authors propose adapting the model's objective so that the generated samples lie close to the boundaries of the real data distribution instead of overlapping it. The objective is to generate data around low data density regions δX around the real dataset.

The authors use the discriminator score to identify the domain of δX and then estimate it with the generator. At the end of the training phase, the synthesized points must enclose the entirety of the data. This goal is achieved by replacing the typical loss of the generator with the *Generator Encirclement Loss*, which penalizes points generated inside or far away from the real distribution. *Generator Dispersion Loss* is also introduced to guarantee that the generated points enclose the whole distribution and not just a single area (similar to mode collapse in other methods). It maximizes the distance between points by penalizing the generator if the synthesized points are too close to each other. The resulting loss function is a weighted sum of the two proposed ones. Similar to the generator, the loss function of

the discriminator is also modified to prioritize classifying real data correctly by reducing the second term of the original discriminator function (refer to equation 2.2).

To evaluate the developed mode, the authors tested its performance on a synthesized 2D dataset. Next, the proposed solution was tested on image and tabular datasets and other state-of-the-art models. FGAN outperformed all baselines for anomaly detection on both types of datasets.

MENSA [32] is an autoencoder-based GAN architecture used to detect intrusions on next-generation Electrical Grids (also known as Smart Grids). Furthermore, the proposed model can also detect and classify different classes of cyberattacks that often occur on the TCP (Transport Communication Protocol) and the DNP3 (Distributed Network Protocol 3) protocols.

The architecture consists of a Generator-Encoder and a Discriminator-Encoder. The first receives input noise samples and inflates them to produce samples that resemble the desired data to learn the normal distribution. The Discriminator-Encoder then compresses the synthesized samples into a single point that indicates if the sample is from the real dataset or is a fake. For the detection phase, the Latent Model is derived from the initial layers of the discriminator. The generator, having learned the normal distribution of the data, tries to reconstruct samples from the real dataset. These are then passed to the Latent Model that calculates the Adversarial Loss score by comparing the real sample with the reconstructed one. Note that the generator will fail to reconstruct abnormal samples, as they were not a part of the training process.

The classification model is derived from the previous architecture, in which the Discriminator-Encoder also learns to classify the attack class of a given sample. The generator learns to generate samples based on the class labels conditionally. Similar to the previous implementation, the discriminator receives the synthesized samples and identifies them as real or fake, but this time, it also tries to determine the class label.

The proposed solution was tested on real Smart Grid datasets and several other intrusion detection methods. The Accuracy, True-Positive and False-Positive rates and F1-Score were calculated for each solution. MENSA outperformed other methods on all datasets except for one.

Liu et al. (2022) [33] propose a deep feature enhanced generative adversarial network to improve fault detection performance in roll bearing imbalanced datasets. New and preexisting methods are introduced to solve mode-collapse during training and enhance the feature learning of the network, which aim to increase the overall detection performance of the architecture. The adopted methodology can be seen in Fig. 3.5.

A new loss function is designed for the generator with a *pull-away* term. It measures the distance between the generated samples inside a given batch and penalizes the generator if the batch's samples are too similar. As a consequence, this solves the mode-collapse

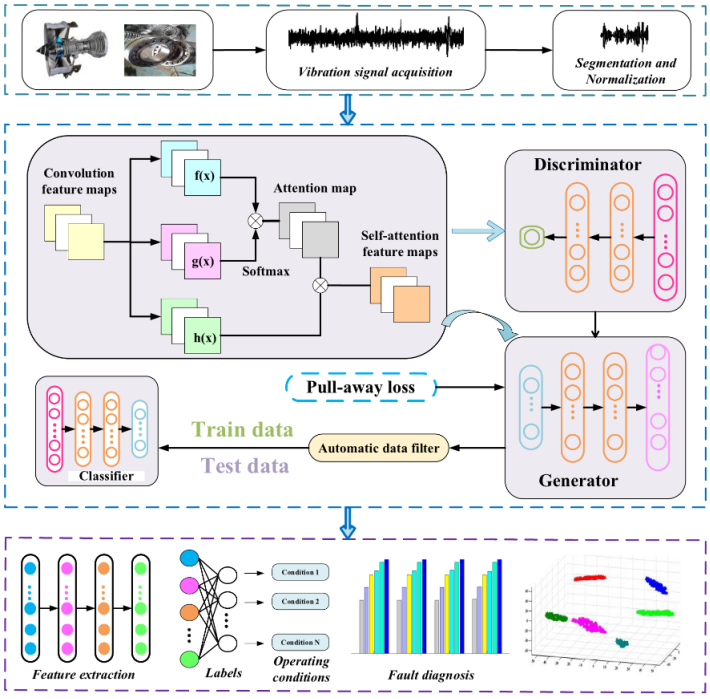

Figure 3.5: Methodology for anomaly detection in roll bearing datasets. Taken from [33].

problem during the training phase. A self-attention model is introduced to the discriminator and the generator so they can learn the features of the original vibration signals more deeply. The self-attention feature maps capture local details and global information in every layer of the network.

During training, the generator synthesizes signals from random noise fed to the discriminator along with an actual signal. The discriminator must then distinguish which one is real and which one is fake. With the proposed mechanisms, the generator must synthesize samples as far as possible to each other to confuse the discriminator. The training phase stops when three criteria are met. These are defined by the *automatic data filter*, which evaluates the accuracy and diversity of the generated samples based on discriminator probability, Kullback-Leibler (KL) divergence and maximum mean discrepancy. When these criteria surpass a predefined threshold, the training phase is concluded. The fault detection phase uses a classifier trained on the generated data.

The quality of synthesised signals was compared with other standard generative models using KL divergence and maximum mean discrepancy of the *automatic data filter*. The proposed method surpassed the others in these metrics. Next, three experiments were set up to evaluate the detection quality of the proposed method. In each experiment, three classifier models were used. In the first, the classifiers learned to detect anomalies from the original signals dataset. In the next, a WGAN-GP architecture was used to generate a more balanced signals dataset. Finally, the three classifiers were trained on data synthesized by the proposed method in the last experiment. This methodology was performed on laboratory

and locomotive roll-bearing datasets. Every model trained on the generated data (for both datasets) from the designed solution outperformed the models from the other experiments.

DOPING [34] is an adversarial autoencoder that aims to improve the performance of unsupervised anomaly detection by oversampling *infrequent normal samples*. With this, the authors intend to reduce the number of false positives that usually occur on datasets with normal samples close to the classification boundary (close to but not anomalies).

In the training phase, the autoencoder receives samples from the entire dataset distribution. The latent vectors generated by the encoder, E, from the data samples, X, are saved for the next stage, in which only the latent variables Z at the tail-end distribution of the normal data are sampled into a pool of Z_{edge}. From this pool, z_{edge} variables are then sampled randomly and interpolated to generate new latent vectors z_{new}. In the next stage, de decoder D synthesizes minority samples from the z_{new} latent vectors. Later the synthesized examples can be used in conjunction with the original ones for anomaly detection.

An isolation forest anomaly detector was used to evaluate the effects of the oversampling method on the detection performance. The authors used three synthesized cluster datasets with outliers, the popular MNIST image dataset, and four real medical record datasets. The results showed that the outlier detection method in conjunction with DOPING achieved better results than the baseline (detection without DOPING).

Bot-GAN [35] aims to improve the detection performance of botnets in network-flow data. The architecture is similar to the one from vanilla GAN. However, the authors chose a botnet detection model as a discriminator. Like vanilla GAN, it receives examples from the training dataset and synthesizes ones from the generator. The main difference is that the discriminator classifies each sample as normal (from the training dataset), an anomaly or fake (either synthesized or from the dataset). The generator's objective is to synthesize more samples similar to the ones in the dataset to assist with the training of the detector.

To test the proposed approach, choose a botnet dataset and perform some preprocessing to normalize the formats of the entries and map the features into vectors. The resulting feature maps are then scaled so that each value falls between [-1,1]. The model was evaluated on precision, accuracy, false positive rate, recall and F1-Score. The results show that it is possible to improve the performance of the classification model by enhancing its training with GANs.

Yuan et al. (2020) [36] propose employing GANs to learn the normal conditions of smart meter operations to detect power outages in electrical grid zones. In addition, the authors propose circumventing the problems of other models applied to this problem, which are the assumption that every network node is directly observable. The distribution network is subdivided into zones determined by two neighbouring observable nodes (nodes in which the voltage

and demand). Each zone has its designated GAN, which learns the time-series data collected by the two nodes. Any deviation from each node's normal measured data distribution will be considered an anomaly.

The architecture is very similar to the one on vanilla GAN [11]. The generator's objective is to synthesize Time-Windows (frames with recorded sequential events) with events similar to the ones from the assigned zone. Similarly, the discriminator must distinguish between real and fake Time-Windows. At the end of the training phase, both components captured the normality of the data for the given zone.

In the detection phase, both generator and the discriminator are used. Time-Windows with actual recorded events are fed to the latter, which calculates the Discrimination Loss. Additionally, an inverted mapping of the Time-Window to the latent space is given to the generator, which is tasked with reconstructing it. The weighted sum of the reconstruction error and discriminator loss is used to calculate the Anomaly Score.

The solution is evaluated on data collected over three years by smart meters on a complex power distribution network. The results proved that the proposed approach could reliably detect power outages in real distribution networks. Finally, numerical comparisons are performed to compare the developed method with a preexisting SMV model to detect power outages. The authors conclude that the developed GAN can achieve better results (in terms of accuracy) with a significantly reduced amount of data.

