Autoencoders in Discovering New Physics

Luca Vicaria

10828724

The University of Manchester

BSc Dissertation

November 2023

Abstract

This dissertation presents an in-depth exploration of the role of autoencoders in anomaly detection within the realm of particle physics, aiming to uncover new physics phenomena. Starting with the foundational principles of machine learning, this research delves into the intricate challenges of anomaly detection, focusing on the architecture and functionality of autoencoders. This study rigorously investigates the integration of machine learning in particle physics, finishing with a practical demonstration of an autoencoder detecting anomalies. Through this research, specific anomalies of interest to physicists are identified and analysed, showcasing the potential of autoencoders in revolutionising our understanding of the universe. The findings demonstrate the promising capabilities and future applications of autoencoders in particle physics, suggesting new avenues for exploration and discovery in the field.

1. Introduction

In the ever-evolving world of physical sciences, the advent of machine learning has paved the way for a new era of discovery and innovation. Machine learning's role in identifying intricate patterns and anomalies is particularly interesting and plays a vital role in the physical sciences. This is particularly true in the domain of particle physics where the mysteries of the universe are continually unravelled. The extensive range of applications of machine learning techniques offers a promising avenue for groundbreaking revelations.

The aim of understanding the building blocks of the universe has always been at the forefront of scientific exploration. Diving deep into the subatomic world, scientists have discovered a vast array of particles, each with their own unique properties and behaviours. Yet, among the large range of discoveries, anomalies and deviations from the established models often emerge, challenging our current understanding and hinting at the existence of new physics. These anomalies are often subtle and hidden within vast datasets and therefore require advanced tools for detection and analysis.

Within this field of particle physics, anomaly detection is a critical area of study and plays a crucial role in being able to identify deviations from established models, which could one day revolutionise our knowledge about the universe. However, traditional methods often struggle to handle the complexity and vast amount of data in particle physics experiments. This is where autoencoders, a type of machine learning algorithm, come into play. Their potential capabilities in anomaly detection are extreme, especially in the identification of deviations from the standard model that currently omits gravity and lacks explanations for phenomena like dark matter, dark energy and neutrino mass [1]. This not only signals the emergence of new physics but also underscores an exhilarating frontier in scientific research.

The goal of this dissertation is to bridge the gap between machine learning's potential and the intricate challenges faced in particle physics. Primarily, we will explore and focus on the utilisation of autoencoders in anomaly detection, within the field of particle physics. Additionally, we aim to understand the implications that these techniques have on the discovery of new physics.

In the subsequent chapters, we will delve deeper into the literature surrounding machine learning in particle physics, explore the architecture and functionality of autoencoders, and present a hands-on demonstration of their application in detecting anomalies. Through this the research aims to shed light on the promising capabilities of autoencoders in the realm of particle physics.

2. Fundamentals of Machine Learning

2.1 What is Machine Learning?

Machine learning is a specialised branch of artificial intelligence that relies on specific algorithms and statistical models that allow computers to execute tasks without relying on explicit instructions. Instead, these systems draw on patterns and inferences from data that allow them to learn and adapt in a way that imitates the way humans do [2].

Modern machine learning is transforming the workplace by automating tasks and simulating complex processes, saving significant time and effort. Because these intelligent systems can now perform tasks that once required a human brain, the nature of work itself is beginning to evolve, demanding new skills from the workforce. As a consequence, continuous learning has become essential for employees to keep pace with the ever-changing landscape where machines handle routine and repetitive tasks, resulting in humans focusing on more strategic activities.

2.2 The Fundamental Principles of Machine Learning

In the domain of machine learning there are a few key concepts that are always considered such as data representation, model selection, the loss function, optimisation and generalisation. These five techniques are employed in any form of machine learning.

Data can manifest in various structural forms, from simple vectors to multi-dimensional tensors, but the efficiency and success of algorithmic models is dictated by the chosen data representations. A consequence of the dataset being poorly represented is that the model may fail to recognise important features, leading to inaccurate outcomes or biased predictions due to overlooked variables or relationships [3].

Next is model selection, this is dictated by the intrinsic characteristics of the data and the specificity of the task at hand. A machine learning model is a mathematic framework that learns from data in order to make future predictions or decisions. Ranging from the simplicity of linear models to the depth and complexity of neural networks, it is important that the right model is chosen to ensure the defining features of the dataset are detected. This will allow for the minimisation of captured noise and under or overfitting [4].

Equally critical is the deployment of the loss functions, which are used for determining the accuracy of the algorithm. The loss function quantifies the difference between the predicted values and the actual values, guiding the training process of the algorithm towards improved accuracy. If an inappropriate loss function is chosen, the model might optimise for the wrong objective, leading to suboptimal predictions and potentially misleading results. Example loss functions include the Mean Squared Error (MSE) for regression or Cross-Entropy for classification [5]. Below is the equation for the MSE:

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (x_i - \hat{x}_i)^2, \tag{1}$$

where n represents the number of samples, x_i is the actual input value, and $\hat{x}_i$ is the reconstructed output from the autoencoder [5].

Optimisation refers to the process of adjusting the model parameters to minimize the loss function. It is essential for finding the best version of the model that can make the most accurate predictions. Poor optimisation can lead to either non-converging models, where the solution is never reached, or to local minima, where the model settles for a suboptimal solution that do not represent the best possible outcome [5].

Finally, generalization is the ability of a machine learning model to perform well on new unseen data, not just the data on which it was trained. This is the ultimate goal of a machine learning model, indicating its applicability and robustness in a multitude of real-world scenarios. Without proper generalisation, a model may perform exceptionally well on training data but fail miserably in practical applications, which is known as overfitting. This can render the model useless if not taken into account.

2.3 Types of Machine Learning

Within machine learning there are a range of techniques that exist, however the most prevalent are supervised and unsupervised learning due to their wide range of applications.

