FAKULTÄT FÜR PHYSIK
Institut für Experimentelle Teilchenphysik

Applying the Object Condensation approach to the COCOA-dataset
(Anwendung des Object Condensation Ansatzes auf den COCOA-Datensatz)

Bachelorarbeit
von

Fabian Riemer

am Institut für Experimentelle Teilchenphysik

Reviewer:	Prof. Dr. M. Klute
Second Reviewer:	Dr. J. Kieseler

Bearbeitungszeit: 01.05.2024 – 18.09.2024

Erklärung zur Selbstständigkeit

Ich versichere, dass ich diese Arbeit selbstständig verfasst habe und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, die wörtlich oder inhaltlich übernommenen Stellen als solche kenntlich gemacht und die Satzung des KIT zur Sicherung guter wissenschaftlicher Praxis in der gültigen Fassung vom 24.05.2018 beachtet habe.

Karlsruhe, den 18.09.2024, _____

Fabian Riemer

Als Prüfungsexemplar genehmigt von

Karlsruhe, den 18.09.2024, _____

Prof. Dr. M. Klute

Contents

List of Figures

1. Introduction

In order to further our understanding of particle physics, experiments at high energy and high precision are necessary. The required energies can be achieved by particle colliders, where complex detector systems measure the results of the collision. To test theories, it is often essential to know the properties of particles created at collider experiments. It is impossible to directly measure the quantities of interest. Instead one has to reconstruct the particles based on a multitude of signals in the detector systems. Naturally the best results can be achieved by combining the measurements of all subsystems together. This is commonly done by so-called particle-flow algorithms, which combine the signals of all detector subsystems and reconstruct the particles.

Modern particle physics experiments generate large amounts of data, which are difficult to analyse. Machine learning is a promising field in the extraction of relevant information from large datasets. By training a model to interpret the measured detector hits, one can predict the particles causing the detected showers and their properties. Such a reconstruction could complement or replace current particle-flow algorithms in future experiments. In order to properly train a model, one needs excellent simulations with the ground truth of the particle properties. After sufficient training a neural network is able to reconstruct the particles, which caused the measured showers, and their properties.

Different machine learning techniques have found application in high energy physics early on as described in Ref [1]. Machine learning methods already support detector subsystems in hit reconstruction [2] and track finding [3] or help to identify particles [4][5]. They are also used in the event selection, both as trigger [6] and in final analysis [7]. Even cutting-edge experiments like the Higgs boson discovery make use of machine learning to identify photons and flavour-tag jets [8].

Nevertheless it is important to properly test new machine learning approaches on simulated datasets first. This way the strengths and weaknesses of different methods can be found and the performances can be evaluated in detail, based on the plethora of information available in the simulation.

One promising approach to machine learning is the so-called object condensation approach [9], where the information is aggregated into multiple points by clustering the vertices that belong to the same object together.

Previous studies already investigated the performance of different machine learning algorithms on a dataset simulated by the COCOA-package [10][11]. The paper Ref [10] looked into three algorithms: a modified object condensation (OC) method, a transformer set prediction network with slot attention (TSPN-SA), and a novel hypergraph architecture (HGPflow). All the models were evaluated based on efficiency and fake rate, classification purity, particle angular and momentum resolution, and jet metrics. The comparisons favoured the newly introduced HGPflow algorithm. It even outperformed a traditional particle-flow method in the reconstruction on jet level. COCOA was developed for studies of machine learning algorithms and allows the simulation of a simplified calorimeter system.

While a modified object condensation algorithm was already part of the investigation, this thesis still focuses on a object condensation method, which is closer to the original object condensation paper. An existing pipeline is adapted and optimised for the training of a model on the COCOA-dataset. A new machine learning model is trained to reconstruct particles and their properties from the simulated detector signals and its performance is evaluated. By removing bad events from the training, the previous results are improved upon.

This thesis is structured as follows: Firstly there will be a short summary on the theoretical background in physics needed for this application in chapter 2, followed by an introduction into machine learning and an overview of GravNet and the object condensation method in chapter 3. Next in chapter 4 the dataset used for this thesis is described. Chapter 5 summarises the experimental investigations. Finally in chapter 6 the results will be evaluated and a conclusion is reached in the last chapter 7.

2. Physical Background

In the following sections a basic overview of the Standard Model and collider experiments based on Ref [12] [13] [14] is provided. Additionally the particle-flow algorithm at use at the CMS and ATLAS experiments are explained based on Ref [15][16] and the anti-k_T jet clustering algorithm [17] is introduced.

2.1. The Standard Model of Particle Physics

While there are still some phenomena left to be understood, the Standard Model is our currently best description of reality. The Standard Model is a quantum field theory and describes all known elementary particles as well as their interaction with all fundamental forces except gravity. An overview of the particles and their properties is shown in Figure 2.1. The particles are separated into fermions or bosons based on their spin.

The twelve fermions are divided into quarks and leptons. Only quarks carry a colour charge and therefore interact through the strong force. The six quarks are grouped into three generations, with each generation containing one particle with an electric charge of $\frac{2}{3}$ and one with an electric charge of $-\frac{1}{3}$. The mass of the particles increases in later generations. The particles are called up (u) and down (d) in the first generation, charm (c) and strange (s) in the second generation, and top (t) and bottom (b) in the third generation.

Of the six leptons three carry a negative unit electric charge: the electron (e^-), the muon (μ^-) and the tau (τ^-). The tau is the heaviest charged lepton, and the electron the lightest. The other three leptons are called neutrinos and only interact via the weak interaction since the electric and colour charges are both zero. They are each grouped with another charged lepton into a lepton family. Neutrinos are assumed to be massless in the Standard Model, although multiple observations like neutrino oscillations suggest that they have a small, non-zero mass, which could lead to interesting physics beyond the Standard Model [19].

All charged particles have an antiparticle with opposite physical charges. The antiparticle of the electron is called the positron (e^+) and has an positive electric charge.

The fundamental forces are described by the exchange of gauge-bosons. The massless gluon (g) can carry colour-charges and is the force-carrying particle of the strong interaction. The unified electro-weak-interaction is mediated by the photon, (γ), the Z-boson (Z^0) and the W-bosons ($W^\pm$). Of these three, only the photon is massless and only the W-boson carries a charge of plus or minus one.

Lastly the massive Higgs boson (H), which results from the mechanism, that gives particles mass, is a scalar spin zero particle. It was discovered in 2012 at the Large Hadron collider (LHC)[20].

Quarks can create colour neutral bound states called hadrons. A hadron consisting of a quark and an antiquark is a meson and a hadron made up of three (anti)quarks is a

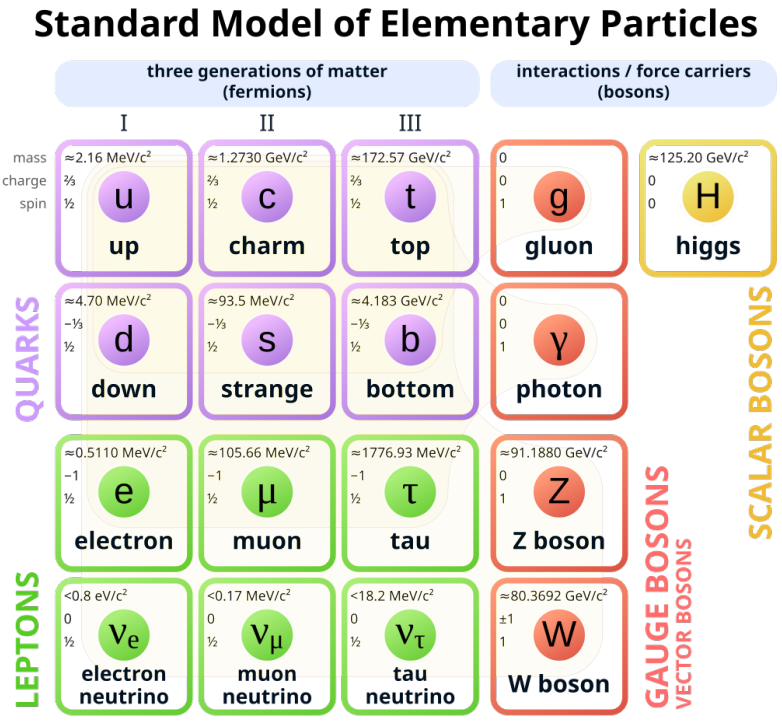


Figure 2.1.: Overview of the particles in the Standard Model. The mass, electric charge and spin of each particle is noted in the top left of each box. [18]

(anti)baryon. The lightest baryon is the proton, consisting of two up-quarks and one down-quark (uud), and the lightest meson is the positively charged pion (π^+) consisting of an up-quark and an anti-down-quark ($u\bar{d}$). Most particles can decay via the fundamental interactions into stable particles.

Although only the electron, the proton, the photon and the neutrinos are stable, some other particles with long decay times also exist long enough to interact with matter. The energy deposited in matter by a charged particle due to ionisation can be described via the Bethe-Bloch equation. Additionally very light charged particles like the electron lose most of their energy above a certain critical energy by emitting bremsstrahlung: the radiation of a photon in the electrostatic field of a nucleus. Since high energy photons can undergo pair production, and create an e^-e^+-pair, which in turn emit photons via bremsstrahlung, a cascade of electrons, positrons and photons is produced in these electromagnetic showers. Hadrons are too heavy to be significantly impacted by bremsstrahlung, but instead can undergo strong interaction with the nuclei of the material, resulting in highly variable hadronic showers with many possible complex end states.

2.2. Collider Experiments

Many recent physical breakthroughs were achieved by collider experiments at particle accelerators, since in order to probe the Standard Model and search for new physics, experiments with extremely high precision at high energies are necessary. Charged particles can be accelerated and shot at a stationary target or collided with another beam of accelerated particles. The advantage of a beam collider is that higher centre-of-mass energies can be achieved.

The particles which are produced by these collisions are measured by complex detector systems. Even new physics models, often have mechanism in which the novel particles decay

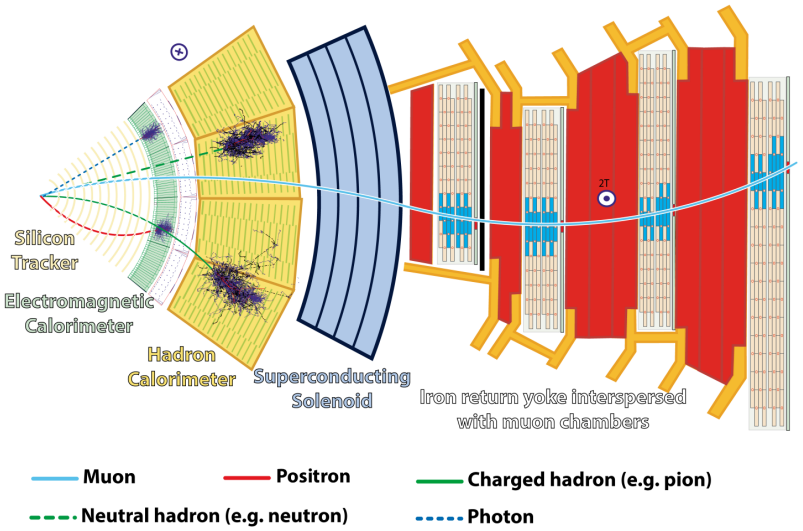


Figure 2.2.: Transverse slice of the CMS detector. The beam line is on the very left. It is surrounded by the tracker, the ECAL and the HCAL. The superconducting solenoid, which creates the strong inner magnetic field is drawn in the middle. Lastly the muon chambers on the outside of the detector are represented on the right. Additionally the paths of different particles are drawn on the detector slice. Adapted from Ref [21]

into known Standard Model particles, allowing a reconstruction based on the detection of traditional particles. The experiments consist of multiple subsystems, like tracking systems, electronic calorimeters (ECAL), hadronic calorimeters (HCAL) and muon chambers. A transverse slice of the CMS detector is shown in Figure 2.2 as an example. Since some interesting interactions are very rare, a high frequency of collisions is necessary. One beam crossing can create many interactions, leading to dense environments with many particles, which need to be identified and reconstructed to test new theories. In order to best utilise all the available information, one has to combine the measurements of all subsystems.

