Tau Reconstruction at FCC-ee using Machine Learning based on Full Detector Simulation

Bachelor's Thesis of

Gregor Brodbek

At the KIT Department of Physics
Institute of Experimental Particle Physics

First examiner:	Prof. Dr. Markus Klute
Second examiner:	Dr. Roger Wolf
Advisor:	Dr. Xunwu Zuo

01. September 2024 – 26. February 2025

Karlsruher Institut für Technologie
Institut für Experimentelle Teilchenphysik
Postfach 6980
76128 Karlsruhe

Tau Reconstruction at FCC-ee using Machine Learning based on Full Detector Simulation (Bachelor's Thesis)

I declare that I have developed and written the enclosed thesis completely by myself. I have not used any other than the aids that I have mentioned. I have marked all parts of the thesis that I have included from referenced literature, either in their original wording or paraphrasing their contents. I have followed the by-laws to implement scientific integrity at KIT.

Karlsruhe, 26. February 2025

. . . . . .

(Gregor Brodbek)

Abstract

Tau lepton reconstruction and identification is a key challenge in collider experiments due to their extremely short lifetimes, causing them to decay before reaching the detectors. Identifying tau leptons traditionally involves reconstructing their well studied decay modes using physics motivated variables and particle flow algorithms. This thesis implements a machine learning algorithm based on the Geometric Algebra Transformer architecture, which processes the low-level detector signals directly to identify tau decay modes without any prior reconstruction. The training dataset is produced through full simulation of electron positron collisions in the proposed Future Circular Collider, generating $Z \rightarrow \tau^+\tau^-$ events that are detected by a detector modeled after the CLIC Like Detector concept.

The results demonstrate promising performances in identifying tau decay modes, especially in leptonic decays and background processes. While it achieves lower efficiencies in the distinction of similar hadronic decays, it can compete with and even outperform efficiencies of explicit reconstruction approaches. This showcases the potential of machine learning approaches in tau identification.

Zusammenfassung

Die Rekonstruktion und Identifizierung von Tau Leptonen stellt eine zentrale Herausforderung in Teilchenbeschleunigerexperimenten dar, da diese aufgrund ihrer kurzen Lebensdauer bereits vor Erreichen der Detektoren zerfallen. Herkömmliche Rekonstruktionsansätze basieren auf physikalisch motivierten Variablen und particle flow Algorithmen, um die Tau Zerfälle anhand ihrer bekannten Zerfallskanäle zu identifizieren. In dieser Arbeit wird ein machine learning Algorithmus basierend auf dem Geometric Algebra Transformer Modell direkt mit den Rohdaten des Detektors trainiert, um Tau Zerfälle zu identifizieren. Der Trainingsdatensatz besteht aus $Z \to \tau^+\tau^-$ Ereignissen, die durch Full Simulation von Elektron-Positron Kollisionen im geplanten Future Circular Collider erzeugt und von einem nach dem CLIC Like Detector Konzept modellierten Detektor erfasst werden.

Die Ergebnisse dieses Ansatzes zeigen vielversprechende Leistungen bei der Identifikation von Tau Zerfällen, insbesondere bei den leptonischen Zerfällen und den Hintergrundprozessen. Obwohl das Modell bei den schwieriger zu unterscheidenden hadronischen Zerfällen niedrigere Effizienzen erreicht, erzielt es auch hier Resultate, die mit expliziten Rekonstruktionsansätzen mithalten können und diese zum Teil sogar übertreffen. Diese Ergebnisse zeigen das Potential von ML Ansätzen bei der Tau Identifizierung.

Contents

List of Figures

Abbreviations

AUC	area under the curve
BSM	beyond the Standard Model
CERN	European Organization for Nuclear Research, *Conseil Européen pour la Recherche Nucléaire*
CIEMAT	Centre for Energy, Environmental and Technological Research, *Centro de Investigaciones Energéticas, Medioambientales y Tecnológicas*
CLD	CLIC Like Detector
CLIC	Compact Linear Collider
CNN	convolutional neural network
ECAL	electromagnetic calorimeter
EDM4HEP	Event Data Model for High Energy Physics
ETP	Institute for Experimental Particle Physics at KIT, *Institut für Experimentelle Teilchenphysik*
FCC	Future Circular Collider
FN	false negative
FP	false positive
FPR	false positive rate
GATr	Geometric Algebra Transformer
GELU	gaussian error linear unit
GNN	graph neural network
GridKa	Grid Computing Centre Karlsruhe
HCAL	hadronic calorimeter
LEP	Large Electron-Positron Collider
LHC	Large Hadron Collider
LSTM	long short-term memory
MC	Monte Carlo
ML	machine learning

MLP	multilayer perceptron
NN	neural network
PF	particle flow
ReLU	rectified linear unit
RNN	recurrent neural network
ROC	receiver operating characteristic
SM	Standard Model
TN	true negative
TOpAS	Throughput Optimized Analysis System
TP	true positive
TPR	true positive Rate

1. Introduction

Particle physics aims to explain the fundamental particles and interactions that make up matter. High-energy colliders like the Large Hadron Collider (LHC) at the European Organization for Nuclear Research (CERN) produce these particles in a controlled environment, enabling precise studies of their properties. The Future Circular Collider (FCC), a proposed collider that is currently being studied, will provide vast new data samples to improve measurements and potentially discover new physics [1].

In its initial phase, the proposed FCC-ee will operate as a high-precision electroweak factory, producing large samples of Z, W and Higgs bosons. With an estimated 6×10^{12} Z bosons producing about 2×10^{11} $Z \rightarrow \tau^+\tau^-$ events, the FCC can provide precise measurements of tau properties, including mass, lifetime and branching ratios [2]. Additionally, it will offer opportunities for precise polarization reconstruction, allowing for detailed studies of spin related properties.

In the last decades machine learning (ML) has been established in high energy physics in different forms. For tasks including track reconstruction, jet tagging or event classification, different ML techniques have been successfully employed [3].

Due to the short lifetime of tau leptons, they cannot be measured directly in particle detectors. Traditionally, the tau leptons are reconstructed by identifying their well known decay products, which are reconstructed from detector signatures using algorithms like the particle flow (PF). This thesis implements a machine learning (ML) based approach that directly utilizes the detector signatures to identify tau decay modes, bypassing the PF reconstruction. To achieve this, the Geometric Algebra Transformer (GATr) architecture is employed, which effectively represents geometric data and uses the versatility and efficiency, that are provided by Transformers [4].

The dataset that is generated to train the ML model consists of $Z \rightarrow \tau^+\tau^-$ samples, including the main tau decay modes (>1% branching ratio), and two background processes in the form of Bhabha scattering and $Z \rightarrow q\bar{q}$ decays, where q represents a quark from the set (u, d, s, c, b). The data is produced by full simulation of the electron-positron beam collisions in the FCC-ee at Z resonance ($\sqrt{s} = 91$ GeV) and the interaction in a detector modeled after the CLIC Like Detector (CLD) concept.

The results are discussed and evaluated based on different metrics, such as confusion matrices and receiver operating characteristic (ROC) curves, showing that the ML approach achieves promising results. The performance is validated by a comparison to an explicit reconstruction.

The thesis is structured as follows: Chapter 2 gives an overview of the relevant physics concepts, which is followed by a theoretical overview of machine learning and the application of different architectures in particle physics in Chapter 3. Chapter 4 highlights the importance of simulation data in physics and explains the simulation workflow in more detail. Chapter 5 presents and discusses the training process and results, followed by a conclusion in Chapter 6.

2. Physics Background

2.1. Standard Model of Particle Physics

The Standard Model (SM) is a widely accepted and proven theory of particle physics, successfully explaining the majority of experimental data. It describes all elementary particles and most of their interactions, including the fundamental electromagnetic, weak and strong forces. The following will give a short overview over the SM based on Ref. [6].

The SM consists of 17 fundamental particles, that are shown in Figure 2.1. They are categorized by their spin into two groups, the fermions with half-integer spins and the bosons with integer spins. Fermions are considered the building blocks of matter, consisting of three generations of up-type quarks, down-type quarks, charged leptons, and neutral leptons. While the neutral leptons, also called neutrinos, are massless in the SM, the other fermions increase in mass with later generations. Bosons, on the other hand, are the mediators for the interactions between particles. The massless photon (γ) and gluon (g)

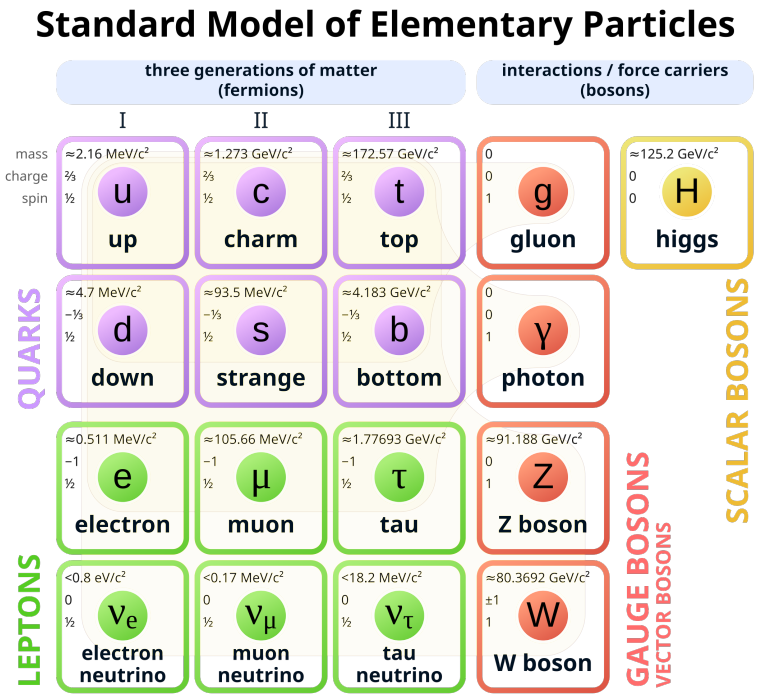


Figure 2.1.: Overview of the Standard Model of particle physics [5]

are the mediators for the electromagnetic and strong force, respectively, while the massive $W^{\pm}$ and Z^0 bosons are the mediators for the weak force. The last particle is the Higgs boson, an excitation of the Higgs-field, which is responsible for the Higgs-mechanism that gives particles their masses. Additionally, each particle has an antiparticle with opposite charge and opposite fermion number, but otherwise the same properties. Particles that are neutral in charge and quantum numbers are their own antiparticles. This includes photons, Z bosons, gluons and the Higgs boson.

Out of these particles, only the quarks and antiquarks can interact with the gluon (except for the self-interaction of gluons). Quarks possess a so-called color charge, the strong force counterpart to the electric charge of the electromagnetic force. There are three different color charges, conventionally called red, green and blue and their anticolors antired, antigreen and antiblue. In nature, only color neutral systems are found, resulting either from three different colored quarks bound together, from a quark antiquark pair with opposite color charges, or, in more exotic cases, from combinations such as tetraquarks (two quarks and two antiquarks) or pentaquarks (four quarks and one antiquark) [7]. These bound systems are called hadrons, which are further specified as baryons for the systems of three quarks and mesons for the quark antiquark pairs. The atomic nuclei, protons and neutrons, are for example baryons formed from the two lightest quarks, the up-quark (u) and down-quark (d). The lightest mesons are called pions and are composed of u and d quarks and include the charged $\pi^{\pm}$ and neutral π^0 mesons.

While the SM is a great success in describing the behavior of particles, it is not yet complete. The most prominent feature not explained in the SM is the forth fundamental force, gravity. Even though the idea of a gravity mediating boson, the graviton, exists, it has not been found so far. Luckily, the gravitational force is the weakest out of the four fundamental forces and therefore negligible in the dimensions of particle physics [8]. Other interesting topics that go beyond the Standard Model (BSM) are the explanation of dark matter, the unbalance of matter to antimatter or the existence of neutrino mass.

For this thesis, the tau lepton is of special interest, so the next section will provide a more detailed overview of it's properties.