2.3.1 Supervised Learning

This is when the model is trained on a labelled data, which means that the data includes inputs and the correct outputs, allowing the model to continuously adjust and improve [6]. Supervised learning can be further broken down into two subcategories; classification and regression.

Classification is the process of taking an input and mapping it to a discrete label and is therefore used when some input needs to be sorted into a specific category, e.g. true or false. Prominent algorithms in this category include linear classifiers, which establish decision boundaries in feature space (feature space is a multidimensional space where each feature of the input data is represented by a dimension); decision trees, which segment the space based on feature values; and Naïve Bayes classifiers, which apply probabilistic models based on Bayes' theorem [7] [8].

Conversely, regression techniques are used with the aim of forecasting a continuous dependent variable from a set of independent variables [9]. For instance, linear regression, a fundamental regression technique, predicts an outcome based on the linear relationship between input variables. It is extremely valuable when forecasting outcomes such as housing prices based on property features or stock prices based on historical data [10].

However, supervised learning is not perfect. Because the datasets are labelled, they have a higher likelihood of being subject to human error or bias which would result in the model being trained incorrectly.

2.3.2 Unsupervised Learning

In contrast, unsupervised learning is when the model is trained on data that is not labelled, meaning that it must determine the structure and patterns of the data independently. This type of learning can be broadly categorised into clustering, association and dimensionality reduction [11].

Clustering is the process of grouping a set of data points in such a way that objects in the same group, or cluster, are similar according to some specified metric and the main goal of this is to discover the natural structures within the data. This is particularly useful for exploratory data analysis such as grouping genes with similar expression patterns. A common algorithm in clustering is K-means, which splits the data into clusters with the nearest mean, grouping data points based on their proximity to the nearest cluster centre. [12].

In contrast, association is a rule-based method that aims to uncover relationships and patterns (associations) between items within the dataset. It’s often used in market basket analysis, which finds associations between products that frequently occur together in transactions. An example of this is the Apriori algorithm [13].

Finally, dimensionality reduction focuses on the simplifying the input data by reducing the number of dimensions, or features, without losing the intrinsic structure of the data. This technique is beneficial for visualising high-dimensional data, increasing the speed of learning algorithms, and reducing noise. This makes dimensionality reduction extremely useful for data analysis and is frequently used in particle physics. Principal Component Analysis (PCA) is one of the most widely used methods for dimensionality reduction; it transforms the data into a set of linearly uncorrelated variables known as principal components, ranked by variance [12].

Unsupervised learning algorithms can uncover hidden patterns in data without needing labelled datasets, but they also have their own set of challenges. One major issue is that because there are no predefined labels, the algorithms used must be much more complex in order for them to still determine the underlying patterns of the data to a high standard of accuracy.

2.3.3 Other Machine Learning Types

Two less popular types of machine learning are semi-supervised and reinforcement learning. Semi-supervised learning combines labelled and unlabelled data for training, offering a cost-effective approach when labelled data is scarce. It leverages labelled examples for explicit guidance while using the larger pool of unlabelled data to improve model generalization [3]. On the other hand, Reinforcement Learning focuses on agents learning to make sequences of actions in an environment to maximise the total sum of rewards over time. It is employed in scenarios where actions have consequences, like robotics or game playing, and involves a continuous process of trial-and-error to develop optimal decision-making policies [11].

In conclusion, machine learning can be applied in a numerous number of ways, making it applicable in almost every professional field. This characteristic makes it extremely valuable and the centre of modern research.

3. Neural Networks

Neural networks are a branch of machine learning that consist of a large series of interconnecting neurons in such a way that is inspired by the human brain. They are structured in layers of neurons consisting of an input layer, that receives the initial data, several hidden layers that process the data and finally an output layer that delivers the final prediction [14].

3.1 Neutrons: The Building Blocks

In the realm of neural networks, neurons are the core processing units, analogous to a linear regression model where the relationship between input variables and the expected outcome is determined. Here, neurons receive a vector input, and based on internal parameters (weights and biases), they generate a scalar output. To illustrate, a neuron's operation can be summarised by the following equation:

$$\sum \omega_i x_i + bias = \omega_1 x_1 + \omega_2 x_2 + \omega_3 x_3 + \ldots + bias \, , \quad (2)$$

$$output = \mathrm{f(x)} = \begin{cases} 1 \; if \; \sum_i \omega_i x_i + bias \geq 0 \\ 0 \; if \; \sum_i \omega_i x_i + bias \leq 0 \end{cases} , \quad (3)$$

where ω_i represents the weight of input x_i [14].

Weights signify the influence of inputs; the larger the weight, the more significant its impact on the output, whereas the bias acts as an adjuster that shifts the activation function, helping the model to better fit the data. A neuron calculates its output by multiplying the inputs by their respective weights, adding a bias, and passing the result through an activation function that dictates whether a neuron will activate and "fire," sending data forward or not. The choice of activation function is crucial as they have the ability to introduce non-linearity into the model, which is important as real-life data is often non-linear, allowing the network to capture complex patterns within the data [14].

3.2 Basic Architecture and Data Processing

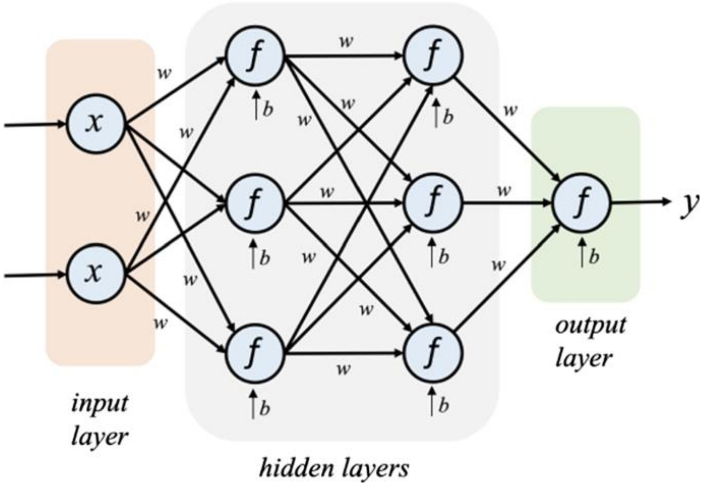

Figure 1. Basic Neural Network Architecture. [15] H. T. Ünal and F. Başçiftçi, "Evolutionary design of neural network architectures: a review of three decades of research," Artif Intell Rev 55, 2022, pp. 1727. [Online]. Available: https://doi.org/10.1007/s10462-021-10049-5. Accessed: Oct. 23, 2023.