Interesting quantities include the energy and transverse momentum p_T of each particle. Often one also wants to know what type of particle is created by the collision. The position inside a collider detector is measured in angular coordinates by the azimuthal angle ϕ and the pseudorapidity η, which describes the angle relative to the beam axis.

Since initial particles created by a collision might decay quickly into secondary particles, particles travelling in the same direction are sometimes grouped together into jets. Depending on the experiment, one might be less interested in the metrics of the individual particles, which reach the detector, but instead might want to investigate the jet properties, to find out more about the initial particles.

The following sections quickly introduce the typical subsystems in a detector at a collider experiment.

2.2.1. Tracking

In order to measure the momenta of charged particles, the inner region of most particle detectors is filled with a strong magnetic field, which forces these particles onto a curved track. The track is commonly measured by layers of sensors with low material budget, for example silicon. When an energetic particle passes through a correctly doped and biased semiconductor, it creates electron-hole pairs and a small electric current can be measured. With multiple layers of high granularity detectors, one can connect the points where a particle was measured and approximate the curvature of the track. From this curvature, the momentum of the particle can be calculated. Gaseous detectors, in which the incoming

particles ionise a gas, leading to localised ions and electrons with can be measured, like multi-wire proportional chambers, micropattern gaseous detectors, or drift chambers can also be used to measure the tracks of a particles.

Some particles lose very little energy in matter, if they have the right energy. Since muons created at the LHC are such minimum ionising particles, and interact very little with matter, they are not stopped in the calorimeters. Therefore special muon detectors are used behind the calorimeters to still measure and identify the muons.

2.2.2. Calorimeter

In order to measure the energy of the particles in a collision, calorimeters are used. Since electromagnetic showers have a shorter lateral profile than hadronic showers a composite calorimeter is used in most experiments. It consists of an inner ECAL and an outer HCAL.

Calorimeters are commonly made from organic scintillators. The molecules in the scintillators are raised into excited states by the deposited energy and emit light, when they decay back to their ground-state. By measuring the intensity of the emitted light with photomultipliers, one can measure the deposited energy.

Another method to detect the energy is to build calorimeters with heavy liquid noble gasses. Incoming radiation can ionise the material, leading to a measurable electric current proportional to the incoming energy. The anode and cathode for measuring the current inside the detector function as absorber and can help to induce showers in the calorimeter.

ECALs induce electromagnetic showers and absorb the energy of electrons or photons. The stopping power is described by the radiation length X_0, which is defined as the mean penetration depth at which a high energy electron lost $\frac{1}{e}$ of its energy.

Since hadrons travel a relatively long distance between interactions, characterised by the nuclear interaction length λ_{int}, HCALs have to be large. In order to reduce the size, a heterogeneous structure is often used. The calorimeter consists of alternating layers of dense absorber material, in which showers are created, and active material, where the energies are sampled. While the energy resolution is naturally worse, it is often an effective measure to save space and money.

Calorimeters have to be large enough to contain the particle showers, common orders of magnitude are 25 X_0 for an ECAL and 11 λ_{int} for a HCAL. Space and cost efficient detectors use materials with short X_0 and λ_{int}. In a composite calorimeter, the ECAL should ideally have a high λ_{int}, so that most hadrons only interact in the HCAL. This can be achieved by materials with a high atomic number Z.

2.2.3. Particle Flow

The measurements of the individual detector subsystems have to be combined to reconstruct each particle. This can be done by so-called particle-flow algorithms.

In the first step of the current global particle-flow algorithm in use at the CMS experiment, which reconstructs all particles in an event simultaneously, physics objects are built from the subsystems. This includes tracks, energy clusters in the ECAL and HCAL, and muon signatures. Topological clusters are created by connecting neighbouring calorimeter cells with deposited energy higher than a certain threshold value.

Additionally, cells are selected as seeds, if the deposited energy is higher than all their neighbouring cells. The measured energy of each cluster has to be calibrated since not all of the energy of a particle can be measured. If there is just one seed in a cluster, all the measured energy in the cluster is assigned to it. If multiple seeds share the same cluster, the

energy of the other cells are distributed between all the seeds. In both cases the position is determined by energy-weighted averages in an iterative procedure.

Lastly the physics objects in close spatial proximity are linked together and the connected blocks are interpreted. Since muons have the cleanest signature tracks and energy clusters in close proximity to the track in the muon chambers are assigned to muons. Next isolated ECAL clusters are assigned to electrons or photons depending on whether they are linked to a track. Clusters in the HCAL are assigned to neutral and charged hadrons depending on the existence of a linked track.

Opposed to the described global algorithm, the particle-flow algorithm in use at ATLAS reconstructs the particles in multiple steps. First charged particles, which leave clear signatures in the tracker, are linked to close clusters in the calorimeters. The energy measured by the tracker is then subtracted from the calorimeters, leaving only the energy deposited by neutral particles. Now photons and neutral hadrons can be reconstructed from the energy left in the ECAL and HCAL.

There are many more particle-flow approaches for the reconstruction [22][23].

2.2.4. Anti-k_t jet clustering algorithm

Often it is interesting to cluster individual particles together and investigate the properties of the resulting jets. A popular jet clustering algorithm is the anti-k_t jet clustering algorithm. It clusters particles according to the following distances based on the transverse momentum p_T, the difference in position $\Delta R^2 = \Delta\eta^2 + \Delta\phi^2$ between particles and a parameter R_{param}, which controls the effective size of the created jet cones:

$$d_{ij} = \min(p_{Ti}^{-2}, p_{Tj}^{-2}) \frac{\Delta R_{ij}^2}{R_{\text{param}}} \tag{2.1}$$

$$d_{iB} = p_T^{-2} \tag{2.2}$$

After calculating all distances, the minimum distance is determined. If the smallest distance is d_{ij} between two particles, they are clustered into a single pseudo-jet. If the distance to the beam d_{iB} is the smallest, the particle i is declared as a jet and removed from the list. This process is repeated until there are no particles left.

Since particles with low p_T (soft), will cluster to particles with high p_T (hard), before hard particles are clustered among themselves, soft radiation does not modify the shape of the jet. Nevertheless the algorithm remains flexible to the introduction of hard radiation.

Since future experiments like the high luminosity LHC [24], will create large amounts of new data with huge pile-up, it is necessary to rethink the reconstruction approach. A promising approach to handle the new data is via machine learning.

3. Machine Learning

Machine Learning (ML) is a complex field, that has received a lot of attention in the last years. In the following sections an introduction into machine learning sourced from Ref [25] [26] is presented and an explanation of GravNet based on Ref [27] and the object condensation approach adapted from Ref [9] is given.

3.1. Basics of Machine Learning

Artificial neural networks are inspired by the way human brains work. The human brain consists of a lot of neurons, which are connected with synapses. The strength of these connections determines whether a neuron activates when its neighbours activate. Crucially the strength of the synaptic connections changes based on external stimuli, enabling the brain to learn an optimal configuration, to achieve certain tasks.

Similarly, artificial neural networks consist of interconnected layers, which compute an output based on their inputs, which can come from another layer, and a set of weights. By optimising the weights for each layer, complex functions can be modelled.

A multilayer neural network consists of an input layer, an output layer and a number of hidden layers in the middle. A common architecture is a multilayer perceptron (MLP), a fully connected feed-forward network, meaning that the whole output of a layer is connected to the input of the sequentially next layer. A representation of a fully connected neural network with one hidden layer is shown in Figure 3.1. The different inputs for the network are called features. Mathematically a fully connected feed-forward network can be represented as a combination of tensors. A single layer k consists of an activation function Φ^k and a set of weights represented by a matrix W^k. The input x^k, which is a vector, is transformed into the output vector y^k as follows

$$y^k = \Phi^k(W^k x^k) \tag{3.1}$$

The output is taken as the input for the next layer: $x^{k+1} = y^k$, until the information arrives at the output layer.

It is essential to add a non-linear activation function to the outputs of the layers, since otherwise the model could be reduced to a simple linear combination of the inputs regardless of the depth. Popular activation functions, which can be applied element wise, include the sigmoid, the hyperbolic tangents (tanh) and the rectified linear unit (ReLU). The ReLU, which has a sharp bent at zero, is sometimes replaced by the exponential linear unit (ELU), which slowly decreases for negative values. The sigmoid and hyperbolic tangents have the advantage, that the result will be constrained between zero or minus one and one respectively. This helps the network keep the weights contained. A similar benefit is also achieved by preprocessing the input to normalise all features.

Most machine learning goals can be formulated as a loss optimisation problem. The loss is a function, which takes the output of the model and assigns it a value of how good

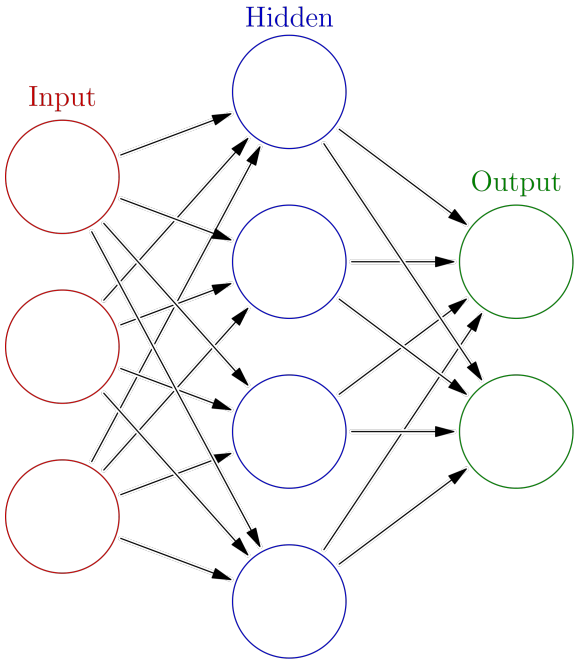


Figure 3.1.: Visual representation of a fully connected neural network consisting of an input layer, a single hidden layer and an output layer. [28]

it is. In a supervised learning, where the ideal outputs are known, this can simply be understood as the absolute difference between the prediction of the model and the true value of the sample. By computing the gradient of the loss with respect to the weights, and implementing a gradient descent algorithm, one can slowly change the weights of each layer to minimise the loss.

Since directly deriving an analytical expression of the gradient for each weight is impractical, backpropagation is used. Firstly in the forward phase, the input is passed though the network until an output is predicted. After calculating the loss, the gradient of the loss with respect to the weights of each layer is calculated numerically in the backward phase. It is called backwards phase, because the gradients are calculated in the reverse order compared to the information flow in the forward phase. By cleverly using the chain rule, one can significantly simplify the necessary calculations. Lastly the weights are adjusted in the negative direction of the gradient.

$$W^k_{\text{new}} = W^k_{\text{old}} - \alpha \frac{\partial L}{\partial W^k_{\text{old}}} \tag{3.2}$$

The learning rate, which controls the step-size of the optimisation, is denoted as α.

