TECHNISCHE UNIVERSITÄT WIEN

HEPHY INSTITUT FÜR HOCHENERGIEPHYSIK

DIPLOMARBEIT

Identifying Leptons with Dynamic Graph Convolutional Neural Networks at the CMS Experiment

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Physikalische Energie- und Messtechnik

eingereicht von

Andreas GRUBER, BSc
Matrikelnummer: 01622522

ausgeführt am Institut für Hochenergiephysik
der Österreichischen Akademie der Wissenschaften

Betreuung:
Betreuer: Privatdoz. DI Dr. Robert SCHÖFBECK
Mitwirkung: Dr. Suman CHATTERJEE, PhD

Wien, 20. Februar 2023

Unterschrift Verfasser

Unterschrift Betreuer

Abstract

Identifying leptons correctly is an important task in high-energy physics experiments. Leptons can serve as probes for many particles of interest, including the Higgs boson and electroweak vector bosons. To achieve this, the Compact Muon Solenoid (CMS) experiment at the Large Hadron Collider (LHC) is designed to detect muons and electrons generated in proton-proton collisions efficiently.

Usually, lepton identification at the CMS experiment is done by applying a series of conditions on different variables or, more recently, by simple machine-learning-based techniques like boosted decision trees (BDTs). Leptons from different origins are separated by conditions on variables describing the hadronic activity around them. In this thesis, we alter a Dynamic Graph Convolutional Neural Network (DGCNN), originally designed for jet identification, to employ it in identifying leptons in the CMS experiment. This DGCNN draws its strength from employing the so-called EdgeConv operation, which incorporates information from other particles in the vicinity of the lepton with the aim of learning their correlations and developing a measure of distance between the particles in a latent space during training.

We demonstrate that our model can outperform CMS's best current lepton identification methods by almost an order of magnitude. We also analyze the performance dependency on different parameters and calibrate constant signal efficiency. The performance is independent of the number of simultaneous particle collisions, indicating good performance in high-pileup conditions.

Contents

1 Motivation

The Large Hadron Collider (LHC), along with its experiments, is the biggest particle physics experiment ever built [1]. Owing to the sheer size of the experiments, sophisticated readout systems and algorithms are implemented, which allow to reconstruct physical particles from their signatures in different detectors. However, these systems and algorithms are not perfect – a significant fraction of the uncertainties in measurements performed using LHC data stem from uncertainties in particle identification and reconstruction. Therefore, every improvement in particle identification can directly translate into more accurate physics results and, therefore, a deeper understanding of the world we live in.

Traditionally, commonly used algorithms identify particles by using conditions on different variables characterising those. For example, an electron would be identified by matching a track in the inner tracker with a cluster in the electromagnetic calorimeter – if the momentum of the track is consistent with the energy of the cluster, with some margin to account for measurement errors and bremsstrahlung, the particle will very likely indeed be an electron. This approach generally works quite well, as the LHC experiments could announce the finding of the Higgs boson already in 2012. Scientists always seek to improve performance by improving existing algorithms or employing new technologies. One of these new technologies is machine learning (ML), which has boosted technological progress in various fields in the latest years. As the LHC generates vast amounts of data, employing ML technologies in particle physics seems like a natural match. Many different algorithms and techniques have been proposed in recent years, and a good fraction of those manages to outperform classical approaches.

Inspired by these considerations, we attempt to employ a Dynamic Graph Convolutional Neural Network (DGCNN) [2] to improve lepton identification at the Compact Muon Solenoid (CMS) experiment in this thesis. The technical details of the CMS experiment and the particle identification algorithms are discussed in chapter 2, while the basics of ML and the DGCNN are described in chapter 3. Our attempts prove successful – we show that our approach outperforms classical lepton identification techniques in chapter 4. Finally, we conclude this thesis in chapter 5.

2 CERN, LHC and CMS

The European Organization for Nuclear Research (*Conseil européen pour la recherche nucléaire*, CERN), is a research organisation based in Geneva, Switzerland and home to a range of particle accelerators, the largest of which is the LHC. Founded in 1954, CERN has been a driver of cutting-edge technologies and the site of some of the biggest discoveries in particle physics, the latest one being the discovery of the Higgs boson in 2012 [3, 4, 5]. An image of an event where a Higgs boson is produced and decays to a pair of photons is included in Fig. 1. Other significant discoveries include, for example, the direct CP violation in kaon decay [6] and the existence of the W and Z bosons [7, 8]. Furthermore, CERN is famously known as the birthplace of the World Wide Web [9].