AMBi-GAN [37] is bidirectional LSTM GAN for anomaly detection in industrial multidimensional time-series data. Unlike univariate time series, multivariate time series consists of multiple measurements in a given time step. The authors propose solving the difficulties of other methods in extracting temporal information, feature extraction and lack of large amounts of labelled data.

The architecture consists of a discriminator and a generator using the same bidirectional LSTM network (AMBi-LSTM). An attention mechanism is also proposed to learn the importance of each time-series element. It calculates a given sample's weight to determine how much it should affect the parameters in the network.

In the training phase, each sample is extracted using a sliding window that subdivides the whole dataset into equal-length subsequences (each with multiple values for a given time). The generator's goal is to generate windows with the same distribution as the original ones. The discriminator receives real and false samples and must distinguish between them.

When both components have reached an equilibrium, anomaly detection can be performed. Random samples are extracted from the testing dataset and fed directly to the discriminator. At the same time, the samples suffer from inverted mapping into the latent space, so they can be given to the generator, whose goal is to reconstruct them as well as possible. Next, the *Discriminator Score* is calculated by combining the loss from the discriminator and the reconstruction error from the generator.

AMBi-GAN and the other two baselines were evaluated on precision, recall and F1-score on three time-series datasets. The proposed solution outperformed the other baselines on every metric. In addition, several variations of AMBi-GAN were developed, from changing the number of hidden layers of AMBi-LSTM to assess which one performed better on the chosen datasets.

TAnoGAN [38] was designed to detect anomalies in industrial time-series data with a small number of samples. The model consists of a generator G and a discriminator D, with both architectures based on LSTM networks. In the training phase, G learns to generate realistic time-series sequences from a latent space distribution, and D distinguishes fake samples from real ones. The samples consist of small time series sequences extracted from the dataset with a sliding window method.

In the detection phase, real-time-series samples are mapped to the latent space and then reconstructed by G. Anomaly detection is done by evaluating the reconstruction error of the synthesized sample with the original one. Mapping from the original sample to the latent space is done iteratively. First, a random sample z^i from the latent space z is chosen and fed to G, which generates a fake sample. The resulting fake data $G(z^i)$ is compared to the original sample x with a point-wise dissimilarity measure L_R. Next, the parameters of z^i are updated and thus, in the next iteration, the reconstructed sample will more closely resemble the authentic one. This process is repeated Λ times (a predefined parameter). At the end of the inverse mapping, the final z^i is compared to the original sample, and the anomaly score is obtained with the weighted sum of L_R and the discriminator loss L_D.

To deal with the small number of samples in the datasets, the authors varied the number of hidden layers in each architecture. It was observed that discriminators with many hidden layers easily overfitted the data. In contrast, generators with small numbers of hidden layers failed to synthesize realist time series sequences.

The solution is evaluated on a large number of time-series datasets along with other unsupervised state-of-the-art anomaly detection methods. The performance (measure in accuracy, recall, F1-score, AUC, and Cohen Kappa Score) demonstrates that TAnoGAN is more suited than other models to detect anomalies in small time-series datasets.

MinLGAN [39] aims to detect outliers by generating both normal and abnormal samples during the training phase. The authors employ minimum likelihood regularisation to the generator, G, to produce more abnormal samples and prevent them from converging to the normal distribution of the data. This solution ensures that the performance of the discriminator, D, does not deteriorate in the final phases of training as it receives samples from the generator that are increasingly closer to the real distribution of data. The regularization is done by adapting the loss function of the generator with a KL divergence measure. It penalizes the generator when the produced samples are distributed close to the dataset and, thus, prevents the convergence of G with the original data distribution.

To deal with the instability of the discriminator in the early phases of training resulting from the data's randomness, the authors propose *Ensemble learning*. Two ensemble methods are presented, which are bagging and boosting. The latter consists of a set of models trained from random subsamples of the training dataset. In contrast, the models are obtained in the former by emphasizing training samples that other models misclassified. The authors independently trained a set of D models in line with this. The outputs of each discriminator (before sigmoid activation) are combined into a single value in two different score functions for anomaly detection (one is scaled to account for the minimum and maximum ranges of the outputs).

In the experimentation phase, three GANs were created. One baseline MinLGAN and two other models with ensemble learning and the score functions that were defined. All of them were trained along with five other unsupervised anomaly detection approaches on an image and several tabular datasets. Despite providing good results in all datasets, the authors pointed to the difficulty felt in the training phase due to the complexity of the proposed approach.

ATTAIN [40] is an architecture that makes use of GANs to detect anomalies in cyber-physical systems (CPS). Unlike other methods, ATTAIN can learn data distribution at runtime without needing static data. This allows it to adapt to previously unseen novel attacks continuously.

The solution consists of a Digital Twin Model, a digital replica of a real system, and a Digital Twin Capability; the GAN used for anomaly detection. The first model is built with historical and real-time measurements from sensors and actuators, while the detector only learns from real-time data. The generator's objective is to produce samples from the latent distribution into the original data distribution. The function of the discriminator is to distinguish between normal, and attack samples that come either from real-time or are synthesized by the generator. Therefore, the output of the discriminator consists of four categories.

The generator, G, is composed of an input layer which encodes discrete values into one-hot vectors; a Graph Convolutional Layer (GCN) that is tasked with learning the independent relationships between sensors and actuators; a pooling layer which collapses the outputs of the previous layers; and an LSTM layer with the function of retaining the temporal features of the data. The discriminator receives both real and generated samples, which are concatenated and suffer a linear transformation. Next, the resulting vector is passed through a *tahn* activation function before being passed to the next layer. The following step calculates a ground truth label for the received fake sample. It is first given to the Digital Twin Model that predicts its state, and later, the hamming distance d between the predicted and real state is obtained to help in the labelling process. The discriminator determines if the sample is real or fake. In the case of the latter the d measure from the previous step is compared to determine if the example is a regular or attack adversarial sample. The loss between the

ground truth and the likelihood (obtained by softmax of the output of D) is calculated in the final layer.

The designed model was tested with two other baselines on three intrusion detection datasets. The performance metrics were outlier precision, recall and F1-score. A comparison of all the results shows that using the digital twin model to guide the training of the GAN improved the overall outlier detection capability of the model compared to the other solutions. However, the authors recognize the possible threats to validity as they could not test the solution on a real CPS system.

3.5 Analysis

In Table 3.2, a brief analysis of the SLR results will be carried out to study the effectiveness of the chosen methodology. Aside from the paper name and year, four other categories were identified to compare the evaluated solutions:

- **Training Objective:** this category aims to describe the training approach for a given model. With this, the goal is to compare models that fit the normal data distribution during the training phase and other alternatives. This allows for a straightforward summary of the training approaches that can be adopted to solve the proposed problem.
- **Anomaly Detection:** in this group, the goal is to identify several of the approaches that can be used to detect anomalies. These methods can be relative to calculating an anomaly score or, in some cases, using classifiers to achieve this goal. The possible values for this category include Reconstruction errors, Discriminator Loss (D loss), and Classifiers.
- **Architecture:** this class aims to analyze the different types of architectures defined in each of the papers. These can be "Normal" in the case they use the vanilla architecture of GANs; "Mixed" when other components like encoders, E, classifiers, C, feature extractors (FE) and Self-attention devices (SA)[2] are used; and autoencoders (AE).
- **Application:** this last category describes the specific problem that each of the designed models tries to solve (i.e. the scenario to which they are applied). Some models have no direct application and can be used in many types of problems, and because of this, they have no assigned application (N/A)

Fig. 3.6 shows the number of analysed papers per year, and Fig. 3.7 shows the year distribution of the papers analysed by hand after the application of the citation filter (Fig. 3.1). The majority of the papers that were chosen were from 2020. Even after the restrictions applied to the search query, a significant portion of the results is still comprised of GANs for image synthesis.

[2]Module used during the training phase to make the discriminator and generator learn the data features more thoroughly.

Table 3.2: List of reviewed papers.

Paper	Year	Training Objective	Anomaly Detection	Architecture	Application
MAD-GAN [19]	2019	Normal	Reconstruction + D loss	Normal	Time Series
ALAD [21]	2018	Normal	Reconstruction	Mixed (E)	N/A
USAD [22]	2020	Normal	Reconstruction	AE	N/A
MO-GAAL [23]	2020	Outlier Generation Division Boundary	Classifier	Mixed (C)	N/A
IGAN-IDS [24]	2020	Outlier Oversampling	Classifier	Mixed (FE + C)	Intrusion Detection
Jiang et al. [25]	2019	Normal	Reconstruction	AE	Time Series
TadGAN [27]	2020	Normal	Reconstruction + D loss	Mixed (E)	Time Series
adVAE [28]	2020	Normal	Reconstruction	AE	N/A
Blance et al. [29]	2019	Outlier Oversampling	Classifier	AE	High Energy Physics
FGPAA [30]	2020	Normal	Reconstruction	Mixed (AE)	Roll Bearing Fault
FGAN [31]	2019	Normal Division Boundary	D loss (adapted)	Normal	N/A
MENSA [32]	2021	Normal	D (initial layers)	AE	Intrusion Detection
Liu et al. [33]	2022	Normal	Classifier	Mixed (C + SA)	Roll Bearing Fault
DOPING [34]	2018	Minority Oversampling	Classifier/Model	AE	Performance Improvement
Bot-GAN [35]	2018	Normal	Classifier (D)	Normal	Bot Detection
Yuan et al. [36]	2020	Normal	Reconstruction	Normal	Power Outage Detection
AMBi-GAN [37]	2021	Normal	Reconstruction + D loss	Normal	Industrial Time Series
TAnoGAN [38]	2020	Normal	Reconstruction	Normal	Time Series
MinLGAN [39]	2018	All Data	D loss	Normal	N/A
ATTAIN [40]	2021	Normal	D Loss	Normal	Cyber-physical Systems

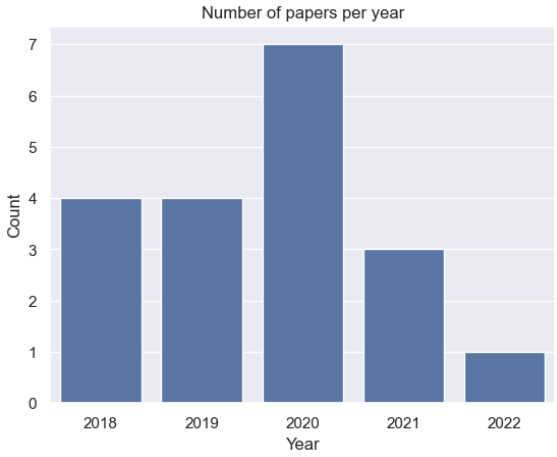

Figure 3.6: Number of reviewed papers per year.

