Department of Physics and Astronomy
Heidelberg University

Bachelor Thesis in Physics
submitted by

Jan Hendrik Rüschkamp

born in Münster (Germany)

2023

DarkCLR

Self-Supervised Anomaly Search for Semivisible Jets

This Bachelor Thesis has been carried out by Jan Hendrik Rüschkamp
at the
Institute for Theoretical Physics in Heidelberg
under the supervision of
Prof. Dr. Tilman Plehn

Abstract

Experiments such as Large Hadron Collider (LHC) provide us unique avenues to understand nature at its fundamental level. For example, data produced by the LHC could potentially hold intriguing signatures for some of the open mysteries such as dark matter. To help uncover the underlying structure of such large amounts of data, machine learning proved to be fruitful. This thesis studies the possibility to detect elusive dark-jets, simulated through a model that could explain the observed dark matter within the universe, by self-supervised learning methods. Doing so, DarkCLR is introduced, a method which is data-driven and uses augmentations of the background data to build a representation space that is invariant under symmetry transformations and at the same time discriminative to certain, beyond the standard model inspired, augmentations. A normalised autoencoder trained on background representations then computes anomaly scores in the representation space.

Zusammenfassung

Experimente wie der Large Hadron Collider (LHC) bieten uns einzigartige Möglichkeiten, die Natur auf ihrer fundamentalen Ebene zu verstehen. So könnten die vom LHC erzeugten Daten möglicherweise verblüffende Hinweise auf einige der offenen Rätsel wie die dunkle Materie liefern. Um die zugrunde liegende Struktur solch großer Datenmengen aufzudecken, hat sich maschinelles Lernen als wertvoll erwiesen. In dieser Arbeit wird die Möglichkeit untersucht, illusorische dunkle Strahlen, die durch ein Modell simuliert werden, das die beobachtete dunkle Materie im Universum erklären könnte, durch selbstüberwachte Lernmethoden zu entdecken. Zu diesem Zweck wird DarkCLR eingeführt, eine Methode, die datengesteuert ist und Erweiterungen der Hintergrunddaten verwendet, um einen Repräsentationsraum aufzubauen, der unter Symmetrietransformationen invariant ist und gleichzeitig bestimmte, über das Standardmodell hinausgehende Erweiterungen diskriminieren kann. Ein normalisierter AutoEncoder, der auf Hintergrundrepräsentationen trainiert wurde, berechnet dann Anomalie-Scores in dem Repräsentationsraum.

Contents

1
Introduction

The Standard Model of particle physics has proven to be a highly successful theory, providing a comprehensive and internally consistent explanation for the elementary particles observed to date, as well as three of the four fundamental forces. Many of the predictions made by the Standard Model have been confirmed experimentally, further validating its accuracy. After the experimental discovery of Higgs boson in 2012 the Standard Model became complete in terms of particle content [1, 2].

The Standard Model precisely describes most of the observed phenomena at the fundamental level. However it cannot account for the existence of dark matter. Initially postulated by Zwicky [3] in 1930 and subsequently explored by Rubin and Ford [4] in the 1970s, the first evidence of dark matter emerged from the study rotational curves of galaxies. Observations showed that stars in spiral galaxies maintain a near-constant speed, regardless of their radial distance from the galaxy's center – a finding at odds with the predictions based on visible matter alone. The latter would propose a decrease in speed with increasing distance from the center. This inconsistency suggests the presence of an unobserved mass – now referred to as dark matter – which provides the additional gravitational force necessary to maintain the observed stellar velocities.

Apart from rotational curves, phenomena such as gravitational lensing [5] and the patterns discerned in the cosmic microwave background radiation [6] strongly endorse the existence of dark matter. Subsequent to these findings, a multitude of dark matter candidates have been proposed within the scientific community [7–10].
This thesis narrows its focus to one such proposal, an augmentation of the Standard Model that postulates a "dark sector" with only faint interaction with Standard Model particles, thereby preserving the observed constant dark matter density.

Despite several efforts focused on identifying the source of dark matter it remains undiscovered. It is plausible that this lack of success should not be attributed to the limitations of current particle accelerator technology but because the signs of new physics are

concealed within the copious data collected by colliders such as the Large Hadron Collider. Machine learning has proven to be an efficient tool in decoding these large datasets.

This thesis presents a mapping that learns to construct a latent space from jet constituents. Employing a self-supervised contrastive methodology, the mapping is trained to create a representation space that encapsulates the differences between semi-visible jets and background Quantum chromodynamic jets. The demarcation task between these categories is undertaken through an unsupervised approach, without the reliance on specific instruction or labeled data.

The training process aims to maximize the distinction between semi-visible jets and background Quantum chromodynamic jets, exploiting the specific structure and patterns within the jet constituents. In doing so, the mapping seeks to reveal and employ the inherent features that distinguish these jet types, without explicit knowledge of the signal dataset.

The structure of this thesis is as follows: The fundamentals of semivisible jet physics are recapitulated in chapter 2, and a groundwork for machine learning methods is established in chapter 3. The concept behind Dark Contrastive Learning of Representations (DarkCLR) is then introduced in chapter 4, where the specific methodology used is described. Subsequently, the results that have been achieved are presented and discussed in chapters 6 and 7.

2

Physics background

This chapter is a basic introduction to the scientific ideas and the variables we use in this research. It starts with a short overview of the Standard Model (SM) of particle physics, focusing on quantum chromodynamics (QCD) - the theory that explains the strong force. This is succeeded by a brief introduction to the concept of semi-visible jets. The chapter concludes with an examination of the kinematic variables utilized throughout this thesis.[1]

2.1 Standard Model of particle physics

Within the paradigm of the SM of particle physics, all recognized elementary particles and three of the fundamental forces are depicted within the context of quantum field theory (QFT). These forces encompass electromagnetism, weak interaction, and the strong force, with gravity being the notable exception.

Elementary particles are primarily classified into two groups. The first group consists of fermions, spin-$\frac{1}{2}$ particles which comprise the observable matter in the universe. These fermions can be further subdivided into quarks and leptons, as illustrated in Fig. 2.1 .The second group embodies the bosons, which are the carriers of the fundamental forces. As spin-1 particles, they mediate these forces, earning them the moniker of gauge bosons within QFT. Each fundamental force is associated with a specific boson: the photon mediates electromagnetism, the $W^{\pm}$ and Z bosons mediate the weak interaction, and gluons mediate the strong interaction. To complete the SM the Higgs boson, an unique spin-0 particle, is responsible for giving mass to the other elementary particles.

[1] In this thesis natural units will be used with $c = \hbar = 1$

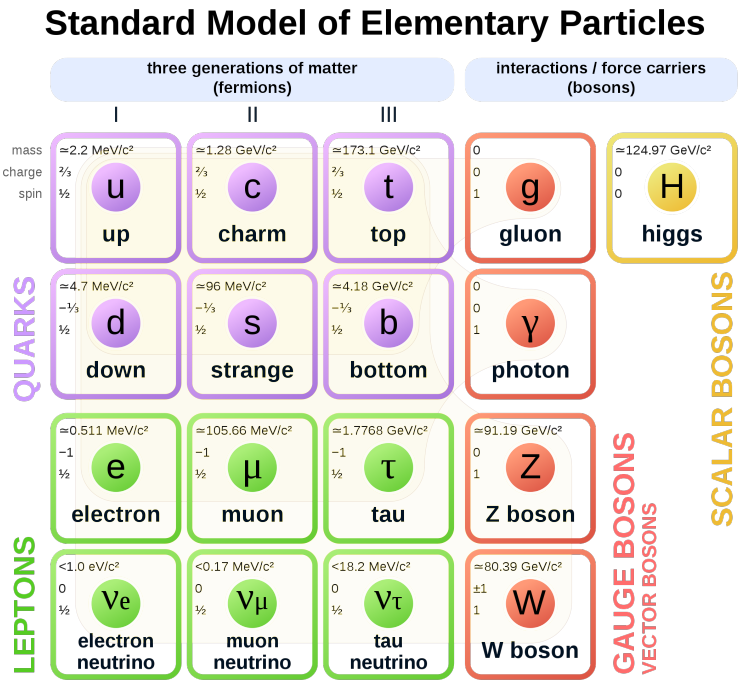


Figure 2.1: The Standard Model of particle physics, which describes all known elementary particles and three of the four known fundamental forces. Taken from [11].

2.2 QCD and jets

Quantum chromodynamics (QCD) is a key component of the SM which provides a theoretical framework to understand the behaviors and strong interactions of quarks and gluons. At high energy scales, quarks exhibit weak interactions that can be effectively analyzed using perturbative methods. However, as the energy decreases and approaches a low energy scale, the interaction strength between quarks increases. Consequently, the traditional perturbative approach breaks down and confinement emerges. The phenomenon of color confinement, a distinctive property of QCD, mandates that quarks and antiquarks coexist in color-neutral states, collectively known as hadrons.

Hadrons can be further classified based on their spin. Those with half-integer spin are identified as baryons, consisting of three quarks each bearing an unique color charge. In contrast, hadrons with integer spin, known as mesons, comprise a pair consisting of a quark and an antiquark.

During high-energy particle collisions, generated quarks or gluons undergo a process referred to as hadronization. Given that quarks and antiquarks can only maintain stability as hadrons, the potential energy between two quarks intensifies as their separation

increases. This results in the formation of hadrons along with quark-antiquark pairs which materialize from the vacuum. The generation of these quark-antiquark pairs may recur several times before the quarks eventually form a stable bound state.
For subsequent analysis, the resulting hadrons must be aggregated by jet clustering algorithms such as anti-k_T [12], producing so-called jets. Jets resulting from QCD processes that do not yield particles categorized as "signals" of interest are collectively referred to as QCD background.

2.3 Semivisible jets

Dark showers, also referred to as semi-visible jets, are postulated by theories that propose extensions to the SM via the addition of a new sector, termed the dark sector. An example of such a theory is the Hidden Valley Model. Given the vast array of Hidden Valley models, this chapter will primarily concentrate on the specific model that governs the creation of the semi-visible jets used as signals within this thesis.
The model of interest here is characterized by a strongly coupled $SU(3)_d$ dark sector comprised of fermions, linked to the SM via a portal. For the purposes of this thesis, the conduit between the SM and the dark sector is assumed to be a Z' mediator. This mediator has the capacity to couple to both, SM quarks $(q, \bar{q})$ and dark sector quarks $(q_d, \bar{q_d})$ [13]. The concept is visualised in Fig. 2.2. This is facilitated by the introduction of a weakly interacting $U(1)'$ Gauge group. The weak nature of this interaction is inferred from the absence of any detectable signatures attributable to the vanishing quarks at the LHC.

Once a $q_d, \bar{q_d}$ pair is produced via a Z', the model permits the initiation of a shower in the dark sector. At this juncture, a critical variable r_{inv} is introduced. r_{inv} quantifies the fraction of particles within a dark shower that does not decay back into SM particles relative to the total number of particles within the dark shower. The variable r_{inv} cannot be considered an independent variable due to its correlation to the physics within the dark sector. It is influenced by various other variables such as the mass of dark mesons or the coupling strength, which will be described in more detail in chapter 5. Nevertheless, the characteristic of r_{inv} is crucial to determine the visibility or invisibility of the dark shower constituents.

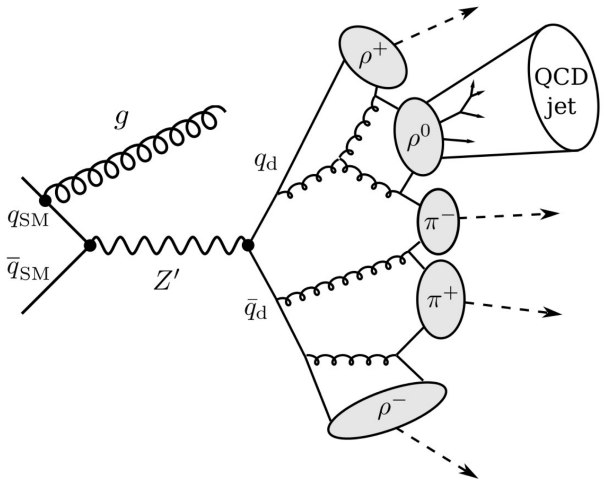


Figure 2.2: Schematic illustration of a shower within the dark sector from a decay of Z' produced in association with a gluon. In this configuration a $Z' - \rho_d^0$ mixing exist that enables the ρ_d^0 to decay into SM-quarks. Taken from [13]

2.4 Kinematic variables for jets

To fully understand the dataset described in the ensuing sections, it is important to first understand some basic physics concepts and notations that we use to measure the discussed phenomena.