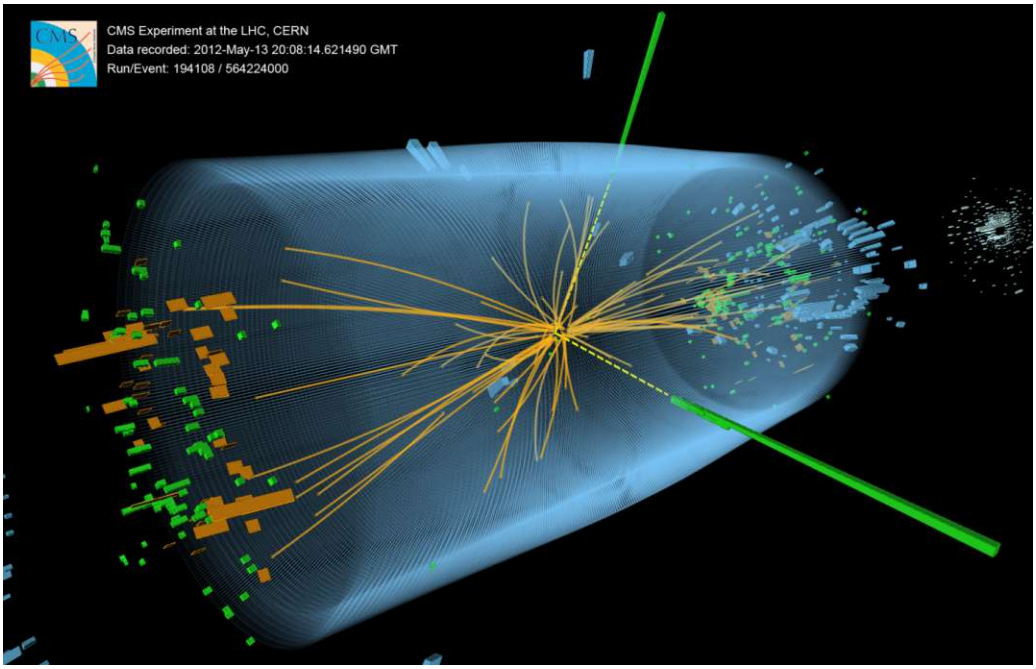


Figure 1: Schematic view of an event recorded with the CMS detector in 2012. The particle decaying to a pair of photons shows properties consistent with a Standard Model Higgs boson. Image taken from Ref. [10].

In 2022, seven accelerators and two decelerators were operated. As the experiments gradually move to use bigger accelerators and therefore higher energies, the older machines are still used as pre-accelerators. Particles are first accelerated in the linear accelerator Linac4 to an energy of 160 MeV and subsequently injected into the Proton Synchrotron Booster (SPB), which accelerates the beam to 1.4 GeV [12] and produces bunches. It is

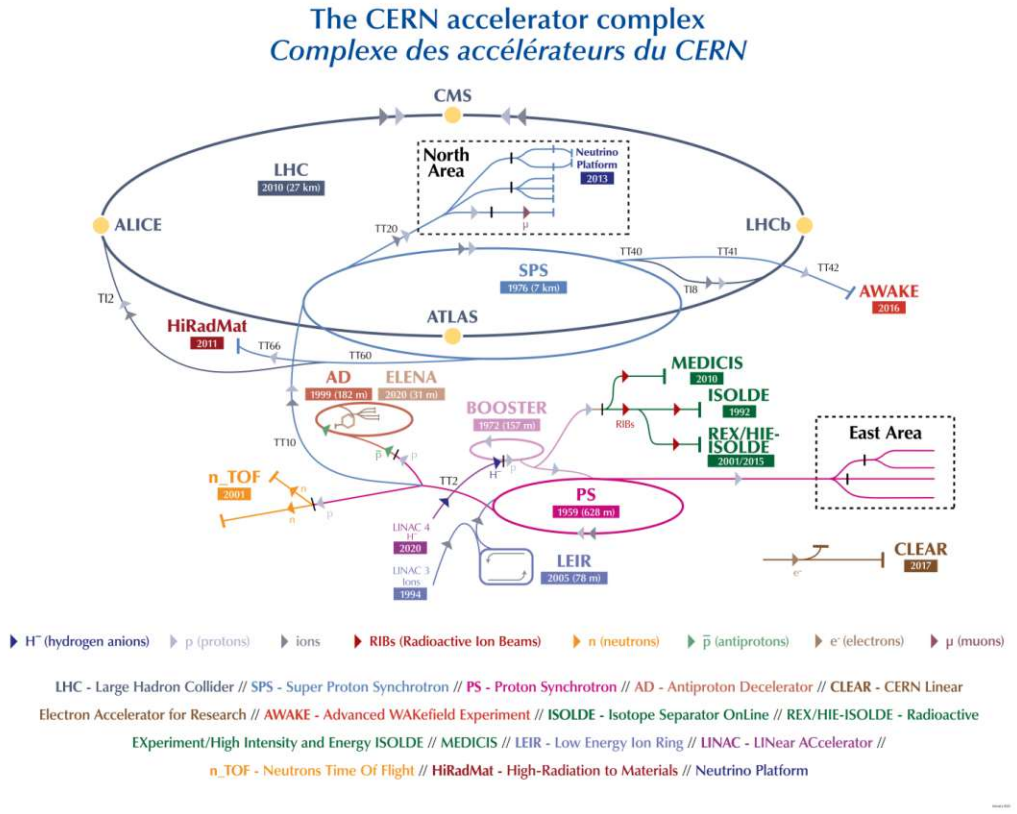


Figure 2: CERN accelerator complex. Image taken from Ref. [11]

then injected in the Proton Synchrotron (PS), which increases the beam energy to 25 GeV. The Super Proton Synchrotron (SPS) then accelerates the protons to an energy of 450 GeV, where they are finally injected into the LHC [13]. The LHC accelerates them to the final energy of 6.8 TeV (at the time of writing this thesis). The whole accelerator complex can be seen in Fig. 2.

Contrary to popular belief, the LHC is actually not perfectly circular – it consists of eight curved segments and eight straight ones, these are the places where the two beams are crossed and the big experiments are situated. Fig. 3 shows this concept.