Further analysis can be done with regard to the search questions defined in sec. 3.1. For **SQ1**, it was shown that several solutions apply GANs or variations in anomaly detection in tabular datasets. Surprisingly, some models worked on image and tabular datasets with the proper adjustments. Additionally, 40% use either an autoencoder or parts of it to help with anomaly detection in complex data types and 9/20 use some variation of reconstruction errors to determine if a sample is anomalous.

As for **SQ2**, it was shown that most methods (75%) learn the normal distribution during the training phase and then detect outliers. 20% generate outliers that are used to train classifiers for detecting anomalies. Only [39] makes use of both abnormal and normal data distribution in the training phase. This indicates that most GAN-based solutions for anomaly detection can generate realistic data with the same distribution as the original data. However, not all use the discriminator to distinguish between normal and abnormal samples, as they employ classifiers.

3.6 Threats to SLR

The chosen limit for the minimum citations might have been too restrictive, especially for the papers recently published. The adaptation of inclusion criteria I4 (Table 3.1) to allow for fewer citations in earlier papers was made to try and mitigate this risk. Despite this, the number of analysed papers for 2022 was significantly smaller than the previous years, which introduces the risk that some of the newer published relevant papers might have been overlooked.

Furthermore, as only one search engine was used for the SLR, there is also the risk that some relevant papers from other platforms were not encountered during this process.

Another threat may arise due to the terms used to search for relevant papers. After analyzing multiple articles, various synonyms for both GANs and anomalies were collected to reach the most number of papers possible. Despite this, there exists the possibility that some authors didn't use any of these terms to characterize their approach in the title, abstract or keywords of the document. This way, there is a small risk that some relevant papers might have eluded the search query.

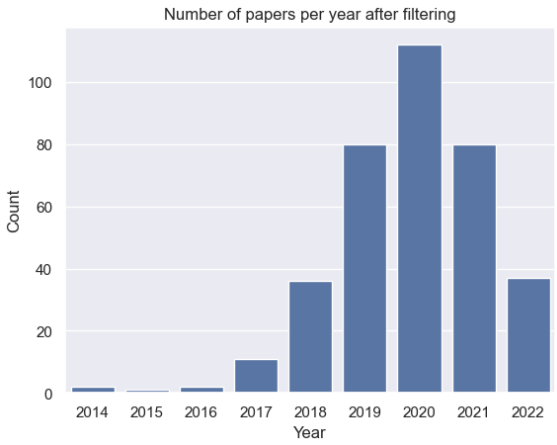

Figure 3.7: Number of reviewed papers after citation filter step (Fig. 3.1)

Similarly, by prohibiting documents that referenced images, or anything other than tabular data, some relevant papers might have been excluded. This can occur, for instance, in architectures that were firstly designed for image anomaly detection but were also suited for tabular datasets.

3.7 Summary

This chapter explained the methodology used to aggregate the papers relevant to the problem in question. First, a set of search questions were defined (sec. 3.1 followed by the respective query (sec. 3.2) that was used on Scopus. The selection criteria and the processing pipeline were delined in Section 3.3. A summary of each of the selected papers is provided in Section 3.4. Next, a categorization and summary of the entire process are performed in Section 3.5, followed by the threats to the SLR process (sec. 3.6).

Chapter 4

Research Proposal

This chapter will provide an overview of the problem this thesis is trying to solve. First, a brief analysis of the input data used by all methods will be presented in Section 4.1. Next, the existing methods for initial flow estimation will be presented in Sections 4.2 and 4.3. In Section 4.1, a preliminary analysis of the input data used in both approaches is undertaken. With the problems identified in the previous section, a hypothesis and research questions are proposed in Section 4.4, intending to solve these issues. Finally, the methodology and development plans are presented in Sections 4.5 and 4.6, respectively.

4.1 Magnetogram Data

The data used by all the methodologies consists of files with 640 entries ordered as the distance to the Sun. The data is divided into two parts: the input and the output. The input data comprises the magnetic field amplitude, B, the flux tube inclination, α, and the radial coordinate, R. The output data includes the number of charged particles per unit volume, n, the velocity, v, and the temperature, T. Table 4.1 presents the input and output data columns.

Some preliminary analysis of the input data for the model was carried out to demonstrate the existence of anomalies in the data used to train the prediction model of the previous section. The radial profiles of the magnetic field in the function of the radial coordinate radius, R, can be seen in Fig. 4.1. It is possible to observe some extreme variations in the data, which indicate the existence of abnormal samples. Additionally, the distribution of magnetic field, B, and flux tube inclination values, α, are expressed in Figure 4.2. There exist a large number of out-of-distribution samples in the dataset used for training the prediction model.

A similar process was carried out on the output data (refer to Table 4.1) to assert if it contained abnormal values that would also affect the training of the ML solution. Significant fluctuations can be observed in all instances of the output columns in Figure 4.3. The distribution for the output values can also be seen in Figure 4.4.

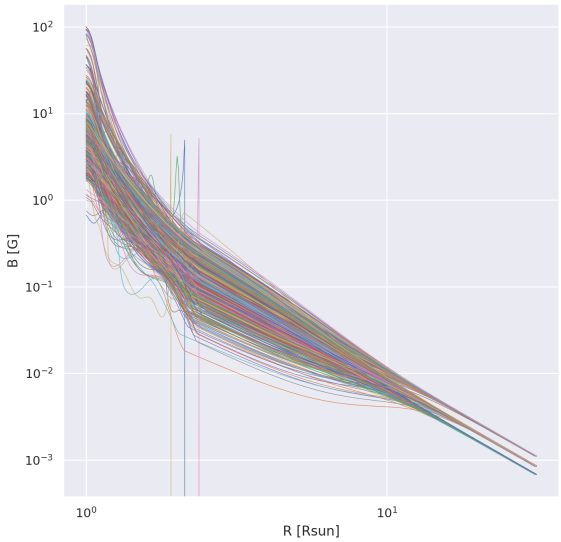

Figure 4.1: Radial profiles of the magnetic field amplitude for the dataset flux-tubes.

4.2 MULTI-VP

MULTI-VP [1] is a global MHD model that simulates the three-dimensional structures of the solar wind. In addition, it also estimates the conditions at the Sun's chromosphere, transition region, corona, and low heliosphere. The model computes many one-dimension solar wind solutions from full flux-tube geometries and heating functions. Background magnetic field geometries are extrapolated from publicly available magnetogram data. The method can estimate solar wind profiles across the Sun's entire atmosphere up to 30 solar radii. The results directly link the geometry of magnetic flux tubes in the lower corona with the distributions of fast and slow solar wind flows. MULI-VP proved faster than other MHD models and did not suffer from cross-field diffusion effects.

Table 4.1 presents the data used as input to the model. A representation of the data processing of MULTI-VP can be seen in Figure 4.5. As previously stated, MULTI-VP takes as input partial flow definitions of the solar wind along with initial expert estimations and derives better solutions during simulation.

4.3 ML for Initial Flow Estimation

Due to the complexity and a large number of calculations, MULTI-VP, like other MHD simulations, still takes a long time to reach solar wind solutions. Furthermore, the need for initial expert guesses also significantly delays the process. These factors directly affect the prediction capability and preparation for extreme solar events. Recently in [3], it has been proved that machine learning techniques can accurately produce good initial flow estimations that MULTI-VP can later use. The authors also proved that the quality of the flow estimations is directly linked to the total execution

Figure 4.2: Distribution of input values

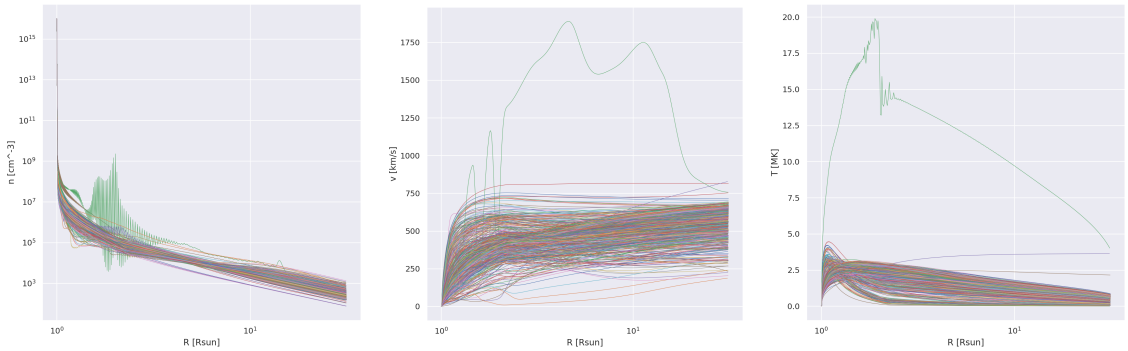

Figure 4.3: Plots of output variables in relation to R.