2.2. The Tau Lepton

The tau lepton is the most massive lepton with a mass of $1777\,\mathrm{MeV/c^2}$ and a mean lifetime of 290×10^{-15} s [9]. Due to its short lifetime, the tau lepton decays very quickly, making it impossible to directly detect in a particle detector. Identifying a tau lepton can still be achieved by reconstructing it from the particles produced in the decay, as the decay modes have been studied thoroughly. A selection of the main decay modes of the tau lepton and their corresponding branching ratios are listed in table 2.1. From the table it is evident that the tau mainly decays into one of the other leptons or in either one or three charged pions, mostly through the intermediate $\rho^{\pm}$ or $a_1^{\pm}$ particles. In some rare cases the decay produces

Table 2.1.: Decay Modes of the τ^--Lepton [9]

Decays	Branching Ratio in %
$\tau^\pm \rightarrow \mu^\pm \nu_\mu \nu_\tau$	17.39 ± 0.04
$\tau^\pm \rightarrow e^\pm \nu_e \nu_\tau$	17.82 ± 0.04
$\tau^\pm \rightarrow \pi^\pm \nu_\tau$	10.82 ± 0.05
$\tau^\pm \rightarrow \rho^\pm \nu_\tau \rightarrow \pi^\pm \pi^0 \nu_\tau$	25.49 ± 0.09
$\tau^\pm \rightarrow a_1^\pm \nu_\tau \rightarrow \pi^\pm 2\pi^0 \nu_\tau$	9.26 ± 0.10
$\tau^\pm \rightarrow a_1^\pm \nu_\tau \rightarrow \pi^\pm \pi^\mp \pi^\pm \nu_\tau$	9.02 ± 0.05
$\tau^\pm \rightarrow a_1^\pm \nu_\tau \rightarrow \pi^\pm \pi^\mp \pi^\pm \pi^0 \nu_\tau$	4.49 ± 0.05
total	94.29 ± 0.17

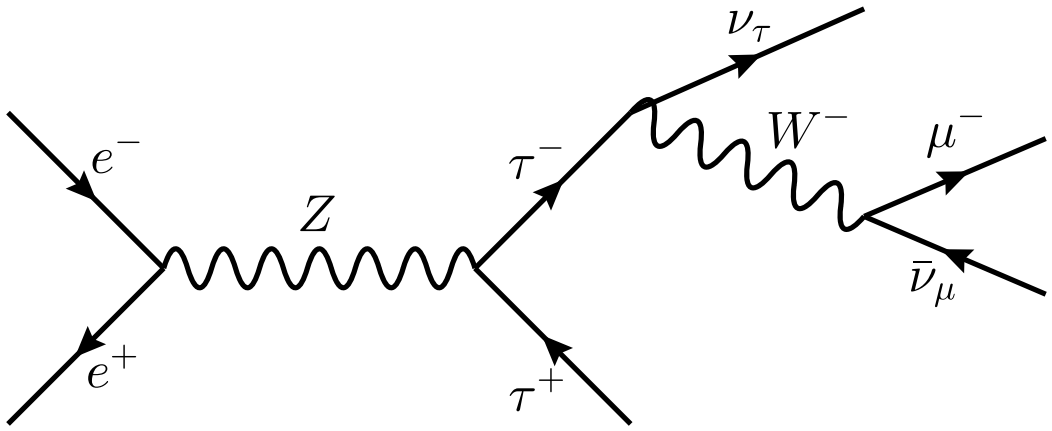
a charged kaon [9]. The neutrinos will be excluded when talking about the decays, as they are not measured in the detector.

2.3. Physics Processes

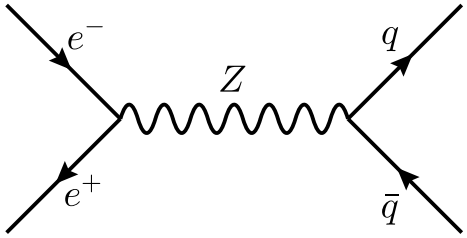
In the training of the neural network (NN) in this thesis, there are three processes that are considered. The beginning is a collision of an electron and a positron beam at a center-of-mass energy of 91 GeV. This is equivalent to the mass of the Z Boson. In the first considered process, a Z boson is created by the collision, which then further decays into a tau-lepton pair. The taus decay further via different decay modes. These data samples are referred to as the signal samples. A Feynman diagram of the creation of a tau pair and its decay is shown in Figure 2.2a, a selection of hadronic decays is displayed in Fig. 2.3. The second process is the creation of a Z boson, that decays into quark pairs of all possible flavors, except the top quark. This process is one of the background samples. The second background process is the Bhabha scattering of the electron-positron beams. These processes are displayed in Figure 2.2b and 2.2c, respectively.

2.4. The Future Circular Collider

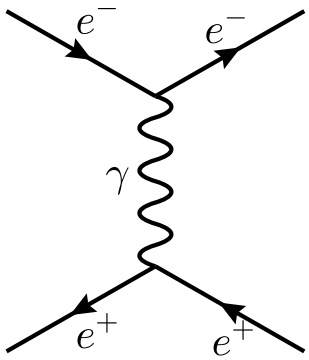
The particles, interactions and predictions of the SM are backed up by experimental findings in particle accelerators around the world. The current biggest particle accelerator is the Large Hadron Collider (LHC) at the European Organization for Nuclear Research, *Conseil Européen pour la Recherche Nucléaire* (CERN) near Geneva in Switzerland. Since 2008, collision experiments are carried out at the almost 27 km long accelerator, where proton-proton beam collisions reach center of mass energies of up to $\sqrt{s} = 13.6\,\text{TeV}$ [11]. These high energies allow the production and detection of the most massive particles like the W and Z Boson or the top quark. The biggest achievement of the LHC was the discovery of the Higgs Boson in 2012 [12], the last missing particle of the SM, that was previously introduced



(a) Feynman diagram of the creation of a tau pair, with one tau further decaying in a muon



(b) Z boson decaying into a quark pair



(c) Bhabha scattering of the electron positron beams

Figure 2.2.: Feynman diagrams of the physics processes underlying the generated data

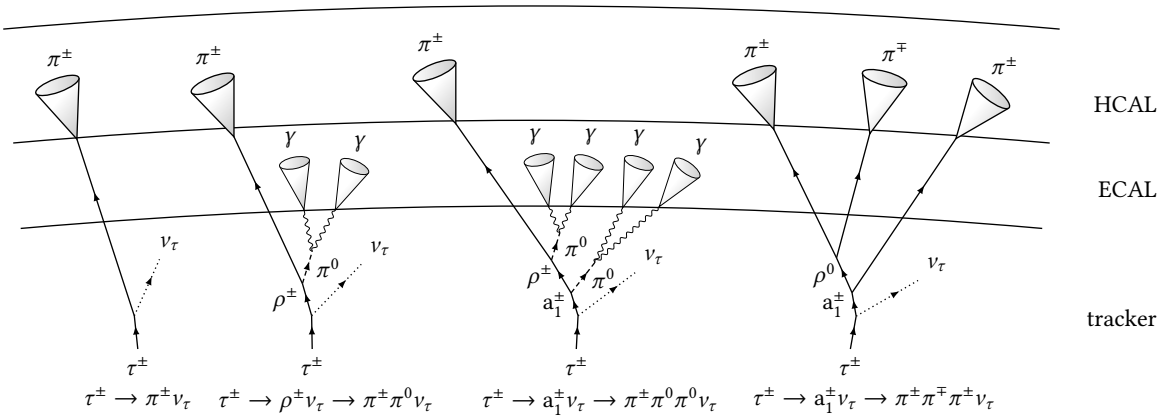


Figure 2.3.: Hadronic tau decays and their interaction with the detector parts [10]

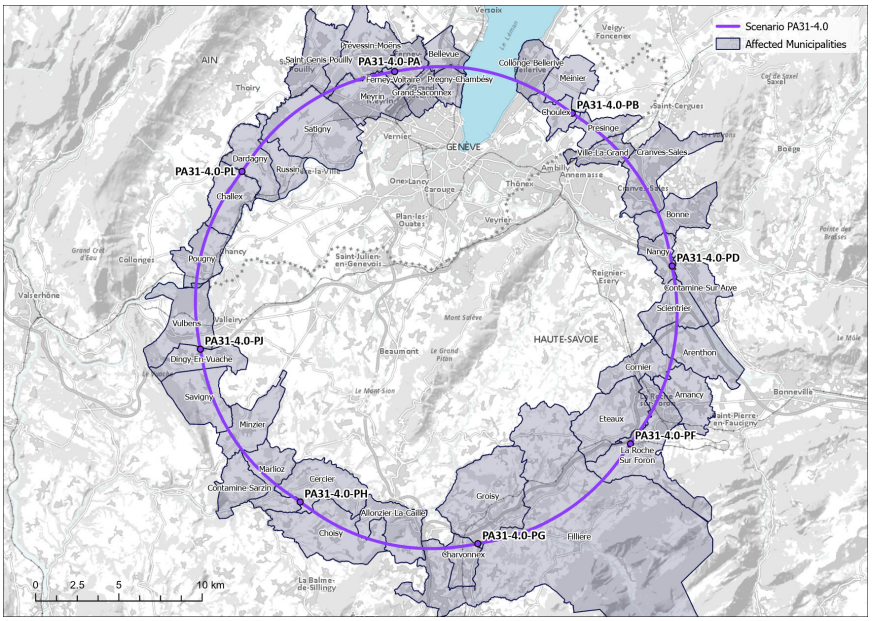


Figure 2.4.: Proposed schematic of the FCC and its surface points [13]

by theorists. To continue the search for BSM physics at high energies, further improve measurements of the properties of massive particles and achieve higher luminosities, the concept of the Future Circular Collider (FCC) was introduced in 2013 [1]. It includes a circular accelerator with a circumference of 90.7 km, aiming to reach energies of 100 TeV with proton-proton collisions [13]. The FCC is proposed to be located in the border region of France and Switzerland with seven surface points in France and one surface point in Switzerland. A schematic of the location of the FCC and its surface points is shown in Figure 2.4. The current, tentative timeline expects the construction of the FCC to start in the 2030s, taking about 15-20 years to be ready to be used. In a first phase of about 15 years the FCC is proposed to run as a lepton collider, colliding electron positron beams, referred to as FCC-ee. After the first phase, the FCC-ee is converted to a hadron collider (FCC-hh), that is planned to operate for 25 years [13]. This thesis works with simulations of the FCC-ee phase running at Z Boson resonance with a center-of-mass energy of $\sqrt{s} = 91$ Gev.

The FCC-ee phase is primarily designed to perform high precision measurements of the W, Z and Higgs bosons. With 6×10^{12} visible Z decays, the FCC-ee is expected to produce five orders of magnitudes more Z events than the Large Electron-Positron Collider (LEP). Among these are approximately 2×10^{11} $Z \rightarrow \tau^+\tau^-$ decays, allowing precise measurements of tau properties [2]. Additionally, the highly boosted tau leptons will enable easier lifetime measurements and vertex reconstruction [14].

2.5. The CLD Detector

One of the three detector concepts that are currently under study for operation at FCC-ee is the CLIC Like Detector (CLD), that is used for data simulation in this thesis. Like the

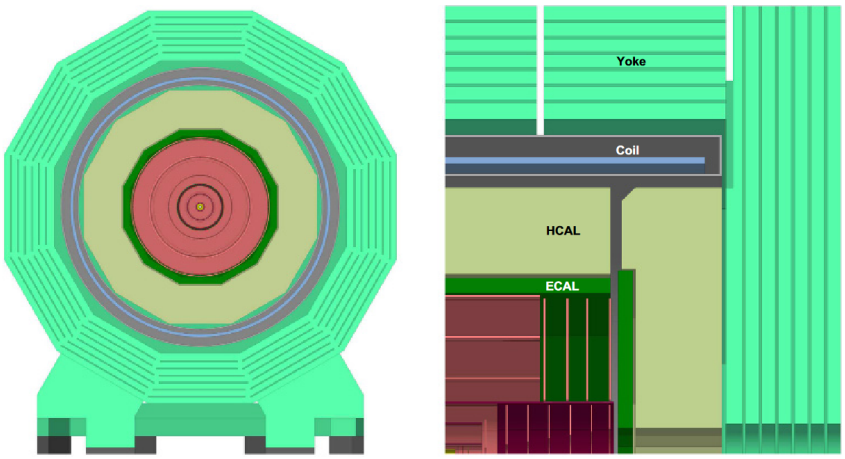


Figure 2.5.: Cross section of the CLD detector [1]

name suggests, the CLD detector is based on the Compact Linear Collider (CLIC) detector, but adapted to the properties for the new FCC-ee collider. The CLD detector consists of a silicon pixel vertex detector and a silicon tracker at the core. Full detector simulations have shown, that a single point resolution for the vertex detector of $3 \times 3\,\mu\text{m}^2$ and for the tracker of $5 \times 5\,\mu\text{m}^2$ to $7 \times 90\,\mu\text{m}^2$ can be expected. The tracker is surrounded by an electromagnetic calorimeter (ECAL) made from silicon-tungsten, with a scintillator-steel hadronic calorimeter (HCAL) following. Outside the HCAL, a superconducting solenoid provides a magnetic field of 2 T. The outermost layer of the detector consists of muon chambers, for a more precise detection of muons [1]. The layout of the detector is shown in Figure 2.5.