In order to iteratively improve the set of weights, one needs a large set of training data. After enough training, the model can predict the correct output for a similar dataset, which it has not seen before. Ideally the network is able to generalise to some extend and complete the task it was trained to do on inputs, which deviate slightly from the training.

A common task for a network is to classify the input as one of a number of classes. This can be achieved by predicting a number $\nu_1 \dots \nu_k$ for each class, which can be transformed into a probability $p_1 \dots p_k$, that the given input belongs to each class, via the softmax function.

$$p_i = \frac{e^{\nu_i}}{\sum_{j=1}^{k} e^{\nu_j}} \tag{3.3}$$

The true label can be one-hot encoded, meaning that y_i is one for the true class and zero for all other classes. Now a cross-entropy loss can be applied

$$L = -\sum_{i=1}^{k} y_i \log(p_i) = -\log(p_{\text{True}}). \tag{3.4}$$

3.2. Graph Neural Networks

An important subset of neural networks are graph neural networks (GNN), which work on information encoded as a graph. A graph consists of nodes, which are connected by edges. The strength of the connections between nodes can be qualified as weights of the edges in a weighted graph. Graphs can be represented via an adjacency matrix with a row and column for each node. In an unweighted graph, the entries in the adjacency matrix are either one, if the node belonging to the row is connected to the node belonging to the column, or zero otherwise. If this matrix is symmetric, meaning, that if node A is connected to node B, node B is also connected to node A, the graph is considered undirected.

An important feature of GNNs is that nodes can pass messages with their close neighbours. Each node can aggregate information of all the nodes it is connected to. For example each node can calculate the mean or maximum of a feature of its neighbours and integrate the result into the output of the layer. The advantage of GNNs is that sparse data of different sizes, like detector hits in a collider experiment, can be easily represented.

3.3. GravNet

GravNet [27] is a novel deep learning architecture using GNNs, which can achieve a high performance on irregular inputs, like a particle detector, without preprocessing the input, by learning a space representation for the input.

The initial input of the GravNet-layers is a $B \times V \times F_{\text{IN}}$ dataset consisting of a batch with B events, each with V detector hits, which all have F_{IN} features, like their position and measured energy. By using a dense neural network[1] two arrays are created from the input features F_{IN}: a set of coordinates S in the abstract space for each vertex and a learnable representation of the vertex features F_{LR}. Now the vertices are encoded as nodes in a GNN and nodes are connected with their closest N neighbours measured by the euclidean distance d_{jk} in the abstract space. For each vertex the features of its neighbours are weighted by a potential of the distance d_{jk} between the vertices.

$$\tilde{f}^i_{jk} = f^i_j \times \exp\left(-|d_{jk}|\right) \tag{3.5}$$

Now information can be aggregated for each vertex by applying a function, like the mean or the maximum, on the weighted features of the neighbours connected to the vertex. Due to the potential, vertices which are closer contribute more to the aggregated feature. This is the reason for the learnable space S, since vertices, which should exchange a lot of information can be moved close to each other. Finally the resulting features $\tilde{F}_{LR}$ are appended to the input features, fed through another dense layer and transformed into a new set of features for each vertex F_{OUT}, resulting in an output with the dimension $B \times V \times F_{\text{OUT}}$. An illustration of the steps performed by the GravNet layer is shown in Figure 3.2.

[1] A MLP which has the same weights for each V in all B

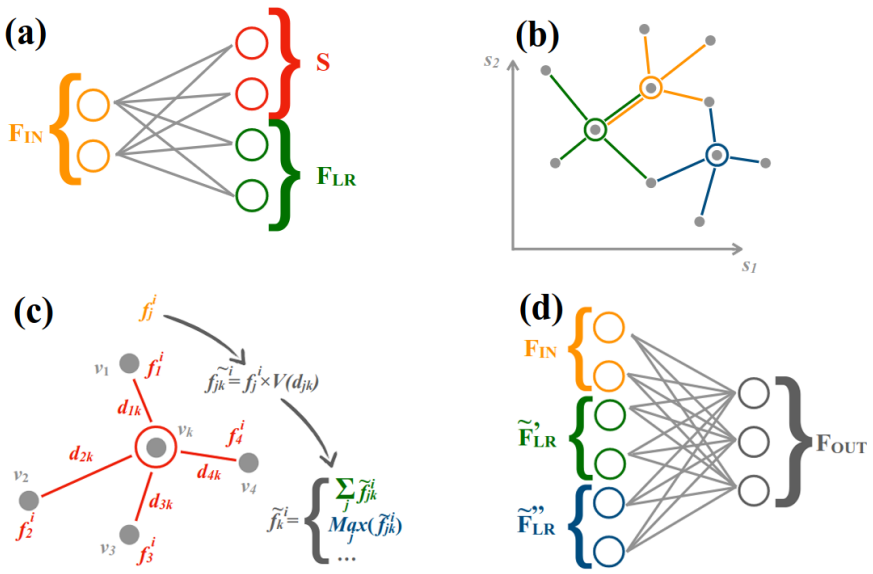


Figure 3.2.: Illustration of the data flow across the GravNet layer. (a) The input features F_{IN} are converted into a set of coordinates S in the clustering space and a learned set of features F_{LR}. (b) Using the position of each vertex according to S, a graph is built and each vertex is connected to its $N = 4$ closest neighbours. (c) For each node the f_j^i features of its neighbours are converted into the $\tilde{f}_{jk}^i$ by weighing them with a potential of the euclidean distance d_{jk}. These weighted features are gathered by the given vertex, resulting in the new features $\tilde{f}_k^i$ for each node for each choice of gathering function. (d) The resulting features $\tilde{F}_{\mathrm{LR}}$ are transformed with the original input features F_{IN} into the output features F_{OUT}. Adapted from Ref [27]

3.4. Object condensation

An innovative method for machine learning is the object condensation (OC) method [9]. For example, it can be used to reconstruct particles from detector signals or recognise objects in images. The basic idea of the object condensation method is to aggregate information about higher object into condensation points. Each individual vertex, a quantum of information like a pixel in an image or a cell in a calorimeter, is assigned to an object that should be reconstructed, like a particle or a shape in an image. The placement of vertices in an abstract clustering space can be learned with an appropriate definition of the loss, so that vertices belonging to the same object are connected, while vertices belonging to different objects are pushed apart. After sufficient training, the network is able to create a cluster and predict relevant properties for each object in a dataset.

For each vertex a scalar quantity β, which can be understood as a measure of the network's confidence, is predicted. Vertices with high β in sufficient isolation are taken as condensation points and the information aggregated into them is taken as the result of the model.

Loss Definition

In order to compute the loss during training, β_i is transformed into a charge q_i

$$q_i = \operatorname{arctan}^2 \beta_i + q_{\min} \tag{3.6}$$

which scales monotonously with β and contains the hyperparameter of a minimum charge $q_{\min}$. Higher values of $q_{\min}$ ensure, that all vertices belonging to a condensation point are clustered together.

An attractive ($\check{V}_k$) and repulsive ($\hat{V}_k$) potential in the abstract clustering space can be defined for each object k. Instead of computing the potentials coming from each vertex belonging to the object k, which would become very computationally expensive, one can approximate the potentials in terms of the distance $||x - x_\alpha||$ between a position x and the vertex α with the highest charge $q_{\alpha k}$ belonging to the object k at x_α:

$$\check{V}_k(x) = ||x - x_\alpha|| \cdot q_{\alpha k} \tag{3.7}$$

$$\hat{V}_k(x) = \max(0, 1 - ||x - x_\alpha||) \cdot q_{\alpha k} \tag{3.8}$$

An example of an effective potential affecting a vertex is shown in Figure 3.3.

In order to compute the loss one needs to know, which vertices i belong to which object k. This information is expressed in the matrix elements M_{ik}, which are one if the vertex i belongs to the object k and zero otherwise in this notation. Putting it all together one arrives at

$$L_{\mathrm{V}} = \frac{1}{N} \sum_{i=1}^{N} q_i \sum_{k=1}^{K} \Big(M_{ik} \check{V}_k(x_i) + (1 - M_{ik}) \hat{V}_k(x_i) \Big) \tag{3.9}$$

Due to the attractive potential $\check{V}_k$, points which belong to the same object are pulled together. The repulsive potential ensures that there are no points belonging to another object in the immediate proximity of one condensation point, influencing the prediction. The strength of the loss is proportional to both the charge of the vertex and the charge of the condensation point.

In order to avoid the trivial minimum of the loss by choosing very small β for all vertices, an additional loss L_β is introduced, to ensure a high charge of at least one condensation point for each object and small β for noise vertices.

$$L_\beta = \frac{1}{K} \sum_{k=1}^{K} (1 - \beta_{\alpha k}) + s_B \frac{1}{N_B} \sum_{i=1}^{N} n_i \beta_i \tag{3.10}$$

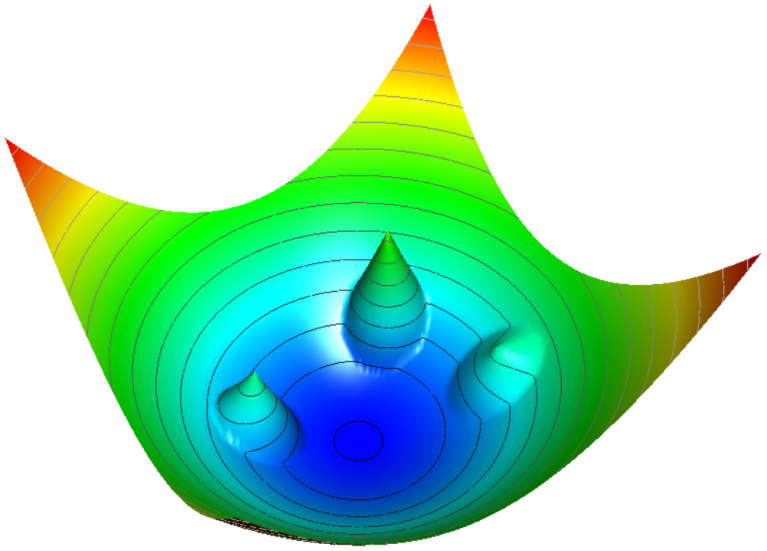


Figure 3.3.: Visualisation of the potential affecting a vertex belonging to an object, whose condensation point is located in the centre. Three condensation points belonging to different objects are located around the centre. [9]

In this notation, n_i is one for noise-vertices and zero otherwise. The strength of the background suppression can be controlled with the hyperparameter s_B.

Other loss terms should also be weighted by the charge of each vertex, so that the vertices where the network is the most sure contribute the strongest.

$$L_p = \frac{1}{\sum_{i=1}^{N} \xi_i} \cdot \sum_{i=1}^{N} L_i \xi_i \tag{3.11}$$

$$\xi_i = (1 - n_i)(q_i - q_{\min}) \tag{3.12}$$

Inference

In the last step of inference, vertices with a β_i higher than a threshold parameter t_β are considered as condensation points. A candidate is only taken as a condensation point if it is not within a distance t_d, controlled by a second parameter, of another candidate with a higher β. Next all vertices within t_d of a condensation point are assigned to it. The properties of the reconstructed object are taken from the prediction of the condensation point.

4. COCOA-dataset

The network presented in this thesis is trained and evaluated on a dataset generated by COCOA. In the following chapter COCOA is introduced and the provided dataset and the used detector geometry is described based on Ref [10] [11]. Lastly the input format for the neural network is described.

4.1. COCOA

The COnfigurable Calorimeter simulatiOn for Ai (COCOA) software package, developed for ML-based studies, allows the simulation of particle showers in a full-cover-calorimeter. Showers are generated by a PYTHIA8 Monte Carlo event generator and their interactions with a configurable detector are simulated in GEANT4.