Figure 1 depicts the architecture of a basic neural network, demonstrating how each node is connected and has an associated weight and bias [15].

The architecture of a neural network is defined by its ability to feed data forward through these layers. As the input is processed, each neuron's output becomes the next neuron's input. This sequential data processing is what categorises the network as a feed-forward network.

3.3 Training and Optimisation of a Neural Network

During training, a neural network undergoes optimization to minimise a predefined loss function that quantifies the error between the models' predictions and the target values. The choice of loss function is dependent on the nature of problem and network architecture. A common technique is a gradient descent-based method, which is where the weights and biases and continuously adjusted with each iteration in order to minimize the loss function [9]. With each iteration, the network gets closer to making more accurate predictions, which is the ultimate goal of its design. The dynamic range of functions and structures within neural networks is what makes them a versatile tool for tackling various problems in fields like anomaly detection, image recognition and beyond.

3.4 Varieties of Neural Networks

Neural networks come in various forms, each designed for specific applications. The Perceptron, conceptualized by Frank Rosenblatt in 1958, stands as the prototype, capable of handling linearly separable classes in a single-layer format [16]. Building on this are Feedforward Neural Networks, which introduce additional layers to accommodate the non-linear nature of real-world data, setting the stage for more complex architectures. Convolutional Neural Networks (CNNs) build on the principles of feedforward networks by incorporating specialized layers that can detect patterns within patterns, a technique essential for the analysis of visual data [17]. Finally, there are Recurrent Neural Networks (RNNs) that add loops in their architecture, allowing them to process sequential data effectively, making them suitable for tasks like time-series analysis or language processing [9]. The diversity of these networks demonstrates the adaptability and robustness of neural networks in addressing a wide range of computational problems.

4. Autoencoders

Within the domain of neural network architectures, autoencoders emerge as a specialised subset tailored for unsupervised learning tasks. It diverges from the conventional neural network design by aiming not to produce an external outcome, but rather to recreate its own inputs. Autoencoders were first conceptualised in the 1980s and historically, they marked a paradigm shift in pre-training deep neural networks, an approach famously articulated by Geoffrey Hinton and Ruslan Salakhutdinov in 2006 [18]. The core objective of an autoencoder is to compress the input data to a lower-dimensional representation and subsequently attempt to reconstruct the original dataset, leveraging its defining features.

4.1 Architecture and Functioning of Autoencoders

Autoencoders are typically identified by their two-part architecture, consisting of an encoder and decoder. The encoder serves to compress the input, funnelling the data into a compact "encoded" state within a latent space. This intermediate form encapsulates all essential features or patterns inherent in the data. The decoder segment aims to decompress this compacted code back into an approximation of the original input, although this reconstruction is intrinsically an approximation, focusing on the salient features of the dataset [3]. Figure 2 illustrates these essential components [19]. This self-supervised mechanism, capable of generating its labels from the input data, distinguishes autoencoders from other neural network models reliant on external labels.

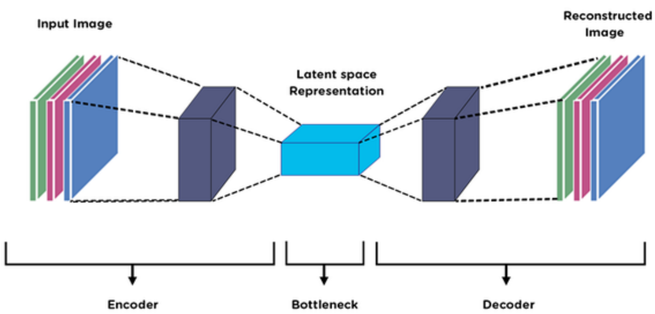

Figure 2. Schematic of Autoencoder. [19] D. Birla, "Basics of Autoencoders," 2019. [Online]. Available: https://medium.com/@birla.deepak26/autoencoders-76bb49ae6a8f. Accessed: Oct. 23, 2023.

4.2 Training and Loss Function in Autoencoders

During training, autoencoders undergo iterative refinement of weights and biases, guided by a loss function in order to minimise the reconstruction error. The quintessential loss function in this context is often the mean squared error, shown in Equation 1, when dealing with continuous input data [20], though other variations may be employed depending on the specificity of the input distribution. To ensure the model does not just simply copy the inputs directly to the output, regularisation techniques are employed in the loss function and by doing this the model learns to preserve the essential patterns of the input data while disregarding the noise.

4.3 Evolution and Varieties of Autoencoders

Since their creation, autoencoders have evolved into various types, each tailored for distinct applications and marked by unique characteristics. Sparse autoencoders, for instance, impose a sparsity constraint on hidden layers to distil the most prominent features of data, while denoising autoencoders are designed to filter out noise and recover a clean signal from corrupted inputs [21]. There are also Variational Autoencoders (VAEs), which push the boundaries further and are particularly useful in particle physics due to their ability to handle the probabilistic nature of data, exemplifying the model's versatility [22]. Finally, there are also Convolutional Autoencoders which take advantage of CNNs to excel in processing structured data like images by finding spatial patterns, a feature very useful in particle physics.