2.1 The Large Hadron Collider – LHC

A 27 km long synchrotron, the LHC is the largest and most energetic particle accelerator ever built. The centre-of-mass energy in the beam collisions can reach up to 13.6 TeV, which

allows physicists to study the predictions of the Standard Model (SM) of particle physics and measure its parameters in never-before achieved accuracy. To collect the maximum accessible amount of data, equally complex and powerful detectors need to be constructed. At the LHC, there are four big detectors, called experiments:

- **ATLAS** (A Toroidal Lhc ApparatuS) is the largest detector operated at LHC. It is a general-purpose detector with a wide range of physical questions to be investigated, including the search for physics beyond the SM [14].
- **CMS** (Compact Muon Solenoid) is another general-purpose detector with the special requirement of reconstructing muon tracks accurately, as the name suggests. The ATLAS and CMS experiments use different detector and magnetic configurations to ensure measurements and results can be cross-checked in a complementary manner by the other experiment [15].
- **ALICE** (A Large Ion Collider Experiment) is a specialised experiment designed to study heavy ion collisions, which might lead to the formation of a quark-gluon-plasma, a state of matter where quarks and gluons do not need to adhere to colour confinement [16].
- **LHCb** (LHC beauty) is designed to study CP violation and its parameters in heavy flavour hadrons (hadrons which contain a bottom quark) and therefore lead to a better understanding of the matter-antimatter symmetry [17].

Besides the four large main experiments, there also exist several smaller experiments. In 2022 there are currently nine experiments in total - but as science evolves, so does the LHC with its upgrades, so this number is bound to change soon again [18].

One of the most important metrics of a particle accelerator is the instantaneous luminosity, which is the rate of potential encounters of colliding protons. It is given by

$$\mathcal{L} = \frac{N_1 N_2 n f}{4\pi\sigma_x\sigma_y}, \tag{1}$$

where $N_{1,2}$ denote the number of protons per bunch, n is the number of colliding bunches, f is the revolution frequency, and $\sigma_{x,y}$ are the spatial extents of the bunches in the plane transverse to the beam direction.
Why the luminosity is important can be seen by considering the collision rate

$$\frac{dN}{dt} \equiv \sigma \cdot \mathcal{L}, \tag{2}$$

where it is clearly visible that the collision rate is directly proportional to the luminosity. This is why a high luminosity is always a desired target - in fact, the next iteration of LHC

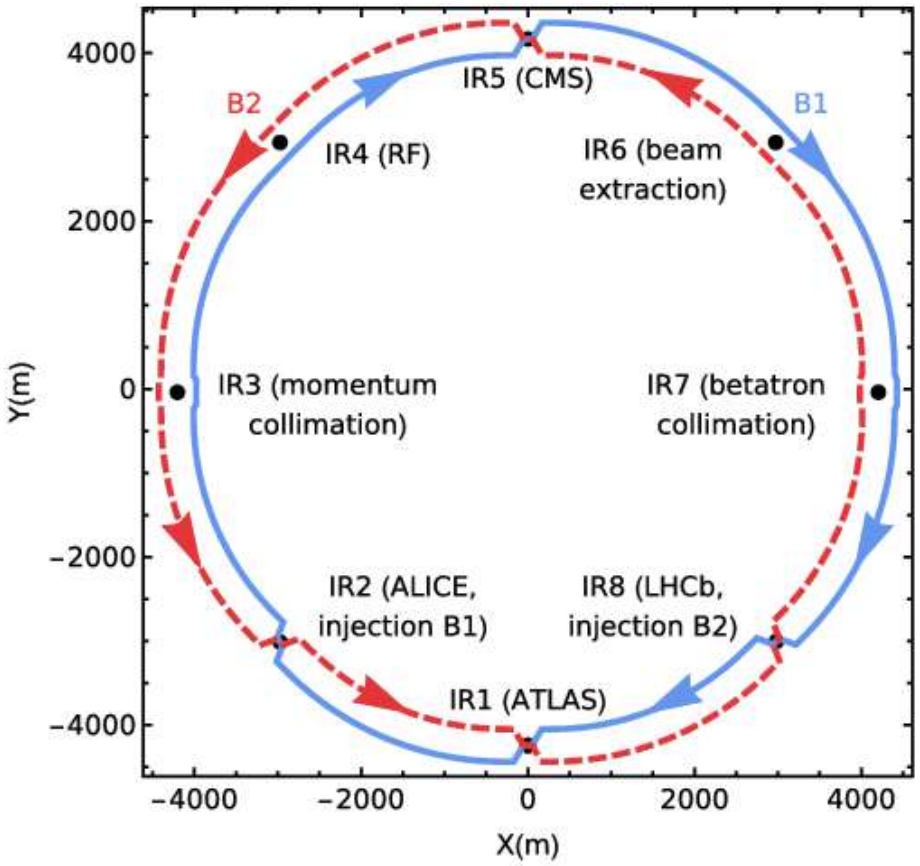


Figure 3: The layout of LHC, showing the interaction points and the location of its biggest experiments. Image is taken from Ref. [19].

running will be called 'High Luminosity LHC' (HL-LHC), where the goal is to increase the instantaneous luminosity by an order of magnitude compared to the LHC's design value of $\mathcal{L} = 10^{34}\,\mathrm{cm}^{-2}\mathrm{s}^{-1}$.