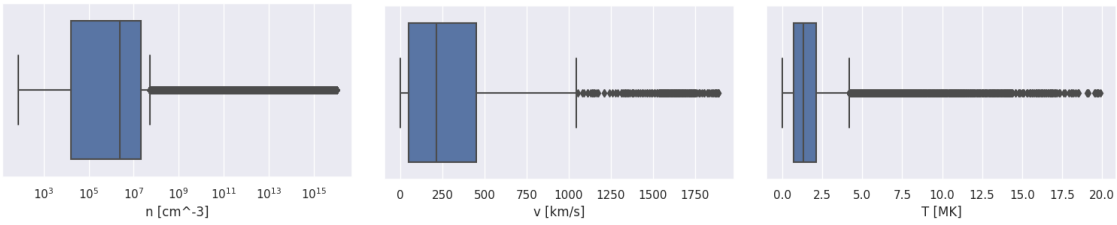

Figure 4.4: Distribution of outputs used during the training of the predictive model.

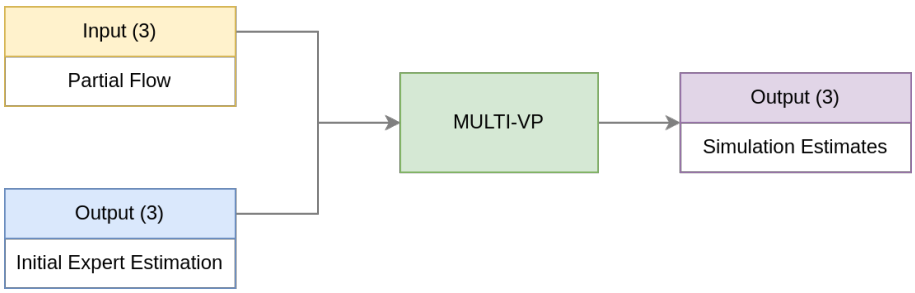

Figure 4.5: MULTI-VP methodology dataflow. The model takes the partial flow and its associated expert initial guess as input and then derives a better solution.

time of the simulation. These allow for faster convergence of the MHD simulation as the initial estimates are closer to the final solution.

The approach, illustrated in Fig. 4.6, uses an ML model to predict the initial expert estimates from the initial partial flow input. Analogous to the method presented in Section 4.2, MULTI-VP takes as input the partial flows along with their initial conditions predicted by the ML model.

An illustration of the training method can be seen in Figure 4.7. In this phase, the model takes as inputs initial partial flows and tries to predict the initial estimations. Next, the predictions for a given flow are compared to those from previous MULTI-VP runs to calculate the prediction's loss and update the model's parameters with backpropagation. The logic behind this was that the model would learn to predict initial estimations closer to the final solution, and thus, the simulation would converge faster.

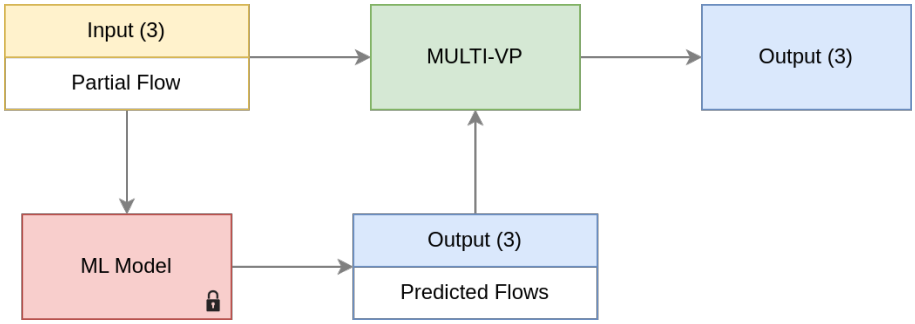

Figure 4.6: ML methodology dataflow. An ML model estimates the initial conditions of a given partial flow. These are then passed to MULTI-VP, approximating them into a final solution.

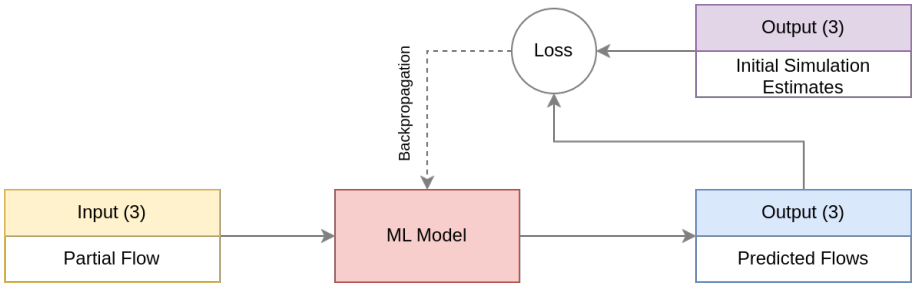

Figure 4.7: Model training method. Takes as input initial partial flows and tries to predict flow estimates close to the ones from previous MULTI-VP simulations.

However, the reduction in execution time was minimal as the model, like many other ANN solutions, was very susceptible to noise in the samples used during training. In [3], the authors posit that the presence of outliers during the training phase hampered the overall prediction quality of the model.

4.4 Hypothesis

Taking these problems into consideration, this thesis aims to evaluate the following hypothesis:

> GAN-based anomaly detection architectures can accurately and automatically detect outliers in the solar wind profiles data used to train the NN model for initial condition estimation and improve the model's predictions.

We will try to validate this hypothesis by (1) verifying that the dataset after outlier removal significantly improves the prediction quality of the RNN and (2) if the predictions with a normalized dataset reduce the time that MULTI-VP takes to reach a solution.

To validate the above hypothesis, the following research questions were defined:

RQ1 *Can GAN-based architectures accurately detect outliers in solar wind profiles?* Considering outlier samples as the positive class, we are mostly trying to reduce the False Negative Rate (FN), which causes the worst effects in the predictive model training. As a secondary priority, we will focus on reducing the amount of False Positives (FP) to ensure that almost no relevant samples are excluded from the training process.

RQ2 *Does the resulting dataset significantly improve the predictive ability of the RNN?* If the resulting dataset after the removal of outliers results in an improvement in the model used to predict initial conditions from input flows. In other words, the mean square distance between the real estimations and the ones predicted by the model is less than with the previous method.

RQ3 *Does the improved predictive ability of the RNN result in a further reduction of execution time for MULTI-VP?* This question aims to clarify if the developed model for initial flow estimation can produce better approximations of solar wind flows and thus reduce the time it takes for MULTI-VP to reach a solution.

4.5 Methodology

To evaluate the integrity of the hypothesis, we propose developing a GAN architecture to detect outliers in magnetogram files with an approach similar to the ones in [19], and [38]. It will use the generator and the discriminator to detect anomalies in the dataset.

The developed anomaly detector will be incorporated into the existing ML methodology expressed in Figure 4.6. The resulting representation of the overall data flow for the proposed approach can be seen in Figure 4.8. The goal is to use a GAN-based solution to detect outlier samples before training the ML models for initial flow estimation. We propose using the input values of the available magnetograms for this step. The detected anomalies will be removed from the dataset, and the RNN model will be trained with the remaining samples. The new predictions will be compared to the ones from the previous method to evaluate the improvement in the prediction quality. Finally, the predictions will be used as initial conditions for MULTI-VP, and the execution time will be compared to the previous method.

The GAN will have a similar architecture to the one in [11]. During training (Fig. 4.9), the generator's goal is to synthesize magnetogram samples close to the original distribution of the dataset. The discriminator will be tasked with distinguishing fake magnetograms from real ones.

For the detection phase, we propose the methodology in Figure 4.10 with the components trained in the previous stage. First, a sample will be extracted from the dataset and passed directly

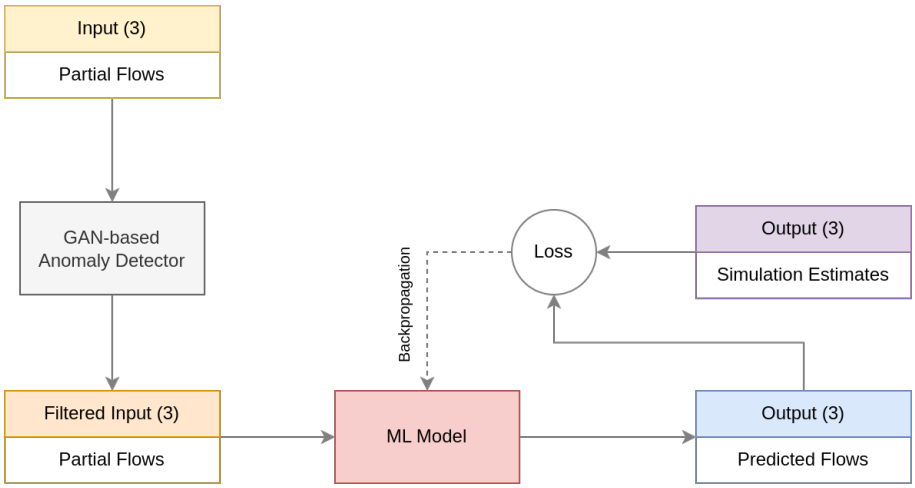

Figure 4.8: GAN-based outlier detection method dataflow.

to the discriminator. If the sample is abnormal, it will most likely have a higher loss than normal ones, as the discriminator only retained the normal data distribution. Next, the sample is mapped into the latent space, where one of the methods from [19] or [38] could be used. The resulting latent sample is then fed to the generator, which is tasked with reconstructing it. In the next step, the reconstructed sample is compared with the real one, and the reconstruction error is calculated. In theory, if the real sample is defective, then the reconstruction error will be much higher when compared to normal examples, as the generator only learned to generate samples from the normal distribution. In the end, discriminator loss and the reconstruction error will be combined to calculate the anomaly score. The example is considered an anomaly if this score exceeds a predefined threshold.