We commence by defining the Lorentz factor $\gamma = (1-\beta^2)^{-0.5}$, where β signifies the particle's velocity in units of the speed of light, c. Leveraging the Lorentz factor along with the rest mass of a massive particle, the relativistic energy $E = \gamma m$ and three-momentum $\vec{p} = \gamma\beta m$ can be effectively described. The combination of these parameters gives us the four-momentum (also known as energy-momentum four-vector) of the particle:

$$p^\mu := (E, \vec{p}) = (E, p_x, p_y, p_z) \quad \text{with} \quad E = \sqrt{|\vec{p}|^2 + m^2}. \tag{2.1}$$

By adopting spherical coordinates with the polar angle $\theta \in [0, \pi]$ and the azimuthal angle $\phi \in (-\pi, \pi]$, the three-momentum $\vec{p} = (p_x, p_y, p_z)$ can be parameterized as

$$\vec{p} = |\vec{p}| \begin{pmatrix} \sin\theta\cos\phi \\ \sin\theta\sin\phi \\ \cos\theta \end{pmatrix}. \tag{2.2}$$

In this context, the z-axis corresponds to the direction of the beam, and the angle θ embodies the deviation between the particle's trajectory, delineated by its three-momentum $\vec{p}$, and the beam direction. In high-energy collision experiments, the laboratory frame and the center-of-mass system (CMS) are intertwined through a Lorentz boost along the beam axis. In order to streamline the analysis, we introduce variables that exhibit simple transformation properties under a Lorentz boost along the designated direction.

$$\text{transverse momentum:} \qquad p_\mathrm{T} = \sqrt{p_x^2 + p_y^2} \tag{2.3a}$$

$$\text{transverse mass:} \qquad m_\mathrm{T} = \sqrt{p_\mathrm{T}^2 + m^2} \xrightarrow{m \to 0} p_\mathrm{T} \tag{2.3b}$$

$$\text{rapidity:} \qquad y = \frac{1}{2} \ln\left(\frac{E + p_z}{E - p_z}\right) \tag{2.3c}$$

$$\text{pseudo-rapidity:} \qquad \eta = \frac{1}{2} \ln\left(\frac{|\boldsymbol{p}| + p_z}{|\boldsymbol{p}| - p_z}\right) = -\ln\left(\tan\frac{\theta}{2}\right) \xrightarrow{m \to 0} y \tag{2.3d}$$

While the transverse momentum p_T and the transverse mass m_T retain their invariance under Lorentz boosts along the beam axis, the rapidity y adds up under such a transformation. For ultra-relativistic particles, which travel at speeds approximating the speed of light (i.e. $|\boldsymbol{p}| \gg m$ and thus $E \approx |\boldsymbol{p}|$), the particle masses can be considered negligible. Hence, the pseudo-rapidity η converges with the rapidity y. Under these conditions, the transverse momentum p_T, the pseudo-rapidity η, and the azimuthal angle ϕ constitute effective measures for the four-momentum of a particle:

$$p^\mu = \begin{pmatrix} m_\mathrm{T} \cosh y \\ p_\mathrm{T} \cos\phi \\ p_\mathrm{T} \sin\phi \\ m_\mathrm{T} \sinh y \end{pmatrix} \xrightarrow{m \to 0} p_\mathrm{T} \begin{pmatrix} \cosh\eta \\ \cos\phi \\ \sin\phi \\ \sinh\eta \end{pmatrix} \tag{2.4}$$

In a symmetric high-energy collision involving two protons A and B (where the proton masses can be overlooked due to the high energies), the four-momenta in the laboratory system are given by:

$$p_A^\mu = (E_\mathrm{B}, 0, 0, E_\mathrm{B}) \qquad \text{and} \qquad p_B^\mu = (E_\mathrm{B}, 0, 0, -E_\mathrm{B}) \quad , \tag{2.5}$$

where E_B represents the beam energy. The square root of the Mandelstam variable s describes the center-of-mass energy E_CM, which denotes the total energy available in the CMS:

$$E_{\mathrm{CM}} = \sqrt{s} = \sqrt{(p_A^\mu + p_B^\mu)^2} = 2E_{\mathrm{B}} \tag{2.6}$$

3

Machine learning overview

In recent years, machine learning has emerged as a powerful tool in various fields, revolutionizing the way we process and analyze data. Its ability to automatically learn patterns and make predictions from vast amounts of information has led to remarkable advancements in diverse domains, including image recognition, natural language processing, and even anomaly detection. Through this chapter the basic tools, needed to understand the idea behind Dark Contrastive Learning of Representations (DarkCLR), are introduced.

3.1 Prelude

The general idea of machine learning is to create algorithms and models that enable computers to learn from data and make predictions or decisions without being explicitly programmed. The process of learning involves iteratively refining the model based on feedback from the data, allowing the system to adapt and improve over time. For a more comprehensive discussion of the described topics see [14].

3.1.1 Neural networks

There are many ways to introduce neural networks, the most common would be comparing them to the human brain [15, 16]. This section follows a slightly more mathematical approach, similar to [14]. Thus we start by considering the neural network as a numerically defined function :

$$f_\theta(x) \approx f(x) \tag{3.1}$$

Here, $f_\theta(x)$ symbolizes the neural network (NN), a composite structure of smaller functional units known as layers. Each layer, denoted by x^n, to establish distinction, can contain numerous nodes or neurons. Nodes, the most elemental components of a NN, receive an input $\boldsymbol{X}$ and produces a scalar output $\tilde{z}$.

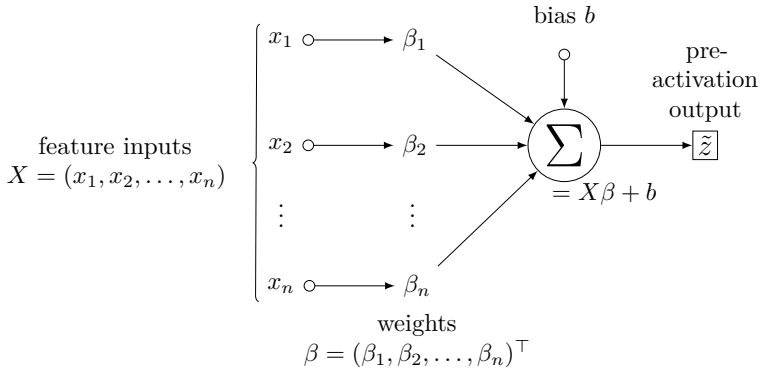


Figure 3.1: Schematic visualisation of a single node.

In a fully connected network, this value is computed by multiplying the node values from the preceding layer, x_i^{n-1}, with a vector, $\boldsymbol{\beta}$, known as the weights, and adding a bias, b, to the result, as illustrated in Figure 3.1. Mathematicly this can be expressed as 3.2. Staying consistent in the notation we realise that $\tilde{z}$ equals the input x_i^n for the next, $n+1$th layer. To better focus on the idea of calculating the output value of a node we will refer to it as $\tilde{z}$.

$$\tilde{z} = \boldsymbol{x}\boldsymbol{\beta} + b = \sum_{i=1}^{n} x_i \beta_i + b \tag{3.2}$$

In eq. (3.1), θ constitutes the sum of weights β and biases b from the nodes, defining the parameters of the function.
To generalize this input-output relation across a layer, we introduce a weight matrix W that encompasses the weight vectors $\boldsymbol{\beta}$ of all nodes within a layer:

$$W = \begin{bmatrix} \beta_{1,1} & \beta_{1,2} & \cdots & \beta_{1,E} \\ \beta_{2,1} & \beta_{2,2} & \cdots & \beta_{2,E} \\ \vdots & \vdots & \ddots & \vdots \\ \beta_{D,1} & \beta_{D,2} & \cdots & \beta_{D,E} \end{bmatrix} \tag{3.3}$$

$$\boldsymbol{B} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_E \end{bmatrix} \tag{3.4}$$

The matrix W carries dimensions $D \times E$, with D and E defining the input and output dimensions of the layer respectively. Similarly, a vector $\boldsymbol{B}$ incorporates all the biases of the layer. Consequently, the output of a whole layer can be formulated as

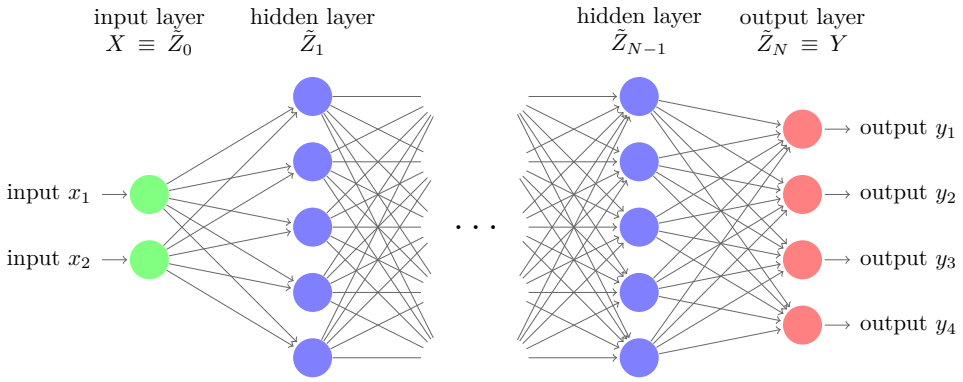


Figure 3.2: Schematic visualisation of a fully connected neural network.

$$\tilde{\boldsymbol{Z}} = W\boldsymbol{X} + \boldsymbol{B}. \tag{3.5}$$

The concept of nodes and layers merge to form a NN by linking successive layers, such that the output of the $(n-1)$-th layer serves as the input to the n-th layer:

$$x \to x^1 \to x^2 \cdots \to x^N \equiv f_\theta(x). \tag{3.6}$$

Within this layered architecture, the first and last layers are identified as the input and output layers, respectively, while the intervening layers are designated as hidden layers. Such a recursively defined network is referred to as feed-forward NN. The example configuration of a fully connected network is visualized in Fig. 3.2. The term "fully connected" characterizes the complete interlinking between nodes of neighbouring layers.

3.1.2 Activation functions

To this point the node (and thus the whole network) describes a linear function. When using the nodes in the described manner there is no point in adding more than one layer due to the fact that it would only be able to describe a linear relation between the input and the output. To make use of more than one layer one need to introduce non linearity to the outputs of the nodes. This is performed by using activation functions ϕ to the output of each node

$$z = \phi(\tilde{z}). \tag{3.7}$$

By introducing non linearity, the network is able to capture more complex relationships. The most prominent example of an activation function is the ReLU (Rectified Linear Unit) [17]. The ReLU function is defined as

$$\phi(\tilde{z}) = \max(0, \tilde{z}). \tag{3.8}$$

Fig. 3.3 shows a graphical representation of the ReLU function.

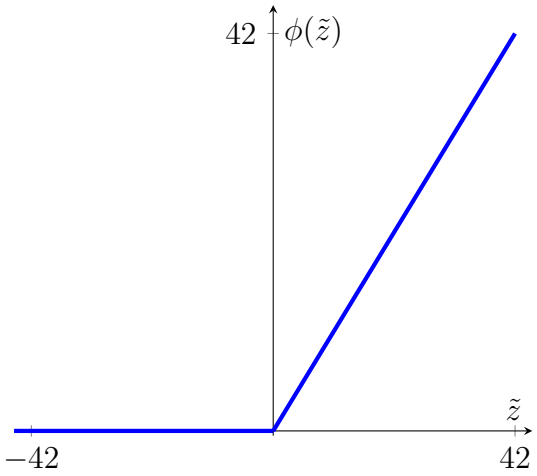


Figure 3.3: The ReLU Activation Function

While the ReLU is a popular choice, it is not the only activation function used in NNs. Other common activation functions include the sigmoid function, hyperbolic tangent (tanh), and the softmax function [17]. The choice of activation function can depend on several factors, such as the specific application, the nature of the data, and the architecture of the network.

3.1.3 Backpropagation and stochastic gradient descent

A NN can be seen as a complex function, which takes inputs, processes them through layers of nodes, and generates outputs. To be useful, the network must be able to adjust its behavior to approximate a target function, typically by learning from data. This learning process involves tuning the network's parameters weights and biases, collectively known as parameters θ, to minimize the difference between the network's predictions and the actual target values. The process used to perform this optimization is known as backpropagation.

To understand backpropagation, we must first introduce a loss function $\mathcal{L}(\theta)$. The loss function quantifies how well the output of our NN, $f_\theta(x)$, matches the actual target output. In other words, it provides a measure of the prediction error of our model.

When a network is being trained on multiple samples simultaneously, i.e. in batches, the loss function is often defined as the mean loss over all samples in a batch:

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}_i(f_\theta(x^{(i)}), y^{(i)}). \tag{3.9}$$

Here, $\mathcal{L}_i(f_\theta(x^{(i)}), y^{(i)})$ represents the loss for a single example, with $x^{(i)}$ being the input and $y^{(i)}$ being the target output. The goal of the learning process is to find the set of parameters θ that minimize this loss function.

To achieve this, typically a method known as stochastic gradient descent is used. At each step, the parameters are adjusted in the direction that most decreases the loss. The gradient of the loss function with respect to the parameters, denoted $\nabla\mathcal{L}$, provides the direction of steepest ascent.Thus the parameters are updated as

$$\theta_{new} = \theta_{old} - \alpha\nabla\mathcal{L}(\theta_{old}). \tag{3.10}$$

Here, α is a hyperparameter known as the learning rate, which determines the step size in the direction of the gradient. Selecting an appropriate value for α is crucial. If α is set too low, the NN might become trapped in a local minimum while if it is too high, the network may oscillate and never converge to the global minimum.

To calculate the gradients, the chain rule of calculus is used. However, due to the complex and nested structure of a NN, naively applying the chain rule can be computationally expensive. This is where backpropagation is used. Backpropagation is an efficient algorithm for computing the gradients of all the parameters of the network with respect to the loss function.The essence of backpropagation is to move backwards through the network, from the output layer to the input layer, applying the chain rule to compute the gradients.

After these gradients are computed, they are used to update the parameters in the direction that minimizes the loss. This process is repeated for several iterations (or epochs) until the loss is minimized to a satisfactory level or no longer decreases significantly.

3.2 Contrastive learning

The intention of this section is to introduce the idea of contrastive learning primarily focusing on SimCLR's foundational concepts [18]. Following this introduction, attention will be directed towards the application of contrastive learning for its use in collider physics.