2.2 The Compact Muon Solenoid experiment – CMS

The CMS detector is situated at Insertion Region 5 (IR5 in Fig. 3), 50 m to 100 m underground close to the French town Cessy, just across the border from Geneva. While other large experiments were constructed in situ, CMS was built on ground level in 15 sections before being lowered in a cavern to the level of LHC and reassembled. The whole detector is 21 m long and measures 15 m in diameter. A photo of the CMS detector before

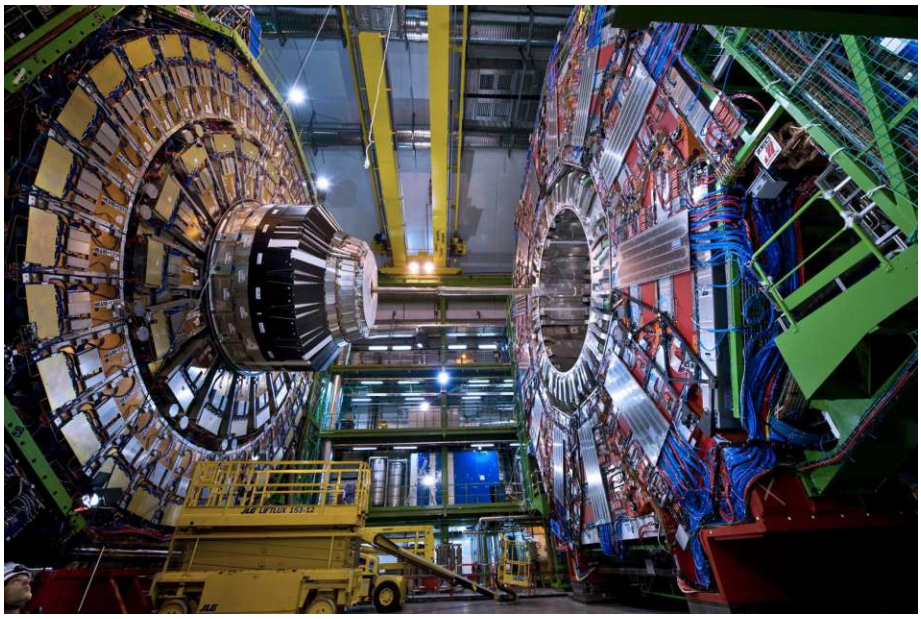


Figure 4: Photo of the CMS detector before it was closed. Taken from Ref. [20].

it was closed is shown in Fig. 4.

Its name comes from the fact that it is quite *compact* for a detector of this enormous capability, that it is designed to detect *muons* very effectively and measure those precisely, and that a strong magnetic field is generated by a superconducting *solenoid*. Muons are interesting for a large number of physical questions, most notably the 'gold plated' Higgs decay channel to four muons, i.e. $h \to Z(\to \mu^+\mu^-)Z^*(\to \mu^+\mu^-)$, as the LHC was constructed primarily to discover the Higgs boson [21]. The solenoid magnet reaches a magnetic field of 3.8 T. This enormous magnetic field is needed for the following reasons: as charged particles experience Lorentz Force

$$\mathbf{F} = q \cdot (\mathbf{v} \times \mathbf{B}) \tag{3}$$

in a magnetic field, the curvature of a particle's path enables the physicist to calculate the momentum of the particle.

The design of CMS is, of course, heavily influenced by the physics goals of the LHC and its respective experiments. For CMS, they are (quoted from Ref. [22]):

- Good muon identification and momentum resolution over a wide range of momenta in the region $|\eta| < 2.5$, good dimuon mass resolution ($\approx 1\,\%$ at $100\,\mathrm{GeV/c^2}$) and the ability to determine unambiguously the charge of muons with $p < 1\,\mathrm{TeV/c}$.

- Good charged particle momentum resolution and reconstruction efficiency in the inner tracker. Efficient triggering and offline tagging of τ 's and b-jets, requiring pixel detectors close to the interaction region.
- Good electromagnetic energy resolution, good diphoton and dielectron mass resolution ($\approx 1\,\%$ at $100\,\mathrm{GeV/c^2}$), wide geometric coverage ($|\eta| < 2.5$), measurement of the direction of photons and/or correct localisation of the primary interaction vertex, π^0 rejection and efficient photon and lepton isolation at high luminosities.
- Good E^T_{miss} and dijet mass resolution, requiring hadron calorimeters with a large hermetic geometric coverage ($|\eta| < 5$) and with fine lateral segmentation ($\Delta\eta \times \Delta\phi < 0.1 \times 0.1$).

2.2.1 The Detector Systems of CMS

CMS is specifically designed to meet these quoted requirements. Its structure can be divided into five distinct layers:

- Tracker
- Electromagnetic Calorimeter
- Hadronic Calorimeter
- Solenoid magnet
- Muon detectors

The innermost region is the tracker, which is responsible for reconstructing charged particle trajectories (commonly referred to as 'tracks') and detecting accurate spatial information of all collision points. To accomplish this, the most sensitive layer can achieve a single point resolution of up to 23 µm [15]. It consists of two types of silicon sensors – the silicon pixels are located in the innermost layers and have to deal with the highest intensity of particles. Silicon microstrips then surround the pixel detectors. With a total detector area of $205\,\mathrm{m^2}$, the CMS tracker is the world's biggest silicon detector, comprised of 13 000 pixel chips with 1.9 billion pixels and 9.3 million microstrip sensors [24]. The data gathered by the tracker is then used to reconstruct the tracks, the primary vertices, where the particle collisions took place, and secondary vertices from the decay of short-lived collision products.