3. Machine Learning

Machine Learning (ML) has become a powerful tool in particle physics. Its ability to recognize patterns in large and complex datasets proved to be very useful while processing the large amount of data from collider experiments. Unlike traditional processing approaches on hardware like CPUs, that focus on few, powerful cores, neural networks (NNs) are computational models that consist of many interconnected processing units called nodes. This structure makes them predisposed for the utilization of GPUs. In the subsequent sections, an overview of machine learning is given based on the Refs. [15] and [16].

3.1. Structure and Nodes

In its most simple form, a NN consists of an input layer of nodes with the dimension of the input features, any number of hidden layers and an output layer. The nodes of each layer are connected with all nodes from the previous and the next layer. Figure 3.1 shows the architecture of such a fully connected neural network, also called a multilayer perceptron (MLP). This structure can be expanded by adding more layers and more nodes, to improve the ability to recognize more complex patterns. However, including more layers and nodes will increase the training time and needed resources. It is also crucial, that the model recognizes only general patterns in the training data rather than memorizing it in great detail. Training overly complex models for too long (or on very small datasets) carries the risk of overfitting, meaning the model performs well on the training data, but struggles to generalize to similar, unseen datasets. Other network structures like the convolutional neural network (CNN) and the graph neural network (GNN) and their application in particle physics will be discussed in section 3.3.

Each node in a neural network processes the incoming signals from the nodes of the preceding layer, where their strength influences the strength of the activation of the node. Accordingly, the new signal is distributed to the next connected nodes. The strength of each incoming signal is multiplied by weights, that are adjusted during the training process using an algorithm called backpropagation. A node with a set amount of input features x_i and weights w_i computes its output by the sum of the weighted inputs. The output signal is then determined by an activation function ϕ. The output $\hat{y}$ for a single node would therefore look like this:

$$\hat{y} = \phi\left(\sum_i x_i w_i\right) . \tag{3.1}$$

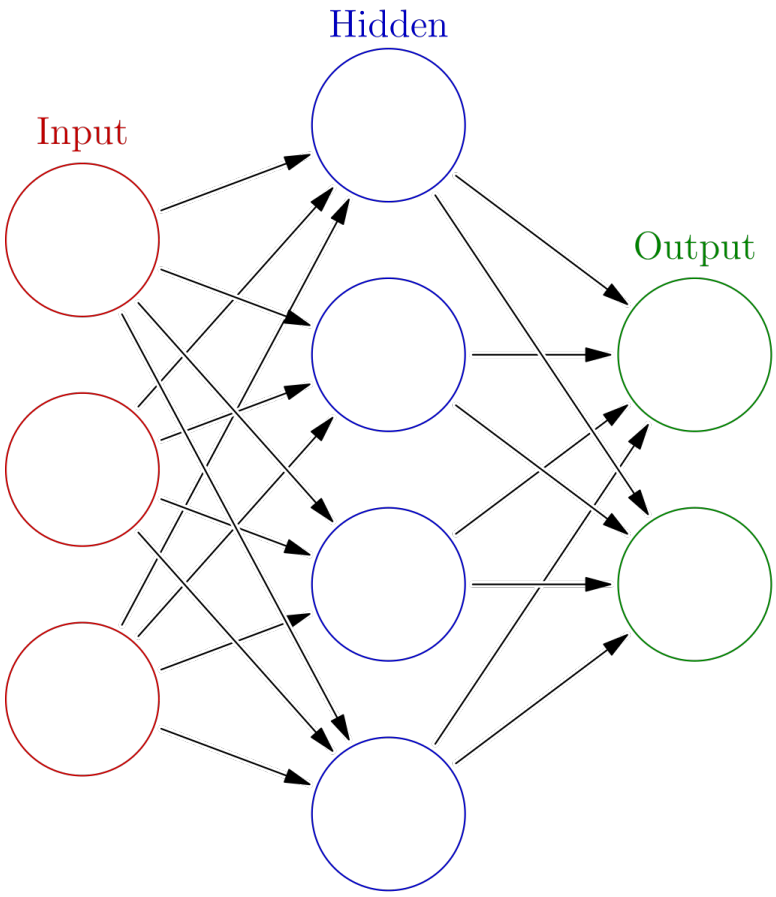


Figure 3.1.: Structure of a simple fully connected neural network [17]

Common suitable activation functions are the sigmoid function $f_{sig}(x) = \frac{1}{1+e^{-x}}$, that scales the input in the range between 0 and 1 or the rectified linear unit (ReLU), which outputs the input directly if it is positive or 0 if it is negative $f_{ReLU}(x) = \max(0, x)$. Without a non-linear activation function, the transformations done by multiple layers can always be reduced to a single layer of nodes, severely limiting the learning potential of the network.

3.2. Adapting the Weights

By default, the initial weights of the connections in the NN are assigned randomly. Consequently, the model will give a random output in the first iteration of the training process. In supervised learning[1], the training data includes truth labels that are assigned beforehand. From these truth labels y_i and the prediction value $\hat{y}_i$, a loss function is calculated, that

[1]There are several other learning methods like unsupervised or reinforcement learning, that will not be discussed further in this thesis.

quantifies the difference or error between the predicted value and the truth value. The loss function could for example be the mean squared error

$$L = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \,. \tag{3.2}$$

The goal of the training is to minimize this loss function by adjusting the weights and therefore minimize the error between prediction and actual event. A common optimization method that achieves this is called gradient descent. In gradient descent, the negative gradient of the loss function with respect to the weights is calculated, indicating the direction of the steepest descent. The weights are then shifted one step in that direction. The size of the step is determined by the learning rate, a customizable hyperparameter of the NN.
After each step, the next iteration is performed with the new weights, approaching a minimum iteratively. This minimum is often a local minimum and not the global minimum of the loss function. The learning rate determines the time and precision, with which the minimum is approached. While a higher learning rate allows the model to change its weights faster, it lacks precision or might even fail to converge entirely. A lower learning rate converges to the minimum slower but more precise. To get the best of both worlds, there are optimizers that utilize variable learning rates to achieve a fast but precise result.

In conclusion, a neural network processes input information by converting it to mathematical representations, making predictions and evaluating them with a loss function. By repeated computations and adjustments, the parameters are then optimized to the given task, gradually approaching the optimal configuration. This enables the network to learn general patterns and make accurate predictions on similar data.

3.3. Machine Learning in Particle Physics

In the last years, the application of machine learning in particle physics has grown steadily. An important part of the processing of collider data is filtering the interesting signal events, that are wanted for a particular task, from the uninteresting events. When working with the large amounts of data produced by modern particle accelerators, ML algorithms can save a lot of time and are therefore well suited for these types of classification tasks. A common example for the application is jet tagging, which refers to the classification of particle jets based on the initial quark, that produced the jet. While classically the distinction was based on physics motivated observables like displaced vertices for specific particles only found in specific flavored jets or energy spectra of different width, that can be used to distinguish gluon initiated jets from quark initiated jets, ML algorithms can find patterns directly at the lowest level of observables produced by the detectors [18]. Different neural network architectures excel at varying tasks in particle physics. Next to the fully connected neural networks, commonly used structures are convolutional, recurrent or graph neural networks. They have individual ways of processing the input data, which makes them useful for certain tasks, that will be discussed in the subsequent sections.

3.3.1. Convolutional Neural Networks

convolutional neural networks (CNNs) use a convolution kernel to find patterns and therefore excel at image processing and recognition. In the learning process, the kernels of the different layers are adapted to the different patterns, that are important to distinguish different inputs. Pooling layers in between the convolutional layers help downsize the images to the most important features and make the pattern recognition independent from their spatial position. In particle physics, CNNs can for example be trained to classify signal from background data by feeding image representations of signal and background data [19].

3.3.2. Graph Neural Networks

graph neural networks (GNNs) handle graph like data, with nodes that represent entities and edges, that connect the nodes representing their interactions or relationships. Unlike CNNs and recurrent neural networks (RNNs) which work with structured data like grids (of pixels) or sequential data, GNNs can handle more abstract, sparse or irregular data best represented as graphs. The nodes in the GNN are represented as feature vectors in a high dimensional embedding space based on the input features. Nodes with similar input features are therefore placed close to each other. The more the input features differ, the further apart the nodes are placed. Each node is connected with its neighboring nodes by edges, that can be directed, only allowing the exchange of information in one direction or undirected. In weighted graphs, there are additional weights for the edges, determining their strength. At each layer, information is exchanged between the neighboring nodes. The nodes then aggregate the received information and combine it with their own features. With the next iteration at the next layer, the updated information is exchanged again. This means that with each additional layer, the nodes receive information from nodes one connection further away. An example of the message passing in a GNN is shown in Figure 3.2, where the gray highlighted nodes and the thickened arrows show, how the information from specific nodes propagates through the graph. During training, the embeddings of the nodes in the embedding space are optimized to more accurately capture and represent the input features important to the given task.
In particle physics, GNNs have been successfully used for tasks like jet tagging or event classifications, being on par or better than traditional approaches [20] [21].

3.3.3. Recurrent Neural Networks

recurrent neural networks (RNNs) are suited to process sequential data, where the output of each step is influenced by information from the previous steps. They are set up similar to a feed forward NN, but with the addition of a feedback loop that introduces a hidden state, that influences the next iteration. While feed forward NNs only use the current input and the weights from previous training to make a decision, the hidden state, that is adapted after every iteration, contains additional information from the previous iterations. The

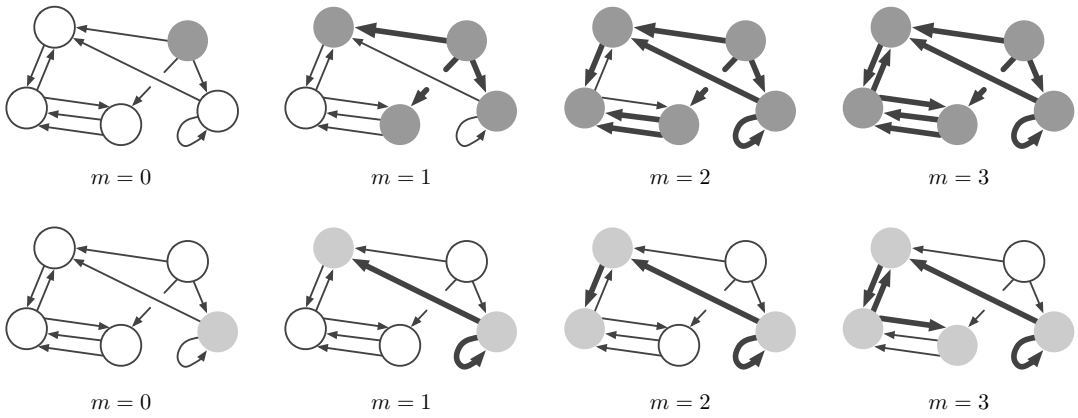


Figure 3.2.: Message Passing in a GNN for two specific nodes: the upper-right node in the upper row and the lower-right node for the lower row over four iterations. Gray shaded nodes represent those updated after receiving the messages from the starting nodes, highlighted edges trace the path where the message is passed [22].

architecture of such a RNN is shown in Fig. 3.3, with the feedback loop on the left side and the unfolded representation on the right.

While the hidden state can capture information from the previous inputs, long term dependencies are often neglected. This issue can be resolved by long short-term memorys (LSTMs), a form of RNN, that can selectively store only the more important information for the task at hand for longer periods of time. In LSTMs, the nodes are replaced by long short-term memory (LSTM) cells, that introduce a second so-called cell state, that functions as a kind of long term memory. Therefore, LSTMs have both the hidden state, that stores information from short term dependencies and the cell states for important long term dependencies, hence the name long short-term memory (LSTM) [23].