4.4 Applications and Challenges of Autoencoders

The practicality of autoencoders extends to a myriad of industries and research fields. They are instrumental in anomaly detection within cybersecurity, noise reduction in signal processing, and are very efficient at encoding for data compression [23]. However, autoencoders are not without their challenges; they sometimes struggle with local minima, require substantial data for training, and the interpretability of the features within the latent space can be elusive [24]. As such, they remain a popular subject of ongoing research, continually being refined and enhanced to overcome these limitations.

In summary, autoencoders serve as a powerful tool for capturing and encoding significant data patterns, facilitating applications such as anomaly detection, denoising, and dimensionality reduction. Their ability to adapt to various data types underscores their value across diverse applications, despite the challenges that persist in their development and utilization.

5. Anomaly Detection using Machine Learning

5.1 Introduction to Anomaly Detection

Anomaly detection is the process of identifying events that do not align with historical data and is a fundamental aspect of machine learning [25]. It is employed in a vast range of applications such as healthcare, where it is used to aid in the early detection of diseases by identifying abnormal patterns in patient data, and in cybersecurity, where it is crucial for identifying unorthodox attacks that deviate from known patterns [26].

This range of applications highlights the method's usefulness in a variety of domains. However, implementing effective anomaly detection methods proves to be a constant challenge. Extremely complex algorithms are required in order to be able to discern subtle patterns hidden in datasets, a task for which traditional statistical methods are often inadequate.

In scientific research, especially in particle physics, anomaly detection is not just a tool for data analysis; it is a gateway to discovery. Identifying anomalies in experimental data can lead to the uncovering of new physical phenomena, challenging existing theories, and providing empirical evidence for theoretical predictions. This chapter focuses on the use of autoencoders, a type of neural network, for anomaly detection in the context of discovering new physics.

5.2 Autoencoders in Anomaly Detection

5.2.1 Why are Autoencoders useful?

Autoencoders are uniquely adept for anomaly detection within particle physics for a range of factors. Particle physics is characterised by its production of complex and high-dimensional data, autoencoders can effectively condense this data into a lower-dimensional latent space, which not only simplifies the data but also preserves the essential features. This allows for a comprehensive analysis and the detection of anomalies [3]. This is particularly true for convolutional autoencoders, which are a powerful mechanism for the analysis of image data, like that obtained from the Large Hadron Collider.

Through their specialised architecture, convolutional autoencoders are extremely capable of parsing complex image data and highlighting subtle irregularities. Because of this, they can efficiently handle the spatial hierarchies in image data, making them ideal for tasks like anomalous jet tagging [27]. This plays a pivotal role in uncovering new insights in particle physics.

Crucially, autoencoders also have the capability for unsupervised learning, a trait that is extremely beneficial in particle physics where labelled data for rare anomalies is often not possible or very limited. This feature allows autoencoders to learn from the data without requiring explicit labelling, making them particularly suitable for exploring uncharted aspects of particle behaviour and interactions.

5.2.2 How to train Autoencoders for Anomaly Detection

In order to specialise an autoencoder for anomaly detection, there a few crucial training aspects that need to be considered. The first step to consider is the pre-processing of data. This process typically includes cleaning, normalising and sometimes transforming the raw data into a more suitable format for analysis [28].

In this specific context, the cleaning of data refers to the removal of any anomalous data points to ensure that the autoencoder is trained on non-anomalous or 'normal' data. This process allows the autoencoder to learn how to compress and reconstruct what is considered standard behaviour or patterns in the data. This means that when it does encounter an observation that deviates significantly from this norm, it will be poorly reconstructed and therefore highlighted as a potential anomaly.

An appropriate data transformation in particle physics would be the conversion of detected signals into structured formats like image grids, which convolutional autoencoders are well suited to handle due to their use of convolutional layers that are adept at capturing local spatial patterns within images [29].

Another crucial aspect of training is the optimisation of the loss function, which typically includes a reconstruction error term that quantifies the difference between the input and its reconstructed output. Therefore, when the reconstruction error is larger than some arbitrary threshold value, it is an indicator that the datapoint is a potential anomaly, making this metric a key aspect in model training.

Regularisation within the loss function is also vital to prevent overfitting and to ensure that the autoencoder captures key data features. Furthermore, the use of Receiver Operating Characteristic (ROC) curves and threshold setting is instrumental in evaluating the model's performance. ROC curves, plotting the true positive rate against the false positive rate, aid in identifying an optimal threshold for anomaly detection [30]. This threshold is determined based on the reconstruction error, representing the balance between detecting anomalies and maintaining a low rate of false positives. Thus, these components work symbiotically during training and validation to fine-tune the autoencoder for effective anomaly detection.

5.3 Applications in Physics

The use of autoencoders in anomaly detection opens a gateway to discovering new physics in particle experiments. By finely tuning these models to distinguish from expected particle interactions, autoencoders become extremely good at spotting data points that exhibit unusual behaviour, potentially indicative of phenomena not accounted for in existing theories. This ability to identify and flag such anomalies as outliers allows for further investigation, which could lead to groundbreaking discoveries in particle physics, offering insights into previously unexplained or unobserved aspects of the subatomic world.

An example case of autoencoders being used for anomaly detection in particle physics is the work done with the ATLAS experiment at CERN's Large Hadron Collider. Here, autoencoders were utilised to analyse jet images, helping to differentiate between those resulting from standard processes and potential new physics events. Specifically, they were used to distinguish between top and QCD jets, which are defined in the following chapter. This application demonstrates the effectiveness of autoencoders in identifying anomalies [31].