The electromagnetic calorimeter (ECAL) serves to measure the energy of electrons and photons. It is made up of 61 200 niobium doped lead tungstate ($PbWO_4$) crystals, which serve as scintillators. These scintillators emit light when passed through by electrons and photons, while silicon avalanche photodiodes or vacuum phototriodes generate an electrical

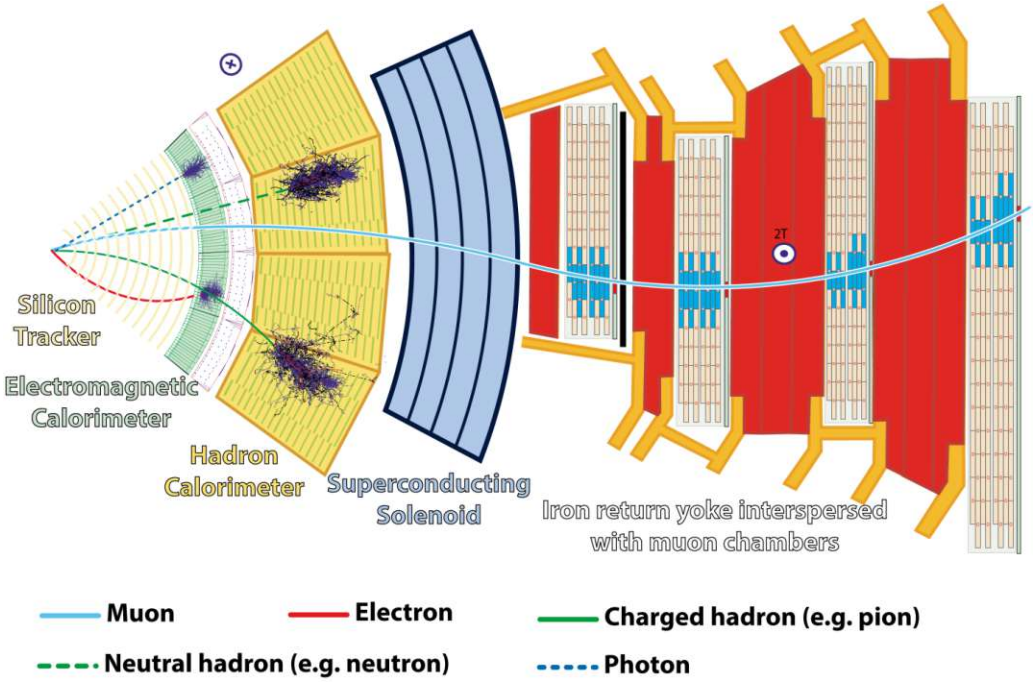


Figure 5: Cut through a slice of the CMS detector on a plane transverse to the beam direction. Image taken from Ref. [23]

signal from these light pulses, which are then further passed to the electronic readout systems.

Next in order is the hadronic calorimeter (HCAL), which serves to detect particles interacting via strong interaction, i.e. hadrons. It is made up of layers of dense material (steel or brass) to induce high interaction rates, interspersed with plastic scintillators that allow for reading out the signals from particle showers. Wavelength-shifting fibres are then used to connect the HCAL to the electronic data-collecting systems consisting of silicon photomultipliers and hybrid photodiodes.

The solenoid magnet is the central part around which the CMS detector was designed and built. The main idea behind using such a powerful magnet can be seen from Eq. 3: Lorentz force is proportional to the magnetic field B, so large B should allow for greater bending of particle's trajectories and, therefore, combined with the highly accurate tracker measurements, allow for efficient and highly accurate measurements of the momenta of charged particles. The superconducting magnet used by CMS is designed to produce a magnetic field of up to 4 T, for which a current of 19 500 A would be needed. This has been scaled down to 3.8 T in operation to maximise longevity. Still, a magnetic field of

3.8 T requires a current of 18 160 A, which equates to a stored energy of 2.3 GJ. To ensure safety during operation, dump circuits are installed to dissipate this energy safely should the magnet quench.

The final component of CMS are the muon detectors which are accompanied by the return yoke. As muons have higher mass than electrons, they can penetrate matter much more deeply and will therefore produce just a small signal in ECAL and HCAL. This is why the muon detectors make up the final layer of the experiment. At the edge of CMS, all other particles have likely been stopped by the calorimeters, so muons are the only particles likely to produce a signal here. To identify muons, CMS uses three distinct technologies:

- The Drift Tube (DT) system detects muons in the barrel part of the detector. A DT is a chamber filled with gas with a positively charged wire in the middle - when a muon passes through, it ionises the gas, electrons knocked out of gas atoms are accelerated ('drifting') towards the wire, creating an electric impulse. CMS features 250 of these DT chambers.
- Cathode strip chambers (CSCs) are used in the endcap region, where the magnetic field is not uniform, and particle density is larger. The working principle is similar to that of the drift tubes, only that there are multiple wires on perpendicular planes, which allow for better position resolution in an environment with high particle multiplicity.
- Resistive plate chambers (RPCs) are used in parallel to the first two detector systems - they allow for very fast readout (faster than the 25 ns frequency of the bunch crossings in LHC). This provides an effective trigger for detecting which bunch crossing the detected muon stems from. The RPCs consist of two parallel highly resistive plates with opposite charges, separated by a thin layer of gas. These plates are transparent to electrons emerging from the ionisation of the gas, which external metallic strips can finally detect. These detectors allow measurements with very good time resolution.

Furthermore, the CMS is divided into two different regions with different requirements: The barrel region, where the detectors are arranged cylindrically around the interaction point (the walls of the CMS 'cylinder'), and the endcap regions, which are located closely around the beam (bottom and top of the 'cylinder'). As particles produced often still carry a significant momentum in the beam direction, particle intensity is much higher in the endcap regions than in the barrel region. An illustration of the CMS detector in the barrel region can be seen in Fig. 5.