The ability of LSTMs to selectively store only the most important information for the task at hand, e.g. which particles have the biggest contribution on the jet identification, and the

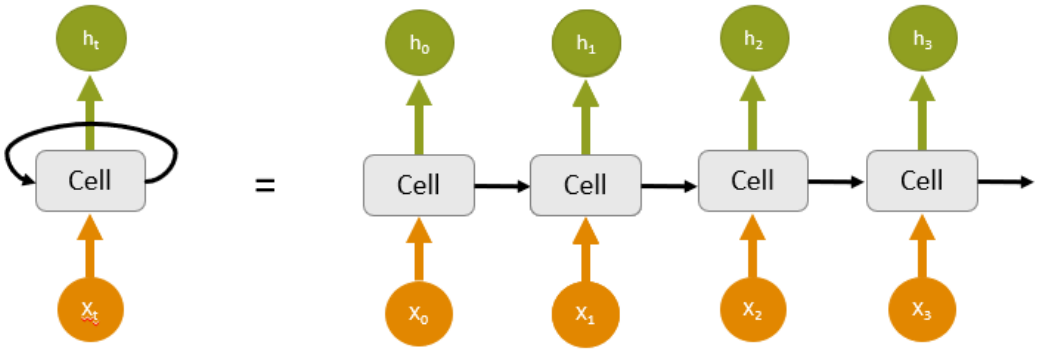


Figure 3.3.: Representation of a RNN with a feedback loop (left) and unfolded (right) [23]

ability to handle sequences of varying lengths, like different amounts of particles in a jet, make them a candidate for jet tagging algorithms [18].

Another concept of ML that was explored with RNNs is the attention mechanism. In the previous discussed approaches to the processing of sequential data, the data of the whole sequence is compressed into one final state, the context vector, that is used to generate an output. However, this approach can lead to a loss in information from the input sequence, especially for longer or more complex input sequences with long term dependencies. This is where attention comes in as a way for the model to focus on the most important parts of the input sequence. It basically gives the model the ability to access all hidden states of the different parts of the input sequence and produce an output, that does not have to rely on a very compressed context vector.

3.3.4. Transformer

The Transformer architecture was developed to address the challenge of capturing long term dependencies, while also reducing the computation times through parallelized computations. It takes the general structure of a RNN with an attention mechanism, but removes the feedback loop, leaving a ML model that is solely dependent on attention. In the following, a basic explanation on the functionality of a transformer will be given based on Ref. [24].

A common example of a sequential input are sentences, where the words need to be translated into a numerical representation to be processed by a ML model. This step is called embedding. Since Transformers do not use the recurrent structure of RNNs, they rely on positional encoding to include information about the order of the input. After embedding and positional encoding, the input is processed using self-attention. The respective input tokens (e.g. words in a sentence) are compared to each other to determine the relationship between them. This is achieved by comparing queries, keys and values, which are vectors generated for each input token by multiplying the embeddings with respective weight matrices. The relationship between words is determined by scores, that are calculated by different methods like the dot product of queries and keys or additive attention. For each input, these values are then summed up, normalized and used to scale the corresponding values. The original embedded and positional encoded values are then added to the self attention values, creating residual connections, and produce a representation, that contains information about its relations to the other inputs.

In a Transformer, multiple of these self-attention cells are used simultaneously, which is called multi-headed attention and allows different attention heads to focus on different relations. The attention cells are then connected to a fully connected feed forward NN. This is the main building block of the Transformer, its whole structure is shown in Fig. 3.4. A similar block is used to embed the output, with an additional multi-head attention block, that compares the input and the output representations after their respective self-attention layers.

Compared to the RNNs and LSTMs architectures, a Transformer does not need to wait for the previous input tokens to be processed. The keys, queries and values can be calculated

simultaneously for all tokens, which allows the model to be parallelized, for example on GPU cores, which greatly reduces running times.

3.4. Geometric Algebra Transformer

As the name suggests, the GATr combines the transformer architecture with the concepts of geometric algebra, making it perfectly suited for geometric data. It is capable to represent geometric objects and transformations, as well as identify common patterns like relative distances, while being equivariant to the Euclidean group E(3). Combined with the great efficiency and scalability provided by the Transformer architecture, the GATr becomes a universal solution for many problems involving geometric data, that arise in physics, chemistry or robotics related fields and many more [4].

Geometric algebra is based on two operations, addition and the geometric product. The geometric product of two vectors a, b is defined as the sum of the interior and the exterior product

$$ab = a \cdot b + a \wedge b \tag{3.3}$$

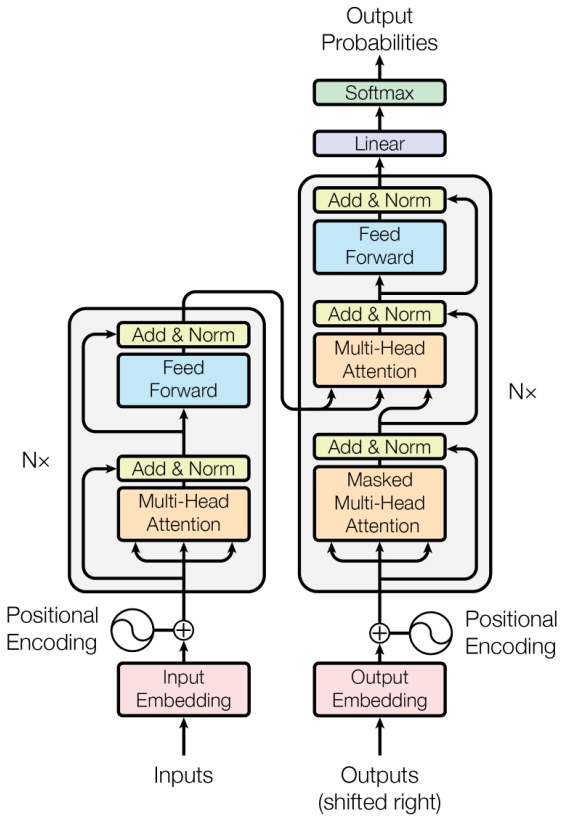


Figure 3.4.: Architecture of the Transformer model [24]

The result is the sum of a scalar and a bivector (like a vector, where the magnitude is the area of the bivector and the direction is either clock- or anticlockwise), which is generally called a multivector. Multivectors consist of the products of the basis vectors, so a multivector in a three dimensional geometric algebra with the basis vectors $\vec{e}_1$, $\vec{e}_2$, $\vec{e}_3$, would look like this

$$x = x_s + x_1\vec{e}_1 + x_2\vec{e}_2 + x_3\vec{e}_3 + x_{12}\vec{e}_1\vec{e}_2 + x_{13}\vec{e}_1\vec{e}_3 + x_{23}\vec{e}_2\vec{e}_3 + x_{123}\vec{e}_1\vec{e}_2\vec{e}_3 \tag{3.4}$$

where $x_s, x_1, \ldots, x_{123}$ are real coefficients. So a multivector is the sum of elements of different dimensionalities or grades, in this case of scalars (grade 0), vectors (grade 1), bivectors (grade 2), and a trivector (grade 3) [25].

In the case of the GATr, the objects of interest are in the three space dimensions, with operators that change the objects position and orientation in the three dimensional space, like translations or rotations. To achieve this, a three dimensional geometric algebra is insufficient, as its multivectors can represent only linear subspaces, that either pass through or rotate around the origin. This is solved by introducing a fourth homogeneous coordinate $x_0\vec{e}_0$, that results in the projective geometric algebra $\mathbb{G}_{3,0,1}$ with a 16-dimensional geometric algebra, whose multivectors can represent any points, lines, planes, rotations, translations, and reflections in three dimensions.

The GATr follows the plane geometric algebra approach, where planes are the basis for the construction of all other objects [4]. Here, planes are represented by vectors. Lines and points are constructed as the intersections of two and three planes, respectively. The exterior product of two plane representing vectors results in their intersecting line, so the representation of lines are bivectors (the exterior product of two vectors) and the representation of points are trivectors (the exterior product of three vectors). The reason why planes are chosen as the basis for the geometric algebra is, that transformations can be realized by a number of reflections on planes. So while a single reflection on a plane just results in a regular planar reflection, performing two planar reflections result in either a translation, when the two planes are parallel, a rotation around the common line when the planes are intersecting or a line reflection, in case the planes are orthogonal. Point reflections, glide reflections or rotoreflections can be achieved by three consecutive plane reflections, depending on the alignment between the planes. Any general motion in 3D space of a rigid body is possible with four reflections [26]. In conclusion, geometric algebra is a mathematical framework, where geometric objects like planes, lines, and points and geometric transformations like translations, rotations or reflections can be represented as multivectors.

The architecture and processing of the GATr Model is shown in Fig. 3.5. The raw inputs are preprocessed in geometric representations and embedded in the geometric algebra in the form of multivectors and scalar inputs. The GATr is composed of N transformer blocks, that are built similar to a regular Transformer. A transformer block consists of linear layers with learnable weights, normalization layers, an attention head with dot-product attention, a bilinear layer for better grade mixing when calculating geometric products, a gaussian error linear unit (GELU) layer to introduce non-linearity and two residual connections. Additionally, auxiliary scalar representations are used to represent non-geometrical data. In linear layers, they mix with the multivector representations, in the self-attention layer

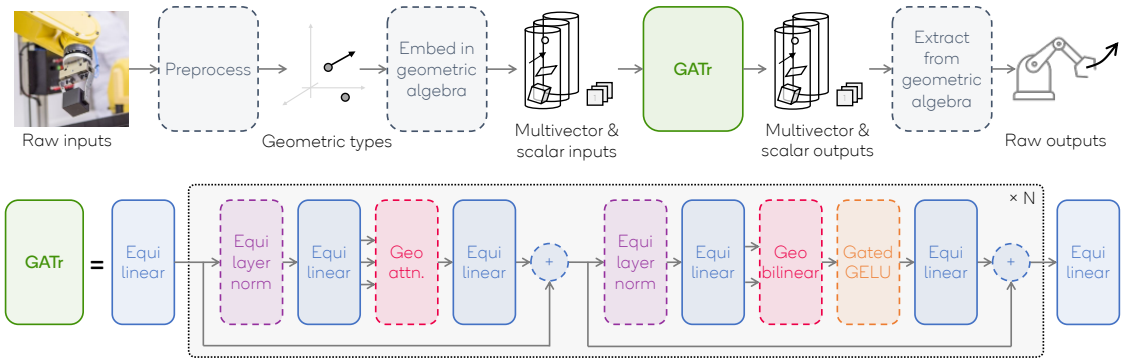


Figure 3.5.: Architecture and processing of the GATr. Squares with solid lines have learnable parameters. [4]

the attention is calculated separately for the scalars and multivectors and then summed up [4].

In conclusion, the GATr combines the precise representation of geometric data in geometric algebra with the efficiency of Transformers, making it a great tool for many fields, including physics. By naturally handling for data like point clouds in particle collisions, it becomes the architecture of choice for the work presented in this thesis.

4. Data Generation with Full Simulation

Before building a new accelerator and detector, it is standard practice to simulate data for the proposed setup. Data simulation is critical to guide detector design, optimization, and preliminary software development. By simulating the behavior of physics processes of interest, whether within the SM or BSM, in detectors, key features to reconstruct and profile these processes can be identified, and the detector design can therefore be purposefully optimized. An example of this is the Higgs boson, whose existence and properties were predicted by theorists well ahead of its discovery. Theoretic predictions about the Higgs bosons energy range and decay processes guided the detector design to focus on capturing these predicted attributes, which was successful in the end. A major and relevant application of simulation data is reconstruction algorithm development, especially in machine learning, which is also the subject of this thesis. Simulation data are essential for training and testing models beforehand, as the performance of the algorithms can easily be evaluated. During the simulation process, every resulting particle, interaction, and detector hit can be identified and tracked back to its true origin. This is needed for supervised learning of neural networks, that rely on comparing their prediction to the assigned truth label.