5.4 Challenges and Limitations

Autoencoders in anomaly detection face several challenges and limitations. Many of these key challenges are influenced by the nature and quality of the training data. A significant issue is the requirement for a broad and representative dataset of 'normal' instances. Particularly in complex fields like particle physics, obtaining such data is challenging and the idea of what constitutes 'normal' data can be fluid. Furthermore, autoencoders are highly sensitive to the assumption that training data is free of anomalies. The inclusion of anomalies in the training set can severely reduce the autoencoders' ability to identify new anomalies, as they become adept at reconstructing both normal and anomalous data [32]. A deficiency in the diversity and size of the input data is also critical to the autoencoders' performance because may lead to a poor representation of the data and miss out key features. This would lead to an increased number of false positives or negatives.

Furthermore, autoencoders assume that the reconstruction error incurred by anomalies is higher than that of normal samples. However, this is not always the case, especially when standard autoencoders generalize well even for even anomalies data points, an issue that becomes pronounced in data-rich environments like particle physics. Such overfitting can lead to ineffective anomaly detection, as the model may fail to distinguish between normal and anomalous instances [33].

In anomaly detection, autoencoders often struggle with model complexity and computational power challenges. Complex datasets require these models to have a high number of latent dimensions for accurate anomaly identification, this significantly increases the computational demands. Balancing the complexity of the autoencoder with available computational power therefore becomes a crucial and challenging aspect of their application in anomaly detection.

Another limiting factor is the interpretability of autoencoders in anomaly detection. Their complex and often unclear internal mechanisms make it hard to understand how they distinguish between normal and anomalous data. This lack of clarity can reduce trust and reliability in a practical setting, where understanding the basis for decisions is crucial [24].

In conclusion, there is a large range of challenges and limitations for autoencoders in anomaly detection, ranging from data quality and diversity to the intrinsic limitations in their design. These challenges are particularly important in specialized areas like particle physics, highlighting the need for advanced strategies to optimize the use of autoencoders in such nuanced applications.

5.5 Future Directions and Emerging Trends

The field of anomaly detection using autoencoders is constantly evolving due to new trends and advancements. One important area of progress focuses on the improvement of architectures. Advanced techniques like autoencoders (VAEs) and generative adversarial networks (GANs) are gaining prominence. These techniques are more adept for handling complex, high dimensional data and therefore lead to better efficiency in detecting anomalies [3].

Another significant aspect is the integration of Explainable AI (XAI) into autoencoders. This integration is particularly relevant in sectors such as healthcare and physics where understanding the decisions made by AI systems is critical. By incorporating XAI we can gain insights into why specific data points are flagged as anomalies thereby increasing trust and reliability in using autoencoders for anomaly detection [4].

Computational advancements also play a role in this field. The increasing availability of high-performance computing resources and cloud platforms is expected to alleviate the challenges associated with training autoencoder models. These developments will consequently contribute to expanding the application of autoencoders across many domains, ultimately enhancing their capabilities for detecting anomalies [31].

Lastly, it is worth noting the growing collaboration between academia and industry as a trend. This partnership is expected to foster the exchange of ideas and resources resulting in the development of efficient autoencoder models. Such collaborations are instrumental in pushing the boundaries of anomaly detection using autoencoders.

5.6 Conclusion

Autoencoders, a useful tool in handling complex, high-dimensional data, are pivotal in anomaly detection. Their utility expands across a large range of fields like healthcare and cybersecurity but they have also proven to be particularly useful in particle physics.

The ability of autoencoders to unearth subtle anomalies in data is unparalleled. However, challenges still remain. These include managing diverse data sets, grappling with the complexity of models, and enhancing the interpretability of the results. These challenges are constant reminders of the necessity for continued innovation and refinement in this field.

Looking ahead, we see the emerging trends that are shaping the future of anomaly detection in machine learning. These include advancements in autoencoder architectures, the creation of explainable AI concepts, computational leaps, and the beginning of cross-disciplinary collaborations. These developments are not just expanding the capabilities and applications of autoencoders in anomaly detection; they are also allowing for a deeper integration of these models across various scientific and practical areas.

In conclusion, the journey of anomaly detection, especially through the use of autoencoders in machine learning, is a path of constant discovery and innovation and will continue to prosper.

6. Practical Implementation of Code

In this following chapter we will break down the python code of a convolutional autoencoder and discuss the function of each section, as well as provide its output in order to show the models performance.

Within this demonstration we will use Quantum Chromodynamics (QCD) and top jet data, which are pivotal for exploring the fundamental forces that govern particle interaction. QCD is the theory that explains the strong force, a fundamental force which binds quarks and gluons into hadrons. QCD jets emerge from such hadronization and typically involves light quarks or gluons scattered at high energies. These are common in high-energy physics experiments and form simple patterns [34]. In contrast, top jets originate specifically from top quarks, the heaviest known quarks, which decay rapidly before hadronizing, often into a W boson and a bottom quark. The decay leads to complex, identifiable patterns, distinguishing top jets from the more frequently occurring QCD jets [35].

We will train the autoencoder on QCD jet data and then test it with a combination of both QCD and top jet data. This approach enables the autoencoder to learn and identify the 'normal' data patterns inherent in QCD jets as these events are particularly abundant within collider experiments and are thus well-understood. Consequently, when we then expose it to a mixed dataset containing both QCD and top jet data, the autoencoder is more adept at detecting anomalies. These anomalies are represented by the top jet data, which differ significantly from the QCD data the model was trained on as these events are much less common. This differential testing strategy effectively evaluates the model's ability to distinguish between 'normal' QCD jets and 'anomalous' top jets, thereby enhancing its utility in anomaly detection within particle physics data.

The QCD and top jet data was obtained through a series of complex simulation and reconstruction processes. Initially, the directional, timing, and energy deposition information of each particle collision event was condensed and the area with the most significant energy accumulation within the hadronic calorimeter was found. This energy cluster is named the 'leading fat jet' and is the focal point of the event’s energy distribution. The simulation of these events involved the use of the Pythia software for the detailed modelling of hadronization, the process by which quarks and gluons transform into hadrons. Subsequently, the Pythia 8 SlowJet program was employed for the clustering of particles, to accurately reconstruct the jet profiles. This methodology allowed for the differentiation of events into two classes: those characteristic of top quark jets and those typical of QCD jet background [36].