As with any experiment in which multiple people are working, a mutual coordinate system has to be adopted. In CMS, this was chosen with the origin at the nominal collision point, shown in Fig. 6. The x-axis points radially inward towards the centre of

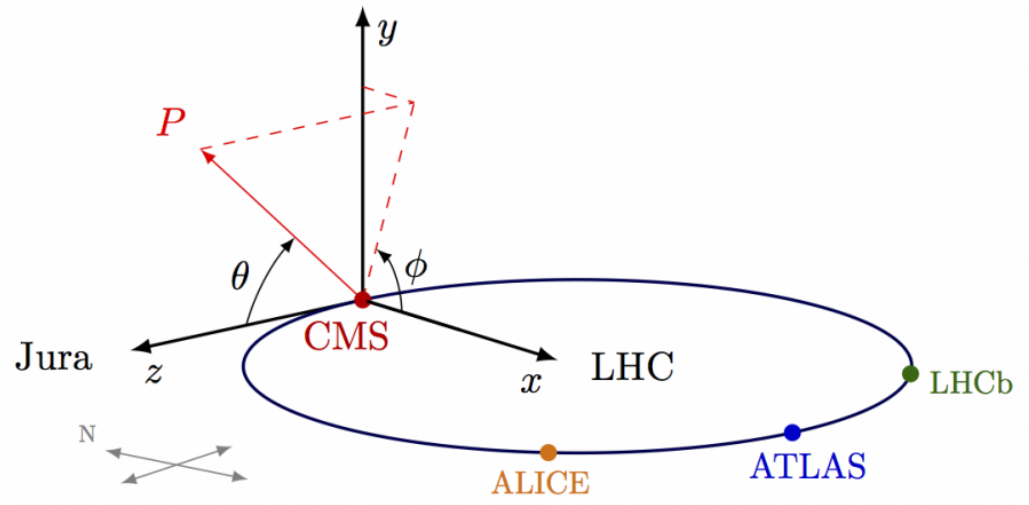


Figure 6: Depiction of the CMS coordinate system, taken from Ref. [25].

LHC, the y-axis upwards, and the z-axis in the general beam direction. The azimuthal angle ϕ is measured from the x-axis, while the polar angle θ is measured from the z-axis. An important variable constructed from θ is the pseudorapidity, which is defined by $\eta = -\ln\tan(\frac{\theta}{2})$ [15]. Another variable often encountered is the transverse momentum p_T - the fraction of the total momentum of a particle in the plane transverse to the beam direction, i.e. the x-y plane.

2.2.2 Trigger

Each collision produces about 1 MB of data [26] in the CMS detectors – with about one billion (10^9) expected proton-proton interactions per second expected at the nominal luminosity, the generated data would be impossible to readout in real-time, let alone be stored for later analyses [27]. Additionally, most events do not involve interesting physics processes, as there will be a lot of 'graze shots' in the proton collisions, which will have too low interaction energy to produce particles and effects of interest. A sophisticated trigger system is used in CMS to sort out these events beforehand. It is implemented as a two-level system – the first level (L1), comprised of custom hardware processors, uses information from the calorimeters and muon detectors to select events at a rate of around 100 kHz. The second level, known as the high-level trigger (HLT), consists of a farm of commercial processors running a version of the full event reconstruction software optimised for fast processing that reduces the event rate to around 1 kHz before data storage. The L1 trigger logic is mostly implemented in Application Specific Integrated Circuits (ASICs) and Field Programmable Gate Arrays (FPGAs). These units have to store and process the

information and give a trigger signal in a defined timeframe of 3.2 µs. Interestingly, due to these time constraints, the L1 trigger can only use signals from the calorimeters and muon systems but no data from the tracker [28].

Contrary to L1 with its home-grown custom data processing systems, HLT employs a large number of commercially available processors in a big computing farm. This allows for scaling up the system as the costs of processors and computing power will probably further sink in the future. It also ensures full flexibility to change the configuration of the Data Acquisition System (DAQ), should the requirements for the DAQ change or new technological developments open up new possibilities to enhance performance even further. The HLT is designed to reduce the output rate further by a factor of ~ 1000, which yields the final output rate of 100 Hz to 1 kHz, which is a manageable load for offline storage.

2.2.3 Reconstruction

The data which is finally stored offline needs to be treated further to allow for analysis to be done – this step is called reconstruction. In CMS, it is done using the particle-flow algorithm [29]. It uses a holistic approach by combining the information of all detector components, while in previous approaches, each subdetector would use its own signals for analysis. The PF algorithm aims to reconstruct muons, electrons, neutral hadrons, charged hadrons and photons.

As CMS was designed to measure muons with great accuracy, muon reconstruction is the 'easiest' part of the PF algorithm. Tracks from the inner tracker and hits in the muon system are reconstructed independently at first [30]. As for all charged particles, the track in the inner tracker is reconstructed using a Kalman-Filtering (KF) algorithm. This approach gives rise to three different categories of muons:

- Standalone muons are constructed using only signals from the muon detectors.
- To build tracker muons, all tracks reconstructed using only information from the tracker with a p_T larger than 0.5 GeV and total momentum p larger than 2.5 GeV are extrapolated to the muon system. If a signal from at least one muon segment matches the extrapolated track, the particle will be classified as a tracker muon.
- Global muons are basically constructed in reverse fashion as the tracker muons ('outside-in') – each standalone muon track is matched to a tracker track. This approach offers better efficiency for high p_T (>200 GeV) muons.

Owing to the specialized design of CMS, usually more than 99% of muons can be reconstructed as global or tracker muon, the majority even as both [29].