4.1. Types of Simulations

Generally, a distinction between two types of simulations is made in particle physics. Fast simulations are first and simplified simulations, valued for their fast computation times. After generating particle collisions using Monte Carlo (MC) simulations, the tracks and properties of the resulting particles are Gaussian smeared to emulate the detector response. Alternatively, a full simulation involves generating particles, that then go through detailed modeling of their interactions with the detector material, simulating processes such as energy deposition, scattering and further decays to reproduce an accurate detector response. The detector hits are then used to reconstruct tracks and particles or are utilized otherwise. The benefit of this method is that the result closely resemble real data, accounting for factors such as the full interaction of incoming particles with detector material, realistic resolution and efficiency of detection units, as well as noise and other imperfections [27].

The goal of this thesis is training a tau lepton tagger with full simulation data, that can distinguish tau events that originate from a Z boson decay from other background events, like the decay of a Z boson into different quark pairs or bhabha scattering of the beams. Furthermore, a classification into the different decay modes of the tau is aimed for. In the following, the general process of the data simulation is presented.

4.2. Simulation Process

The first step in the data simulation is the MC Simulation of the collision of the electron-positron beams, that are intended to run at the first stage in the FCC-ee phase. These collisions occur at a center-of-mass energy of about 91 GeV, corresponding to the Z boson resonance. The MC simulation is run using Pythia8 [28], where various settings can be adjusted, specifying which processes should be generated. For the tau signal simulations the creation of two taus that decay further according to the different decay modes listed in section 2.2 is simulated. After all the particles are generated, the next step is to simulate the interaction with the detector. In this case, the CLD detector described in section 2.5 is modeled by the Geant4 toolkit developed by CERN [29]. Geant4 simulates all interactions of the MC particles with the detector material, giving all hits and detector signals as outputs. In a third step, all the information available from the detector simulation, which resembles the data collected by real detectors, are used to form tracks and calorimeter clusters, and subsequently particle objects. The reconstruction is performed using a Pandora reconstruction algorithm [30]. In the training performed in this thesis, the information from the reconstruction is not used. The last step runs a Python script that extracts all the information that is used in the training like the coordinates of the detector hits, the energy deposition, and the momentum and writes them into a ROOT file. This step also simplifies the data format by removing object-oriented structures and organizing the data into simple variables in a flat format. Furthermore, the script analyses the individual events, assigning them a label based on the true information obtained by the MC simulation. The events are classified as either one of the tau decay modes, background or undefined (for any of the tau decay modes, that are not used in the training. See the used decay modes in table 2.1). This general workflow as well as the mentioned python script is based on the work of Dr. Dolores Garcia (CERN), that can be found in Ref. [31].

4.3. Data Evaluation

Before starting with the training, the data was evaluated to verify, that the simulations delivered the expected results. While most of the data looked promising, there were a few outlier found, that contained a lot of particles from the MC simulations but caused very few hits in the detector simulations. Those are discussed in more detail in the appendix A.1. The hits of one of the regular events are displayed in a 3D plot in 4.1. The hits are clearly distributed around two clusters, that originate from one of the taus each. The sizes of the data points correlate logarithmically to the deposited energies with the hits. Small points offside the main clusters are considered detector noise.

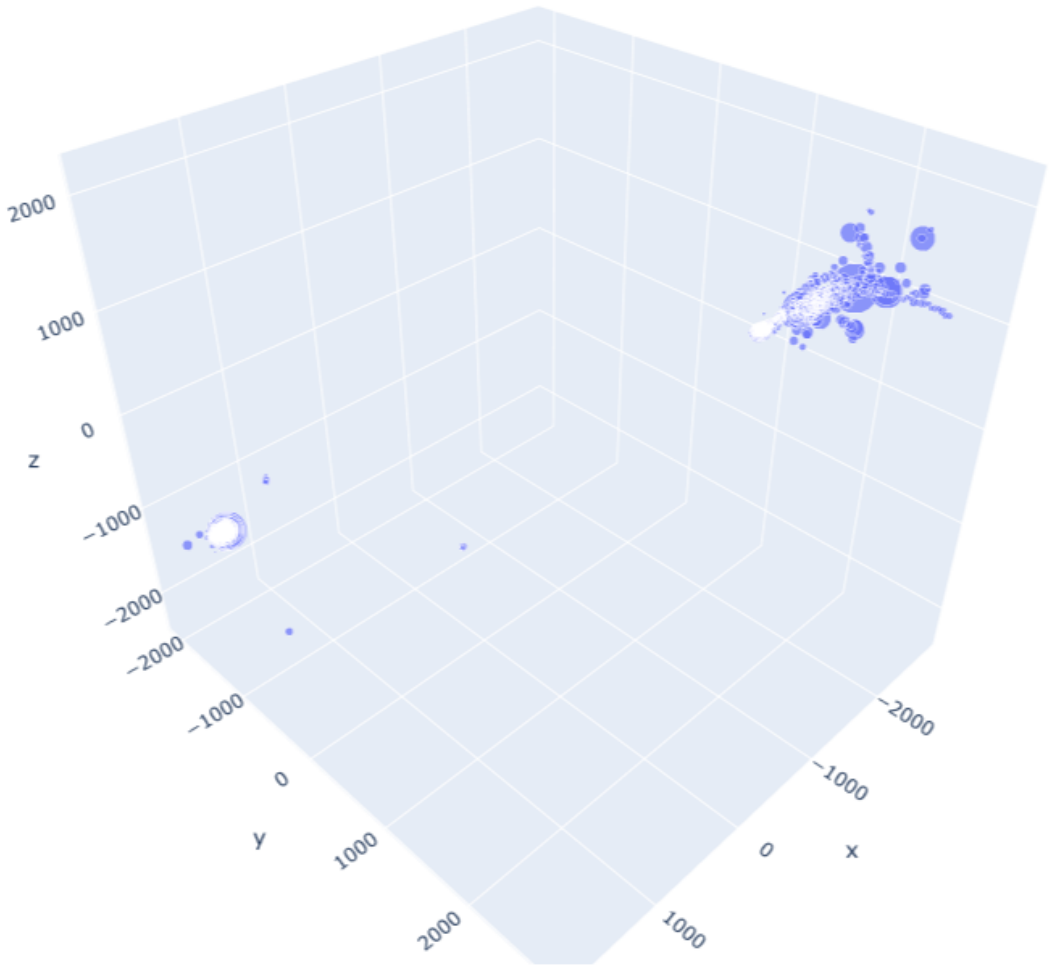


Figure 4.1.: Detector hits of a regular signal event

5. Training and Evaluation of the Neural Networks

The training of the neural network is performed in python using the PyTorch Lightning module [32]. The workflow for that is adapted from the work of Dr. Dolores Garcia (CERN), that can be found on Github [31]. Data generated as described in chapter 4 is used for training and testing the models.

5.1. Network Configuration

For the work in this thesis, an NN on the basis of the GATr architecture is utilized. The network features a multivector input channel and a single output channel, with 16 hidden channels in between. Aditionally, three scalar input channels are included alongside 64 hidden channels and a single output channel. The network comprises ten transformer blocks using the default parameters for the linear layers and the self-attention head. To produce the final output, a MLP readout layer is implemented. This layer consists of an input layer of 20 nodes, two hidden layers with 20 nodes each, and an output layer, where the amount of nodes varies, depending on the amount of classes considered.

The input data consists of the coordinates of the detector hits, the corresponding momentum and energy of each hit and the information, at which detector part the hit was detected. The coordinates are embedded as multivector representations in the GATr, hit type, momentum and energy of the hits are processed as scalars. Predefined labels are used as the ground truth of the tau decay modes in the supervised learning process. To ensure equal impact of the different classes independent of their commonness, their influence on the loss calculation is scaled by the inverse of the branching ratio.

5.2. The TOpAS Cluster

While the first smaller trainings were performed on the GPUs of the ETP machine, bigger trainings with more data required more and faster computing power. The Throughput Optimized Analysis System (TOpAS) Cluster of the Grid Computing Centre Karlsruhe (GridKa) provides 32 Nvidia V100 graphics cards and over 1500 CPU cores with 3 GB of RAM each. Using this computation center greatly improved training times and also solved

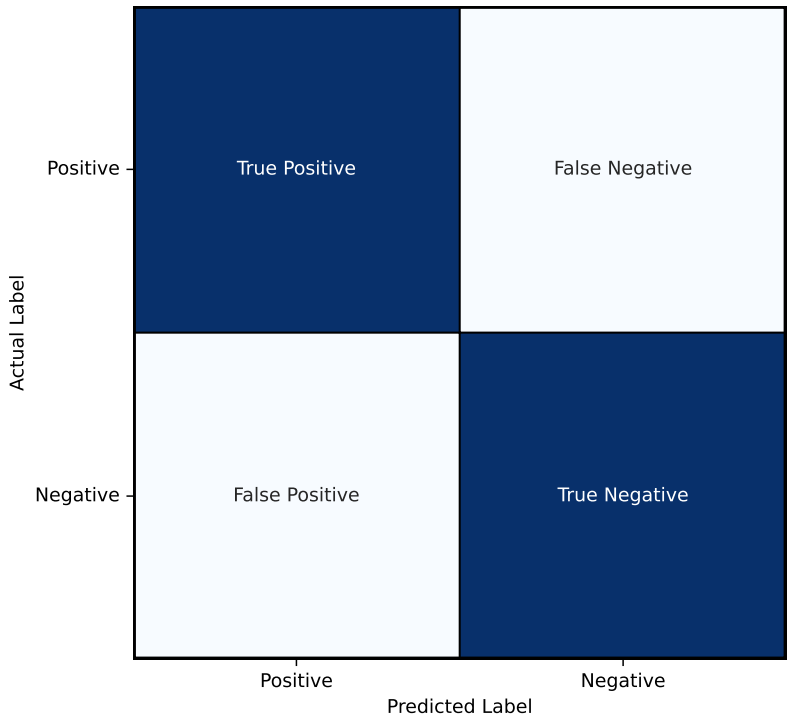


Figure 5.1.: Example confusion matrix for a binary classification

a minor GPU issue, where the VRAM utilized by the training exceeded the available VRAM of the ETP GPUs.

5.3. Evaluation Metrics for Neural Networks

Various metrics can be used to assess and compare the performance of ML algorithms. This section highlights the evaluation techniques applied to compare the NNs developed in this thesis.

The results are presented in a confusion matrix, where the rows represent the true decay modes and the columns show the predictions made by the NN. The entries of the matrix are normalized by the rows for better comparability. In the case of binary classification, the confusion matrix resembles Fig. 5.1. The instances correctly identified as positive are called true positive (TP), while true negative (TN) refer to the corresponding negatives that are correctly identified. Conversely, misclassified outcomes are categorized as false positive (FP) and false negative (FN). The commonly used metrics are derived from these

confusion matrix entries. One such metric is the sensitivity or true positive Rate (TPR), that measures the probability of correctly identifying a positive outcome:

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} . \tag{5.1}$$

Comparably, the false positive rate (FPR) measures the ratio of falsely identified positives out of all positive classifications

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{TP} + \mathrm{FP}} . \tag{5.2}$$

The precision describes the quote of correctly identified outcomes of all positive outcomes, or in short

$$\mathrm{Precision} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} . \tag{5.3}$$

A good comparison between models can be made by comparing the ROC curve, that is created by plotting the TPR on the y-axis against the FPR on the x-axis for different thresholds. Thresholds refer to the decision boundary of the NN. By default, the NN outputs probabilities for both classes and the threshold determines, at which probability a particular class is selected. When the probabilities of the decisions are known, for each threshold the TPR and FPR can be computed and plotted, forming the ROC curve [33].

For multiclass classification tasks, similar metrics can be evaluated. A class can be either compared to all other classes resulting in a one-vs-all ROC curve, or to one specific class, resulting in a one-vs-one ROC curve.

To compare two models against each other, the area under the curve (AUC) can be calculated. For a model that simply guesses the outcome, the ROC curve would just be a straight line with a slope of 1. The AUC would therefore be AUC = 0.5. Generally, the higher the AUC, the better the performance of the model.

5.4. Initial Implementation

In its initial state, the script that is adapted from Ref. [31] performs a binary classification between the decays $\tau \to \pi^{\pm}\pi^{0}$ and $\tau \to \pi^{\pm}$. To test the script in the initial state, a training with 10000 of the simulated signal events is performed for 15 epochs, and then evaluated with another 10000 events.

The confusion matrix for this evaluation run in Fig. 5.2 shows, that even a training with a very limited amount of data and training time achieves a good result for this task. This initial step shows that the adapted NN and script are generally capable for this task and can be expanded to more decay modes.