A COCOA-dataset [29], which contains 60 649 simulated events of a single quark jet, was provided and is used in this investigation. It has the cell-wise detector information, like position and deposited energy, as well as truth particle information belonging to each cell and track. In addition to the training set a testing set with 38 922 single quark events is provided, as well as a testing set with 38 295 showers originating from a single gluon. The distributions of the dimensionality for the events can be seen in 4.1.

The simulated initial particles have an energy in the range $[10, 200]$ GeV and their angular coordinates are distributed uniformly in the ranges $\eta \in [-2.5, 2.5]$ and $\phi \in [-\pi, \pi]$.

4.2. Detector

The detector is designed to be similar to detectors in use at current collider experiments and is loosely inspired by the ATLAS calorimeter. The inner 150 cm of the detector have a strong uniform magnetic field, which is stopped by four 1.1 cm layers of iron. The detector is split up into a barrel for $|\eta| < 1.5$ and two endcaps for $1.5 < |\eta| < 3.0$ each consisting of six layers. The inner three layers simulate a homogeneous electromagnetic calorimeter (ECAL) with radiation length of $X_0 = 2.5\,\text{cm}$. The outer three layers simulate a hadronic calorimeter (HCAL) with an active and a passive material resulting in a nuclear interaction length of $\lambda_{\text{int}} = 26.6\,\text{cm}$ by only sampling some of the detected energy. Noise and smearing are added at random. Instead of a complicated tracker, the position of the truth particle at the beginning of the calorimeter is taken, and its true energy is smeared for an approximation of a momentum measurement.

4.3. Input format

The network is given B events in each batch as input. Each event consists of V vertices. A vertex can be a single detector cell hit or a track. It has the following F_{IN} features: position and deposited energy for hits, and position, charge, and measured momentum for tracks. In the implementation the vertex information from all event is concatenated

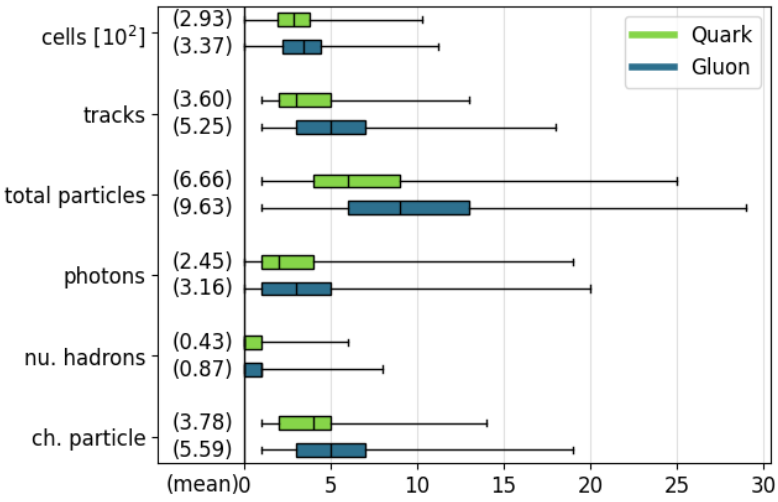


Figure 4.1.: Boxplots of the number of entries within one event for the quark test dataset (green) and the gluon test dataset (blue). The mean value is written on the left.

and can be separated by a rowsplits array, which contains the event boundaries. While the network only has access to the stated features, other truth information is stored for evaluation. This includes which particle made a given track or deposited the majority of the energy in a given cell encoded as a truth-index, and its properties like its initial energy or angular coordinates. After selecting a single event via the rowsplits, vertices belonging to the same particle can be grouped using the truth-index.

5. Experimental Investigations

5.1. Filtering the dataset

After encountering issues during training, an error was found in the provided dataset. The simulation wrongfully assigned detector hits to the same particle, although they are positioned on opposite sides of the detector. Such an event is displayed in Figure 5.1.

The authors of the dataset were informed and they were able to replicate the problem. No corrected dataset was provided in time for this thesis. Therefore studies were performed to identify events that are less affected by the issues for the training. By requiring that 90% of the energy deposited by a particle in the detector is within close proximity of the mean position of all detector hits of the particle, the problematic events could be removed. With a generous slice, which allows a difference in ϕ and η of 0.5, half of the dataset (29674 events) remained.

During the investigation, another problematic event was found. A single event in the training dataset contained a track at a position 1×10^{12} mm outside the detector. Since this track is definitely unphysical, the event was also removed.

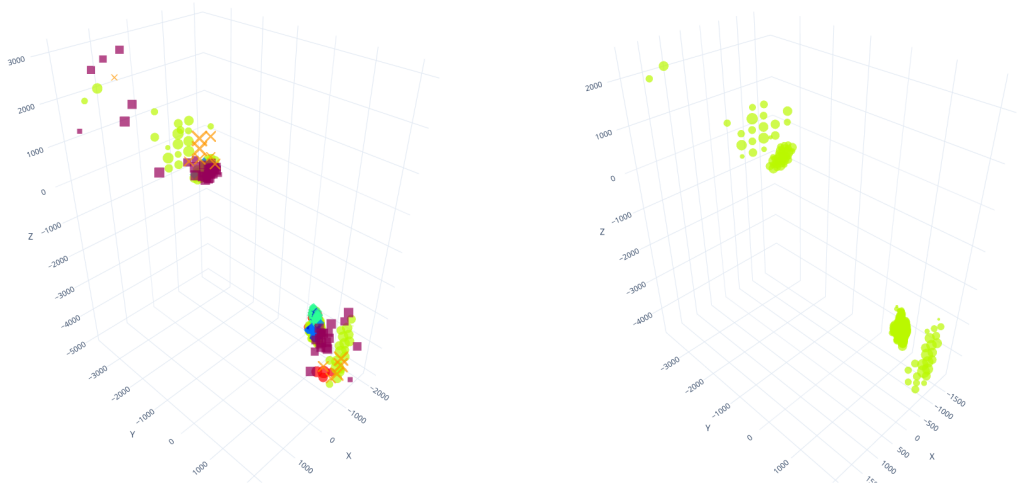
These cuts are only made on the training set; the testing set is not changed to ensure a fair comparison to Ref [10].

5.2. Technical Implementation

The training of the reconstruction model was implemented in Python using TensorFlow [30], Keras [31], the DeepJetCore framework [32] and the Adam optimiser [33]. A lot of the infrastructure already existed, because the project [34] developed for the analysis of the HGCAL could be utilised. The code of the final project can be found online [35].

A pictorial representation of the network architecture is shown in Figure 5.2. All dense layers have a dimension of 64 and are activated by the ELU activation. A small value for q_{min} of 0.1 is chosen, since here the focus is on predicting the objects themselves directly, and the association of individual hits to the object is secondary. The condensation space has three dimensions for easy visualisation. The training is performed for 1200 epochs with a decreasing learning rate in later epochs over three days.

Opposed to the modified object condensation approach in Ref [10], here the original object condensation approach is followed. This means, the values for β are trained in an unsupervised manner. Initially the β-values for tracks were forced to be one, which helped the network reconstruct charged particles, but further testing showed, that the network can achieve similar performance without this manual change to the algorithm.



(a) All particles in the event

(b) Only energy deposits by one particle

Figure 5.1.: Example of corrupted event. All detector hits are displayed at the position of the corresponding cell. The scale is proportional to the deposited energy. Points are colour and symbol coded based on the truth particle which deposited the most energy in the cell. Purple squares are cells where noise is the dominant contributor.

5.3. Reconstruction goals

The model is given the cell-wise information of position and deposited energy and the track information of position, momentum and charge and is tasked with reconstructing the truth particles of the shower. The loss function has three main components next to the clustering described in section 3.4: position-, classification- and energy-loss.

The position-loss rewards the network for correctly predicting the positional quantities ϕ and η of the truth particles, by adding a loss term proportional to the square of the difference in these quantities. Due to the modular nature of the angle ϕ, an encoding via the x- and y-coordinates on the unit circle corresponding to ϕ is chosen.

The goal of the classification-loss is to predict what type of particle the truth particle is. A simplified classification with only three classes is adopted. All particles are assigned to one of three groups: charged particles, neutral hadrons and photons. All points are assigned one of these three groups or classified as background noise. This classification is trained via a categorical cross-entropy loss. Since the number of particles in each category is unevenly distributed as seen in Figure 4.1, the classification loss is weighted by the frequency of each class.

Lastly the energy-loss penalises the model for not predicting the true energy of each particle. By applying a loss proportional to the square of the difference of the predicted and the true energy, the model learns to predict the true initial energy of the parent particle.

For these high energetic particles, the momentum is approximated as the energy and the transverse momentum p_T is calculated from the predicted energy E and pseudorapidity η

$$p_T = \frac{E}{\cosh \eta}. \tag{5.1}$$

5.4. Using the TOpAS cluster

Since the training of a machine-learning model requires a lot of computing power and a significant speed boost can be achieved by utilising a GPU, it was necessary to make use

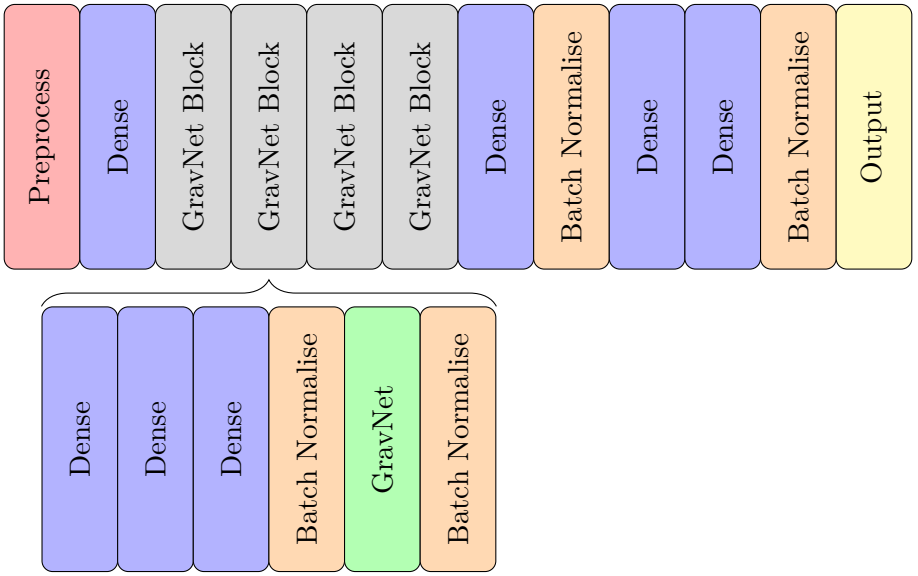


Figure 5.2.: Illustration of the network architecture. The network consists of a preprocess layer, which normalises all input features, followed by one dense layer and four blocks. Each block is made up of three dense layers, a batch normalisation layer, a GravNet layer and another batch normalisation layer. The last block is followed by another dense layer, a batch normalisation layer, further two dense layers, another batch normalisation layer and the output layer.

of the computing cluster TOpAS (Throughput Optimized Analysis System) at GridKa. For this purpose a script was created, that automatically submits a job to TOpAS.

By using the scheduling system HTCondor, the submission script was an important step in the training pipeline. It transfers all needed files to the cluster, sets up the required environment, and starts the training. The progress can be monitored with wandb [36]. After the execution, all created files are transferred back to the local machine for further analysis.

The submission pipeline is not limited to training scripts. All kinds of resource intensive jobs can be submitted to TOpAS.