Electron reconstruction is somewhat more challenging, as electrons, owing to their lower mass, lose a sizable portion of their energy as bremsstrahlung – up to 86% of the energy can be lost before reaching the ECAL [31]. To allow for accurate measurements, it is therefore critical to detect the bremsstrahlung photons and attribute them to the corresponding electron. As was the case for muons, tracks in the tracker and ECAL energy clusters are built separately from each other and combined in a later step. For cluster building in the barrel region of the ECAL, at first, a seed crystal is identified. This is the crystal which detects the largest amount of energy in excess of a threshold $E_{T,seed} > E_{T,seed}^{min}$. Afterwards, arrays of 5×1 crystals in $\eta \times \phi$ are added N_{steps} times in both directions of ϕ until they meet the energy threshold E_{array}^{min}. Finally, they are connected into a so-called supercluster, a weighted average of all the previously reconstructed clusters. This superclustering leads to better and more accurate energy resolution than using readout clusters of a fixed size [31]. In the endcap regions, a slightly different algorithm is used owing to the altered geometric conditions. During track reconstruction, as for all charged particles, the KF algorithm is applied. However, this can produce suboptimal results for electrons, as the KF approach does not account for changes in curvature due to bremsstrahlung. The KF tracks which do not meet certain criteria, indicated by an unusually poor fit result, are revisited by a dedicated Gaussian-Sum Filter (GSF) algorithm, tailored to account for the energy losses through bremsstrahlung [32]. However, this algorithm requires higher computing time, which is why this algorithm is not deployed by default.

Photons have similar detector signatures as electrons, but as they are uncharged, they will not induce a signal in the tracker. Therefore, hits in the ECAL, which could indicate both electrons and photons, with no matching inner track will be reconstructed as isolated photons. As hadrons can also leave signals already in the ECAL, the ratio between ECAL and HCAL energy must be similar to what is expected for an isolated photon shower.

Hadrons are identified by energy clusters in the HCAL - at first, all HCAL clusters, which have not yet been used in reconstructing another particle, give rise to a hadron. When a matching inner track can be identified, it is identified as charged hadron, otherwise, the algorithm will detect a neutral hadron [29]. Furthermore, if there is an energy excess in an HCAL cluster over the energy of the matched track, this will give rise to an additional neutral hadron. No attempt is made to separate the specific types of hadrons further than that.

Jets are showers of particles in a narrow cone. Particles produced in the proton-proton collisions are likely to be unstable and will further decay. All of those decay products will carry a fraction of the initial momentum. They will therefore be moving in similar directions, creating jets – a large number of particles near each other.

2.2.4 Simulation

Another important part is the simulation of the collisions and the behaviour of the detectors as they are hit by the created particles. It is an essential tool for interpreting the data collected by the CMS experiment and making new discoveries. This is discussed extensively in Ref. [33].

According to Ref. [34], simulation can generally be described in the following steps:

- Initial State: Inside the colliding protons, the momentum distributions of the constituent quarks and gluons, commonly referred to as partons, are described by parton density functions (PDF).
- Hard scatter: A probabilistic distribution of the particles produced in the parton-parton collisions is calculated using perturbation theory.
- Parton shower: The scattered quarks and gluons radiate secondary quarks and gluons, which are coloured particles themselves and can therefore further emit quarks and gluons.
- Hadronization: While the quarks and gluons lose energy, the strong interaction becomes finally strong enough for these particles to combine to form colourless hadrons.
- Underlying Event: During the collision of proton bunches, also other partonic collisions involving low-energy transfer happen apart from the hard scattering process; these processes produce soft hadrons all over the event.
- Unstable particle decay: A significant fraction of the hadrons produced in hadronisation are unstable and decay after a short time.

Commonly used general-purpose event generators include, e.g. MADGRAPH5 aMC@NLO [35] and POWHEG [36]. Both of these generators were used in this thesis to generate the hard interaction, while the subsequent steps were simulated using PYTHIA8 [37]. The interaction of the generated particles with the detectors is simulated using the GEANT4 software package [38].

3 Machine Learning, Neural Networks and ParticleNet

Since the invention of mechanised calculators and eventually computers, scientists have theorised about ML, or as it is better known in public by the more sensationalist term Artificial Intelligence (AI). At the beginning an almost entirely theoretical field, several developments have driven the theory and practical use of ML since the beginning of the twenty-first century and make it now one of the most rapidly advancing fields of modern science. According to [39], the first works and models emerged as soon as 1957. However, computing power was rare in those days, and interest in the field declined, followed by increasing interest again in the 1990s. However, funding and expectations soon declined again – these two drops in interest are known as the *Winters of AI*. The *Gold Rush of ML* emerged in the first decade of the 21st century due to three at first glance independent developments [39]:

- Big Data: never before known amounts of data were available in machine-readable data formats – a prerequisite to enable powerful ML technologies.
- Computational progress: The ongoing development of ever more powerful Graphics Processing Units (GPUs), as well as the accompanying software to distribute huge amounts of data among several processing nodes.
- Algorithms: The development of new ML algorithms, in particular Deep Learning (DL) and Deep Neural Networks (DNNs).

These developments have enabled ML applications to outperform traditional algorithms and even humans in many tasks. For example, in 2015 AlphaGo, an ML software tailored to play the traditional board game Go, beat a human for the first time using ML techniques – an achievement thought to be at least a decade away for traditional algorithms [40]. Nowadays, ML has become so important that large IT companies like Meta, Alphabet and Amazon (each ranking under the top 10 most valuable companies worldwide [41]) are investing billions of dollars each year in the technology.