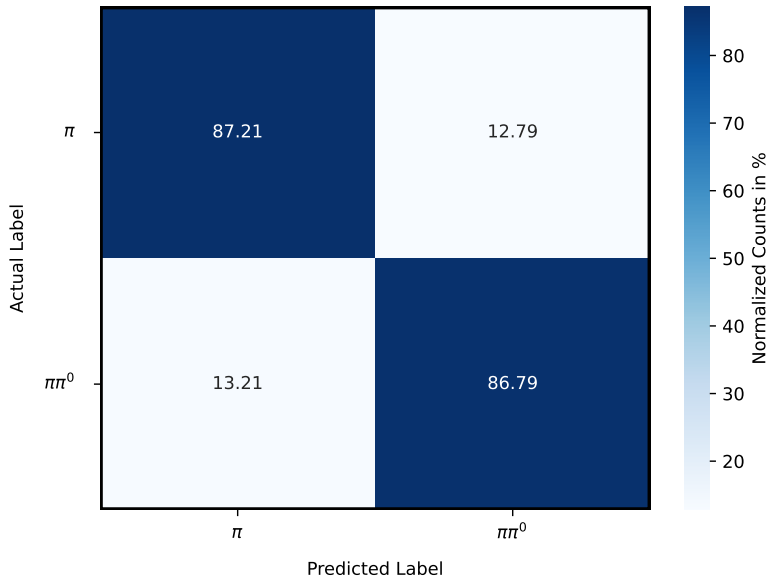


Figure 5.2.: Confusion matrix for the binary classification

5.5. Multiclass Classification

The main goal for this thesis is to implement a multiclass classification model, that distinguishes between the tau decay modes listed in table 2.1 and two background processes: $Z \rightarrow q\bar{q}$ and Bhabha scattering. Initially, the data has to be relabeled to include these additional classes and the script has to be modified to the multiclass classification task. All tau decays not relevant to this task, which includes all tau decays with a branching ration smaller than 1 %, are excluded from the training set. The dimensions of the input layer of the NN and the MLP readout layer are increased to the number of classes. With these changes, another model is trained with 50000 signal events for 50 epochs. The starting learning rate was kept at 0.001.

Fig. 5.3 shows the confusion matrix of a set of 50000 signal events, that were processed by this model. Especially the performance for the leptonic decays $\tau \rightarrow e$ and $\tau \rightarrow \mu$ is very good, while the efficiency of the $\tau \rightarrow \pi 2\pi^0$ is rather low with only 46 %.

5.6. Addition of Background

With the multiclass classification working, two new classes are introduced. In addition to the classification of the tau decay modes, the model is supposed to distinguish background processes that appear in the beam collisions or from the Z decays. The relevant background processes that are included in this step are the decay of a Z boson into a quark pair, including all quarks except the top quark, and Bhabha scattering of the beams. To address an issue, where the model occasionally classified all events as a single class, additional weighting

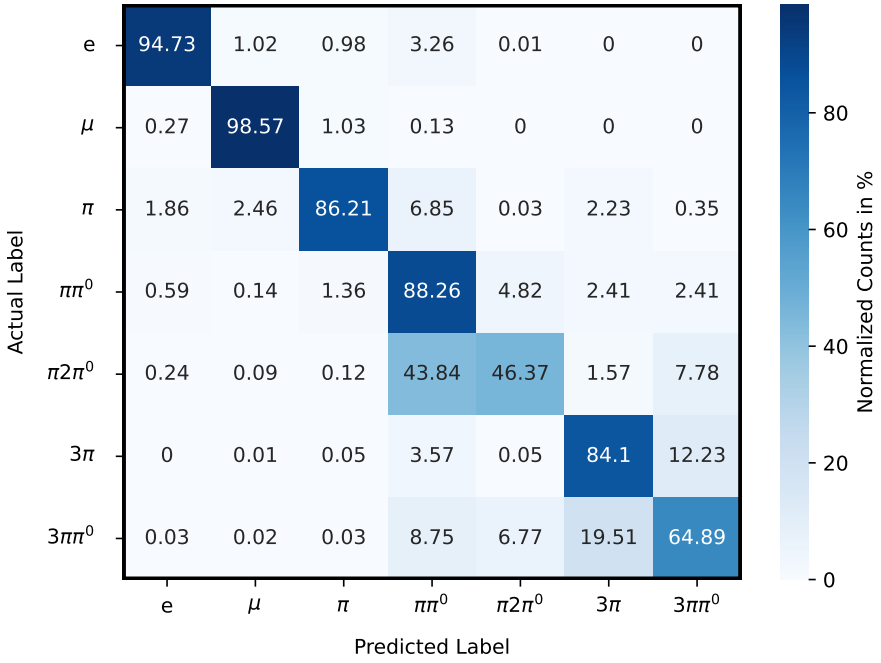


Figure 5.3.: Confusion matrix for the multiclass classification

factors for the events are introduced. Without these weights, all events are treated equally in the loss calculation, heavily favoring the events that occur more often, leading to the described problem. The weights are applied as factors of the reciprocal value of the branching ratio of the tau decays to ensure, that the total loss is equally influenced by each class. The weight for the background classes is determined in the same way, depending on the amount of background events used in the training. With this problem out of the way, a network is trained with 50000 signal events, 20000 $Z \longrightarrow q\bar{q}$ events and 10000 Bhabha events.

This model, whose performance is shown in Fig. 5.4, shows a strong efficiency for labeling the leptonic decays and the added background events. The efficiency for the $\pi 2\pi^0$ decay remain at a rather low 51 %, with the $3\pi\pi^0$ decay also being at the lower end with 71 % efficiency. For the tau polarization studies, the method used to reconstruct the ρ ($\pi^{\pm}\pi^0$) and a_1 ($\pi^{\pm}2\pi^0$) is the same, so the high misclassification rate between these two is not a significant issue.

This performance can also be seen in the ROC curves, that are shown in Fig. 5.5. While these ROC curves are generally pretty good, the one-vs-all ROC curves of the e, μ, Zqq and Bhabha classes particularly stand out in Fig. 5.5a with an AUC of 1. The worst performing class is the $\pi 2\pi^0$, who performs especially bad against $\pi\pi^0$ and $3\pi\pi^0$, as can be seen in the one-vs-one ROC curves in Fig. 5.5b. All one-vs-one ROC curves can be found in Appendix A.2.

From a physics point of view it makes sense, that the decay modes that differ only from the amount of π^0 are harder to distinguish. With a branching ratio of 98.8 % and a mean lifetime of 8×10^{-17} s the π^0 decays into two photons almost immediately in most cases

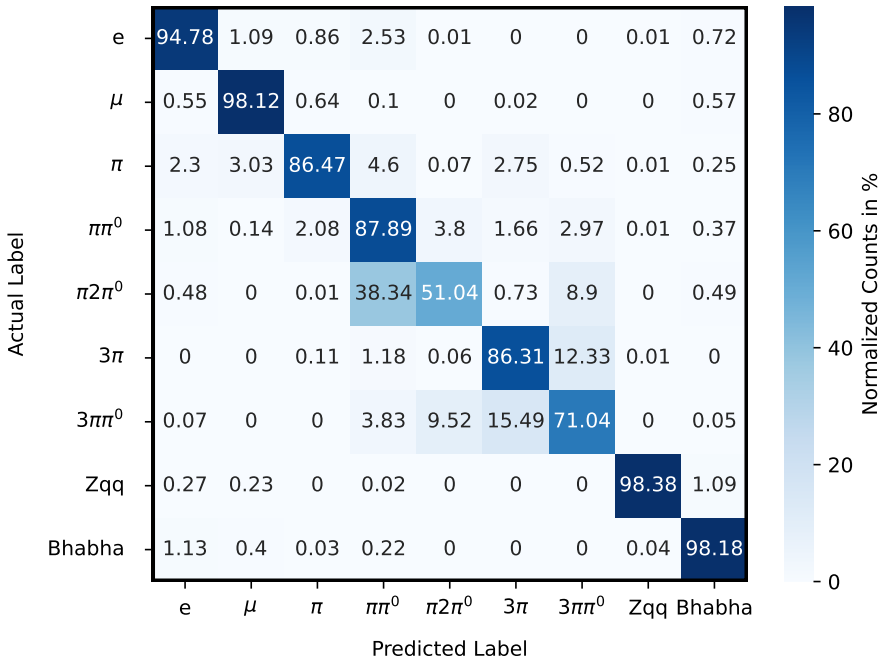
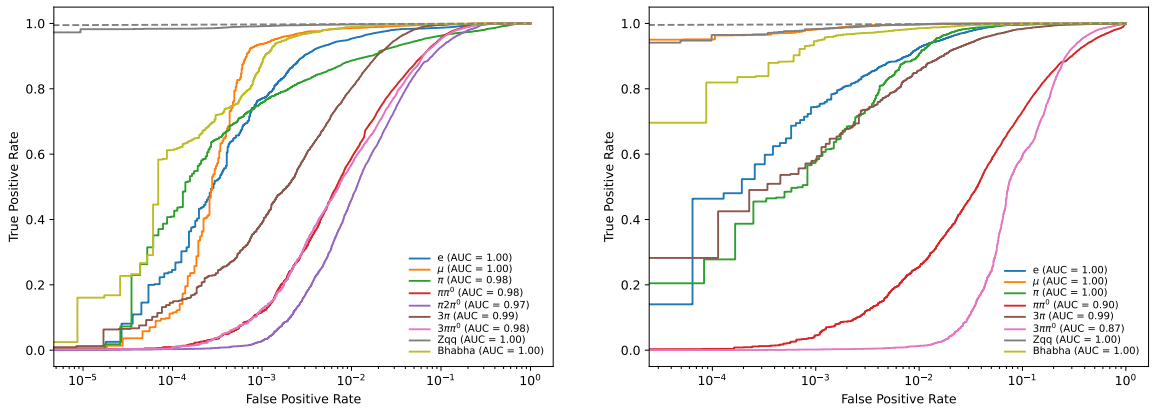


Figure 5.4.: Confusion matrix for the multiclass classification including background



(a) One-vs-all ROC curves

(b) One-vs-one ROC curves for $\pi 2\pi^0$

Figure 5.5.: ROC curves for the multiclass classification

[9]. So in the detector, the difference between these decay modes are additional photon induced showers. In $Z \rightarrow \tau^+\tau^-$, the tau leptons and subsequent pions are highly boosted, causing the photons from the π^0 to be tightly collimated. This can make the resulting hit clusters hard to differentiate, as they may largely overlap, causing two or more clusters to potentially not be recognized as distinct photons.

Similarly, the good efficiency for the leptonic decays can be justified. Muon detection is very clear in detectors like the CLD due to their dedicated muon chambers, that are only reached by muons. Given the detector information about where the hits where registered,

Table 5.1.: Explicit reconstruction of hadronic tau decays

Actual Decays	Reconstructed Tau ID						
	h	h+γ	h+2γ	h+3γ	h+4γ	3h	n
$\pi^{\pm}$	0.81	0.03	0.00	0.01	0.01	0.00	0.13
ρ ($\pi^{\pm}\pi^0$)	0.03	0.21	0.59	0.07	0.01	0.00	0.09
a_1 ($\pi^{\pm}2\pi^0$)	0.00	0.02	0.09	0.31	0.39	0.00	0.10
a_1 ($\pi^{\pm}\pi^{\pm}\pi^{\pm}$)	0.02	0.00	0.00	0.00	0.00	0.74	0.16

it should be relatively easy for the NN to recognize muon events. Electrons (and positrons) are the only particles in these decays, that generate a track, an adjoining shower in the ECAL and no energy deposits in the HCAL, making these easy to distinguish from pions, that mainly induce showers in the HCAL and only mild showers in the ECAL, and photons that do not leave a track.

5.7. Comparison to Explicit Tau Reconstruction

Another approach is explicit reconstruction using the Pandora particle flow reconstruction algorithm. The algorithm works by tracking the paths of the particles through the detector. Momentum and energy measurements are then taken from the exact detector component where the highest accuracy is expected, such as the tracker for the momentum of charged particles or the ECAL for photon energy [34].