The results will be evaluated in an initial step by comparing the distance between the predicted initial estimation of the ML model and the real ones. Similar to the work carried out in [3], we will measure the Mean Squared Error (MSE) between the two. Subsequently, the filtered inputs from the outlier detection step will be fed to MULTI-VP along with the initial flow estimations from the ML model. Then the simulation execution times when using the proposed methodology will be compared with the previous implementations to assert if there was a significant reduction.

4.6 Work Plan

In Figure 4.11, an illustration of the proposed work plan is given. In the initial part of development, a more in-depth analysis of the input data will be performed. In addition, we will try to apply some clustering methods to determine if any subgroups in the dataset would facilitate the anomaly detection method.

In the next stage, a GAN architecture will be designed to detect outliers in magnetogram data. Pytorch [41] will be the tool used for this goal. Some time was also allocated to implementing GANs with this framework. After developing a suitable method for outlier detection on the given

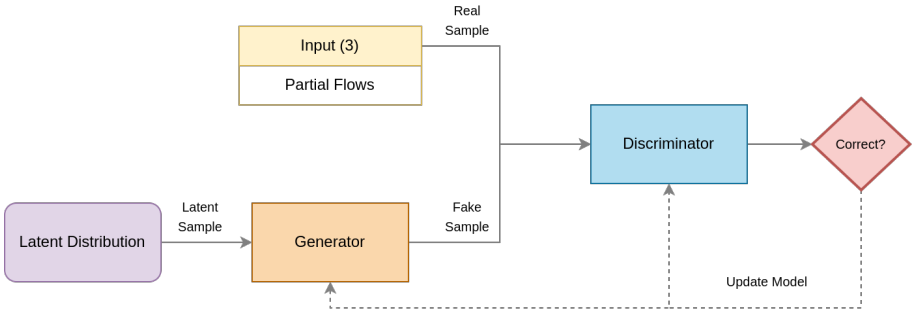

Figure 4.9: Proposed training method.

dataset, an extensive analysis of the performance of the ML model with the data without outliers will be done. This part will be used to validate the hypothesis defined in Section 4.4. During this time, some adjustments might have to be made to the initial solution to overcome any shortcomings on these tests. As was already stated, we will use the MSE between the actual estimations and those predicted with the new method.

After finetuning, the newly improved estimations from the previous step will be used on MULTI-VP to evaluate if the chosen implementation resulted in a shorter execution time. The final stage will be dedicated to writing the dissertation and accommodating any delays from the previous steps.

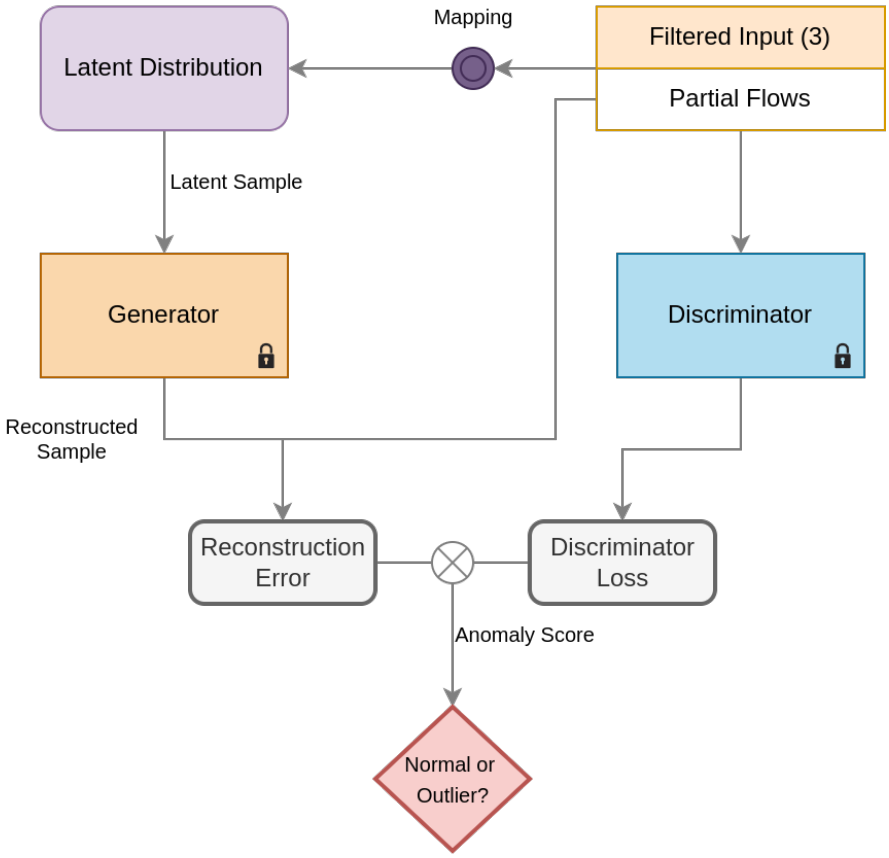

Figure 4.10: Proposed method for anomaly detection in magnetogram data.

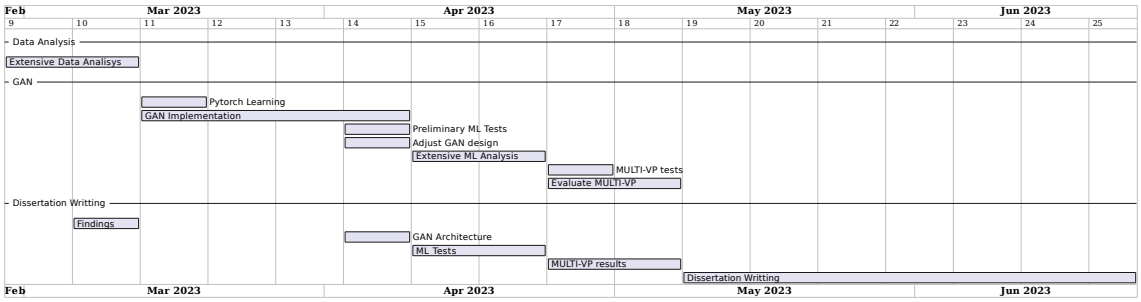

Figure 4.11: Proposed development plan.

Exploratory Data Analysis

An extensive analysis of the data that will be used during the development process was carried out. This chapter presents the results of this analysis, which are used to guide the development of the system.

4.7 MULTI-VP Dataset

The data used by MULTI-VP and, consequently, the RNN prediction model consists of magnetogram data from the Wilcox Solar Observatory. Each file contains 12 columns representing measurements of the magnetic field in the solar atmosphere at different heights. Every variable comprises 640 points (abscissas) measured at different radial distances from the Sun (up to 30 Solar radii). In addition, the data is distributed evenly throughout five profiles, each consisting of solar wind measurements at different surface locations. For this work, only six columns will be used, as the others are derivations of these and, thus, are obtained with these columns.

The data columns can be seen in Table 4.1. These are divided into two parts: the input and the output. The former comprises the set of variables used by the simulation to approximate solar wind conditions, and the latter the initial expert guesses needed to kickstart the multiple flux simulation. The input data includes the magnetic field amplitude, B, the flux tube inclination, α, and the radial coordinate, R. The output data consists of the number of charged particles per unit volume, n, the velocity, v, and the temperature, T.

Table 4.1: Data columns of magnetogram used by MULTI-VP.

Input (Partial Flows)			Output (Estimations)		
$R[R_{sun}]$	$B[G]$	$\alpha[deg]$	$n[cm^{-3}]$	$v[km/s]$	$T[MK]$

A joint plot of the input variables can be seen in Fig. 4.12. From these plots, it can be concluded that several input files constitute anomalous data. This is clear from the graph of B, with files that significantly deviate from the rest.

As previously explained, MULTI-VP requires initial expert guesses to kickstart the simulation. These guesses consist of the output variables presented in Table 4.1 and, during the simulation, are approximated to better solutions. In line with the work carried out in [3], we will be using the outputs of previous simulations as initial guesses (refer to chapter 4 for more details).