The fundamental concept of contrastive learning originates from image classification, with the objective of devising an algorithm capable of self-supervised image clustering. Central to this idea is the function $f(\cdot)$, which maps the data space $\mathcal{D}$ into a representation space $\mathcal{R}$. This function undergoes optimization through a loss function that can be interpreted as an optimization task executed within the representation space $\mathcal{R}$. The first research introducing the contrastive learning concept [18] utilizes a data space $\mathcal{D}$ constituted of various images I of different objects. The loss function was designed based on the principle of positive and negative pairs.

$$\begin{aligned} \text{Positive pairs:} \quad & \{(I_i, I_i')\} \\ \text{Negative pairs:} \quad & \{(I_i, I_j)\} \cup \left\{(I_i, I_j')\right\} \quad \text{for} \quad i \neq j. \end{aligned} \tag{3.11}$$

Positive pairs are composed of the original image I_i and a corresponding augmented variant I_i'. Image augmentation encompasses transformations like rotation, color channel omission, and pixel blurring. The negative pairs are formed by the original image I_i and an arbitrary different image I_j or augmented image I_j' from the batch. In this unlabeled data approach, a pseudo-class is constructed for each distinct, positive image pair. The basic idea of contrastive learning is to bring the positive pairs as close as possible to each other and to ensure the opposite for the negative pairs. This is achieved via the loss function which is defined as:

$$\mathcal{L} = -\log\left\{ \frac{e^{\left[\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_i')\right]/\tau}}{\sum_{j \neq i \in \text{batch}} \left[e^{\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j))/\tau} + e^{\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j'))/\tau} \right]} \right\}, \tag{3.12}$$

where $\boldsymbol{z} \in \mathcal{R}$ are representation vectors with their indices aligned with the preceding definition in eq. (3.11). $\tau > 0$ is a scalar hyper-parameter, referred to as temperature. In addition the cosine similarity sim measures the similarity between two vectors $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$:

$$\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j) \coloneqq \frac{\boldsymbol{z}_i \cdot \boldsymbol{z}_j}{\|\boldsymbol{z}_i\| \|\boldsymbol{z}_j\|} = \cos\theta_{ij}. \tag{3.13}$$

As is evident, minimizing the contrastive loss requires maximizing the similarity between similar samples (numerator) and concurrently minimizing the similarity of dissimilar samples (denominator). These individual task are also referred to as uniformity and alignment. For a deeper analysis of this classical approach of contrastive learning on jets we refer to [19].

3.2.1 Contrastive learning for anomalous jets

When training a contrastive network for an anomaly detection of jets task the network will only be trained on the background QCD data. In this case the original idea explained in the previous section breaks down as the network creates sub-classes within the QCD data and optimises only regarding features of the background. By doing so it is no longer ensured that a signal will be mapped to an out-of-distribution point within the representation space.

To help the network with the creation of an out-of-distribution signal point [20] introduces the idea of negative augmentations. As we are no longer referring to images as input we change the notation, now including the data vector $\boldsymbol{x} \in \mathcal{J}$ for the jet-data space. In this case we can now define two kinds of augmentations:

1. **Physical augmentations**
   For now, these augmentations can be understood as transformations that are desired to be mapped to the same point within the latent space[1].

2. **Anomalous augmentations**
   These augmentations mimic potential anomalies. The objective here is to ensure that the representation space is highly distinctive to these augmented instances.

With the help of this definition we are now able to construct a new pseudo label similar

to the ones described in eq. (3.12).

$$\begin{aligned} \text{Augmented pairs:} \quad & \{(\boldsymbol{x}_i, \boldsymbol{x}_i^*)\} \\ \text{Positive pairs:} \quad & \{(\boldsymbol{x}_i, \boldsymbol{x}_i')\} \\ \text{Negative pairs:} \quad & \{(\boldsymbol{x}_i, \boldsymbol{x}_j)\} \cup \left\{(\boldsymbol{x}_i, \boldsymbol{x}_j')\right\} \quad \text{for} \quad i \neq j. \end{aligned} \tag{3.14}$$

The augmented pair is now consisting of an original jet and an anomaly-augmented version of itself. Therefore the newly defined pairs enable us to incorporate a discriminative element into the loss function,

$$\mathcal{L}_{\text{AnomCLR}} = -\log\left\{ \frac{e^{\left[\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_i') - \text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_i^*)\right]/\tau}}{\sum_{j \neq i \in \text{batch}} \left[e^{\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j)/\tau} + e^{\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j')/\tau} \right]} \right\}. \tag{3.15}$$

With the introduction of anomaly-pairs, we enhance the mapping $f(\cdot) \coloneqq \mathcal{J} \to \mathcal{R}$ onto data features that lie beyond the background distribution. While the contrastive learning component of the loss function continues to optimize for alignment and uniformity, the anomaly-pair term disrupts this uniformity. Consequently, the representation space does not exhibit an uniform distribution for the background data, as certain regions will encapsulate features from the anomaly-augmented data. This suggests that anomalous data sharing similarities with features generated by anomaly-augmentations should be considered out-of-distribution within this representation space.

As detailed in [20], it's also feasible to remove the denominator of eq. (3.15), as the discrimination within the background data is not critical for anomaly detection. This makes the use of a less computationally demanding loss function possible:

$$\mathcal{L}^{+}_{\text{AnomCLR}} = -\log \mathrm{e}^{\left[\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_i') - \text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_i^*)\right]/\tau} = \frac{\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_i^*) - \text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_i')}{\tau}. \tag{3.16}$$

In this case the network is trained on mapping the physical augmented representation vectors $\boldsymbol{z}'$ to the same region as the original jet representation $\boldsymbol{z}$ while pushing away the augmented versions $\boldsymbol{z}^*$. As the trade off between nominator and denominator has vanished in this version it is also possible to remove the τ dependency.

3.3 Self attention

This section is meant to introduce the idea of self attention in general[1]. The main idea of self-attention is to capture the relationships between different constituents within a jet by computing attention scores. These attention scores are then used to create the final output. For the usage of self-attention in our case the input for the self-attention block is a batch of previously embedded jets[2]. That means that each jet at this stage consists of $n_{constits}$ constituents while each constituent corresponds to a d-dimensional vector. As for the next steps the batchsize is redundant we will leave this dimension out of the calculations for the further part of the chapter. Furthermore we will use index notation similar to [21]. We define two kind of indices:

$$\begin{aligned} \text{physical indices: } & i, j, k, \ldots \in \{1, 2, \ldots, n_{\text{constit}}\} \\ \text{model indices: } & \alpha, \beta, \gamma, \ldots \in \{1, 2, \ldots, n_{\text{model dimension}}\} \end{aligned} \tag{3.17}$$

Thus the input can be written as : $x_{i,\alpha}$ where i goes from 1 to n_{constit} and α from 1 to $n_{\text{model dimension}}$. The single-head self attention [22] for an input can now be constructed in three steps:

1. To begin with we want to measure the relation between x_i and a given x_j, embedded in the d-dimensional latent space. Instead of the standard scalar product $x_{i\alpha}x_{j\alpha}$, we introduce learnable latent-space transformations $W^{Q,K}$ to the elements

$$q_{i\alpha} = W^{Q}_{\alpha\beta}x_{i\beta} \quad \text{and} \quad k_{j\alpha} = W^{K}_{\alpha\beta}x_{j\beta} \tag{3.18}$$

   and use the directed scalar product.

$$A_{ij} \sim q_{i\alpha}k_{j\alpha} \tag{3.19}$$

   to encode the relation of x_j with x_i through k_j and q_i. At this moment the matrix A_{ij} encodes the dependencies of the constituents of one jet towards each other.

2. The next step is to think about normalisation as it is preferred to train with values $\approx \mathcal{O}(1)$. Thus we want each row of our matrix to be normalised. This leads to the

[1] Further specification for our implementation is done in section 4.3
[2] The technicalities of the mapping processed used are specified in section 4.2

following conditions:

$$A_{ij} \in [0,1] \quad \text{and} \quad \sum_j A_{ij} = 1. \tag{3.20}$$

Additionally, we want to include that A_{ij} will scale with the dimension of our model d. We therefore divide it through its square root. Combining these steps we can formulate the condition

$$A_{ij} = \text{Softmax}_j \frac{q_{i\alpha} k_{j\alpha}}{\sqrt{d}} \quad \text{with} \quad \text{Softmax}_j\left(x_j\right) = \frac{e^{x_j}}{\sum_k e^{x_k}}. \tag{3.21}$$

This leads to a spelled out definition as:

$$A_{i,j} = \text{Softmax}_j \left(\frac{W^Q_{\delta\gamma} x_{i\gamma} W^K_{\delta\sigma} x_{j\sigma}}{\sqrt{d}} \right) \tag{3.22}$$

3. Now that the network has constructed a basis to evaluate the relation between two input elements x_i and x_j, we use it to update the actual representation of the input information. To do so the attention matrix A_{ij} is combined with the input data. The output is then again transformed in latent space through another learnable matrix W^V,

$$x'_{i\alpha} = A_{ij} W^V_{\alpha\beta} x_{j\beta}. \tag{3.23}$$

In this form we see that the self-attention vector x' just follows from a general basis transformation with the usual scalar product, but with an additional learned transformation for every input vector.

To summarize our output of the self attention block it is helpful to have a look at the indices. For the weight matrices $W^{(V,Q,K)}_{\alpha\beta}$ both of the indices live in the model dimension d. Therefore the learning happens within the non physical space. On the other hand the attention matrix $A_{i,j}$ has only indices for the physical constituents of the jet. This makes sense as the idea of attention towards each other should be independent from the choice of the model dimension.

The self-attention mechanism alone faces the issue where each element in a sequence tends to predominantly attend to itself [22]. This problem can be addressed by incorporating multiple heads into the network. By performing several self-attention operations

in parallel, each with distinct learned weight matrices , the outputs are concatenated and passed through a final linear layer. In practice, the full calculation for all constituents, all attention heads, and for an entire batch is carried out in parallel with tensor operations.

3.4 Transformer encoder

Building upon the explained concept of self-attention, we can delve into the architecture of the transformer encoder, utilized for the mapping process. Our focus will be exclusively on the encoder component from [22], specifically the setup detailed by [19]. The setup essentially performs a sequence-to-sequence operation, comprising N structurally identical, successive blocks. These consecutive blocks incorporate three primary stages. The initial stage involves applying a multi-headed self-attention mechanism to the input constituent, with the outcome being combined with the input in a residual manner. This joint output then undergoes layer normalization, a well-established technique for stabilizing the NN's learning process[23]. Following this, the normalized output traverses through a residual feed-forward network, where each constituent is processed independently. This entire sequence within the transformer-encoder block undergoes repetition for N iterations. Subsequent to these iterations, the output is subjected to another round of layer normalization.

3.5 Autoencoder as anomaly score

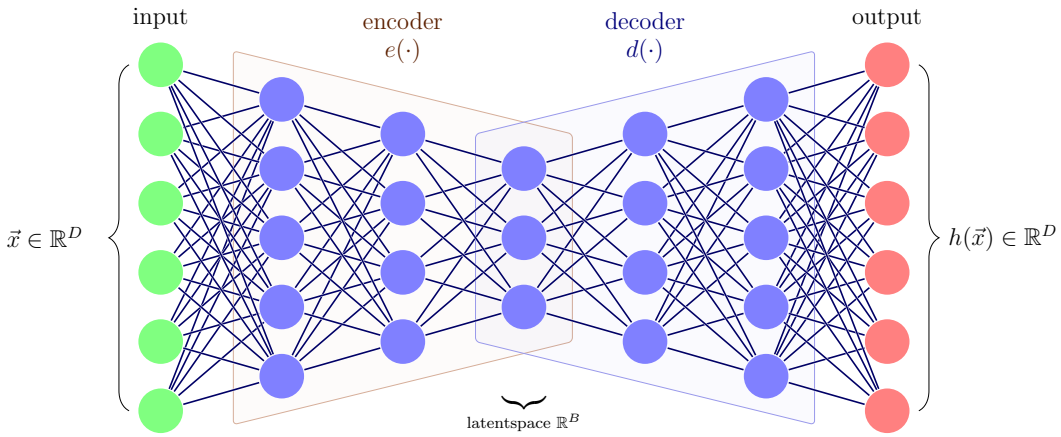


Figure 3.4: Sketch of an autoencoder

An Autoencoder(AE) is a type of NN widely used for unsupervised learning tasks, particularly in the field of dimensionality reduction and anomaly detection. The fundamental concept behind an AE involves two crucial steps:

encoding and decoding.

In the encoding phase, a high-dimensional input data point is mapped to a compressed latent space representation using a NN known as an encoder. Subsequently, in the decoding phase, the compressed latent space representation is mapped back to a reconstructed version of the original input data using a NN called a decoder.
Formally, let the input data dimension be denoted by D, and the desired dimension of the compressed representation, or bottleneck, be denoted by B. The encoder network can now be defined as: $e : \mathbb{R}^D \to \mathbb{R}^B$. The decoder network performs the reverse operation by mapping the compressed latent space representation back to a reconstructed version of the input data thus being defined as $d : \mathbb{R}^B \to \mathbb{R}^D$. Consequently, the AE itself can be represented as the composition of the encoder and decoder functions, denoted as :

$$h = d \circ e : \mathbb{R}^D \to \mathbb{R}^D. \tag{3.24}$$

When an input vector $\boldsymbol{x} \in \mathbb{R}^D$ is fed into the AE, it produces a reconstructed version $h(\boldsymbol{x}) = \boldsymbol{x}'$. The training objective of the AE is to reconstruct the input as best as possible. In our case we evaluate this by the mean squared error loss function between the input and its corresponding reconstructed output:

$$\mathcal{L}_{MSE}(\boldsymbol{x}, \theta) = (\boldsymbol{x} - \boldsymbol{x}')^2, \tag{3.25}$$

where θ represents the learnable parameters of the AE. In the ideal case where the AE can perfectly reconstruct the inputs, which must be possible when $B \geq D$, the function $h_\theta(\cdot)$ reduces to the identity function. However, when the bottleneck dimension B is smaller than the input dimension D, the AE may not be capable of precisely reconstructing all the original features present in the data. Consequently, the AE is encouraged to learn to reconstruct only the relevant features in the data.

To now use the AE for anomaly detection the training of the network is solely done on background data. Therefore the AE will be able to reconstruct the most important

features of this data to a good extend. If now, after the training, instances of different, anomalous features are given to the AE, it is expected to exhibit a larger reconstruction loss, i.e., $\mathcal{L}(\boldsymbol{x}_{background}, \theta) \leq \mathcal{L}(\boldsymbol{x}_{signal}, \theta)$. This reconstruction loss can then serve as an anomaly score, aiding in the identification of anomalous or unusual data points.

3.5.1 Normalising autoencoder

When an AE is used for an anomaly detection task, learning is performed using only the background data. The problem is that a small reconstruction error from a simple AE can, but certainly does not always, correlate with a large probability of the data space. To visualise this problem Fig. 3.5 is helpful.