In Figure 3 we begin by importing the QCD and Top jet data from an external model and define the input layer of the autoencoder. Line 4 sets autoencoder up to handle images that are 25x25 pixels.

```
from DataLoader1 import load_data3
x_train, x_test, y_train, y_test = load_data3()

# Define the structure of the autoencoder
input_img = Input(shape=(25, 25, 1))
```

Figure 3. Importing Data and Initialising Autoencoder Structure.

Next, the encoder is created and consists of two convolutional layers (Conv2D) as well as a maximum pooling layer (MaxPooling2D). Each convolutional layer extracts features from the input dataset while the pooling layers reduce the spatial dimensions of the extracted features. The encoder is shown in Figure 4.

```
# Encoder
x = Conv2D( filters: 16,  kernel_size: (3, 3), activation='relu', padding='same')(input_img)
x = MaxPooling2D( pool_size: (2, 2), padding='same')(x)
x = Conv2D( filters: 8,  kernel_size: (3, 3), activation='relu', padding='same')(x)
encoded = MaxPooling2D( pool_size: (2, 2), padding='same')(x)
```

Figure 4. The Encoder.

In Figure 5, the decoder is built. This designed to reconstruct the input image from the encoded features. This is done using the same structure as the encoder but instead of max pooling layers it uses upsampling layers (UpSampling2D) to increase the spatial dimensions of the encoded features.

```
# Decoder
x = Conv2D( filters: 8,  kernel_size: (3, 3), activation='relu', padding='same')(encoded)
x = UpSampling2D((2, 2))(x)
x = Conv2D( filters: 16,  kernel_size: (3, 3), activation='relu', padding='same')(x)
x = UpSampling2D((2, 2))(x)
```

Figure 5. The Decoder.

In Figure 6 we add a cropping layer (Cropping2D) in order to adjust the output size so that it matches the input image size. This is necessary because the combination of convolutional, pooling, and upsampling layers can lead to output dimensions that are slightly different from the input dimensions. The model is then compiled using the Adam optimizer. This stands for Adaptive Moment Estimation and is an iterative optimisation algorithm that is used to minimise the loss function. We then use the mean squared error (MSE) as the loss function, which is typical for reconstruction tasks like the one performed by an autoencoder.

```
# Add Cropping2D layer to adjust the output size to match the input size
cropped_decoded = Cropping2D(cropping=((1, 2), (1, 2)))(x)

autoencoder = Model( input_img, cropped_decoded)
autoencoder.summary()

# Compile the autoencoder using Adam optimizer and MSE loss
autoencoder.compile(optimizer=Adam(learning_rate=0.001), loss='mse')
```

Figure 6. Finalising the Autoencoder Model and Compiling.

As shown in Figure 7, we then train the model using the 'fit' method and set the number of epochs to 50 for more accurate results. After training, the model is used to predict the test set, which contains both "normal" and "anomalous" data, and the reconstruction error is calculated.

```
# Assuming you've split your data and x_train contains only "normal" data
# Train the autoencoder
autoencoder.fit(x_train, x_train, epochs=50, batch_size=128, shuffle=True, validation_split=0.1)

reconstructed = autoencoder.predict(x_test)
reconstruction_error = np.mean(np.power(x_test - reconstructed, 2), axis=(1, 2, 3))
```

Figure 7. Training the Autoencoder and Making Predictions.

Finally, in Figure 8, the reconstruction error is used to determine a threshold for anomaly detection, the model's performance in detecting anomalies is evaluated using the Receiver Operating Characteristic (ROC) curve and the area under the ROC curve (AUC), which are common metrics for evaluating classification models.

```
# Determine the threshold from the normal training data
# For example, use the 95th percentile of the normal data's reconstruction error
threshold = np.percentile(reconstruction_error, 95)

# Evaluate anomaly detection performance
y_test_binary = np.argmax(y_test, axis=1)

# Compute ROC curve and ROC area
fpr, tpr, thresholds = roc_curve(y_test_binary, reconstruction_error)
roc_auc = auc(fpr, tpr)
```

Figure 8. Anomaly Detection and Evaluation.

Figures 9 and 10 show the reconstruction error and ROC curve respectively which are the final outputs of the code.

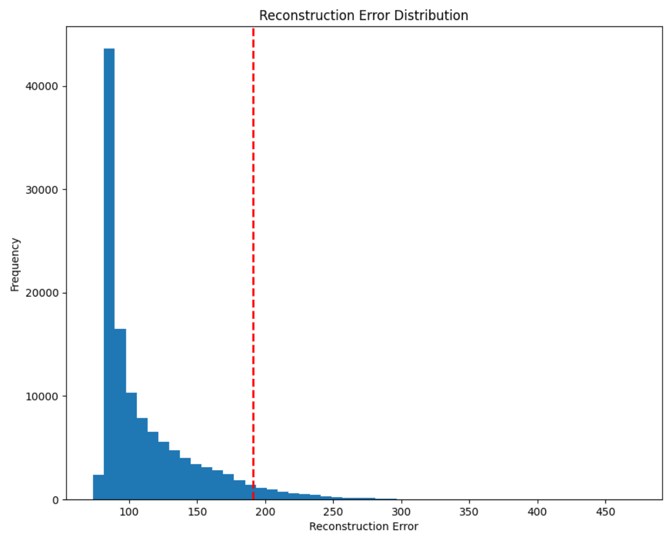

Figure 9. Frequency against Reconstruction Error.

The majority of data points have a reconstruction error between 0 and 100, this suggests that in most instances the model can accurately reconstruct the input data with relatively low error. We also see some data points with a larger reconstruction error that is pass our threshold value, indicating where the model has identified top jet data.