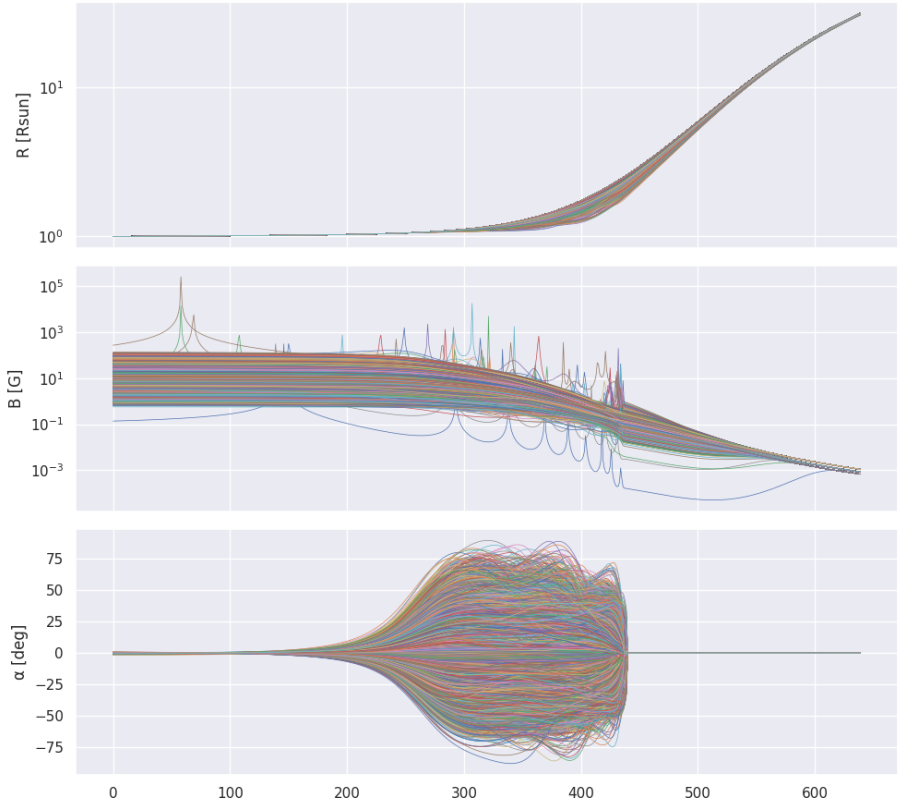

Figure 4.12: Joint plot of the input variables of each file used in this work. The first row is the plot of the radial coordinate radius, R, the second the magnetic field, B, and the last the flux tube inclination α. All are plotted in function of position in the magnetogram file.

Fig. 4.13 shows the joint plot of the output variables. Similar to the input plots, several faulty predictions can be seen in all variable plots, which are the result of simulations carried out on anomalous inputs.

A preliminary statistical analysis of the data can be seen in tables 4.2. In it, the mean, standard deviation, minimum, maximum, and quartiles of each variable are presented.

Out of the three input columns used, the magnetic field (B) has the highest standard deviation, which might indicate that this column is more prone to anomalies than the others. The radial coordinate (R) has the lowest standard deviation, which is expected, as it is a constant value for each file. The output variables have a similar standard deviation, with the number of charged particles per unit volume (n) having the highest and the temperature (T) the lowest. The velocity (v) has a standard deviation similar to that of the temperature.

In addition to these statistics, the correlation between the variables can be seen in Fig. 4.14. There is virtually no correlation between the input variables (R, B, and α), which is expected, as they are independent of each other.

The output variables (n, v, and T) are somewhat correlated. The temperature, T, is positively correlated with the velocity of the solar wind, v, which is expected as with higher wind speeds the particles there is a tendency for higher temperatures. The opposite is true for the volume density

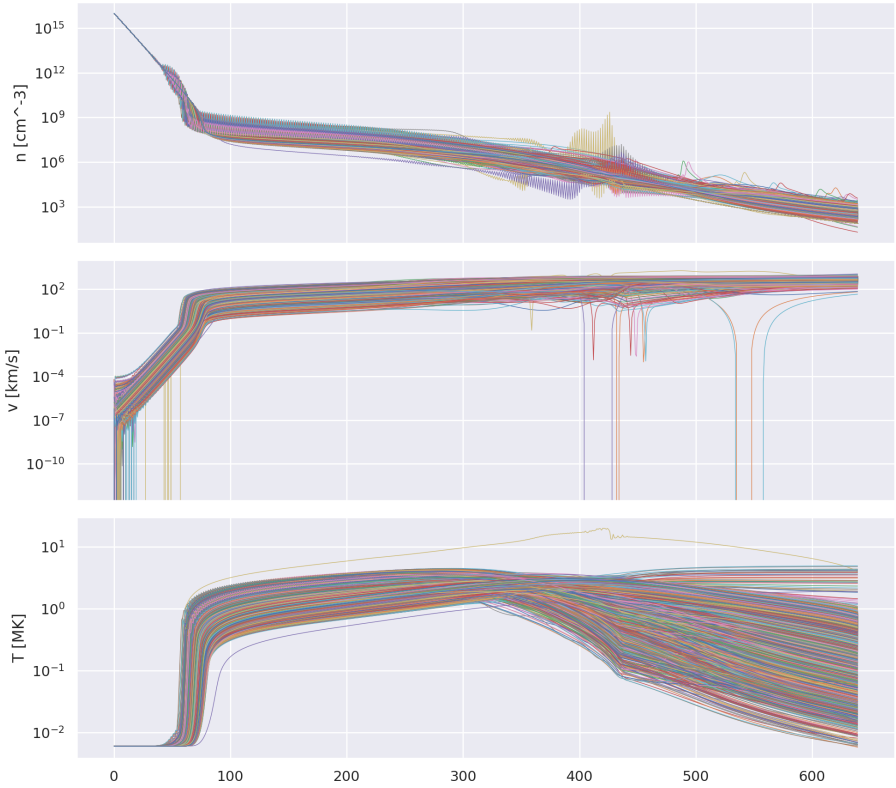

Figure 4.13: Joint plot of the output variables of each file used in this work. The first row is the plot of the number of charged particles per unit volume, n, the second the velocity, v, and the last the temperature, T. All are plotted in function of position in the magnetogram file.

of the solar wind, n, which is negatively correlated with the temperature. This is expected as with higher densities, the particles will have less kinetic energy and, therefore, a lower temperature. The velocity of the solar wind is also slightly negatively correlated with the volume density.

A high correlation between the radial coordinate R and the velocity v can be observed, as the velocity is expected to increase with distance from the Sun. Additionally, the temperature of the solar wind drops as the distance to the Sun increases, which is reflected in the negative correlation between R and T.

A distribution plot of each of the input variables can be seen in Fig. 4.15. The values of the radial coordinate R range from 1 to about 31.5 as was already seen by looking at Table 4.2, with most values being concentrated around 1.

Table 4.2: Statistical Analysis of the dataset.

	R [Rsun]	B [G]	α [deg]	$n[cm^{-3}]$	$v[km/s]$	$T[MK]$
mean	4.755	5.471	1.885	8.630e+13	2.553e+02	1.384
std	7.165	9.178e+01	1.472e+01	6.839e+14	2.148e+02	8.968e-01
min	1.000	5.122e-05	-8.763e+01	1.973e+01	-6.757e-03	5.765e-03
25%	1.021	4.051e-02	-1.079e-01	1.622e+04	4.926e+01	7.179e-01
50%	1.151	2.095	0.000	2.351e+06	2.110e+02	1.337
75%	4.250	5.582	9.997e-01	2.132e+07	4.508e+02	2.098
max	3.150e+01	2.470e+05	8.925e+01	1.010e+16	1.889e+03	1.990e+01

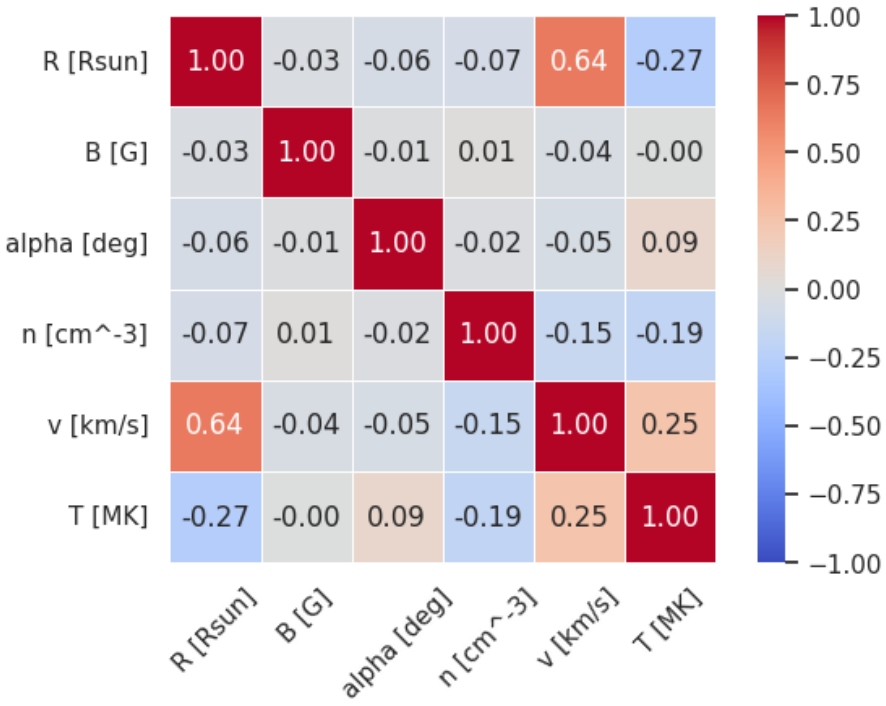

Figure 4.14: Correlation plot of all variables used in this work.