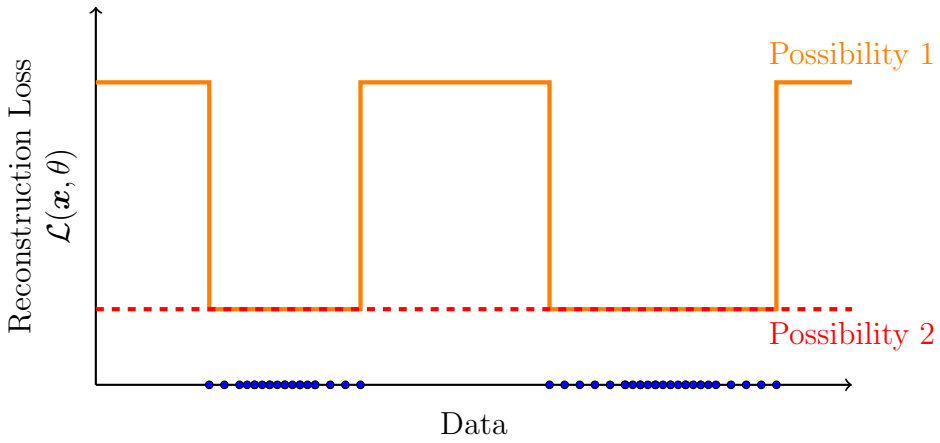


Figure 3.5: Problem of standard Autoencoder. Tye blue points indicate an one dimensional training data while the orange and red curves describe potential losses for the whole data space.

The sketched two possibilities correspond to the following two cases:

Possibility 1 : The AE is able to reconstruct the training data but fails in doing so for data points that lie outside of the trained distribution. In this case the reconstruction error would correspond to a higher likelihood.

Possibility 2 : In this case the AE is able to reconstruct all of the dataspace to the same reconstruction loss. The reconstruction error clearly does not correspond to a higher likelihood.

As a solution [24] proposes a new approach where the loss of the AE is modified so that it includes a normalisation, thus defining the model as a normalised autoencoder(NAE).

For the implementation on physical behavior this abstract will closely follow [25]. To introduce the normalisation we make use of energy-based models. We start by defining a probalistic loss using a Gibbs distribution to mimic a probability density region over phase space,

$$p_\theta(x) = \frac{e^{-\frac{E_\theta(x)}{T}}}{Z_\theta} \quad \text{with} \quad Z_\theta = \int_x dx e^{-\frac{E_\theta(x)}{T}} \tag{3.26}$$

where Z_θ can be seen as a partition function. We will set the temperature T to 1 as we will be able to change the shape of the distribution with the help of other hyperparameters later on (section 4.5.1). The energy function itself can be chosen as any non-linear function mapping a point to a scalar value $E_\theta(\boldsymbol{x}) : \mathbb{R}^D \to \mathbb{R}$. To now connect the idea of a probabilistic loss with an AE we set

$$E_\theta(\boldsymbol{x}) = \mathcal{L}_{MSE}(\boldsymbol{x}, \theta), \tag{3.27}$$

as defined in eq. (3.25). Here $\boldsymbol{x}$ is a vector of the data space we want to evaluate.
As the NAE can be seen as a probabilistic model it is trained to maximize the likelihood of the data. Therefore the loss function is the negative log-likelihood of the data. For one sample x we can thus follow:

$$\mathcal{L}(x) = -\log p_\theta(x) = E_\theta(x) + \log Z_\theta. \tag{3.28}$$

As explained in section 3.1.3 we now need to compute the gradient of the loss in order to be able to minimize it:

$$\begin{aligned} \nabla_\theta \mathcal{L}(x) &= \nabla_\theta E_\theta(x) + \nabla_\theta \log Z_\theta \\ &= \nabla_\theta E_\theta(x) + \frac{1}{Z_\theta} \nabla_\theta \int_x dx e^{-E_\theta(x)} \\ &= \nabla_\theta E_\theta(x) - \int_x dx \frac{e^{-E_\theta(x)}}{Z_\theta} \nabla_\theta E_\theta(x) \\ &= \nabla_\theta E_\theta(x) - \left\langle \nabla_\theta E_\theta(x) \right\rangle_{x \sim p_\theta}. \end{aligned} \tag{3.29}$$

The first term in this expression can be obtained using automatic differentiation from the training sample, while the second term is intractable and must be approximated. Additional details on how that approximation is performed within this research will be provided in Section 4.5.1.

The total loss can be defined as the expectations over the per-sample loss:

$$\mathcal{L} = \langle E_\theta(x) + \log Z_\theta \rangle_{x \sim p_{\text{data}}} . \tag{3.30}$$

Regarding the total gradient this allows us to rewrite the gradient of the loss as the difference of two energy gradients

$$\langle \nabla_\theta \mathcal{L}(x) \rangle_{x \sim p_{\text{dara}}} = \langle -\nabla_\theta \log p_\theta(x) \rangle_{x \sim p_{\text{dara}}} = \langle \nabla_\theta E_\theta(x) \rangle_{x \sim p_{\text{dara}}} - \langle \nabla_\theta E_\theta(x) \rangle_{x \sim p_\theta} . \tag{3.31}$$

The initial component samples from the training data, while the second component samples from the model. Depending on the energy's sign in the loss function, the contribution from the training dataset is termed as positive energy, whereas the contribution from the model is termed as negative energy.

4

Methodology

In this section, we will explain the methods and steps used to generate the results discussed in chapter 6.

4.1 Pre processing

As our goal is to detect an unknown signal we want to alter the data as little as possible to leave the signature of the signal the same. Therefore the used pre processing should be minimal. Initially, we compute the transverse momentum (p_T), pseudorapidity (η), and azimuthal angle (ϕ) for each constituent of a jet. Handling varying lengths of input makes it necessary to zero pad the jets with fewer constituents than $n_{constit}$. We then perform the only significant preprocessing action, which is to shift the jet toward the p_T-weighted centroid of the constituents:

$$\left(\sum_{i\in \text{ jet}} \eta_i p_{\mathrm{T},i}, \sum_{i\in \text{ jet}} \phi_i p_{\mathrm{T},i}\right) = (\eta_0, \phi_0) \stackrel{!}{=} (0,0)\,. \tag{4.1}$$

This measure is crucial as the augmentations employed for contrastive learning (see section 4.3) are predicated on the jets being centered. To ensure that the zero-padding introduced at the beginning does not influence the output of the mapping, we implement a masking. The information flow from zero-valued constituents is halted by setting their associated attention weights to zero. This is achieved technically by adding negative infinity to the attention weight prior to softmax normalization (eq. (3.22)). In addition, we exclude zero constituents from the sum over constituents to ascertain complete invariance of the transformer to zero-padding.

As a final step we divide all p_T through $p_{\mathrm{T,max}}$ to ensure that the $p_\mathrm{T} \in [0,1]$.

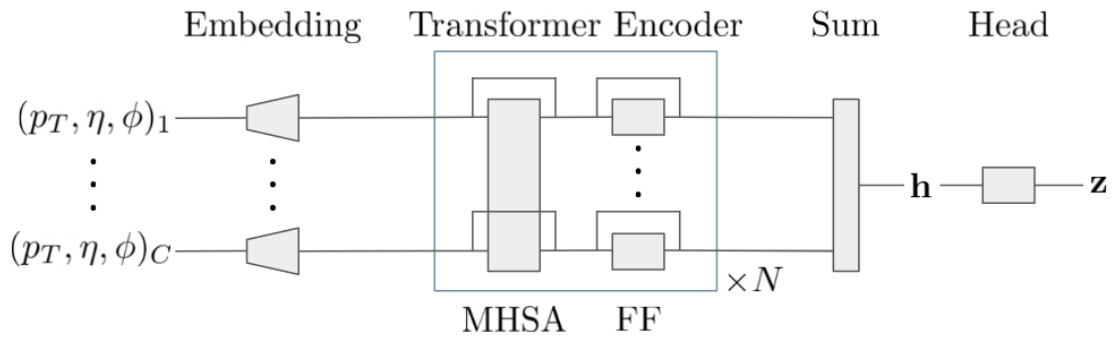


Figure 4.1: Illustration of the mapping architecture as presented in [19]. MHSA stands for multiheaded self-attention, and FF for a feed-forward block, as defined in section 3.4.

4.2 Mapping towards representation space

As described within the introduction to contrastive learning section 3.2, the main goal of the network's training is to improve a certain mapping. The mapping used within this thesis was introduced in [19] and is defined by:

$$f : \mathcal{J} \longrightarrow \mathcal{R} \tag{4.2}$$

with $\mathcal{R}$ being the high-dimensional representation space of dimension d and $\mathcal{J}$ being the jet data space. As illustrated in Fig. 4.1 the first step of the mapping is a learnable embedding layer that maps a single constituent of a jet, consisting of $[p_T, \eta, \phi]$, to the d dimensional vector space. As this is done for each constituent of a jet the output of this step is the vector $\boldsymbol{x}_{j,\beta}$ that will serve as input for the self attention block (similar as in section 3.3) within the transformer encoder, which describes the next step of the mapping process.

As described in sections 3.3 and 3.4 the output of this block looks like $x'_{i\alpha}$, when the batchsize is neglected. It is crucial to take a step back and think about eq. (3.23) in more detail for a good understanding of the subsequent summing step, also visualized in Fig. 4.1:

$$\begin{aligned} x'_{i\alpha} &= A_{ij} W^V_{\alpha\beta} x_{j\beta} \\ &= A_{ij} \nu_{j,\alpha}. \end{aligned} \tag{4.3}$$

Assuming that $\nu_{j,\alpha}$ represents a set of j vectors, we can deduce that a summation over the index i of the output will still encapsulate nearly all of the information regarding the dependencies within the constituents. Thus, the output of the transformer encoder block is:

$$h_\alpha = \sum_i x'_{i\alpha}. \tag{4.4}$$

Since the summants within the sum can be permuted, this introduces a permutation invariance for the constituents of one jet [19]. Important to keep in mind here is the fact that, as explained in section 4.1 the vectors originating from a zero-padded constituent will be added as a zero to this sum.

As last step a head network is added to provide additional representational power [19]. The output of the head network, $\boldsymbol{z} \in \mathcal{R}$, will then be used as input for the contrastive loss.

4.3 Contrastive learning of jets

To now shape this mapping $f : \mathcal{J} \longrightarrow \mathcal{R}$ in a way useful for our evaluation we make use of the idea of contrastive learning as described previously in section 3.2.1. To utilise this method we need two kind of augmentations for the loss function $\mathcal{L}^{+}_{\text{AnomCLR}}$(eq. (3.16)) on which we improve our mapping: **physical** and **anomalous** augmentations.

4.3.1 Symmetry inspired physical augmentations

The symmetry inspired physical augmentations describe transformations that the data should be invariant to. For the use within DarkCLR we stay close to [19] thus introducing the following augmentations:

Rotations in the η-ϕ plane In order to attain rotational symmetry around the jet axis, we generate augmented jets by randomly rotating each jet by an angle sampled from the interval $[0, 2\pi]$. It is important to recognize that rotations in the η-ϕ plane plane do not preserve the jet mass, as they are not Lorentz transformations. However, for narrow jets where the jet radius (R) is relatively small (i.e., $R \lesssim 1$), the corrections to the jet mass due to these rotations are negligible and can be disregarded.

Translation in the η-ϕ plane This augmentation shifts all of the constituents of a jet by the same random distance. The shifts themselves are constrained in each direction by the $\pm$ width of the distribution of either η or ϕ.

Permutation invariance As seen in section 4.2 our mapping to the representation space $f : \mathcal{J} \longrightarrow \mathcal{R}$ includes a self attention mechanism. Thus, as seen in eq. (4.4) the mapping includes a summation over all the constituents within one jet. Because this operation introduces an invariance to the swapping of summants, our network is also invariant under the permutation of constituents within one jet.

4.3.2 Theory inspired physical augmentations

Adding to the physical augmentations inspired by the underlying symmetry of jets one could also employ theory inspired augmentations as done in [19].

Collinear splitting This augmentation does not follow directly from symmetries but can be inspired by theory. Collinear splittings lead to divergences in perturbative quantum field theory. However, these divergences are typically removed because of the limited angular resolution of a detector, which fails to distinguish between two constituents with $p_{\mathrm{T},a}$ and $p_{\mathrm{T},b}$ when their separation ΔR_{ab} is much less than 1. To embody this property, we propose collinear augmentations by picking and splitting the components in such a way that the total p_{T} remains constant in a very small region of the detector, i.e

$$p_{\mathrm{T},a} + p_{\mathrm{T},b} = p_{\mathrm{T}}, \quad \eta_a = \eta_b = \eta, \quad \text{and} \quad \phi_a = \phi_b = \phi. \tag{4.5}$$

Low-p_{T} modifications We are aware that the low-p_{T} parts of jets, even those made by the same process and under the same conditions, can differ due to soft gluons. We use this knowledge to introduce another theory inspired augmentation of the low-p_{T} parts. We create the modified low-p_{T} jets by adding noise to the locations of these parts:

$$\eta' \sim \mathcal{N}\left(\eta, \frac{\Lambda_{\mathrm{soft}}}{p_{\mathrm{T}}} R\right) \qquad \text{and} \qquad \phi' \sim \mathcal{N}\left(\phi, \frac{\Lambda_{\mathrm{soft}}}{p_{\mathrm{T}}} R\right). \tag{4.6}$$

Here, $\mathcal{N}(\mu, \sigma)$ is a gaussian distribution with a standard deviation that depends on p_{T}. This means the standard deviation of the smeared positions is inversely proportional to the transverse momentum p_{T} of the particle (a higher p_{T} would mean less dispersion), and directly proportional to the jet radius R. In our studies, we've picked Λ_{soft} to be

100 MeV.

4.3.3 Anomalous augmentation

The anomalous augmentation are supposed to mimic the potential signal in order to create a distinction for similar inputs within the representation space.