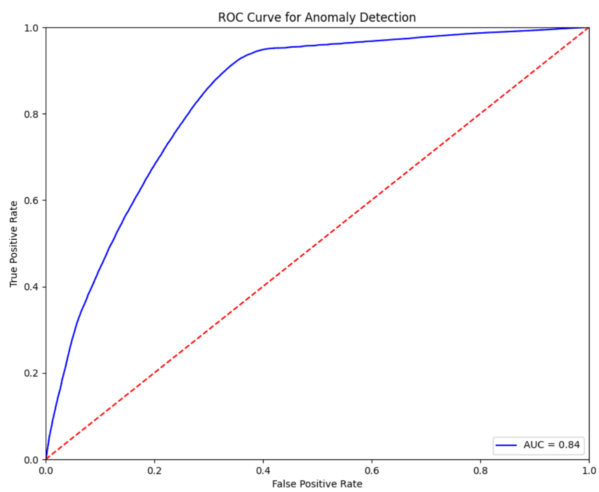

Figure 10. ROC curve.

From the ROC curve we can see that the model has an area under the curve (AUC) of 0.84, this suggests the model is effective at distinguishing between the two classes (QCD and top jets).

An AUC of 1.0 represents a perfect model whereas an AUC of 0.5 would suggest that the model has no discriminative ability, equivalent to randomly guessing. However, the model is not perfect, if it was then we would expect the blue curve to peak in the top left-hand corner, where the true positive rate is high and the false positive rate is low. This could be due to a range of reasons but the most probable is that the model is underfitting the data and therefore unable to accurately capture the complex features of the dataset.

7. Conclusion

This dissertation has systematically explored the role of autoencoders in anomaly detection, particularly within the field of particle physics. This study began with a foundational understanding of machine learning principles, leading to an in-depth analysis of anomaly detection techniques with a specific focus on autoencoders. The research effectively bridged the gap between machine learning potential and the complex challenges inherent in particle physics.

Key findings include the capability of autoencoders to effectively condense high-dimensional particle physics data into a more manageable form while preserving essential patterns, a feature that is crucial for comprehensive analysis and anomaly detection. This study also highlighted the adaptability of autoencoders for various data types, despite challenges in model complexity, data diversity, and interpretability. Notably, the application of autoencoders in the ATLAS experiment at CERN's Large Hadron Collider underscores their effectiveness in identifying anomalies and contributing to significant discoveries in the field.

This study discussed the limitations and challenges of autoencoders, such as their sensitivity to training data anomalies and the balance between model complexity and computational power. Looking forward, the evolving trends in autoencoder architecture, computational advancements and cross-disciplinary collaborations offer promising avenues for enhancing the capabilities of autoencoders in anomaly detection.

Overall, this dissertation has provided a comprehensive overview of the use of autoencoders in anomaly detection within particle physics, establishing their potential for groundbreaking discoveries and underlining the necessity for ongoing research and refinement in this exciting field of study.

Word count: 5996

References

[1] M. Farina, Y. Nakai, and D. Shih, "Searching for New Physics with Deep Autoencoders, 2018, pp. 16-17. [Online]. Available: https://arxiv.org/pdf/1808.08992.pdf. Accessed: Nov. 24, 2023.

[2] "Machine Learning," IBM. [Online]. Available: https://www.ibm.com/topics/machine-learning. Accessed: Oct. 13, 2023.

[3] G. Karagiorgi, G. Kasieczka, S. Kravit, B. Nachman, "Machine Learning in the Search for New Fundamental Physics", D. Shih, 2021, pp. 2-6. [Online]. Available: https://arxiv.org/pdf/2112.03769.pdf. Accessed: Oct. 13 2023.

[4] S. Raschka, "Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning," 2018, p. 4. [Online]. Available: https://ar5iv.labs.arxiv.org/html/1811.12808. Accessed: Oct. 13 2023.

[5] P. Mehta, C.H. Wang, A. G. R. Day, C. Richardson, "A high-bias, low-variance introduction to Machine Learning for physicists," 2019, pp. 21-49. [Online]. Available: https://arxiv.org/pdf/1803.08823.pdf. Accessed: Oct. 13 2023.

[6] "Supervised Learning," IBM. [Online]. Available: https://www.ibm.com/topics/supervised-learning. Accessed: Oct. 13, 2023.

[7] "Choosing the Right Classification Model," MathWorks. [Online]. Available: https://www.mathworks.com/campaigns/offers/next/choosing-the-best-machine-learning-classification-model-and-avoiding-overfitting.html. Accessed: Oct. 13, 2023.

[8] D. Mladenić, J. Brank, M. Grobelnik, N. Milic-Frayling, "Feature selection using linear classifier weights: interaction with classification models," 2004, pp. 234-236. [Online]. Available: https://dl.acm.org/doi/pdf/10.1145/1008992.1009034. Accessed: Oct. 14, 2023.

[9] G. Carleo, I. Cirac, K. Cranmer, L. Daudet, M. Schuld, N. Tishby, L. Vogt-Maranto, L. Zdeborova, "Machine learning and the physical sciences," 2019, pp. 4-5. [Online]. Available: https://arxiv.org/pdf/1903.10563.pdf. Accessed: Oct. 14, 2023.

[10] J. Brownlee, "Linear Regression for Machine Learning," 2020. [Online]. Available: https://machinelearningmastery.com/linear-regression-for-machine-learning/. Accessed: Oct. 14, 2023.

[11] P. Mehta, C.H. Wang, A. G. R. Day, C. Richardson, "A high-bias, low-variance introduction to Machine Learning for physicists," 2019, p. 4. [Online]. Available: https://arxiv.org/pdf/1803.08823.pdf. Accessed: Oct. 14, 2023.

[12] P. Mehta, C.H. Wang, A. G. R. Day, C. Richardson, "A high-bias, low-variance introduction to Machine Learning for physicists," 2019, pp. 63-70. [Online]. Available: https://arxiv.org/pdf/1803.08823.pdf. Accessed: Nov. 8 2023.