5.5. GPU Optimisation

One of the problems during training was a long execution time for individual steps, resulting in a long and inefficient training. Upon a closer examination, the major problem for the delay was found in the loss layer. Multiple events are grouped into batches and the loss is calculated for each batch. The inefficiency arises from the fact that two matrices are calculated for each event in a batch sequentially. For the dataset of the HGCal, the loss layer was originally designed for, this is not a problem, since a single event has more than 100 000 hits and therefore each batch consists of only a few events, so the loop is only executed a few times. Since a single event in the COCOA-dataset contains significantly fewer hits as seen in Figure 4.1, one batch can contain many events and the loop was executed repeatedly. Even though one iteration is quite fast, the inherent inefficiency of loops in Python resulted in a long calculation time for the loss of a batch.

In order to combat this, it was decided to change the implementation of the CUDA-kernels, so that the matrices of multiple events can be calculated at the same time. The main bottleneck was the calculation of the attractive and repulsive potential.

Attractive Potential

For the attractive potential, one needs to know which hits belong to the same particle. This is expressed by a matrix with one row for each particle, containing the indices of all hits made by this particle. Since not all particles have the same number of hits associated, the empty fields are filled with negative one.

The repetition of the same index for a particle in a different event can be avoided by simply offsetting the truth-indices. Now, with unique indices for all particle the matrix can be determined via the same logic as before, but converting all events in a batch at once.

Repulsive Potential

In order to calculate the repulsive potential one needs to know all hits, which do not belong to a chosen particle. Before the changes this was encoded in a binary matrix, with a row for each particle containing a one, if the corresponding hit does not belong to the particle, and a zero, if it does.

To correctly adapt this information for a calculation of multiple events at the same time, one has to compare which event the chosen particle and the hit fall into, because despite the fact that a hit might be due to a different particle, other events should not influence the current event.[1]

Although the old encoding still worked after the changes, it becomes an inefficient representation for the information with a larger batch size, since most entries are zero, because the hits come from a different event than the particle. Furthermore, the dimension of the matrix is $K \times d$ with the total number of particles K. The problem is that the matrix dimension d grows with to the sum of the number of hits V_i of all events in a batch, which can lead to memory problems for large batch sizes.

$$d_{\text{old}} = \sum_{i=1}^{B} V_i \tag{5.2}$$

Therefore it was decided to change the encoding to the same approach as the matrix for the attractive potential. In the new definition, the indices of hits in the same event, but from a different source are stored in each row. Now the number of columns is limited by the maximum number of hits in a single event, instead of the sum of all hits in a batch.

$$d_{\text{new}} = \max(V) \tag{5.3}$$

While this number could theoretically be reduced by the minimal number of hits belonging to a particle in the largest event most of the time, evaluating the minimal dimension would be more resource intensive, than the current implementation.

Improvement

After implementing this logic, adapting the loss calculation to the new definition of the second matrix and comparing the results with the old implementation, an improvement of around one order of magnitude was achieved, depending the batchsize. Now the training of a model on the COCOA-dataset is ten times faster. The training time is therefore reduced from weeks to a few days.

[1] The same selection would have worked for the attractive potential, but it would be less efficient than the chosen offset-method.

5.6. Matching

In order to evaluate the quality of the reconstruction, the predicted particles need to be matched to true particles. In accordance with Ref [10] the predicted particles are matched to the true particles separately for charged and neutral particles, since they can be differentiated by the existence of a track. For charged particles only the difference in position $\Delta R^2 = \Delta\eta^2 + \Delta\phi^2$ is minimised. For neutral particles, the matching algorithm minimises both the difference in position as well as the difference in transverse momentum expressed by the following metric

$$d = \sqrt{\left(\frac{\Delta p_T}{p_{T\text{truth}}}\right)^2 + 5 \cdot \Delta R^2} \tag{5.4}$$

The inefficiencies that arise if the number of true and predicted particles do not match are discussed in section 6.2.1.

6. Evaluation

6.1. Condensation Spaces

A first quick check to see if the model is working as intended is to visualise the abstract condensation space by plotting the coordinates of each vertex in the abstract clustering space, or a projection of the coordinates if a higher dimension is chosen for the clustering space. Two examples of clustering spaces are provided in Figure 6.1.

Figure 6.1a shows an example of a good condensation space. The points are collected in discrete groups and in each group there are only points of the same colour, meaning, that the network is able to correctly separate the detector hits of each particle. Additionally the number of clusters matches the number of true particles in the event.

An example of a suboptimal condensation space is shown in Figure 6.1b. While the network is able to sort the vertices somewhat, one can see, that the majority of the points is positioned in a big cloud. There are no clear condensation points for all objects.

High values for q_{min} ensure, that all vertices are clustered close to their condensation points. For models with low q_{min} it is no problem, if vertices with low β are not positioned close to their condensation point, as long as one exists somewhere.

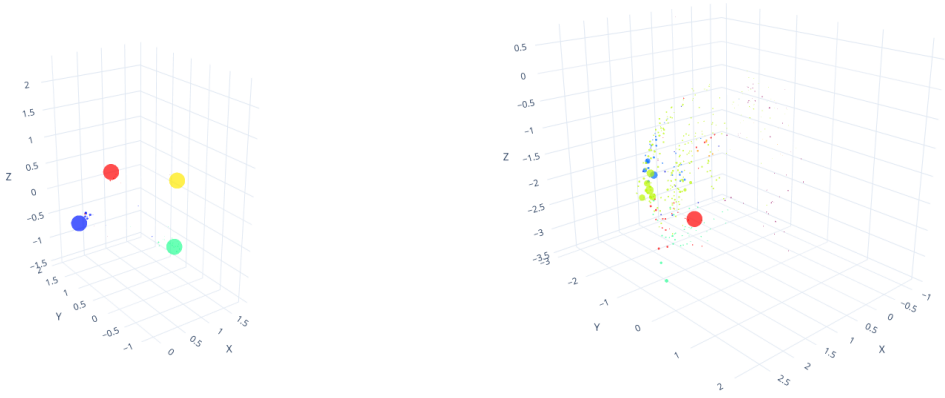
Although the visualisations have limited significance, they can provide insight into what the model is doing. By looking at the condensation spaces for a couple of events, one can quickly asses, whether the model is capable of clustering the input in a sensible manner or whether there is a problem.

6.2. Quark dataset

The evaluation is performed on an independent dataset from the training, by feeding the new dataset through the trained model and comparing the output with the truth information. The cuts used on the training set are not applied to the test set, to ensure a fair comparison with Ref [10].

The model is evaluated on reconstruction efficiency, classification correctness, momentum resolution, particle metrics and jet metrics. Additionally, the model will be evaluated on the provided gluon dataset described in section 4, to test the ability of the model to generalise beyond the training data.

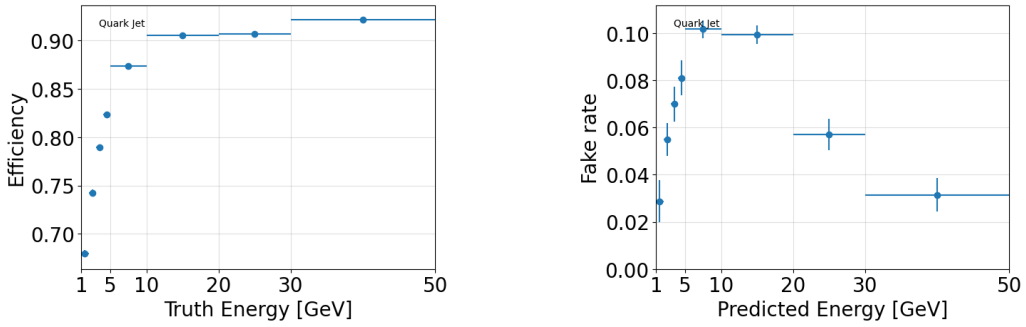
One has to be careful with drawing conclusions from these tests, since the provided dataset has some problems assigning detector hits to individual particles as discussed. Therefore the results, which are all calculated from the given truth information, have limited validity. The jet metrics are the least impacted, but the confusing truth assignment may still skew the results.



(a) Example of a good condensation space

(b) Example of a suboptimal condensation space

Figure 6.1.: Examples of condensation spaces. For one event all detector hits are plotted according to their predicted position in the abstract condensation space. The size of each point is scaled by the predicted β of each point. Points belonging to the same truth particle are coloured in the same colour.



(a) Reconstruction efficiency binned by the energy of the truth particle

(b) Reconstruction fake rate binned by the energy of the fake particle

Figure 6.2.: Efficiency and fake rate of the reconstruction binned by the energy of the truth particle

6.2.1. Efficiency

Important qualities of particle reconstruction are a high efficiency and a low fake rate. Efficiency is defined as the number of correctly reconstructed particles per true particle

$$\epsilon = \frac{N_{\text{Pred}}^{\text{matched}}}{N_{\text{Truth}}}. \tag{6.1}$$

So an efficiency of one would mean, that all particles are reconstructed. While it is trivial to only optimise efficiency, the challenge comes from also having a low fake rate. The fake rate is defined as the number of wrongly predicted particles per predicted particles

$$f = \frac{N_{\text{Pred}}^{\text{unmatched}}}{N_{\text{Pred}}}. \tag{6.2}$$

Here, a fake rate of 0 would mean that no particle was predicted that was not actually there.

As seen in Figure 6.2 the model achieves a high reconstruction efficiency of larger than 90% for particles with an energy higher than 10 GeV. As expected, the efficiency drops off for lower energetic particles, since they leave weaker signatures in the detector. Combined with the low fake rate over all energies, the results are satisfactory.

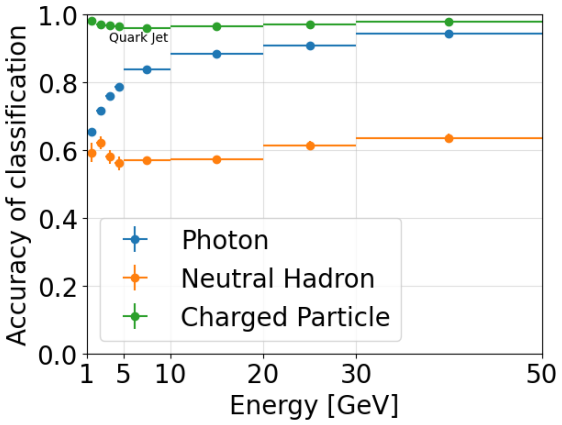


Figure 6.3.: Probability of correctly predicting the class of a particle binned by the energy of the true particle for photons (blue), neutral hadrons (orange) and charged particles (green)

6.2.2. Classification

Another difficult challenge of reconstruction algorithms is to differentiate neutral hadrons and photons, since low energetic neutral hadrons deposite most of their energy in the ECAL and leave similar signatures to photons.

Displayed in Figure 6.3 one can see the accuracy of the classification of particles, which are matched with a truth particle. Although one might assume, that charged particles can always be differentiated by the existence of a track, some charged particles in the dataset are missing tracks, leading to a small performance loss. Otherwise the charged particles are almost always classified correctly.

Due to the imbalance of photons compared to neutral hadrons in the dataset as seen in Figure 4.1, the network misclassified low energetic hadrons as photons, leading to a very bad performance for neutral hadrons at low energies. Weighing the classification loss by the frequency of the different particles led to a slightly worse accuracy for photons but an significant improvement in the accuracy for neutral hadrons. Nevertheless the model continues to achieve higher accuracy for photons but worse accuracy for neutral hadrons compared with the modified object condensation approach presented in Ref [10].

Overall the model showed, that it can differentiate between different particle types. Future work might try to classify the particles into finer groups and identify specific particles.