Dropping constituents As our signal consists of dark jets we can use the fact that different parts of the jet are not be detectable for the detector. Therefore, we introduced an augmentation that mimics the behavior of vanishing jet constituents. In our implementation each constituent has probability $p_{\text{drop}}$of being dropped to 0. For a dropping probability of 30% the change of constituents with a $p_{\text{T}} > 0$ for QCD Jets is shown in Fig. 4.2.

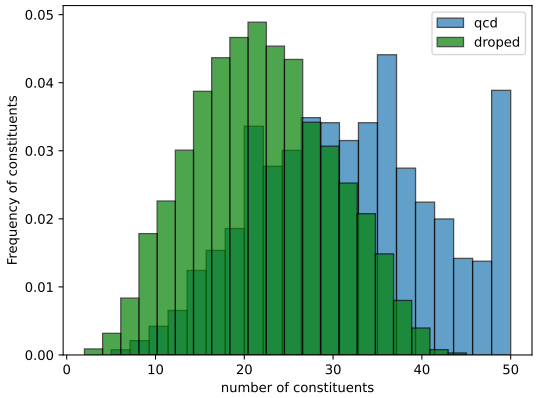


Figure 4.2: Histogram of constituents with $p_{\text{T}} > 0$ for standard QCD jets and their anomalous augmented versions for a $p_{\text{drop}}$of 30%.

4.4 Understanding the representation space

As we implement our mapping in a physical inspired way we also want to evaluate the created representation space under these assumptions. For this purpose we created two main ways of visualising the representation space.

4.4.1 Most important features

We define the most important feature to be on average (over the whole dataset) the feature with the highest reconstruction error, thus being the most relevant for the detection of the signal. A feature is defined as a single dimension within the representation space $\mathcal{R}^{\mathcal{D}}$.

4.4.2 Similarity matrix

In our work, we evaluate contrastive learning using a metric called cosine similarity, as detailed in equation (3.13). To help visualize this similarity, we use a matrix that includes all the similarities of the semivisible signal jets and the QCD background jets.

To create this matrix, we first arrange the vector representations $\boldsymbol{z}$ based on the number of p_{T} values that were greater than zero in the original jet used for their creation. Once the vectors are ordered, we build the similarity matrix, as shown in Figure (4.3).

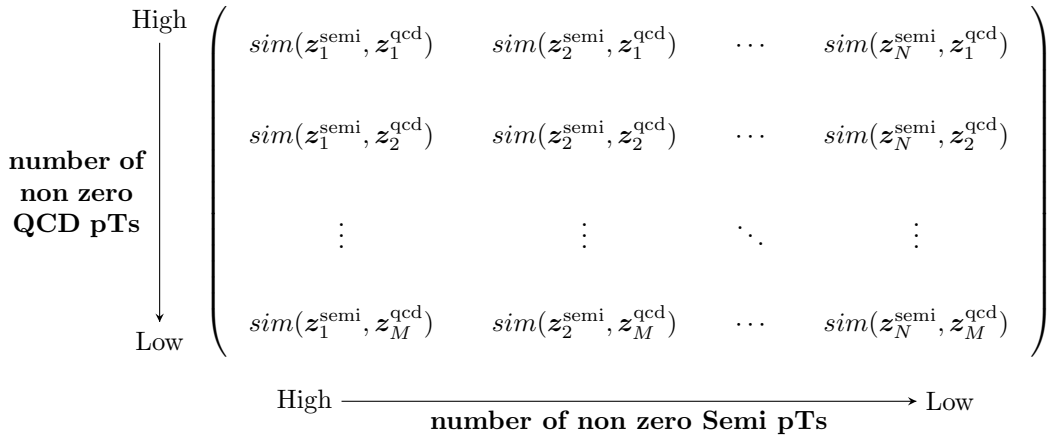


Figure 4.3: Sketch of similarity matrix. The subscript of the representation is not referring to a single entry but to a whole representation vector $\boldsymbol{z}$.

For easier understanding and visualisation, we convert this matrix into a heat map representation. Furthermore we set $N = M$ to have a quadratic form within the plot.

4.5 Evaluation of the representation space

For the evaluation of the representation space we used different approaches. As our goal is to do anomaly detection the most important one will be the normalised autoencoder.

As we further investigated the representation space we also used various classifiers including a linear classifier test, a standard classifier as well as a transformer classifier. These methods are all supervised thus meant to give us an estimate on how much information is introduced within the representation space.
Finally, the method used for perfomance measurement of the presented methods is explained.

4.5.1 Normalised autoencoder with on-manifold initialisation

To make use of the ideas presented in section 3.5.1 we need a method to sample from the model distribution p_θ to calculate the *normalisation term*

$$\langle \nabla_\theta E_\theta(z) \rangle_{z \sim p_\theta} , \tag{4.7}$$

where $\boldsymbol{z} \in \mathcal{Z} \subset \mathbb{R}^{\mathcal{D}}$. As in our case $\mathcal{D}$ is 1000 dimensional it is very hard to find high energy regions within this data space [24]. Thus we use the on-manifold initialisation described in [24].

In this method the fact is utilized that we can find high energy regions of p_θ by using a sufficiently trained AE. This is because a point with small reconstruction (potentially corresponding to a point with high p_θ) will be likely to lay near the decoder manifold defined as

$$\mathcal{M} = \{\boldsymbol{z} | \boldsymbol{z} = d(\boldsymbol{b}), \boldsymbol{b} \in \mathcal{B}\}, \tag{4.8}$$

with $d(\cdot)$ being the decoder defined in section 3.5 and $\mathcal{B}$ being the bottleneck latent space of the AE. As not all points within the manifold $\mathcal{M}$ will have a high energy, a Monte Carlo Markov Chain(MCMC) is initialised within $\mathcal{B}$. This way we first focus on taking samples from a suitable defined distribution in the low-dimensional latent space. To do so we need a probability density defined for this bottleneck latent space. In analogy to eq. (3.26) this is defined as:

$$q_\theta(b) = \frac{e^{-H_\theta(b)}}{\Psi_\theta} \quad \text{with} \quad H_\theta(b) = E_\theta(d(b)), \tag{4.9}$$

where H_θ can be seen as the bottleneck latent space energy. The MCMC are Langevin

Monte Carlo Chains defined as following:

$$\begin{aligned} \textbf{representation chain: } & z_{t+1} = z_t + \lambda_z \nabla_z \log p_\theta(z) \quad + \sigma_z \epsilon_t \\ \textbf{bottleneck latentspace chain: } & b_{t+1} = b_t + \lambda_b \nabla_b \log q_\theta(b) \quad + \sigma_b \epsilon_t \end{aligned} \tag{4.10}$$

with $\epsilon_t \sim \mathcal{N}_{0,1}$, λ being the step size and σ the standard deviation of the noise. To summarize, this means that we start by implementing the latent space chain within the decoder manifold $\mathcal{M}$. When reaching a sufficient density region within $\mathcal{M}$ we proceed to set up the representation chain within this region. Then the final sample is obtained by running a this representation chain.

As we have to sample from a high dimensional representation space it is difficult to cover the whole space with a reasonable length. Thus we use chains of shorter length and tweek the parameters of the chains to give more relevance to the gradients than to the noise. If $2\lambda \neq \sigma^2$ changing these parameters is equal to a change in the temperature within the probability distribution

$$T = \frac{\sigma^2}{2\lambda}. \tag{4.11}$$

To ensure a converging learning we use the same measurements to deny instabilities as in [25].

As a final step we define the model that will be considered within the evaluation. This is done through the use of a dataset consisting of representations created from dropped jets $\boldsymbol{x}_i^*$. The model achieving the best AUC's on this dataset will be defined as final model.

Summary of the NAE Method By introducing a way of using the reconstruction error of an AE as a probabilistic model a normalisation constraint was added towards the standard reconstruction error section 3.5.1. To train this model the gradient of the loss (eq. (3.31)) has to be evaluated. To calculate the positive energy $\langle \nabla_\theta E_\theta(x) \rangle_{x \sim p_{\text{data}}}$ we use the autograd function of pytorch. For the negative energy $\langle \nabla_\theta E_\theta(x) \rangle_{x \sim p_\theta}$ we use the on-manifold initialisation described in this chapter to sample from the model p_θ. By examining the expected loss described in eq. (3.31), we can observe that the training process can be characterized as a minimax problem. In this context, our objective is to minimize the energy of the training samples while simultaneously maximizing the energy of the MCMC samples. As a result we have ensured that our NAE will reconstruct signal

representation vectors $\boldsymbol{z}_{signal}$ with a higher loss than the trained on QCD representations $\boldsymbol{z}_{qcd}$.

4.5.2 Classifier

To perform evaluations in a supervised manner,we utilize classifiers. Typically, when referring to a classifier, we refer to a basic architecture that reduces the dimension of the previous output by a factor of ≈ 4, resulting in an integer output at the final stage.

In the context of a transformer classifier, we define a configuration where we replace the head network of our standard architecture with a classifier. This modified architecture produces an integer value as the output of the network.

4.5.3 Performance measurement

To measure the performance of our method and compare it towards the benchmark we use Receiver Operating Characteristic (ROC) curves. A ROC curve serves as the measure of the performance at varying thresholds. To sketch an ROC curve, two essential parameters are required:

- True Positive Rate (TPR): this represents the ratio of signal events correctly identified as signals by the classifier, otherwise known as the signal efficiency (ϵ_s).
- False Positive Rate (FPR): this signifies the proportion of background events wrongly classified as signals by the classifier, often referred to as the background mistag rate (ϵ_b).

In our evaluation a inverse ROC curve (ϵ_b^{-1} vs. ϵ_s) is used. It plots the reciprocal of the background mistag rate against the signal efficiency. The Area Under the ROC Curve (AUC) quantifies the area beneath the ROC curve and remains constant irrespective of the classification threshold. An AUC value of 0.5 implies that the model assigns class labels to the samples randomly. In general, a higher AUC denotes superior classification performance of the model.

5

Dataset and network settings

The dataset we are using as signal consist of semivisible jets. The specific hidden valley model we use was introduced in section 2.3. To now be more concrete we define:

particle	**mass**
Z'	2 TeV
q_d	500 MeV
π_d	4 GeV
ρ_d	5 GeV

Table 5.1: Defined masses within the model

In this configuration a $Z' - \rho_d^0$ mixing exist that enables the ρ_d^0 to decay into SM-quarks. As this is not possible for the other particles created within the dark shower they will be stable. The detector will thus not be able to detect them. The fraction of these escaping constituents is $r_{\text{inv}} = 0.75$. The signal and background dataset is simulated using Madgraph5 [26] for the hard process and the hidden valley model [27, 28] in Pythia8.2 [29] for showering and hadronization. The jets are reconstructed using the anti-k_T algorithm [12] with $R = 0.8$ in FastJet [30] and fulfill

$$p_{T,j} = 150 \ ... \ 300 \text{ GeV} \qquad \text{and} \qquad |\eta_j| < 2. \tag{5.1}$$

The used hyperparameters are summarized in table 5.2 for the transformer encoder and table 5.3 for the normalized autoencoder. If other hyperparameters were used to produce a result they will be specifically noticed.

Hyper-parameter	Value
Model (embedding) dimension	1000
Feed-forward hidden dimension	same
Output dimension	same
# self-attention heads	4
# transformer layers (N)	4
# head architecture layers	2
Dropout rate	0.1
Optimizer	Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$)
Learning rate	5×10^{-5}
Batch size	128
# constituents	50
# jets	100.000
# epochs	300

Table 5.2: Default configuration of the transformer encoder and the training process.

LMC parameters	latent	input
λ	100	100
σ	10^{-2}	10^{-4}
# of steps	30	30
metropolis	✓	✓
annealing	–	✓

Training Parameters	pre-AE	NAE
Learning Rate	10^{-3}	10^{-5}
Iterations	15k	40k
Batch Size	2048	512
Layers structure	[512:256:128:64:32:16:8]	-

Table 5.3: LMC and training parameters. The temperature is implicitly fixed by the noise and the step size as $T_x = 10^{-7}$ and $T_z \approx 10^{-6}$.

6

Results

This chapter is intended to present the results obtained through the DarkCLR method. For this purpose, the chapter will be divided into two sections. The first part will primarily focus on the visualisation of the representation space, which has been enhanced through our mapping technique coupled with contrastive learning. The second part will delve further into the evaluation of this representation space using an unsupervised method, more precisely a normalised Autoencoders. The $\mathcal{L}^{+}_{\text{AnomCLR}}$loss function will be employed as the default loss.

6.1 Visualising the representation spaces

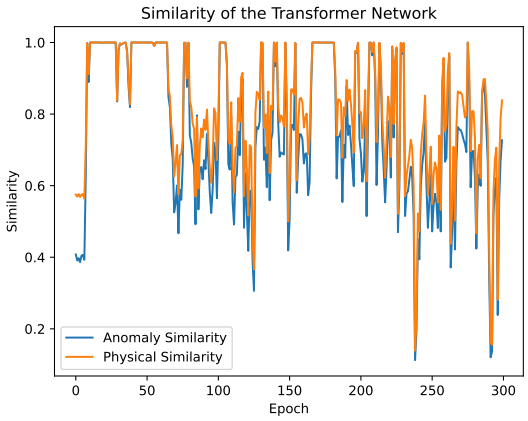
As described in section 3.2, the formation of the representation space $\mathcal{R}$ is significantly influenced by the applied augmentations. Since two different sets of augmentations were used within the research for this thesis, it is useful to distinguish them in two subsections, the *multiplicity*[1] and the *minimal* setup.

6.1.1 The multiplicity setup

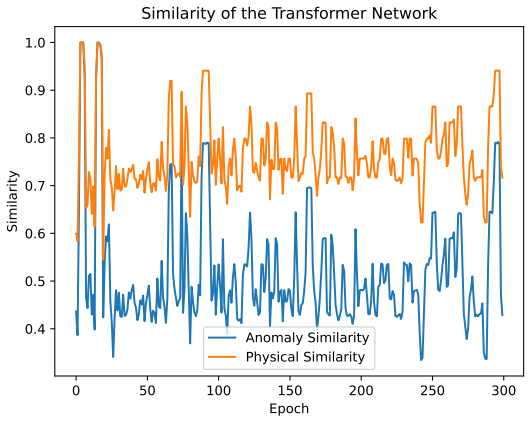
We started our research with using all of the augmentations described in section 4.3, thus using all the positive augmentation from [19].