[13] B. Lutkevich, "Association rules," TechTarget, June 2023. [Online]. Available: https://www.techtarget.com/searchbusinessanalytics/definition/association-rules-in-data-mining. Accessed: Nov. 8 2023.

[14] "Neural Networks," IBM. [Online]. Available: https://www.ibm.com/topics/neural-networks. Accessed: Oct. 11, 2023.

[15] H. T. Ünal and F. Başçiftçi, "Evolutionary design of neural network architectures: a review of three decades of research," Artif Intell Rev 55, 2022, pp. 1727. [Online]. Available: https://doi.org/10.1007/s10462-021-10049-5. Accessed: Oct. 23, 2023.

[16] M. Banoula, "What is Perceptron: A Beginners Guide for Perceptron," Simplilearn, May 10, 2023. [Online]. Available: https://www.simplilearn.com/tutorials/deep-learning-tutorial/perceptron. Accessed: Nov. 9 2023.

[17] P. Mehta, C.H. Wang, A. G. R. Day, C. Richardson, "A high-bias, low-variance introduction to Machine Learning for physicists," 2019, pp. 56-58. [Online]. Available: https://arxiv.org/pdf/1803.08823.pdf. Accessed: Nov. 9 2023.

[18] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," Science, vol. 313, no. 5786, pp. 504-507, 2006.

[19] D. Birla, "Basics of Autoencoders," 2019. [Online]. Available: https://medium.com/@birla.deepak26/autoencoders-76bb49ae6a8f. Accessed: Oct. 23, 2023.

[20] I. Goodfellow, Y. Bengio, A. Courville, “Deep Learning”, MIT Press, 2016, pp. 502-525.
[21] D. Bank, N. Koenigstein, R. Giryes, "Autoencoders," in Machine Learning for Data Science Handbook, 2023, pp. 353-374. [Online]. Available: https://doi.org/10.1007/978-3-031-24628-9_16. Accessed: Oct. 19, 2023.
[22] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," arXiv preprint arXiv:1312.6114, 2013, pp. 1-14. Accessed: Nov. 12 2023.

[23] C. Zhou and R. C. Paffenroth, "Anomaly detection with robust deep autoencoders," in Proc. 23rd ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, 2017, pp. 665-674.

[24] D. F. N. Oliveiraa et al., "A new interpretable unsupervised anomaly detection method based on residual explanation," 2021, pp. 1-3. [Online]. Available: https://arxiv.org/pdf/2103.07953.pdf. Accessed: Oct. 23, 2023.

[25] V. Chandola, A. Banerjee, V. Kumar, "Anomaly detection: A survey," ACM Computing Surveys (CSUR), vol. 41, no. 3, 2009, pp. 1-58.
[26] E. Šabić, D. Keeley, B. Henderson, S. Nannemann, "Healthcare and anomaly detection: using machine learning to predict anomalies in heart rate data," AI & Soc 36, 2021, pp. 149–158. [Online]. Available: https://doi.org/10.1007/s00146-020-00985-1. Accessed: Nov. 8 2023.

[27] T. Finke, M. Krämer, A. Morandini, A. Muck, I. Oleksiyuk, "Autoencoders for unsupervised anomaly detection in high energy physics," 2021. [Online]. Available: https://ar5iv.labs.arxiv.org/html/2104.09051. Accessed: Nov. 8 2023.

[28] A. Singh, "Transforming Data for Machine Learning: A Guide to Cleaning, Normalizing, and Aggregating Your Data," Medium, January 15, 2023. [Online]. Available: https://medium.com/@Ambarish_224/transforming-data-for-machine-learning-a-guide-to-cleaning-normalizing-and-aggregating-your-data-10ca04493905#:~:text=The%20goal%20of%20data%20transformation,machine%20learning%2C%20data%20transformation. Accessed: Nov. 12 2023.

[29] G. E. Hinton and R. R. Salakhutdinov, "Reducing the Dimensionality of Data with Neural Networks," 2006.

[30] "ROC Analysis," IBM. [Online]. Available: https://www.ibm.com/docs/en/spss-statistics/beta?topic=features-roc-analysis. Accessed: Nov. 12 2023.

[31] B. Nachmanm, "Anomaly Detection for Physics Analysis and Less than Supervised Learning," 2020, pp. 7-16. [Online]. Available: https://arxiv.org/pdf/2010.14554.pdf. Accessed: Nov. 12 2023.

[32] L. Beggel, M. Pfeiffer, B. Bischl, "Robust Anomaly Detection in Images using Adversarial Autoencoders," 2020, p.9.

[33] M. S. Battikh, "Latent-Insensitive Autoencoders for Anomaly Detection," 2021, pp. 1-2. [Online]. Available: https://arxiv.org/pdf/2110.13101.pdf. Accessed: Nov. 8 2023.

[34] CMS Collaboration, "Study of quark and gluon jet substructure in Z+'jet and dijet events from pp collisions", 2021, pp. 1-5. [Online]. Available: https://arxiv.org/pdf/2109.03340.pdf. Accessed: Nov. 25 2023.

[35] D. Denisov and C. Vellidis, "The top quark, 20 years after its discovery", Physics Today, vol. 68, no. 4, 2015, pp. 46-52. [Online]. Available: https://pubs.aip.org/physicstoday/article/68/4/46/415101/The-top-quark-20-years-after-its-discovery. Accessed: Nov. 25 2023.

[36] Charanjit K. Khosa, Veronica Sanz, "Anomaly Awareness", 2023. [Online]. Available: https://arxiv.org/pdf/2007.14462.pdf. Accessed: Nov. 24, 2023.

Generative AI Disclosure: I used ChatGPT 3.5 to assist in idea generation, and for feedback on grammar and content. I implemented some of its recommendations.