6.2.3. Momentum resolution

Lastly an important measure is the resolution of the momentum prediction, which is shown in Figure 6.4.

The network tends to predict momenta, which are too high, for particles with low energies and momenta, which are too low, for particles with high energies. This could be addressed by finding an energy dependent correction similar to correction performed for the jet p_T discussed in section 6.2.5. The momentum resolution is better for charged particles than for neutral particles, since they have a track. While the tracker might provide good estimates for the momentum of low energy charged particles, the importance diminishes for higher energies. Therefore it is important to also use calorimeter information for the best estimation of the momentum of charged particles, which the network is capable of doing.

Especially for low energetic neural particles, the relative error is quite high. A small absolute error can lead to a large relative error for low momenta. A reason might be,

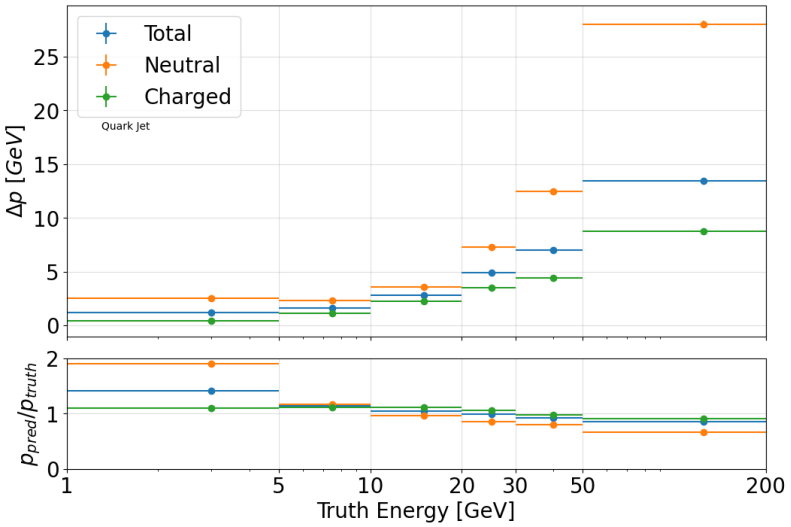


Figure 6.4.: Momentum resolution of the reconstructed particles binned by the momentum of the truth particle. The upper plot shows the absolute deviation in GeV and the lower plot shows the relative deviation from the correct value. The resolution is shown for all particles (blue), neutral particles (orange) and charged particles (green).

that the reconstruction algorithm is unable to separate multiple neutral showers in close proximity and joins them, leading to an underestimation of the total number of particles in an event and an momentum prediction, which is to high for the particle, it was matched to.

Overall the momentum resolution results for high energetic particles are satisfactory.

6.2.4. Particle Metrics

Ideally, the reconstruction should predict accurate values for the properties of each individual particle. Although the dataset does not have a good definition what a truth particle is, the difference in position, expressed via η and ϕ, p_T and particle energy is shown for reconstructed neutral particles in Figure 6.5.

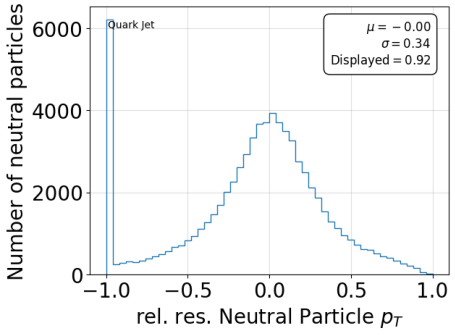
While the network is able to predict the correct initial energy and transverse momentum for most neutral particles to a satisfactory level, it struggles for some and tends to predict energies and momenta, which are significantly to high. This is mostly due to low energetic particles, where a small absolute deviation leads to a large relative error as discussed in section 6.2.3. Other outliers might be due to events with confusing particle definition, for which the evaluation has to be redone once the problems with COCOA are fixed.

The resolution of the angular coordinates for individual neutral particles is worse than the resolution presented in Ref [10] and a slight bias towards low η can be observed. The network is not able to combine the positions and energies of the hits of a particle into a precise prediction of the true coordinates of the particle.

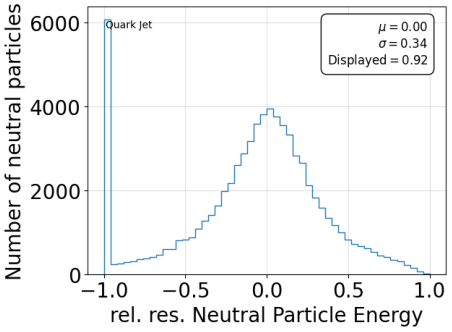
6.2.5. Jet Metrics

Since the provided dataset has some problems assigning detector hits to individual particles, the most important quantities of the reconstruction are jet level metrics. While the average position of the particles in the shower can somewhat easily be predicted from the detector hits, the most interesting reconstruction is of the jet energy or jet p_T.

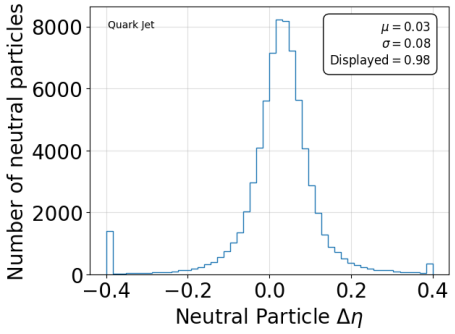
The transverse momentum p_T of the shower is a combination of the prediction of the energy of each particle combined with the prediction of the position of each particle. It is an interesting quantity in many real physics applications.



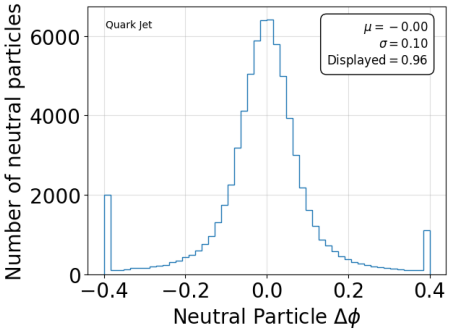
(a) Histogram of the relative residuum of p_T



(b) Histogram of the relative residuum of the energy



(c) Histogram of the residuum of η



(d) Histogram of the residuum of ϕ

Figure 6.5.: Evaluation of the prediction of particle metrics of neutral particles. Outliers are collected in the outermost bins and fraction of correctly displayed events is written in the top right with the optimal fit values for a normal distribution on the clipped data.

Jet Clustering

In order to calculate jet level metrics, the individual particles need to be combined into jets. At first the jets were clustered separately for the predicted and true particles with a anti-k$_t$ algorithm with a radius parameter of 0.4 and a minimum number of two constituents with the FastJet package [37]as described in Ref [10]. The predicted and true jets were matched by minimising the distance in ΔR^2.

Figure A.1 in appendix A shows the number of jet constituents from the clustering algorithm and the total number of particles per event based on the truth information of the quark test set. Comparing the results with the number of jet constituents for the truth jets presented in Ref [10], it seems that the evaluation of the jets metrics is performed under the assumption, that most events consist of a single jet. For a fair comparison, the jet metrics are calculated from all particles in a given event too, instead of applying the clustering algorithm, which increased the number of outliers. The jet p_T is taken as the sum of the predicted transverse momenta of all particles and the angular coordinates η and ϕ are calculated as an momentum weighted average from the predictions of all particles. The jet metrics based on the anti-k_T clustering algorithm can still be found in Figure A.2 in appendix A. The truth information is also calculated similarly as the sum or momentum weighted average of the individual truth particles.

The jet p_t-prediction of the network is further corrected in accordance with Ref [10]. A correction factor for the relative residuum of the jet transverse momentum is calculated in bins of the predicted p_t and a simple linear function is fitted to the results. Now the prediction can be corrected with the results of the fit. In the case of this thesis no significant response offset was found and the improvements from the correction are negligible.

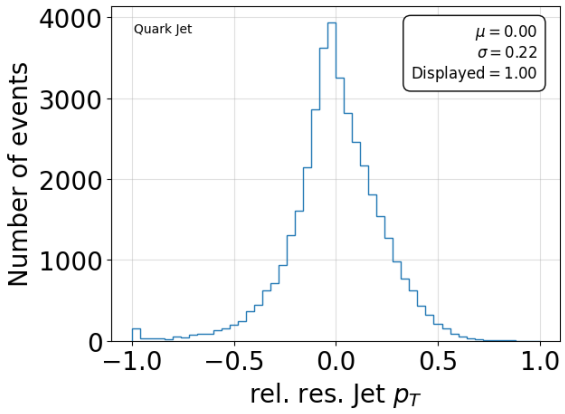


Figure 6.6.: Histogram of the relative residuum of p_T of the whole jet. Outliers are collected in the outermost bins and fraction of correctly displayed events is written in the top right with the optimal fit values for a normal distribution on the clipped data.

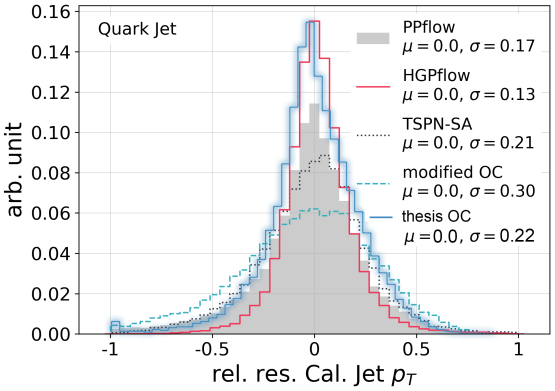


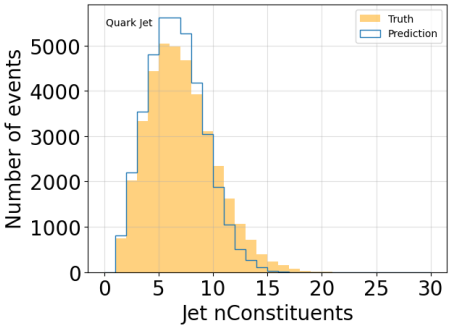
Figure 6.7.: Overlay of the histogram of the relative residuum of p_T of the whole jet from the object condensation model presented in this thesis (dark blue) onto the figure presented in Ref [10]

Results

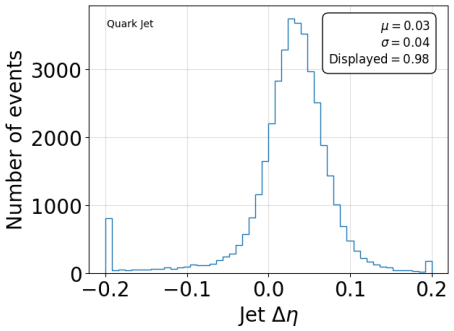
The relative residuum of jet p_T is displayed in Figure 6.6. A normal distribution fitted to the results has no offset of the mean from zero and a standard deviation of $\sigma = 0.22$. Compared with the results presented in Ref [10], the distribution is significantly narrower than the presented modified object condensation algorithm with $\sigma = 0.30$. For illustration the distribution in Figure 6.6 is overlaid into the results shown in Ref [10] in Figure 6.7 by matching the x-axis. It is unclear how the fit values for the presented algorithms in Ref [10] are calculated, since the model presented in this thesis appears to have a narrower distribution than the TSPN-SA model, but a larger fitted σ.

More insight can be gained by also discussing other jet-level quantities like the position or number of particles in the jet. As for the particle prediction, the resolution of the angular coordinates is still capable of improvement. This is expected, since the jet coordinates are calculated from the individual particles. Compared with the results presented in Ref [10], the distributions in Figure 6.8b and 6.8c are significantly wider. The network is not able to predict the position with a high accuracy. The bias for small η-values is still noticeable.