To begin with we visualize the $\mathcal{L}^{+}_{\text{AnomCLR}}$by plotting the similarities. The Fig. 6.1 shows the similarities of the pairs where *Anomaly Similarity* is defined as $\{(\boldsymbol{x}_i, \boldsymbol{x}_i^*)\}$ and *Physical Similarity* as $\{(\boldsymbol{x}_i, \boldsymbol{x}_i')\}$ in a same manner as in eq. (3.14).
The converging value of the $\mathcal{L}^{+}_{\text{AnomCLR}}$[2] is varying between the different dropping probabilities p_{drop}. With p_{drop}we refer to the dropping probability introduced within the anomalous augmentation defined in section 4.3.3. For higher values we can see a quick

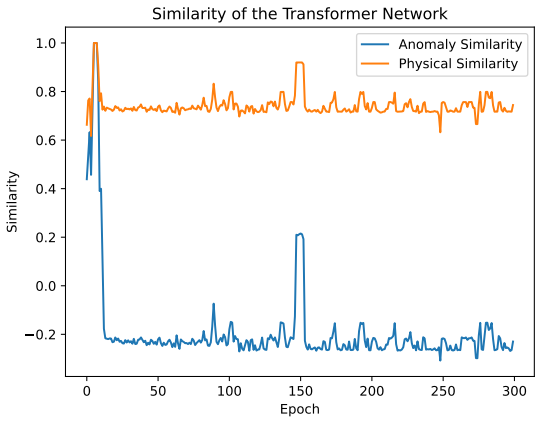
[1] We refer to multiplicity as the number of non zero p_{T} constituents within a jet.
[2] The value of the loss is directly proportional to the difference between the similarities as defined in eq. (3.16).



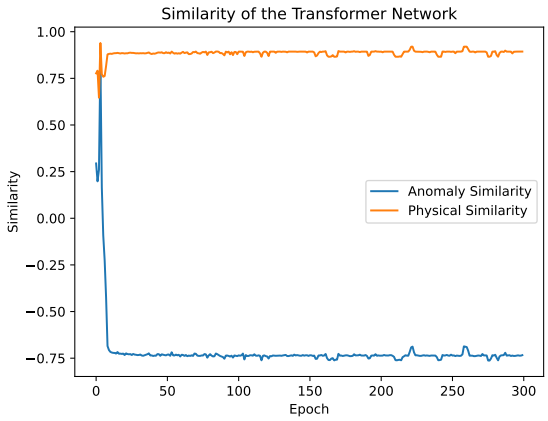
(a) Transformer Similarities for $p_{\text{drop}}= 0.1$



(b) Transformer Similarities for $p_{\text{drop}}= 0.3$



(c) Transformer Similarities for $p_{\text{drop}}= 0.5$

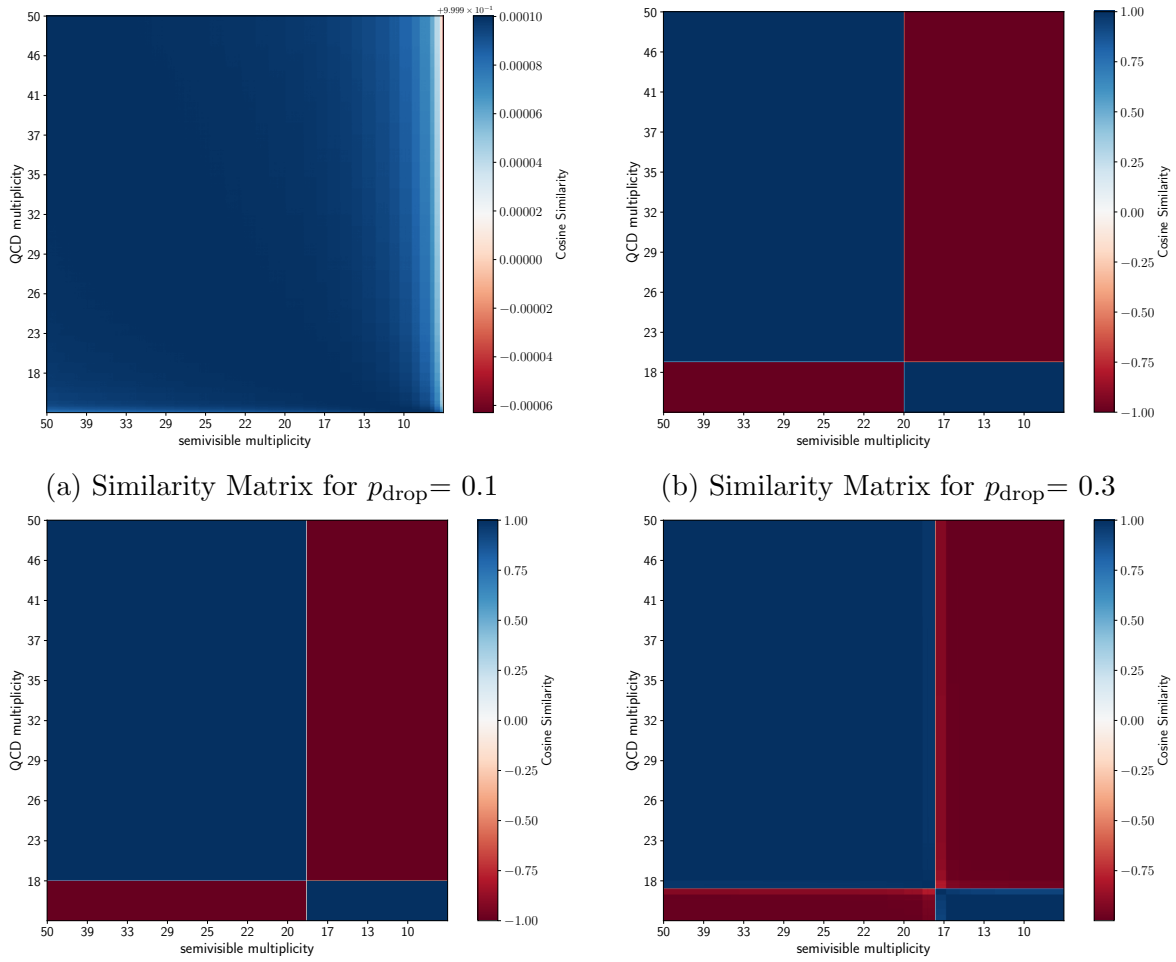


(d) Transformer Similarities for $p_{\text{drop}}= 0.7$

Figure 6.1: Transformer Similarities Plots. The value of the loss is directly proportional to the difference between the similarities. The orange curves describe the cosine similarity between positive augmented pairs while the blue curves do the same for anomalous pairs.

converging behavior. In contrary, for smaller values the learning becomes more unstable. This behavior is expected due to the fact that the transformer encoder is able to directly realise differences in the input shapes due to the masking we implemented, described in section 4.1. Thus it is easier to realise differences within the negative augmented pairs. Accordingly the learning will more quickly and consistently lead to a mapping that maps $\boldsymbol{x}_i^*$ and $\boldsymbol{x}_i$ to different points in space. This therefore explains the dependency of the training to the dropping probability seen in Fig. 6.1.

As a next step we want to validate that the trained mapping is discriminating not only positive augmented samples but also semivisible jets. For this purpose the similarity matrix, definde in section 4.4.2, is used.



(a) Similarity Matrix for p_{drop}= 0.1

(b) Similarity Matrix for p_{drop}= 0.3

(c) Similarity Matrix for p_{drop}= 0.5

(d) Similarity Matrix for p_{drop}= 0.7

Figure 6.2: Similarity Matrix Plots. For $p_{\text{drop}} \geq$ 10% the color blue refers to a cosine similarity of 1, meaning the vectors show in the same direction while. Red corresponds to a -1, indicating the opposite.

Fig. 6.2 reveals two distinct observations. Firstly, it becomes evident that the concept of dropping constituents as negative augmentation proves effective when an adequate dropping probability is employed. Within the representation space, jets with a high number of constituents are consistently mapped to the same point, located in the upper left corner, while jets with a small number of constituents are similarly mapped to a distinct point in the lower right corner. This allows for a distinction of QCD and

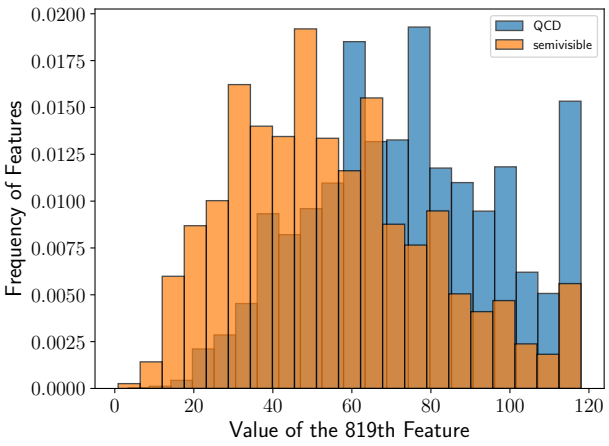
Drop Chance	**Lowest Feature Value**	**Highest Feature Value**
0.1	-121.78	120.68
0.2	-8.23	8.49
0.3	-0.31	0.31
0.4	-0.30	0.30
0.5	-0.31	0.31
0.6	-0.35	0.35
0.7	-0.63	0.62

Table 6.1: Drop chance and its corresponding feature value ranges. A clear increase of the norm can be seen for $p_{\text{drop}} \leq 20\%$, corresponding to a non converging training.

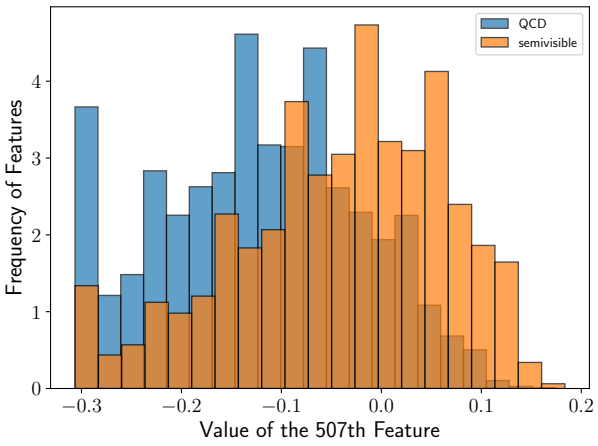
semivisible jets as the they differ in their number of detected constituents. Secondly, a notable phenomenon observed is the occurrence of mapping collapse to some extent. This can be argued by the absence of a well-defined transition zone between the mapping of jets onto the "high" and "low" constituent points. Consequently, a vast majority of jets tend to be mapped onto these two points. In cases where the dropping probability does not allow for sufficient separation within the training, no correlation in regard to the position on the hypersphere can be observed, seen for $p_{\text{drop}} = 10\%$.

To further visualise the representation space the most important features are plotted as explained in section 4.4. They are shown in Fig. 6.3.
Focusing on a single plot of Fig. 6.3 it is apparent that the mapping creates an overlapping but still significant difference within the representations of semivisible and QCD jets. Additional, it is obvious that the distribution shape is similar in all cases. The sole difference lies in the norm of the representation vectors, denoted as $\boldsymbol{z}$. Remarkably, this pattern is observed across all 1000 features for each of the displayed dropping probabilities. The only discrepancy within one representation space seems to be a reflection along the y-axis or variations in the norm.

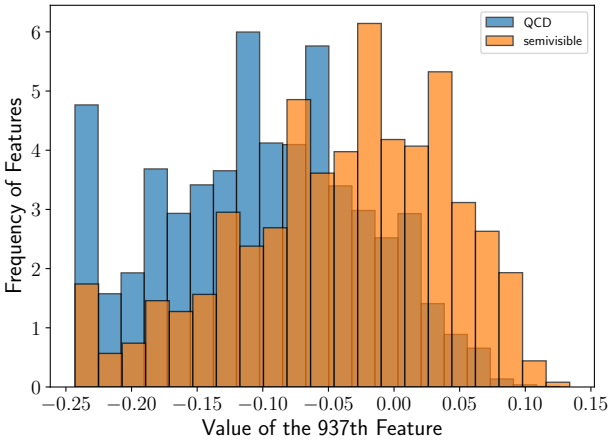
As final step of the visualisation a comprehension of the norm values is given in table 6.1 as the previos paragraph indicates an importance within the norm distribution. Doing so a clear correlation between a non converging of the $\mathcal{L}^{+}_{\text{AnomCLR}}$Loss in Fig. 6.1 and an increasing norm of the representation vectors $\boldsymbol{z}$ is observed.
To ensure that the collapse onto the two points and the degenerated features are not a problem we investigated the representation space $\mathcal{R}$ and the original datasets $\mathcal{J}$ with a



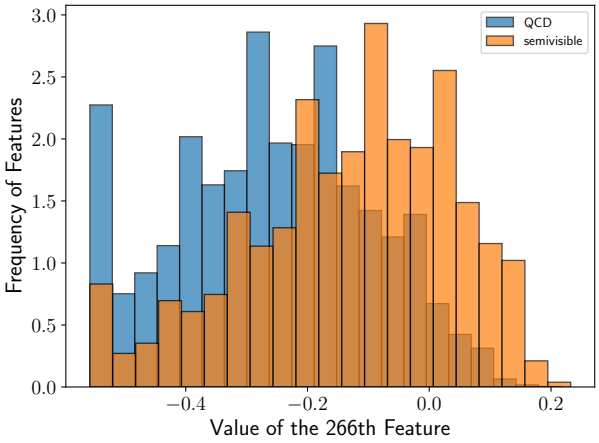
(a) Most important feature for p_{drop}= 0.1



(b) Most important feature for p_{drop}= 0.3



(c) Most important feature for p_{drop}= 0.5



(d) Most important feature for p_{drop}= 0.7

Figure 6.3: Most important feature plots for the multiplicity setup. The only discrepancy within features seems to be a reflection along the y-axis or variations in the norm.

supervised Classifier. This will be further discussed in the evaluation, section 6.2.1.