Table 5.1 shows the results from such an explicit reconstruction done by Dr. Maria Cepeda at CIEMAT. Different hadronic decay modes are simulated and reconstructed as a single hadron with varying amounts of photons, three hadrons or as a neutron. Here, it is assumed that the categories with one and two photons are the ρ ($\pi^{\pm}\pi^0$) decay and the categories with three or four photons are a_1 ($\pi^{\pm}2\pi^0$). These results can potentially be improved further by considering the ρ and π^0 masses and the kinematics of the reconstructed photons. There is also a significant amount of misidentification of pions as neutrons (n), which reduces the efficiency.

When comparing these results to the results of the ML approach, the ML achieves higher efficiencies across all categories, except for the previously highlighted $\pi 2\pi^0$. As previously mentioned, the lower efficiency of the $\pi 2\pi^0$ decay may be attributed to the strong similarity to other decays, particularly the $\pi\pi^0$ decay. However, the better performance of the explicit reconstruction approach in this category suggests that the current ML model has not yet reached the physical limit. This indicates room for improvement, potentially through hyperparameter optimization or feature engineering on the technical side, but likely more effectively by training on a larger dataset.

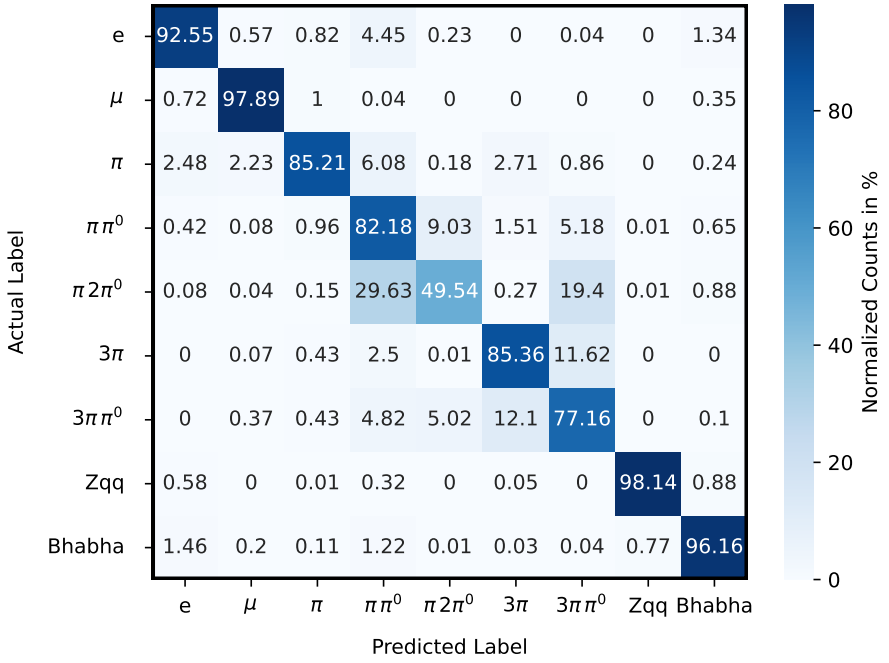


Figure 5.6.: Confusion matrix after doubling the weights of $\pi 2\pi^0$ decays in the loss calculation

5.8. Adaption of Class-Specific Weights for $\pi 2\pi^0$ Optimization

One hyperparameter optimization that was explored to potentially improve the $\pi 2\pi^0$ efficiency was a change to the class-specific weights. Doubling the weight of the $\pi 2\pi^0$ class in the loss calculation aimed to shift the models focus on the correct identification of this decay. With this change, a new model was trained and tested with the otherwise same parameters and datasets as before.

This generates the confusion matrix displayed in Fig. 5.6. It is noticeable, that the results are very similar to the model before in Fig. 5.4, with the $\pi 2\pi^0$ efficiency even slightly worse. This indicates that the model is already capable to capture all necessary patterns, meaning that altering the focus does neither enhance the efficiency for the target decay, nor negatively impact the other classes.

6. Conclusion and Outlook

In this thesis, a machine learning approach to reconstruct tau decay modes is implemented and evaluated. The ML model is based on the GATr architecture, that combines a natural handling of geometric data using a plane based geometric algebra with the efficiency of Transformers. Low-level detector variables simulated in the clean environment of the FCC-ee are processed directly to distinguish between tau decay modes, without relying on particle objects from PF reconstruction.

The training dataset is produced by a full simulation of $Z \rightarrow \tau^+\tau^-$ events at the FCC-ee collider at the center-of-mass energy of 91 GeV, that corresponds to the mass of the Z boson. The CLD detector concept is used for the detector simulation. Background processes are added in the form of $Z \rightarrow q\bar{q}$ events, that often occur in Z boson decays, and Bhabha scattering of the beams.

The model is trained on a dataset consisting of 50,000 signal events and 20,000 and 10,000 background samples of $Z \rightarrow q\bar{q}$ and Bhabha events, respectively. The inputs are only low-level detector variables like the xyz-coordinates, the energy and momentum of the hits as well as the detector part where the hit is registered, without any preprocessing or reconstruction beforehand. The final models are tested on a new dataset of the same size and evaluated based on their confusion matrices, ROC curves and AUC.

The ML approach demonstrates promising results, particularly excelling in classifying leptonic decays and background processes. While it achieves lower, yet still good, efficiencies for the hadronic decay modes, the performance in distinguishing similar classes, especially when the only difference is the presence of a π^0, is less effective. This challenge arises, because the π^0 particles produced in tau lepton decays are highly boosted and therefore produce tightly collimated photons. As a result, the photons leave overlapping hit clusters in the detector, making it difficult to identify individual photons. For further tau polarization studies, the most problematic misclassification rate between the $\rho^\pm$ ($\pi^\pm\pi^0$) and $a_1^\pm$ ($\pi^\pm 2\pi^0$) is not an issue, as the reconstruction method is the same for them. These difficulties are justified from a physics point of view and are also present in traditional approaches like explicit reconstruction. Compared to the results of an explicit reconstruction, the ML approach achieves higher efficiencies in all hadronic decay modes, except for the $\pi 2\pi^0$ class. This suggests that a higher efficiency for this class might be reachable, e.g. through hyperparameter optimization or feature engineering.

The work in this thesis demonstrates the potential of machine learning in tau lepton reconstruction and identification, serving as a proof-of-concept for future applications. The results indicate that ML algorithms, particularly the GATr architecture, can achieve

competitive efficiencies in tau reconstruction. By utilizing low-level detector variables, the model eliminates the need for particle object reconstruction, while still successfully identifying most of the tau decay modes.

Based on the promising result of this thesis, further development can be made. In this thesis, the hit information used for training and testing the model is selected based on the truth information from the MC simulation, ensuring that all hits caused by the decay products of a specific tau are included. However, in a real environment, signals from different sources overlap, making perfect hit assignment impossible. To further validate the models performance, a more realistic hit selection based on clustering and proximity of the hits rather than the truth information could be implemented. Furthermore, applying the training to the standardized EDM4HEP files would facilitate performance comparisons with explicit reconstruction and other ML approaches.

7. Acknowledgments

I would like to thank Prof. Dr. Markus Klute for providing me with this interesting thesis topic and the opportunity to work on it.

Special Thanks to Dr. Xunwu Zuo for the continuous guidance and support throughout the whole process and for always taking the time to discuss my questions and thoughts.

I also want to thank Dr. Dolores Garcia for providing the script that serves as a foundation for this work and Dr. Maria Cepeda for sharing the results for the explicit reconstruction. I'm especially grateful to both of them for taking the time to respond to all the questions I had regarding the materials they shared.

Lastly, many thanks to all the members of the ETP, that were always there to help whenever I had questions.

Bibliography

[1] A. Abada et al. "FCC-ee: The Lepton Collider". In: *The European Physical Journal Special Topics* 228.2 (2019). DOI: 10.1140/epjst/e2019-900045-4. URL: https://doi.org/10.1140/epjst/e2019-900045-4.

[2] CERN. *Machine parameters*. visited on 23.02.2025. URL: https://fcc-ped.web.cern.ch/content/machine-parameters.

[3] Dan Guest, Kyle Cranmer, and Daniel Whiteson. "Deep Learning and Its Application to LHC Physics". In: *Annual Review of Nuclear and Particle Science* 68.1 (Oct. 2018), pp. 161–181. ISSN: 1545-4134. DOI: 10.1146/annurev-nucl-101917-021019. URL: http://dx.doi.org/10.1146/annurev-nucl-101917-021019.

[4] Johann Brehmer et al. *Geometric Algebra Transformer*. 2023. arXiv: 2305.18415 [cs.LG]. URL: https://arxiv.org/abs/2305.18415.

[5] Cush. visited on 16.01.2025. URL: https://commons.wikimedia.org/wiki/File:Standard_Model_of_Elementary_Particles.svg.

[6] Mark Thomson. *Modern Particle Physics*. Cambridge University Press, 2013.

[7] R. Aaij et al. "Observation of $J/\psi p$ Resonances Consistent with Pentaquark States in $\Lambda_b^0 \to J/\psi K^- p$ Decays". In: *Physical Review Letters* 115.7 (Aug. 2015). ISSN: 1079-7114. DOI: 10.1103/physrevlett.115.072001. URL: http://dx.doi.org/10.1103/PhysRevLett.115.072001.

[8] CERN. *The Standard Model*. visited on 13.11.2024. URL: https://www.home.cern/science/physics/standard-model.

[9] S. Navas et al. "Review of particle physics". In: *Phys. Rev. D* 110.3 (2024), p. 030001. DOI: 10.1103/PhysRevD.110.030001.

[10] Izaak Neutelings. *Hadronic tau decay*. URL: https://tikz.net/tau_decay/.

[11] CERN. *Facts and figures about the LHC*. visited on 24.11.2024. URL: https://home.cern/resources/faqs/facts-and-figures-about-lhc.

[12] G. Aad et al. "Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC". In: *Physics Letters B* 716.1 (Sept. 2012), pp. 1–29. ISSN: 0370-2693. DOI: 10.1016/j.physletb.2012.08.020. URL: http://dx.doi.org/10.1016/j.physletb.2012.08.020.

[13] CERN. *Future Circular Collider*. visited on 24.11.2024. URL: https://home.cern/science/accelerators/future-circular-collider.

[14] Mogens Dam. *Tau-lepton Physics at the FCC-ee circular e^+e^- Collider*. 2018. arXiv: 1811.09408 [hep-ex]. URL: https://arxiv.org/abs/1811.09408.

[15] Charu C. Aggarwal. *Neural Networks and Deep Learning. A Textbook*. Springer Cham, 2023. DOI: https://doi.org/10.1007/978-3-031-29642-0.

[16] Andriy Burkov. *The Hundred-Page Machine Learning Book*. Andriy Burkov, 2019.

[17] Glosser.ca. visited on 06.01.2025. URL: https://commons.wikimedia.org/wiki/File:Colored_neural_network.svg.

[18] Spandan Mondal and Luca Mastrolorenzo. "Machine learning in high energy physics: a review of heavy-flavor jet tagging at the LHC". In: *The European Physical Journal Special Topics* 233.15–16 (July 2024), pp. 2657–2686. ISSN: 1951-6401. DOI: 10.1140/epjs/s11734-024-01234-y. URL: http://dx.doi.org/10.1140/epjs/s11734-024-01234-y.

[19] Venkitesh Ayyar et al. "The use of Convolutional Neural Networks for signal-background classification in Particle Physics experiments". In: *EPJ Web of Conferences* 245 (2020). Ed. by C. Doglioni et al., p. 06003. ISSN: 2100-014X. DOI: 10.1051/epjconf/202024506003. URL: http://dx.doi.org/10.1051/epjconf/202024506003.

[20] Bharti Khemani et al. "A review of Graph Neural Networks: Concepts, architectures, techniques, challenges, datasets, applications, and Future Directions". In: *Journal of Big Data* 11 (Jan. 2024). DOI: 10.1186/s40537-023-00876-4.

[21] Jonathan Shlomi, Peter Battaglia, and Jean-Roch Vlimant. "Graph neural networks in particle physics". In: *Machine Learning: Science and Technology* 2.2 (Jan. 2021), p. 021001. ISSN: 2632-2153. DOI: 10.1088/2632-2153/abbf9a. URL: http://dx.doi.org/10.1088/2632-2153/abbf9a.

[22] Peter W. Battaglia et al. *Relational inductive biases, deep learning, and graph networks*. 2018. arXiv: 1806.01261 [cs.LG]. URL: https://arxiv.org/abs/1806.01261.