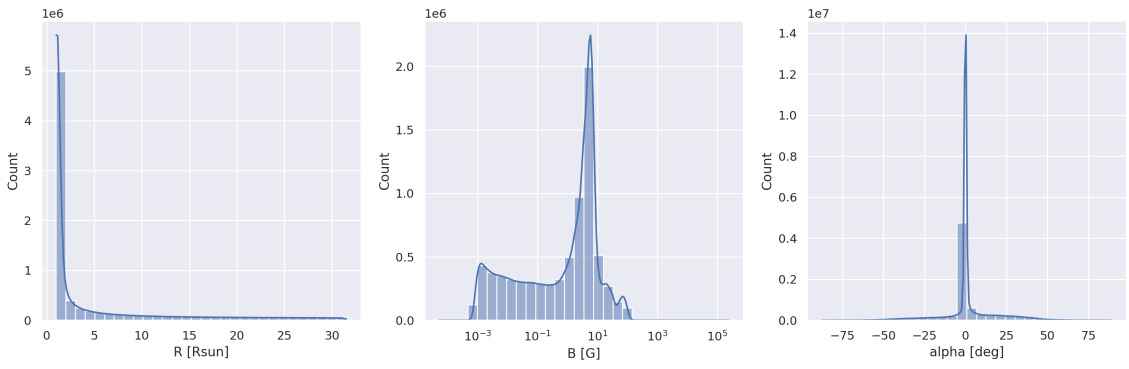

Figure 4.15: Distribution of the input variables

Chapter 5

Research Statement

Chapter 6

Clustering

To better understand the data and to help with the task of finding anomalies, we will be using clustering techniques. These techniques are used to group data points that are similar to each other. The goal is to find groups of data points that are similar to each other, but different from the rest of the data. This is done by finding the distance between each data point and the rest of the data. The distance between two data points is calculated using a distance metric.

In this chapter, the clustering methods that were used will be briefly explained in section 6.1. Next, the dimensionality reduction methods used in some of the tests will be presented in section 6.2.

6.1 Clustering Methods

Clustering is defined as the task of finding and then grouping the data values that are close to each other in some way. Because of this, clustering is not only a useful technique for data analysis but also for the task of anomaly detection, as abnormal samples tend to be far from the rest of the data. In this section, the clustering methods that were used in this work will be presented.

6.1.1 K-Means

The objective of K-means is to find a partition of the data into K clusters, where each data point belongs to the cluster with the nearest mean. The K-means algorithm is an iterative algorithm that starts by randomly assigning each data point to a cluster. Then, the mean of each cluster is calculated and the data points are reassigned to the cluster with the nearest mean. This process is repeated until the data points stop changing clusters.

The number of divisions is defined by the k parameter, which can be determined with the help of the following methods:

- **Elbow Method:** The elbow method is a heuristic method that is used to find the optimal number of clusters. This method plots the sum of squared errors (SSE) for each value of k and then finds the value of k that has the "elbow" in the plot. This value of k is considered

to be the optimal number of clusters. This method presents some disadvantages, as it is not always clear where the "elbow" is, making it a subjective method. Also, the SSE is not a good measure of the quality of the clustering, as it is biased towards clusters with similar sizes.

- **Silhouette Method:** The silhouette method is a method that is used to find the optimal number of clusters. This method calculates the silhouette coefficient for each data point, by evaluating the intra-cluster and inter-cluster distances. Scores for each point can range from -1 to 1, with lower values indicating that the data point is in the wrong cluster and higher values that it is in the correct cluster. The silhouette coefficient for each cluster is then calculated by averaging the silhouette coefficients of the data points in the cluster. Finally, the score for the entire dataset is calculated by averaging the coefficients from every cluster. The optimal number of clusters is the value of k that has the highest silhouette coefficient.

K-means is advantageous when compared to other clustering methods because of its simplicity and the fact that it doesn't require any prior knowledge of the data. However, the disadvantages are that it is biased towards spherical clusters with similar sizes, is sensitive to the initial position of cluster centroids and requires the determination of the number of clusters beforehand.

6.1.2 SOM

This method is based on the idea of self-organizing maps, which are artificial neural networks used to find a low-dimensional representation of a high-dimensional space. It is usually employed to reduce the dimensions of the data to a map, but can also be used as a clustering method, as it groups similar data.

The algorithm starts by randomly assigning each data point to a neuron in the map. Then, the weights of each neuron are updated to be closer to the data point that it represents. This process is repeated until the data points stop changing neurons.

The advantage of this algorithm is that it can map high-dimensional input vectors to a low-dimensional space while preserving the original topology of the data. Unlike other clustering methods, which aren't able to provide good results for data with high dimensionality.[1]

6.1.3 Agglomerative Clustering

Agglomerative clustering is a hierarchical clustering method that starts by assigning each data point to its own cluster. Then, the two clusters that are closest to each other are merged into a single cluster. This process is repeated until all the data points are in the same cluster. These are combined by comparing intra-cluster and inter-cluster distances. Distance between the clusters is calculated using a linkage function, and the distance between the data points in the clusters is

[1] Self-Organizing Maps https://sites.pitt.edu/~is2470pb/Spring05/FinalProjects/Group1a/tutorial/som.html

calculated using a distance metric (normally the Euclidean distance). The size and shape of the clusters are directly influenced by these two parameters.

The number of clusters can be determined with the help of a dendogram plot. This plot shows the distance between the clusters as they are merged. Each leaf in the dendrogram represents a data point that is fused with a similar point into the same cluster as the height of the tree increases. The height of the fusion (on the vertical axis) represents the dissimilarity score between the two clusters. Higher fusions indicate that the clusters are more distinct. As a rule of thumb, the number of clusters can be determined by looking at the height of the fusion(s) that has the most distinct clusters (higher dissimilarity scores).[2]

One of the problems with this method is that there is no clear way to determine the number of clusters, as the dendrogram doesn't always produce clear divisions, making the process of choosing the number of clusters subjective.

6.1.4 DBSCAN

This unsupervised clustering method was first introduced in Ester et al. [42]. It is a density-based clustering algorithm that is used to find clusters of data points that are close to each other. The algorithm assigns "core" labels to the points that have at least $minPts$ points within a distance eps from them. Points that are reachable from a core point, but do not have at least $minPts$ points within a distance eps from them, are assigned "border" labels. These are considered to be a part of the cluster formed by the core point.

The points that are not reachable from any core point are assigned "noise" labels and are considered to be outliers. Because of this, the algorithm provides a basic method for outlier detection.

The algorithm is advantageous because it doesn't require the determination of the number of clusters beforehand, as it can find clusters of any size and is also robust to the existence of outliers. Despite this, it is sensitive to the parameters $minPts$ and eps, with slight changes in these parameters having a significant impact on the results.

6.2 Dimensionality Reduction

In Data Science and Machine Learning, dimensionality refers to the number of features of the dataset. The dimensionality of a dataset can be reduced by removing features that are not relevant to the problem at hand. This can be done by using feature selection methods, which are methods that select the most relevant features for the task at hand. However, this can also be done by using dimensionality reduction methods, which are methods that transform the data into a lower-dimensional space, while preserving the most important information.

[2] Agglomerative Clustering https://www.datanovia.com/en/lessons/agglomerative-hierarchical-clustering/

These processes are useful in these contexts because they can reduce the computational cost of the algorithms, as well as the time needed to train the models. They can also improve the performance of the algorithms, as they can remove noise and irrelevant features from the data.

From the data analysis in Section 4.7, it can be seen that each line, representing a distinct variable, has 640 abscissas which translate to the same number of features. This is an extremely high number of features that can be problematic for some algorithms, as the features are not all equally relevant to the problem at hand. Therefore, it is important to reduce the dimensionality of the dataset, so that the algorithms can be trained more efficiently and efficiently.

In this case, variables with the same repeating values with be considered less important and be discarded by the dimensionality reduction algorithm, giving more emphasis on other variables that have more variation. This will be the case for the R variable which is almost constant throughout every line measurement and thus provides almost no meaningful information.

For this work, we will be using the following dimensionality reduction methods:

- **PCA:** Principal Component Analysis is a linear dimensionality reduction method that uses Singular Value Decomposition to project the data into a lower-dimensional space. It is advantageous because it is fast and efficient, but it is also sensitive to outliers.
- **t-SNE:** t-Distributed Stochastic Neighbor Embedding is a non-linear dimensionality reduction method that is based on the idea that similar data points should be close to each other in the lower-dimensional space. It is advantageous because it can preserve the topology of the data, but it is also computationally expensive.

6.3 Experiments

After the identification of the most common clustering and dimensionality reduction methods, the next step is to apply them to the dataset and evaluate their performance. This section will describe the experiments that were carried out to try to better understand the dataset.

6.3.1 Time Series Methods

The first method to be tested was the *TimeSeriesKmeans* clustering algorithm from *tslearn*[3]. As the name indicates, this algorithm is a variation of the K-means algorithm that is used for time series data. It is advantageous as it can cluster time series data without the need to transform it into a vector, which can be problematic because it can lead to the loss of information. Despite this, the method is still based on the K-Means algorithm presented in Section 6.1.1 and presents the same disadvantages as the original algorithm.

Clustering was conducted on the B and α variables of the dataset ranging from 2 to 6 clusters. It is difficult to determine the number of appropriate clusters for the dataset beforehand as this

[3]TimeSeriesKMeans https://tslearn.readthedocs.io/en/stable/gen_modules/clustering/tslearn.clustering.TimeSeriesKMeans.html

method has no technique for their determination, unlike the regular K-Means. Therefore, the number of clusters was determined by looking at the results and trying to find the number of clusters that best represented the data.

In Figure **??**

6.4 Results

Chapter 7

Conclusions

The necessity to consistently predict the Sun's conditions that lead to the most extreme events has become an increasingly important study. However, technological difficulties make obtaining real-time data from the Sun's surface challenging. Multiple numeric simulators have tried to fill this gap by extrapolating these conditions based on limited observations from Earth. In this dissertation, we have explained the problems (sec. 1.2) associated with these solutions that severely affect the ability to generate solar estimations promptly. These issues included the long execution time of the simulation models as well as the need for initial expert estimations. Additionally, it was posited that the use of machine learning techniques to predict the original conditions suffered greatly from anomalies in the training dataset.