6.1.2 The minimal setup

As will be explained in section 6.2.2 there are good arguments to focus only on symmetry inspired positive augmentations, leaving theory inspired augmentations asides. This setup using less positive augmentations will be referred to as *minimal setup*.

Fig. 6.4 visualises the similarities of a training done with the minimal setup. A clear

change in the loss of the network can be observed in comparison to Fig. 6.1. Explanations for this difference can be the fact that the collinear splitting introduced a multiplicity correlation within the positive augmentations as the augmented jets will have an increased value for multiplicity. Therefore the network is pushed onto solely focusing upon the multiplicity. Supplementary an explanation for this difference could be the fact that the network is able to translate the missing noise from $\boldsymbol{x}_i^{'}$ (due to no Low-p_{T} modification) into a smoother learning.

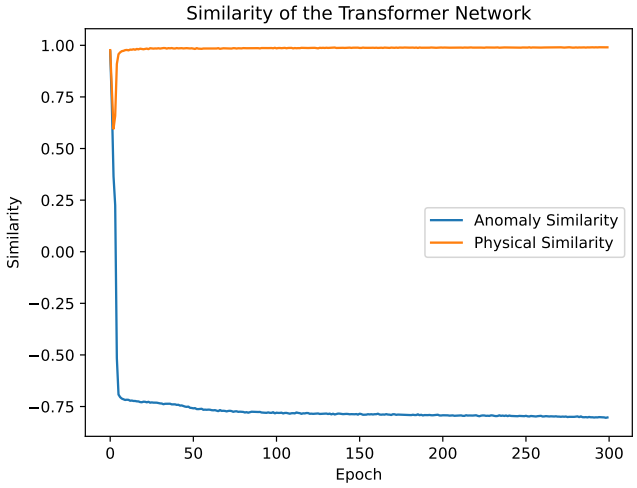


Figure 6.4: Transformer Similarities for p_{drop}= 0.5. The orange curve describe the cosine similarity between positive augmented pairs while the blue curves do the same for anomalous pairs.

In the given scenario for a dropping probability of 50%, the similarity matrix (Fig. 6.5) provides valuable insights into why this specific representation space is more informative compared to the multiplicity-focused counterpart. It is clear that the network is learning new information, as the mapping is not constrained to only two points on the hypersphere $\mathcal{S}^{d-1}$ anymore.

To further analyse the representation space of the minimal setup we can also visualise the individual features, as seen in Fig. 6.6. Here we observe a clear deviation from the earlier examples (Fig. 6.3) as there is a difference within the shape of the histograms. In addition it is also notable that the minimal setup leads to a bigger range of feature values being in an interval of $[-310, 300]$, therefore being significantly higher then the norm values for the multiplicity setup.

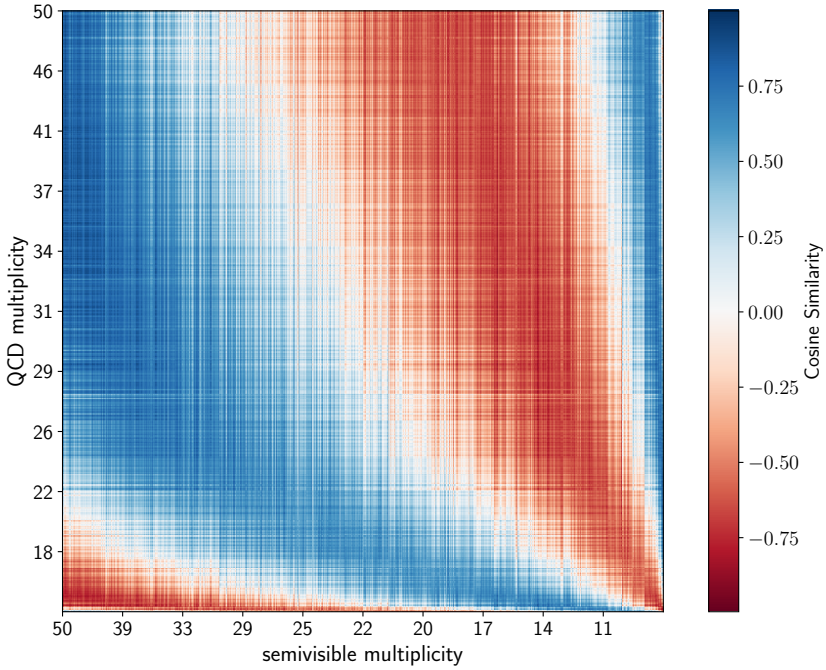
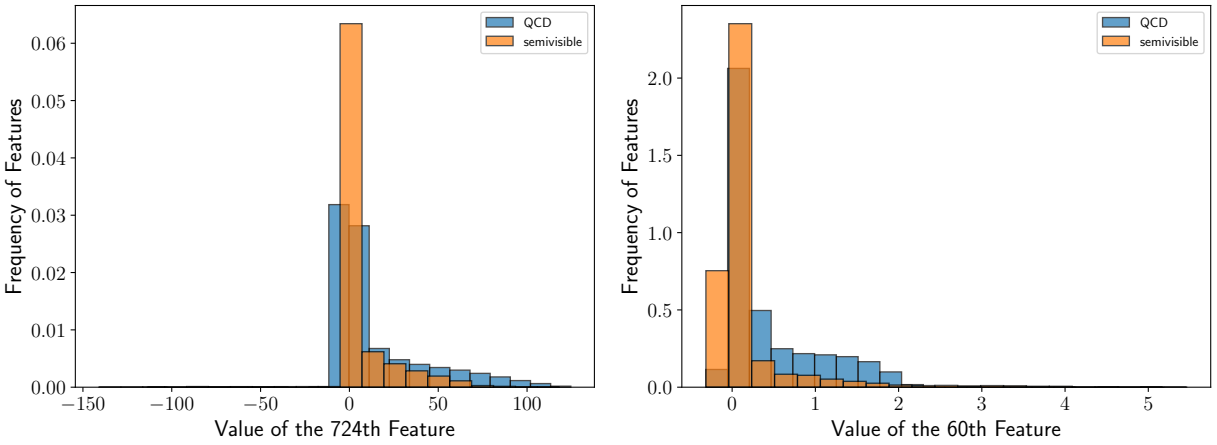


Figure 6.5: Similarity Matrix for $p_{\text{drop}}= 0.5$. It can be observed that the mapping is not constrained to only two points on the hypersphere $\mathcal{S}^{d-1}$ due to a broader range of cosine similarities.



(a) Most important feature

(b) Least important feature

Figure 6.6: Significant features for $p_{\text{drop}}= 0.5$. A difference within the features is observable while showing that the *importance* of a feature correlates to its range of values.

6.2 Evaluation

To start this section we would like set a baseline to compare to. This can be gained by the use of supervised learning. Therefore a transformer classifier is trained on the same, slightly pre processed (according to section 4.1) jet data space $\mathcal{J}$. This achieves an AUC of **0.81** . Thus we can see this value as our upper limit from now on.

Similar to the previous section 6.1 this chapter is also splitted two part focusing on the *multiplicity* and *minimal* setup independently.

6.2.1 The multiplicity setup

For the evaluation 3 runs of the NAE were done to evaluate the representation space for dropping probabilities going from 10% up to 70% in steps of 10%. The results can be seen in Figs. 6.7 and 6.8 .

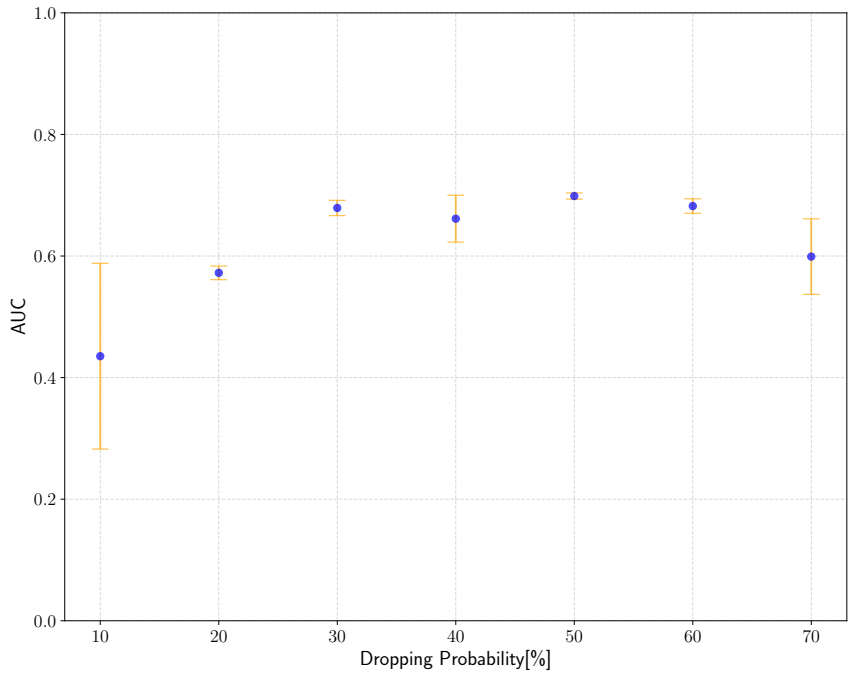


Figure 6.7: AUC of the multiplicity setup for different droppin probabilities . A plateau can be noticed within the range of 30-60%.

Thereby shows Fig. 6.7 that the idea of contrastive learning is working. For a sufficiently high dropping probability p_{drop}, in between 30% and 60%, we see a plateau in terms of the AUC. Here we can deduce that the anomalous augmentations really introduced information useful for the semivisible dataset. This resulted in a representation

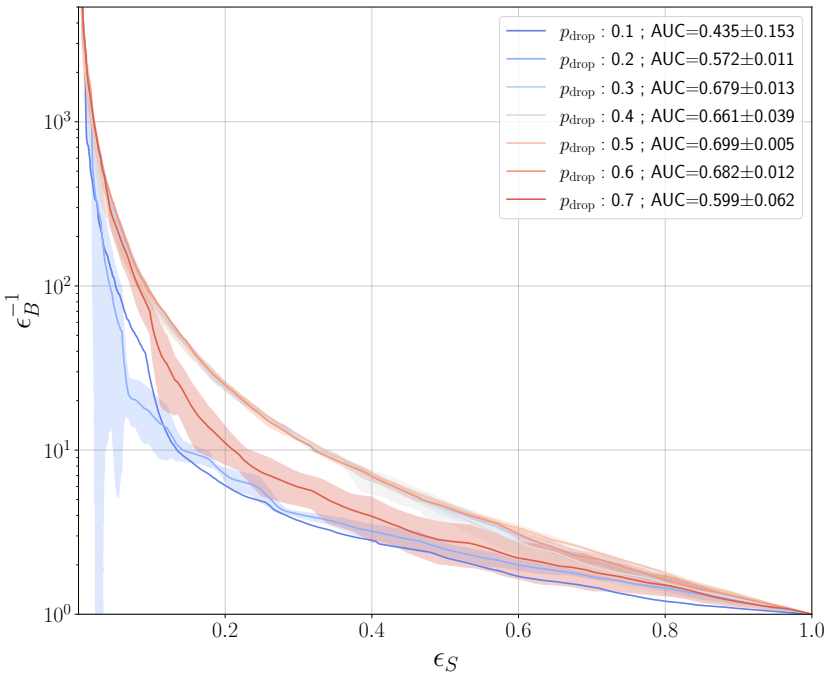


Figure 6.8: ROC Curves for the multiplicity setup for different droprates. The error of 0.1 is not plotted as it high and thus would distort the clarity and interpretability of the plot.

space which produced better and more stable results, also supported by the similarity matrices in Fig. 6.2. At the same time a decreasing performance is observed in the cases of $\leq 20\%$ and $\geq 70\%$. For the case of small p_{drop}this can be attributed to the fact that the difference between the augmented jets $\boldsymbol{x}_i^*$ and the original version of it $\boldsymbol{x}_i$ is not sufficient. This is also supported by the non converging loss seen in Fig. 6.1. For the case where $p_{\text{drop}} \geq 70\%$ we can assume that too much information within the augmented jet is dropped.

Different methods were tried to improve the AUC of the multiplicity focused setup, including changes to the preprocessing techniques, utilization of the $\mathcal{L}_{\text{AnomCLR}}$Loss, and introduction of a new head network. However, these attempts were not successful. An evaluation of the representation space $\mathcal{R}$ was carried out using a supervised classifier to determine if similar results to the baseline could be achieved in general by using the representation space. It was observed that some information was lost during the mapping process, leading to an average AUC value of 0.71 for the supervised method. A value close to the plateau observed in Fig. 6.7.

The subsequent search for the reason of the information loss ended when the the multiplicity of the data space $\mathcal{J}$ was examined. We found that the same AUC value of 0.71 was consistently observed in this aspect as well. This observation led us to the conclusion that the primary feature encoded within the representation space is indeed the multiplicity. This aligns with the description of the representation space's shape presented in section 6.1.1. The collapse phenomenon observed in that section can be attributed to the fact that the transformer enoder model somehow incorporates the shape of the number of constituents within each feature.

After realising this we hypothesised that the reason can be the fact that the positive augmentations used in JetCLR [19] are not positive in our case. That is, we do not want our representations to be invariant to the same theory inspired as in the case of QCD and top jets. Our current hypothesis is that by adding noise and splitting jets, we are destroying information that is important for the recognition of the difference between QCD and semivisible jets. Especially the observed stable loss of Fig. 6.4 supports that we turned the mapping to solely focus on the multiplicity by introducing a multiplicity correlation within the positive augmentations by splitting jets.

6.2.2 The minimal setup

To solve the problem of information loss apparent in the multiplicity set of augmentations we stopped using the theory inspired augmentation, defined in section 4.3.2. Thus the minimal setup is only consisting of symmetry inspired augmentations (i.e. rotation, translation and permutation) in addition to the essential anomalous one.