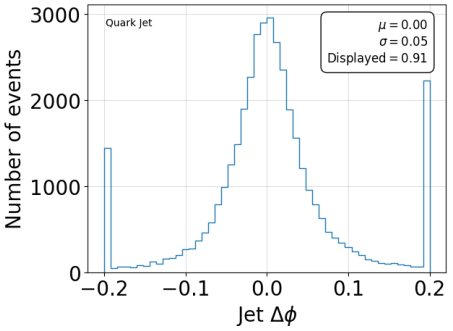
The model achieves a close match to the distribution of the number of particles in each event to the truth as shown in Figure 6.8a. Overall it tends to predict to few instead of too many particles in some showers. Some particles in dense events are not recognised. Nevertheless the close match to the truth information is satisfactory for the number of



(a) Comparison of the number of particles between the prediction and the truth



(b) Histogram of the residuum of average η in the whole jet



(c) Histogram of the residuum of average ϕ in the whole jet

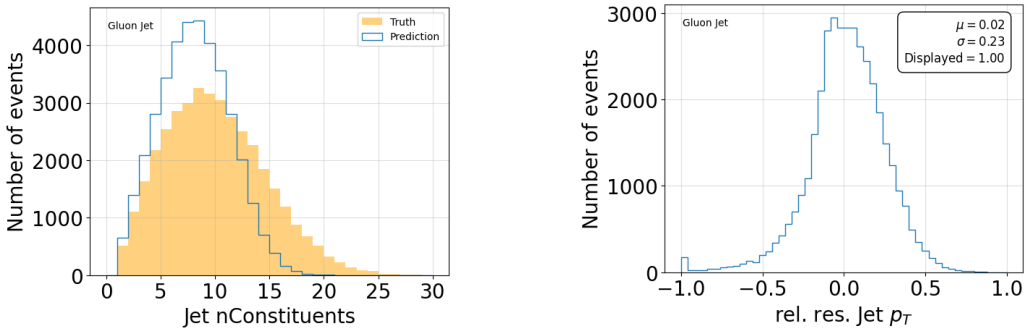
Figure 6.8.: Evaluation of the prediction of jet metrics. Outliers are collected in the outermost bins and fraction of correctly displayed events is written in the top right with the optimal fit values for a normal distribution on the clipped data.

particles in a jet. Especially considering the corrupted hit assignment for some events, the model achieves very good results. The TSPN-SA and the HGPFlow algorithms do not operate on individual cells, but on topoclusters. These algorithms predict the total number of particles in a first step and then match them directly to the truth particles to calculate the loss function. This approach comes with severe issues when trying to apply it to a larger fraction of the detector since it necessarily introduces cross-talk between detector regions. This makes a calibration in data, that often relies on a factorisation assumption between two well separated reconstructed particles, close to impossible. Furthermore, it introduces a significant source of bias from the distribution in the training sample. However, these algorithms can be trained without the cell-truth particle assignment that is broken in the current dataset. Therefore it can be that these algorithms were trained with more correct truth information, and as a result perform slightly better in the setting. It was not investigated, whether these topoclusters are also broken. Nevertheless the similarity to the truth information of the number of particles per jet of all machine learning algorithms is similar. The match to the number of jet is significantly closer than between the truth information and the traditional particle-flow algorithm presented in Ref [10].

The observation that the model tends to reconstruct too few rather than too many particles fits the low fake rate and suboptimal efficiency at lower energies discussed in section 6.2.1.

6.3. Gluon dataset

While an artificial neural network might perform well on data it has seen before, it is an important quality that it can generalise to some extent. In a particle physic application, not every interaction might be included in the training set, but the model should still perform well enough on events, which differ slightly from the training. For this reason,



(a) Comparison of the number of particles between the prediction and the truth

(b) Histogram of the relative residuum of p_T of the whole jet

Figure 6.9.: Results of the prediction of jet metrics for the gluon jets. Outliers are collected in the outermost bins and fraction of correctly displayed events is written in the top right with the optimal fit values for a normal distribution on the clipped data.

another testing set [29] is available, where the showers do not come from quarks, but from gluons with a similar energy. Below, the predicted jet metrics are shown for the evaluation on the gluon testing set.

The trend of not recognising every particle continues for the gluon dataset, but no performance loss for the prediction of jet p_T is observed. The model is capable to sufficiently generalise and predict the correct transverse momentum for differently structured showers. The other plots of the performance on the gluon dataset can be found in appendix B.

6.4. Summary

Overall, the model shows satisfactory performance, particularly in higher-energy regimes. It exhibits a good efficiency and fake rate, as well as improved jet p_T resolution. However, there is room for improvement in the model's sensitivity to low-energy signals and the resolution of the prediction for the angular coordinates of the jets. Until a new dataset is published, it is unclear what the true performance of the object condensation algorithm is and how it compares in detail to the other algorithms presented in Ref [10].

7. Conclusions

In the course of this thesis, an existing training pipeline was adapted for the COCOA-dataset. Due to the difference in the use case, it was necessary to change the implementation of some CUDA-kernels. Afterwards the training of a model was significantly faster. This in combination with the usage of TOpAS enabled the sufficient training of models in time for this thesis, leading to a model, which is able to reconstruct particle showers with a high precision.

It was shown that the object condensation approach to machine learning is able to produce good reconstruction results on the provided dataset. Some of the results presented in Ref [10] were improved upon. Especially the reconstruction of jet p_T showed significant improvement over the modified object condensation algorithm. In the high energy regime the model showed satisfactory efficiencies and energy resolution.

This work uncovered some problems of the current COCOA-dataset. Some events have an unphysical assignment of hits to particles. Nevertheless it still provides a baseline to compare different machine learning algorithms. New innovative algorithms can be tested and compared with each other, since the dataset finds a good balance between being complex enough to be relevant for real applications while being simple enough so that models accurately depict their performance after a manageable computational expense.

If a fixed COCOA-dataset is published and a new model is trained, an improvement in the performance is expected. Until the dataset is updated, the validity of the evaluation remains uncertain.

Future work might apply the object condensation algorithm to more complex datasets to investigate the capability to reconstruct more complex detector environments. With simulations that closer mimic real detectors, the ability of neural networks to find complex connections between inputs might lead to further improvement upon traditional reconstruction approaches. If further tests on simulated data continue to provide results as good or better than the reconstruction algorithms in use, the object condensation approach might be used for reconstruction in real particle physics experiments.

Appendix

A. Jet Definition

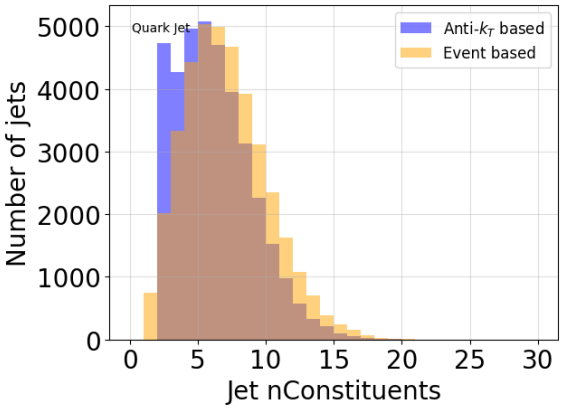
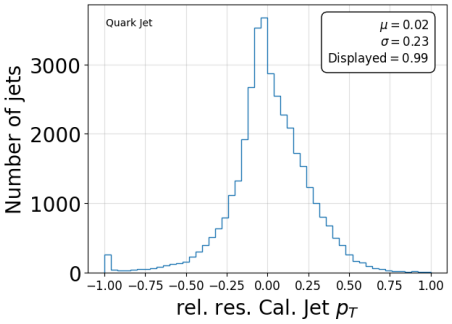
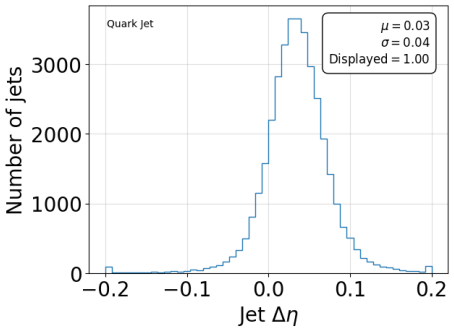


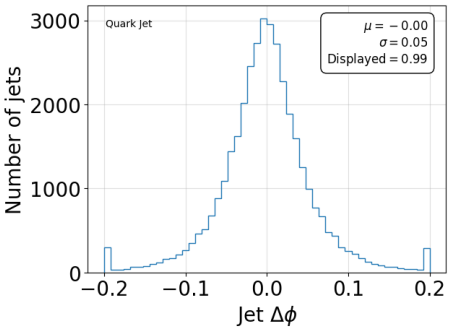
Figure A.1.: Comparison between the number of jet constituents. The results of the anti-k_T algorithm are displayed in blue. The total number of particles per event are displayed in yellow.



(a) Histogram of the relative residuum of p_T of the clustered jet.



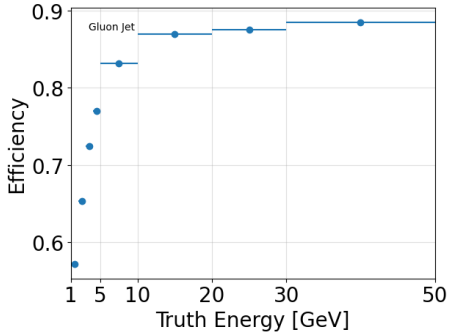
(b) Histogram of the residuum of η in the clustered jets



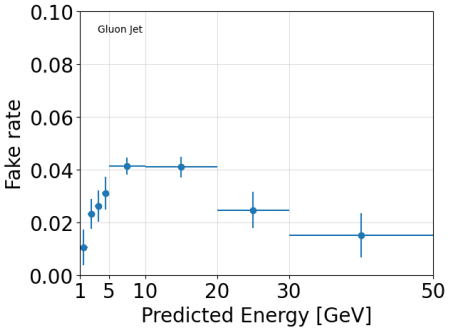
(c) Histogram of the residuum of ϕ in the clustered jets

Figure A.2.: Evaluation of the prediction of clustered jet metrics. Outliers are collected in the outermost bins and fraction of correctly displayed events is written in the top right with the optimal fit values for a normal distribution on the clipped data.