[23] Christian Bakke Vennerød, Adrian Kjærran, and Erling Stray Bugge. *Long Short-term Memory RNN*. 2021. arXiv: 2105.06756 [cs.LG]. URL: https://arxiv.org/abs/2105.06756.

[24] Ashish Vaswani et al. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL]. URL: https://arxiv.org/abs/1706.03762.

[25] Anthony Lasenby Chris Doran. *Geometric Algebra for Physicists*. Cambridge University Press, 2003.

[26] Leo Dorst. *A Guided Tour to the Plane-Based Geometric Algebra PGA*. Version 1.15. 2020. URL: http://www.geometricalgebra.net/.

[27] V. Daniel Elvira. "Impact of detector simulation in particle physics collider experiments". In: *Physics Reports* 695 (June 2017), pp. 1–54. ISSN: 0370-1573. DOI: 10.1016/j.physrep.2017.06.002. URL: http://dx.doi.org/10.1016/j.physrep.2017.06.002.

[28] Christian Bierlich et al. *A comprehensive guide to the physics and usage of PYTHIA 8.3*. 2022. arXiv: 2203.11601 [hep-ph]. URL: https://arxiv.org/abs/2203.11601.

[29] J. Allison et al. “Recent developments in Geant4”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 835 (2016), pp. 186–225. ISSN: 0168-9002. DOI: https://doi.org/10.1016/j.nima.2016.06.125. URL: https://www.sciencedirect.com/science/article/pii/S0168900216306957.

[30] Akshat Dave, Yongyi Zhao, and Ashok Veeraraghavan. *PANDORA: Polarization-Aided Neural Decomposition Of Radiance*. 2022. arXiv: 2203.13458 [cs.CV]. URL: https://arxiv.org/abs/2203.13458.

[31] Dolores Garcia. *PID_GNN*. as of 10.12.2024. URL: https://github.com/doloresgarcia/PID_GNN.

[32] William Falcon and The PyTorch Lightning team. *PyTorch Lightning*. Version 1.4. Mar. 2019. DOI: 10.5281/zenodo.3828935. URL: https://github.com/Lightning-AI/lightning.

[33] Riku Klén Oona Rainio Jarmo Teuho. “Evaluation metrics and statistical tests for machine learning”. In: *Scientific Reports* (2024).

[34] J. S. Marshall and M. A. Thomson. *The Pandora Particle Flow Algorithm*. 2013. arXiv: 1308.4537 [physics.ins-det]. URL: https://arxiv.org/abs/1308.4537.

A. Appendix

A.1. Unusual Events in the Data Generation

Upon further investigations the high amount of particles turned out to be very low energy photons and electrons. While the hits of the regular events occurred in two main clusters, which is expected from the particles generated in the decay of the two taus, the hits for the unusual events are more evenly spread. Figure A.1 shows the distribution of hits in a 3D scatter plot of one unusual event. Interestingly, it is always the same events, that show this weird behavior. The three events with the most particles in every file with 1000 events that were simulated together, are almost always the event numbers 145, 349 and 463, even with different random seeds used. The unusual events make up less than 1 % of the dataset. It is unclear what caused these events, but they were kept in the training dataset, as the majority of the 3D plots appeared normal.

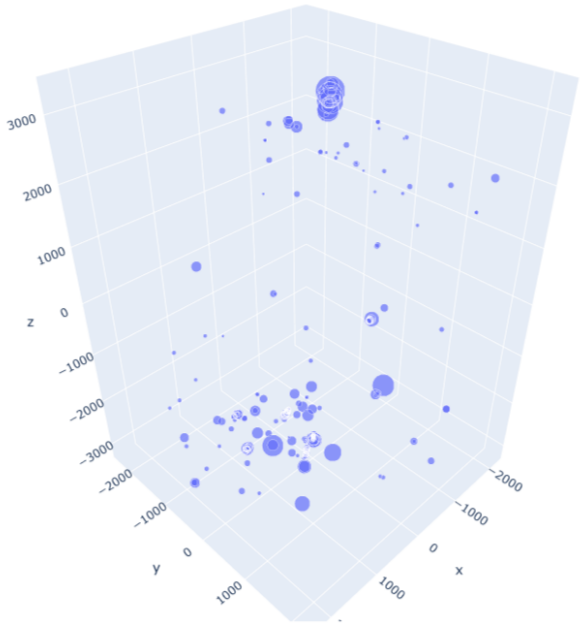
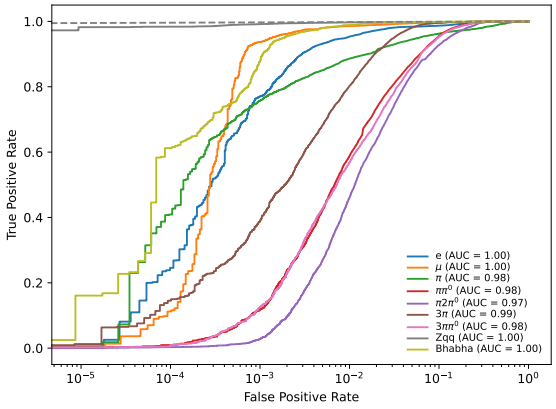
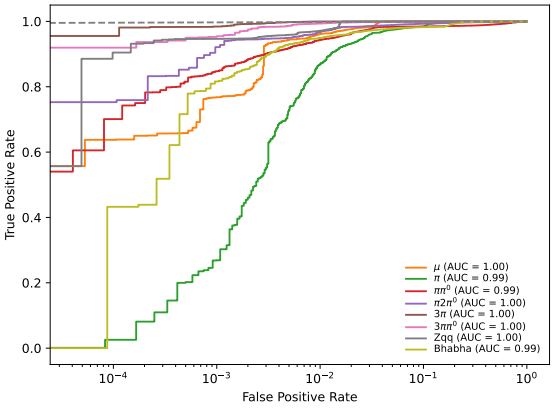


Figure A.1.: 3D plot of an unusual event

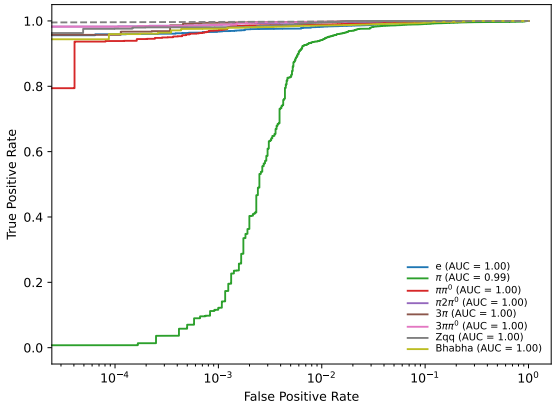
A.2. ROC-Curves for the Final Model



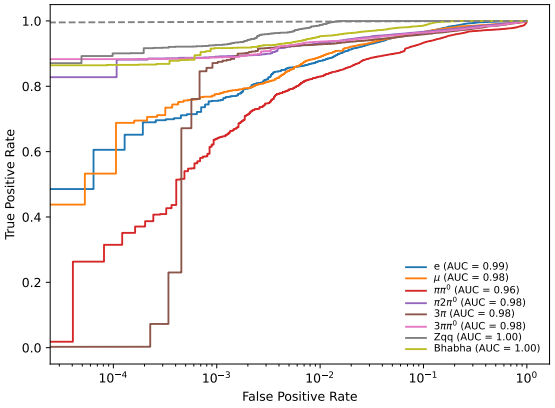
(a) One-vs-all ROC curves



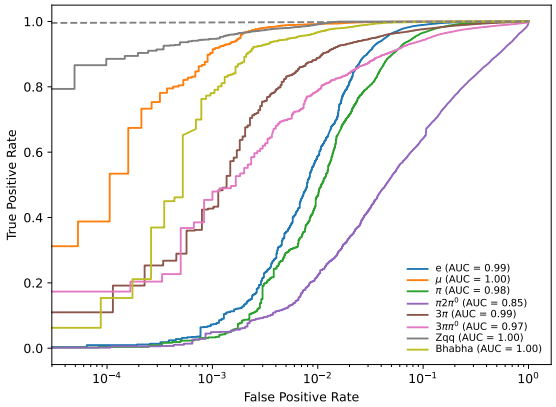
(b) One-vs-one ROC curves for e



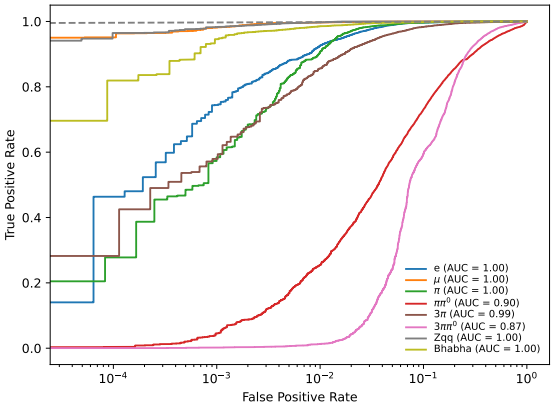
(c) One-vs-one ROC curves for μ



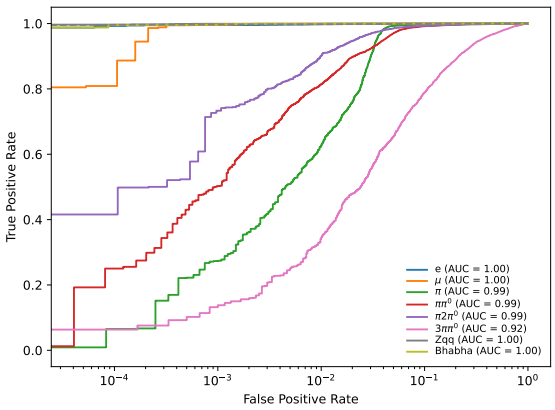
(d) One-vs-one ROC curves for π



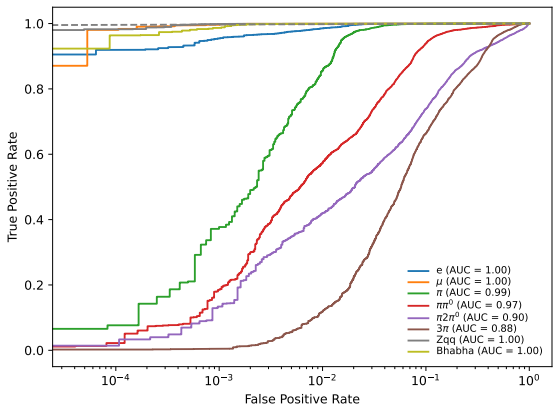
(e) One-vs-one ROC curves for $\pi\pi^0$



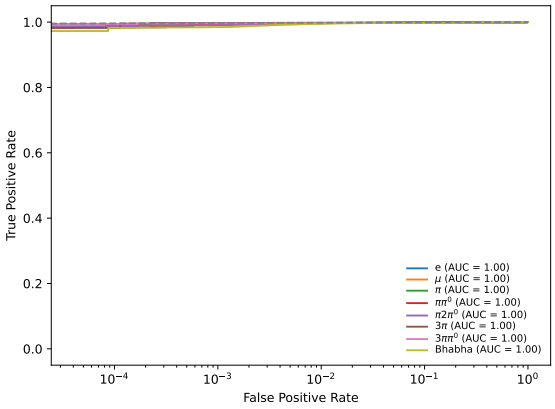
(f) One-vs-one ROC curves for $\pi 2\pi^0$



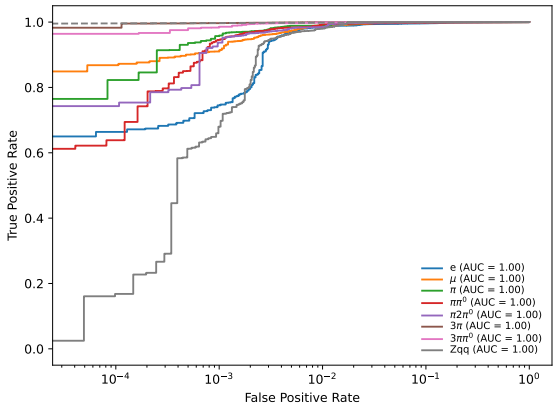
(g) One-vs-one ROC curves for 3π



(h) One-vs-one ROC curves for $3\pi\pi^0$



(i) One-vs-one ROC curves for Zqq



(j) One-vs-one ROC curves for Bhabha

Figure A.2.: All ROC curves for the final model