The ROC curves of five runs of the NAE trained on the same representation space $\mathcal{R}$, created with a dropping probability $p_{\text{drop}}$of 50%, are plotted in Fig. 6.9.
Fig. 6.9 shows that the unsupervised evaluation of this representation space is improving in comparison to the multiplicity setup(Fig. 6.8), reaching an AUC score of $\mathbf{0.767 \pm 0.034}$ [3]. It is apparent that the first runs deviates significantly from the other, probably due to some bad mode configurations. Therefore an evaluation of this representation space seems to be harder for the normalised auto encoder compared to the simpler one visualised in section 6.1.1. In addition, while a compatible result can be reproduced for $p_{\text{drop}}= 0.5$, we have not yet been able to achieve the same for $p_{\text{drop}}=0.3$.

[3] For this evaluation a batchsize of 512 was also chosen for the pre-AE training.

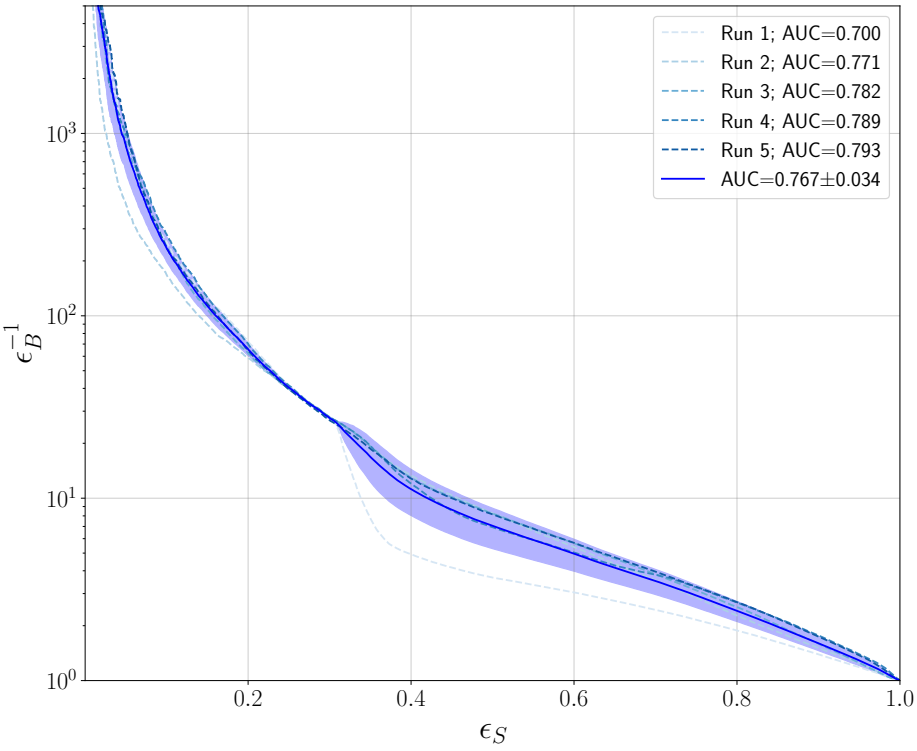


Figure 6.9: ROC Curves for the minimal setup with $p_{\text{drop}} = 0.5$. Run 1 deviates significantly from the other runs, specially for $\epsilon_S > 0.3$.

As the current stage of this project remains ongoing, our primary focus centers precisely on the task of reducing the inconsistency in the evaluation of the more informative representation space and creating a model agnostic model by not being fixed on one dropping probability.It is also crucial to contextualize this value of **0.767 ± 0.034** by noting that no substantial improvements have been made to the hyperparameters of the NAE. Consequently, it is reasonable to expect even higher values can be achieved utilizing this method, potentially consistently surpassing the benchmark value of **0.767±0.005** from [25].

7

Conclusion and outlook

Within this thesis a new self-supervised method for anomaly detection called DarkCLR is presented. Using a contrastive loss function modified for anomaly detection (section 3.2), it is possible to create representation spaces sensitive to signals outside the training distribution. This is additional to the benefits of contrastive learning, namely that the representations are still invariant under symmetry augmentations. Furthermore we investigate possibilities of evaluating these representation spaces with unsupervised methods. We especially focus on the use of a normalised autoencoder.

It was possible to achieve good results on a challenging semivisible jet data set (section 2.3 and chapter 5). Working with a multiplicity focused representation space we can show that the trained mapping of jets to a high dimensional representation space is resulting in a simplified representation space, visualized in Fig. 6.7. When evaluating this representation space, our unsupervised NAE was able to achieve similar AUC values as a supervised classifier, without being constrained to a certain dropping probability within the anomalous augmentation.

By actively shifting away the focus from the number of constituents per jet we were able to create a more informative, therefore more complex representation space. The unsupervised evaluation of this representation space achieves better AUC scores than the multiplicity focused one, increasing from **0.70** to **0.767** for a specific dropping probability. Using this method we achieve similar values as the benchmark [25] while having introduced no non-linear pre processing step.

The evaluation of the minimal representation space introduces some challenges, mainly the dependency on the dropping probability and the inconsistent learning of the normalized autoencoder seen in Fig. 6.9. If these problems can be eliminated it would be very exciting to investigate the model agnostics of the method.

Possible solutions to these challenges are being investigated as we are still working on

the project. Since we have not been able to study the hyperparameters of the normalized autoencoder extensively, we hope to resolve some of the challenges. Moreover it would also be interesting to investigate the influence of pre processing the representations. This could potentially increase the consistency of the NAE by decreasing differences within the input values.

There are also possible enhancements regarding the transformer encoder. One example would be to introduce a logarithmic pre processing to ensure a higher focus upon the smaller p_{T} regions. Another idea is to implement a trade off between the positive and negative augmentation in the $\mathcal{L}^{+}_{\mathrm{AnomCLR}}$(eq. (3.16)) which would enable shifting around with the attention given towards the multiplicity.

Finally other self supervised tools for the evaluation could be examined, for example the maximum likelihood autoencoder introduced in [31].

To conclude, DarkCLR is a promising method for unsupervised, and possibly model agnostic, anomaly detection. Nevertheless, it still has room for improvement, which would be interesting to further research.

References

[1] A. Collaboration, *Observation of a new particle in the search for the standard model higgs boson with the ATLAS detector at the LHC*, Physics Letters B **716**, 1 (2012), https://doi.org/10.1016%2Fj.physletb.2012.08.020.

[2] C. Collaboration, *Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC*, Physics Letters B **716**, 30 (2012), https://doi.org/10.1016%2Fj.physletb.2012.08.021.

[3] H. Andernach and F. Zwicky, *English and spanish translation of zwicky's (1933) the redshift of extragalactic nebulae*, 2017, arXiv:1711.01693 [astro-ph.IM].

[4] V. C. Rubin and J. Ford W. Kent, *Rotation of the Andromeda Nebula from a Spectroscopic Survey of Emission Regions*, apj **159**, 379 (1970).

[5] R. Massey, T. Kitching, and J. Richard, *The dark matter of gravitational lensing*, Reports on Progress in Physics **73**, 086901 (2010), https://doi.org/10.1088%2F0034-4885%2F73%2F8%2F086901.

[6] K. Freese, *Status of dark matter in the universe*, International Journal of Modern Physics D **26**, 1730012 (2017), https://doi.org/10.1142%2Fs0218271817300129.

[7] G. Bertone, D. Hooper, and J. Silk, *Particle dark matter: Evidence, candidates and constraints*, Phys. Rept. **405**, 279 (2005), arXiv:hep-ph/0404175.

[8] J. L. Feng, *Dark Matter Candidates from Particle Physics and Methods of Detection*, Ann. Rev. Astron. Astrophys. **48**, 495 (2010), arXiv:1003.0904 [astro-ph.CO].

[9] B. Carr, F. Kuhnel, and M. Sandstad, *Primordial Black Holes as Dark Matter*, Phys. Rev. D **94**, 083504 (2016), arXiv:1607.06077 [astro-ph.CO].

[10] S. Nojiri and S. D. Odintsov, *Introduction to modified gravity and gravitational alternative for dark energy*, eConf **C0602061**, edited by A. Borowiec, 06 (2006), arXiv:hep-th/0601213.

[11] Wikimedia Commons, *Standard model of elementary particles*, Visited on 06/07/2023, 17 September 2019, https://commons.wikimedia.org/wiki/File:Standard_Model_of_Elementary_Particles.svg.

[12] M. Cacciari, G. P. Salam, and G. Soyez, *The anti-kt jet clustering algorithm*, Journal of High Energy Physics **2008**, 063 (2008), https://doi.org/10.1088%2F1126-6708%2F2008%2F04%2F063.

[13] E. Bernreuther, F. Kahlhoefer, M. Krämer, and P. Tunney, *Strongly interacting dark sectors in the early Universe and at the LHC through a simplified portal*, JHEP **01**, 162 (2020), arXiv:1907.04346 [hep-ph].

[14] T. Plehn, A. Butter, B. Dillon, and C. Krause, *Modern Machine Learning for LHC Physicists*, (2022), arXiv:2211.01421 [hep-ph].

[15] M. A. Nielsen, *Neural networks and deep learning* (Determination Press, 2015).

[16] T. Rashid, *Make your own neural network: an in-depth visual introduction for beginners* (CreateSpace Independent Publishing Platform, 2016).

[17] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, *Activation functions: comparison of trends in practice and research for deep learning*, 2018, arXiv:1811.03378 [cs.LG].

[18] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, *A simple framework for contrastive learning of visual representations*, 2020, arXiv:2002.05709 [cs.LG].

[19] B. M. Dillon, G. Kasieczka, H. Olischlager, T. Plehn, P. Sorrenson, and L. Vogel, *Symmetries, safety, and self-supervision*, SciPost Phys. **12**, 188 (2022), arXiv:2108.04253 [hep-ph].

[20] B. M. Dillon, L. Favaro, F. Feiden, T. Modak, and T. Plehn, *Anomalies, Representations, and Self-Supervision*, (2023), arXiv:2301.04660 [hep-ph].

[21] A. Butter, N. Huetsch, S. P. Schweitzer, T. Plehn, P. Sorrenson, and J. Spinner, *Jet Diffusion versus JetGPT – Modern Networks for the LHC*, (2023), arXiv:2305.10475 [hep-ph].

[22] A. Vaswani et al., *Attention is all you need*, 2017, arXiv:1706.03762 [cs.CL].

[23] J. L. Ba, J. R. Kiros, and G. E. Hinton, *Layer normalization*, 2016, arXiv:1607.06450 [stat.ML].

[24] S. Yoon, Y.-K. Noh, and F. C. Park, *Autoencoding under normalization constraints*, 2021, arXiv:2105.05735 [cs.LG].

[25] B. M. Dillon, L. Favaro, T. Plehn, P. Sorrenson, and M. Krämer, *A normalized autoencoder for lhc triggers*, 2023, arXiv:2206.14225 [hep-ph].

[26] J. Alwall et al., *The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations*, JHEP **07**, 079 (2014), arXiv:1405.0301 [hep-ph].

[27] L. Carloni and T. Sjostrand, *Visible Effects of Invisible Hidden Valley Radiation*, JHEP **09**, 105 (2010), arXiv:1006.2911 [hep-ph].

[28] L. Carloni, J. Rathsman, and T. Sjostrand, *Discerning Secluded Sector gauge structures*, JHEP **04**, 091 (2011), arXiv:1102.3795 [hep-ph].

[29] T. Sjöstrand et al., *An introduction to PYTHIA 8.2*, Computer Physics Communications **191**, 159 (2015), https://doi.org/10.1016%2Fj.cpc.2015.01.024.

[30] M. Cacciari, G. P. Salam, and G. Soyez, *FastJet user manual*, The European Physical Journal C **72**, 10.1140/epjc/s10052-012-1896-2 (2012), https://doi.org/10.1140%2Fepjc%2Fs10052-012-1896-2.

[31] P. Sorrenson, F. Draxler, A. Rousselot, S. Hummerich, L. Zimmermann, and U. Köthe, *Maximum likelihood training of autoencoders*, 2023, arXiv:2306.01843 [cs.LG].

Acknowledgements

First of all, I want to thank my research supervisors Prof. Dr. Tilman Plehn, Dr. Tanmoy Modak and Luigi Favaro for giving me the opportunity to work in a great atmosphere and to gain first experiences in research. Thank you Tilman for answering my rather unusual mail with personal advice and for giving me the opportunity to work on such an interesting interface between machine learning and physics. Thanks a lot, Luigi and Tanmoy, for introducing me to PyTorch and QFT so well – and for your infinite patience and good answers in spite of the amount of questions I had.

Thanks also to Dr. Caroline Heneka for kindly agreeing to be the second examiner for this thesis and for showing interest in the topic.

I would also like to thank the whole Pheno group for always being open for some physics and statistics questions. Additional thanks to Jonas for explaining the idea of a transformer in such a comprehensible way and to Lorenz for answering redundant questions about JetCLR. Also thanks to the S-Fitter office for not only providing shelter while everyone was at the summer school, but also for giving feedback on basic QFT questions and LaTeX settings.

A special thanks goes to Mathias Backes, not only for proof-reading this thesis, but also for the enthusiasm for physics while holding the best tutorial i ever attended and the guidance provided afterwards.

Getting through physics studies required more than just academic and scientific support. Therefore, I would like to thank all those who made sure that I did not go crazy in the KIP basement during exam periods. Thank you Anton, Leo and Leon for putting up with me not only as a friend but also as a roommate, Lenni for listening to my problems while walking the Philosophenweg and Paul for always having an ear for me.

Johanna, I hope you know how much you have helped me through the last weeks, thus just thank you for everything.

Finally, I would like to thank my family for always being there for me and helping me find my way - directly and indirectly.

Declaration

I hereby formally declare that I have composed the present thesis myself and without use of any other than the cited sources and aids [1].

Heidelberg, 20 July 2021 .....

Jan Hendrik Rüschkamp

[1] The only additional aid used was DeepL and GPT 4.0 for spelling correction and improvement of my english.