B. Evaluation on the gluon dataset



(a) Reconstruction efficiency binned by the energy of the truth particle



(b) Reconstruction fake rate binned by the energy of the fake particle

Figure B.3.: Efficiency and fake rate of the reconstruction binned by the energy of the truth particle for the gluon dataset

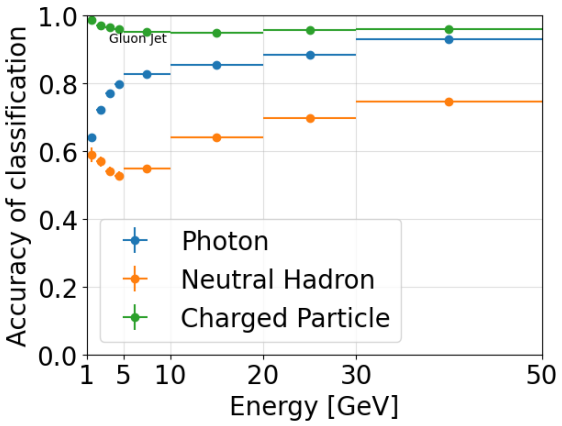


Figure B.4.: Probability of correctly predicting the class of a particle binned by the energy of the true particle for photons (blue), neutral hadrons (orange) and charged particles (green) for the gluon dataset

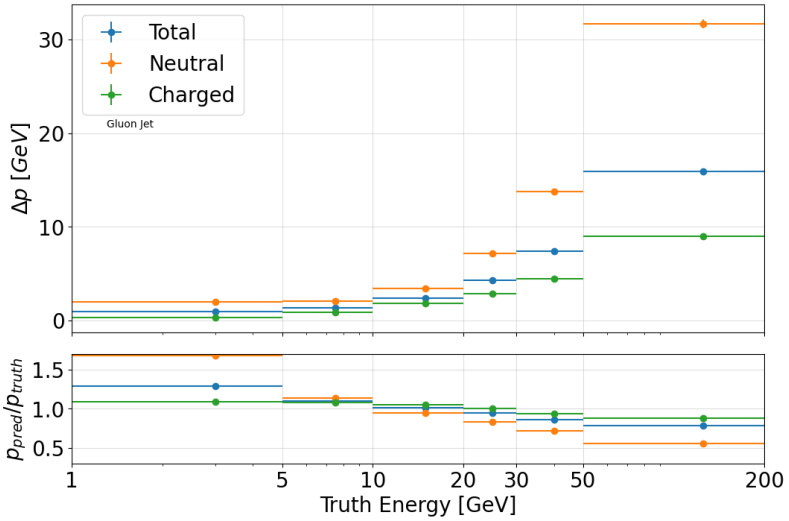
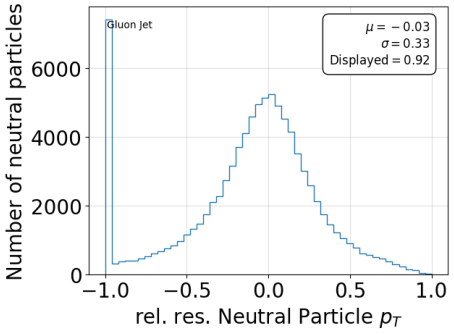
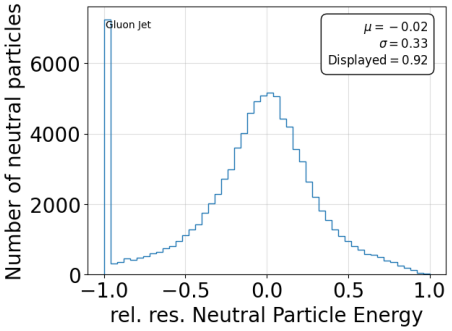


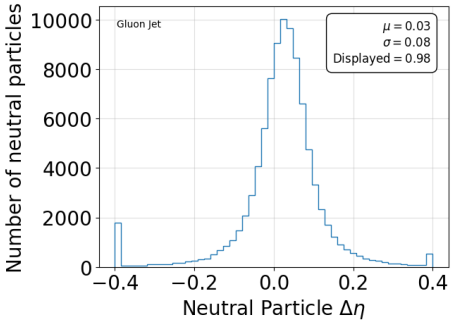
Figure B.5.: Momentum resolution of the reconstructed particles binned by the energy of the truth particle for the gluon dataset. The upper plot shows the absolute deviation in GeV and the lower plot shows the relative deviation from the correct value. The resolution is shown for all particles (blue), neutral particles (orange) and charged particles (green).



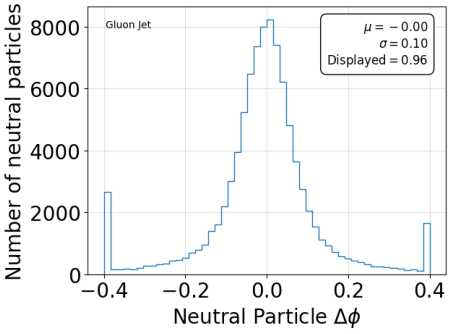
(a) Histogram of the relative residuum of p_T



(b) Histogram of the relative residuum of the energy

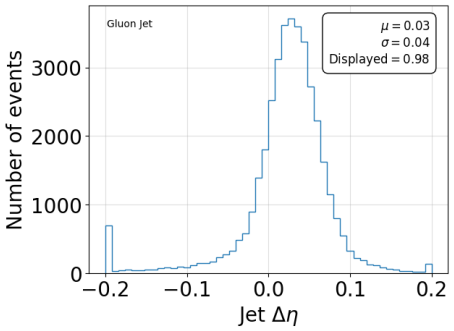


(c) Histogram of the residuum of η

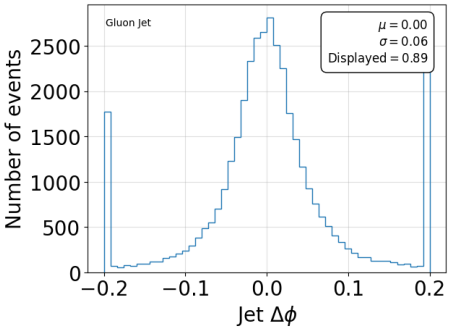


(d) Histogram of the residuum of ϕ

Figure B.6.: Evaluation of the prediction of particle metrics of neutral particles for the gluon dataset. Outliers are collected in the outermost bins and fraction of correctly displayed events is written in the top right with the optimal fit values for a normal distribution on the clipped data.



(a) Histogram of the residuum of average η in the whole jet



(b) Histogram of the residuum of average ϕ in the whole jet

Figure B.7.: Evaluation of the prediction of jet metrics for the gluon dataset. Outliers are collected in the outermost bins and fraction of correctly displayed events is written in the top right with the optimal fit values for a normal distribution on the clipped data.

Bibliography

[1] Dan Guest, Kyle Cranmer, and Daniel Whiteson. Deep learning and its application to lhc physics. *Annual Review of Nuclear and Particle Science*, 68(1):161–181, October 2018.

[2] The ATLAS collaboration. A neural network clustering algorithm for the atlas silicon pixel detector. *Journal of Instrumentation*, 9(09):P09009, sep 2014.

[3] Carsten Peterson. Track finding with neural networks. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 279(3):537–545, 1989.

[4] Halina Abramowicz, Allen Caldwell, and Ralph Sinkus. Neural network based electron identification in the zeus calorimeter. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 365(2–3):508–517, November 1995.

[5] V. M. et al. Abazov. Measurement of $\sigma(p\overline{p} \to z) \cdot \mathrm{Br}(z \to \tau\tau)$ at $\sqrt{s} = 1.96$ TeV. *Phys. Rev. D*, 71:072004, Apr 2005.

[6] J.K Köhne et al. Realization of a second level neural network trigger for the h1 experiment at hera. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 389(1):128–133, 1997. New Computing Techniques in Physics Research V.

[7] T Behnke and D G Charlton. Electroweak measurements using heavy quarks at lep. *Physica Scripta*, 52(2):133, aug 1995.

[8] ATLAS 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):1–29, September 2012.

[9] Jan Kieseler. Object condensation: one-stage grid-free multi-object reconstruction in physics detectors, graph, and image data. *The European Physical Journal C*, 80(9), September 2020.

[10] Francesco Armando Di Bello, Etienne Dreyer, Sanmay Ganguly, Eilam Gross, Lukas Heinrich, Anna Ivina, Marumi Kado, Nilotpal Kakati, Lorenzo Santi, Jonathan Shlomi, and Matteo Tusoni. Reconstructing particles in jets using set transformer and hypergraph prediction networks. *The European Physical Journal C*, 83(7), July 2023.

[11] Francesco Armando Di Bello, Anton Charkin-Gorbulin, Kyle Cranmer, Etienne Dreyer, Sanmay Ganguly, Eilam Gross, Lukas Heinrich, Lorenzo Santi, Marumi Kado, Nilotpal Kakati, Patrick Rieck, and Matteo Tusoni. Configurable calorimeter simulation for ai applications, 2023.

[12] M. Thomson. *Modern Particle Physics*. Cambridge University Press, 2013.

[13] William Frass. *Particle Detectors*. Oxford: Department of Physics at Oxford University, 2009.

[14] Hermann Kolanoski and Norbert Wermes. *Teilchendetektoren: Grundlagen und Anwendungen*. Springer, 2016.

[15] CMS Collaboration. Particle-flow reconstruction and global event description with the cms detector. *Journal of Instrumentation*, 12(10):P10003–P10003, October 2017.

[16] ATLAS Collaboration. Jet reconstruction and performance using particle flow with the atlas detector. *The European Physical Journal C*, 77(7), July 2017.

[17] Matteo Cacciari, Gavin P Salam, and Gregory Soyez. The anti-ktjet clustering algorithm. *Journal of High Energy Physics*, 2008(04):063–063, April 2008.

[18] Wikimedia Commons. File:standard model of elementary particles.svg — wikimedia commons, the free media repository, 2024. [Online; accessed 6-August-2024].

[19] G. Bellini, L. Ludhova, G. Ranucci, and F. L. Villante. Neutrino oscillations. *Advances in High Energy Physics*, 2014:1–28, 2014.

[20] Jana Faltova. Standard Model Higgs results from ATLAS and CMS experiments. Technical report, CERN, Geneva, 2016.

[21] David Barney. CMS Detector Slice. CMS Collection., 2016.

[22] Manqi Ruan. Arbor, a new approach of the particle flow algorithm, 2014.

[23] M.A. Thomson. Particle flow calorimetry and the pandorapfa algorithm. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 611(1):25–40, 2009.

[24] G Apollinari, O Brüning, T Nakamoto, and L Rossi. High luminosity large hadron collider hl-lhc, 2015.

[25] Charu C. Aggarwal. *Neural Networks and Deep Learning: A Textbook*. Springer Publishing Company, Incorporated, 2 edition, 2023.

[26] Bharti Khemani, Shruti Patil, Ketan Kotecha, and Sudeep Tanwar. A review of graph neural networks: concepts, architectures, techniques, challenges, datasets, applications, and future directions. *Journal of Big Data*, 11(1):18, Jan 2024.

[27] Shah Rukh Qasim, Jan Kieseler, Yutaro Iiyama, and Maurizio Pierini. Learning representations of irregular particle-detector geometry with distance-weighted graph networks. *The European Physical Journal C*, 79(7), July 2019.

[28] Wikimedia Commons. File:colored neural network.svg — wikimedia commons, the free media repository, 2024. [Online; accessed 3-September-2024].

[29] Francesco Armando Di Bello, Etienne Dreyer, Sanjay Ganguly, Eilam Gross, Lukas Heinrich, Anna Ivina, Marumi Kado, Nilotpal Kakati, Lorenzo Santi, Jonathan Shlomi, and Matteo Tusoni. Single-jet datasets for particle reconstruction with deep learning, March 2023.

[30] TensorFlow Developers. Tensorflow, July 2024.

[31] Francois Chollet et al. Keras, 2015.

[32] Jan Kieseler, Markus Stoye, Mauro Verzetti, Pedro Silva, Swapneel Sundeep MEHTA, Anna Stakia, Yutaro IIYAMA, Emil Bols, Shah Rukh QASIM, Henning KIRSCHENMANN, Huilin Qu, Marcel Rieger, and Loukas Gouskos. Deepjetcore, February 2020.

[33] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[34] cms pepr. HGCalML. https://github.com/cms-pepr/HGCalML, 2024.

[35] Fabian Riemer. HGCalML. https://github.com/Fantastic-Fabian/HGCalML, 2024.

[36] Lukas Biewald. Experiment tracking with weights and biases, 2020. Software available from wandb.com.

[37] Matteo Cacciari, Gavin P. Salam, and Gregory Soyez. Fastjet user manual: (for version 3.0.2). *The European Physical Journal C*, 72(3), March 2012.