To address these issues, we first studied the current most popular GAN-based techniques for anomaly detection. The state-of-the-art analysis (sec. 3.4) showed that GAN-based anomaly detection techniques performed the task better than other mainstream methods. With this in mind, we proposed a GAN-based anomaly detection technique (sec. 4.5) that detects anomalies in the training dataset used by the prediction models. In the same section, the evaluation methods that will be used for the validation of the results were also explained. Finally, the work plan (sec. 4.6) was presented, which included the tasks that will be performed to achieve the objectives of the dissertation.

References

[1] Rui F. Pinto and Alexis P. Rouillard. A Multiple Flux-tube Solar Wind Model. *The Astrophysical Journal*, 838(2):89, 2017.

[2] D. Odstrčil and V. J. Pizzo. Three-dimensional propagation of coronal mass ejections (CMEs) in a structured solar wind flow: 1. CME launched within the streamer belt. *Journal of Geophysical Research: Space Physics*, 104(A1):483–492, 1999.

[3] Ana Filipa Sousa Barros. Initial Condition Estimation in Flux Tube Simulations using Machine Learning, 2021.

[4] Rainer Schwenn. Space Weather: The Solar Perspective. *Living Reviews in Solar Physics*, 3(1):2, 2006.

[5] Rushil Anirudh, Rick Archibald, M. Salman Asif, Markus M. Becker, Sadruddin Benkadda, Peer-Timo Bremer, Rick H. S. Budé, C. S. Chang, Lei Chen, R. M. Churchill, Jonathan Citrin, Jim A. Gaffney, Ana Gainaru, Walter Gekelman, Tom Gibbs, Satoshi Hamaguchi, Christian Hill, Kelli Humbird, Sören Jalas, Satoru Kawaguchi, Gon-Ho Kim, Manuel Kirchen, Scott Klasky, John L. Kline, Karl Krushelnick, Bogdan Kustowski, Giovanni Lapenta, Wenting Li, Tammy Ma, Nigel J. Mason, Ali Mesbah, Craig Michoski, Todd Munson, Izumi Murakami, Habib N. Najm, K. Erik J. Olofsson, Seolhye Park, J. Luc Peterson, Michael Probst, Dave Pugmire, Brian Sammuli, Kapil Sawlani, Alexander Scheinker, David P. Schissel, Rob J. Shalloo, Jun Shinagawa, Jaegu Seong, Brian K. Spears, Jonathan Tennyson, Jayaraman Thiagarajan, Catalin M. Ticoş, Jan Trieschmann, Jan van Dijk, Brian Van Essen, Peter Ventzek, Haimin Wang, Jason T. L. Wang, Zhehui Wang, Kristian Wende, Xueqiao Xu, Hiroshi Yamada, Tatsuya Yokoyama, and Xinhua Zhang. 2022 Review of Data-Driven Plasma Science, May 2022.

[6] D.N. Baker. What is space weather? *Advances in Space Research*, 22(1):7–16, 1998.

[7] Mark Moldwin. *An Introduction to Space Weather*. Cambridge University Press, 2008.

[8] Eric Priest. *The Solar Wind*. Cambridge University Press, 2014.

[9] Sami K. Solanki, Bernd Inhester, and Manfred Schüssler. The solar magnetic field. *Reports on Progress in Physics*, 69(3):563–668, March 2006. arXiv:1008.0771 [astro-ph].

[10] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, May 2015. Number: 7553 Publisher: Nature Publishing Group.

[11] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks, 2014.

[12] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks, 2016.

[13] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 214–223. JMLR.org, 2017.

[14] Divya Saxena and Jiannong Cao. Generative adversarial networks (gans): Challenges, solutions, and future directions. *ACM Comput. Surv.*, 54(3), may 2021.

[15] Claire Little, Mark Elliot, Richard Allmendinger, and Sahel Shariati Samani. Generative Adversarial Networks for Synthetic Data Generation: A Comparative Study, 2021.

[16] Charu C. Aggarwal. *Outlier Analysis*. Springer New York, 2013.

[17] Xuan Xia, Xizhou Pan, Nan Li, Xing He, Lin Ma, Xiaoguang Zhang, and Ning Ding. GAN-based anomaly detection: A review. *Neurocomputing*, 493:497–535, July 2022.

[18] Victoria Hodge and Jim Austin. A survey of outlier detection methodologies, oct 2004.

[19] D. Li, D. Chen, B. Jin, L. Shi, J. Goh, and S.-K. Ng. MAD-GAN: Multivariate Anomaly Detection for Time Series Data with Generative Adversarial Networks. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11730 LNCS:703–716, 2019.

[20] Houssam Zenati, Chuan Sheng Foo, Bruno Lecouat, Gaurav Manek, and Vijay Ramaseshan Chandrasekhar. Efficient GAN-Based Anomaly Detection, 2018.

[21] Houssam Zenati, Manon Romain, Chuan-Sheng Foo, Bruno Lecouat, and Vijay Chandrasekhar. Adversarially Learned Anomaly Detection. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 727–736, 2018.

[22] J. Audibert, P. Michiardi, F. Guyard, S. Marti, and M.A. Zuluaga. USAD: UnSupervised Anomaly Detection on Multivariate Time Series. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 3395–3404, 2020.

[23] Y. Liu, Z. Li, C. Zhou, Y. Jiang, J. Sun, M. Wang, and X. He. Generative Adversarial Active Learning for Unsupervised Outlier Detection. *IEEE Transactions on Knowledge and Data Engineering*, 32(8):1517–1528, 2020.

[24] Shuokang Huang and Kai Lei. IGAN-IDS: An imbalanced generative adversarial network towards intrusion detection system in ad-hoc networks. *Ad Hoc Networks*, 105:102177, 2020.

[25] W. Jiang, Y. Hong, B. Zhou, X. He, and C. Cheng. A GAN-Based Anomaly Detection Approach for Imbalanced Industrial Time Series. *IEEE Access*, 7:143608–143619, 2019.

[26] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial Feature Learning, 2017.

[27] Alexander Geiger, Dongyu Liu, Sarah Alnegheimish, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. TadGAN: Time Series Anomaly Detection Using Generative Adversarial Networks. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 33–43, 2020.

[28] X. Wang, Y. Du, S. Lin, P. Cui, Y. Shen, and Y. Yang. adVAE: A self-adversarial variational autoencoder with Gaussian anomaly prior knowledge for anomaly detection. *Knowledge-Based Systems*, 190, 2020.

[29] Andrew Blance, Michael Spannowsky, and Philip Waite. Adversarially-trained autoencoders for robust unsupervised new physics searches. *Journal of High Energy Physics*, 2019(10):47, 2019.

[30] J. Wu, Z. Zhao, C. Sun, R. Yan, and X. Chen. Fault-Attention Generative Probabilistic Adversarial Autoencoder for Machine Anomaly Detection. *IEEE Transactions on Industrial Informatics*, 16(12):7479–7488, 2020.

[31] Phuc Cuong Ngo, Amadeus Aristo Winarto, Connie Khor Li Kou, Sojeong Park, Farhan Akram, and Hwee Kuan Lee. Fence GAN: Towards Better Anomaly Detection. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 141–148, 2019.

[32] Ilias Siniosoglou, Panagiotis Radoglou-Grammatikis, Georgios Efstathopoulos, Panagiotis Fouliras, and Panagiotis Sarigiannidis. A Unified Deep Learning Anomaly Detection and Classification Approach for Smart Grid Environments. *IEEE Transactions on Network and Service Management*, 18(2):1137–1151, 2021.

[33] Shaowei Liu, Hongkai Jiang, Zhenghong Wu, and Xingqiu Li. Data synthesis using deep feature enhanced generative adversarial networks for rolling bearing imbalanced fault diagnosis. *Mechanical Systems and Signal Processing*, 163:108139, 2022.

[34] S.K. Lim, Y. Loo, N.-T. Tran, N.-M. Cheung, G. Roig, and Y. Elovici. DOPING: Generative Data Augmentation for Unsupervised Anomaly Detection with GAN. In *Proceedings - IEEE International Conference on Data Mining, ICDM*, volume 2018-November, pages 1122–1127, 2018.

[35] Chuanlong Yin, Yuefei Zhu, Shengli Liu, Jinlong Fei, and Hetong Zhang. An enhancing framework for botnet detection using generative adversarial networks. In *2018 International Conference on Artificial Intelligence and Big Data (ICAIBD)*, pages 228–234, 2018.

[36] Yuxuan Yuan, Kaveh Dehghanpour, Fankun Bu, and Zhaoyu Wang. Outage Detection in Partially Observable Distribution Systems Using Smart Meters and Generative Adversarial Networks. *IEEE Transactions on Smart Grid*, 11(6):5418–5430, 2020.

[37] Fanhui Kong, Jianqiang Li, Bin Jiang, Huihui Wang, and Houbing Song. Integrated Generative Model for Industrial Anomaly Detection via Bidirectional LSTM and Attention Mechanism. *IEEE Transactions on Industrial Informatics*, 19(1):541–550, 2023.

[38] Md Abul Bashar and Richi Nayak. TAnoGAN: Time Series Anomaly Detection with Generative Adversarial Networks. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1778–1785, 2020.

[39] Chu Wang, Yan-Ming Zhang, and Cheng-Lin Liu. Anomaly Detection via Minimum Likelihood Generative Adversarial Networks. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 1121–1126, 2018.

[40] Qinghua Xu, Shaukat Ali, and Tao Yue. Digital Twin-based Anomaly Detection in Cyber-physical Systems. In *2021 14th IEEE Conference on Software Testing, Verification and Validation (ICST)*, pages 205–216, 2021.

[41] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[42] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, page 226–231. AAAI Press, 1996.