ML can be divided into three different categories:

- Supervised Learning: The training data is labelled with truth labels for each input, which the algorithm can see. Training optimises the internal parameters of the model through backpropagation by minimising a given figure of merit, commonly referred to as loss function. Famous examples employing supervised learning are Support Vector Machines [42] and Neural Networks (NNs) [43].
- Unsupervised Learning: In popular culture known as *Netflix algorithm*, unsupervised learning does not have access to truth labels of the input data. It works by clustering data points together which have similarities, and should therefore provide new information about the structure of the data. These techniques are commonly employed

in social media feed algorithms, which should deliver new content which the end user will enjoy. Another field would be fraud detection, where unsupervised learning can detect anomalies in the data. Widely used algorithms include k-means clustering [44], principal component analysis [45] and autoencoders [46].

- Reinforcement learning: Here, the model is not allocated a predefined amount of data to perform its training on but is placed in an unknown, potentially complex environment. By maximizing a figure of merit called *reward*, the algorithm uses trial and error to achieve the programmer's goal. A common use case for reinforcement learning is the development of self-driving cars.

3.1 Introduction to Neural Networks

A NN is a type of ML model designed to mimic the human brain, connecting thousands of artificial neurons. Of course, the human brain can not be recreated artificially (yet), but NNs have proven quite powerful when trained on large amounts of data. Commonly, NNs have such good performance that they are quite prone to overfitting, i.e. learning arbitrary features of the training data that can not be generalized. A lot of optimisation work on NNs therefore goes into preventing overfitting.

Most ML models employ some kind of convolution operation, thus they are denoted convolutional NNs (CNNs). The convolution serves to reduce layer sizes and make the network slimmer, therefore saving computational cost. A convolutional network only connects neighbouring layers; the contrary would be a fully connected network, where each node is connected to each other node. These layers are discussed more intensively in Sec. 3.5.

A dynamic graph algorithm is one where the data it operates on is updated while running the algorithm. This is a widely used technology in both classical algorithms and novel ML applications.

The characteristics listed above form the basis of ParticleNet – a Dynamic Graph Convolutional Neural Network (DGCNN) developed by H. Qu and L. Gouskos (both at CERN) to employ the newest ML research in particle physics. In this case, it is designed for jet tagging in all LHC experiments. It is written in Python and built upon the PyTorch library, an open-source Deep Learning framework [47]. The ParticleNet NN constitutes the basis of the ParticleNetLepton algorithm presented in this thesis.

The ParticleNet algorithm aims to combine two before independent approaches: the image-based representation and the particle-based representation. The image-based approach maps particle hits onto a model of the detector, generating a 3-dimensional image of a jet with no data loss - however, as there are quite few hits per event in comparison

to the detector size, the vast majority of detector cells will be blank, thus wasting a lot of computing resources. Examples of this approach are plentiful and show comparable performance to cut-based approaches using a sequence of simple conditions, e.g. the ones used in Refs. [48, 49].

The particle-based approach aims to use computing resources more efficiently by treating jets as packages of their constituent particles. This is a much more natural representation of a jet. The most advanced of these algorithms is DeepJet [50] - the DeepJet algorithm already was the foundation of several theses in the CMS analysis group of HEPHY trying to employ the capability of DNNs for lepton identification, leading to the DeepLepton algorithm [51]. It shows quite good performance, outperforming traditional cut-based methods. However, this approach also has disadvantages: as the particles of the input data possess no intrinsic ordering, an arbitrary, external structure needs to be imposed onto the data, e.g. ordering of jet constituents by decreasing p_T. The model could learn this structure, so it would learn features that the data does not possess, therefore impairing performance. The ParticleNet algorithm aims to combine the advantages of these by employing the EdgeConvolution operation, discussed in more detail in chapter 3.6.1, first described in Ref. [52]. It is designed to be a convolution-like operation on point clouds, allowing the network to develop its own measure of distance between data points and update it during training.

3.2 Choice of Loss Function: Cross Entropy Loss

The so-called Loss function is of great importance in giving a NN (or any ML model) its performance. It is a measure of the difference between the current prediction of the model and the true labels of the training data - the NN 'learns' through minimizing the loss function. During training, the gradient of the loss function shows the direction (see also section 3.3) in which the parameters have to be tuned to deliver better prediction results. Without a loss function, a NN would just be a computationally expensive random number generator - parameter values would just float around arbitrarily.

In this thesis, we use the *Cross Entropy Loss*, which is a widely used choice for NNs. It is defined by

$$\ell(x, y) = \frac{\sum_{n=1}^{N} l_n}{N} \qquad l_n = -\sum_{c=1}^{C} w_c \ln(p(x_{n,c}))y_{n,c}, \tag{4}$$

where C denotes the number of classes, x is the prediction of the network, y are true values, n is the number of samples in the current mini-batch, w is the weight and p is a probability distribution – in our case the Softmax output distribution described in Sec. 3.5.5.

3.3 Optimiser: Ranger

The optimiser is the other important part in giving the NN its performance, so to speak, the counterpart of the loss function. It operates in the feature space - the space spanned by all tunable parameters. The optimiser updates the parameters of the network in the direction of the inverse gradient of the loss function in the feature space - it *optimises* the parameters towards minimisation of the loss function. This finally allows the network to learn during training.

A simple example of an optimiser is Gradient Descent (GD) - during each training iteration, GD takes a predefined step in the gradient direction. In recent years, a vast amount of different optimisers have been proposed, each with its own strengths and shortcomings. They are interspersed by combined approaches, which merge multiple optimizers with the goal of maximising the strengths and minimising the shortcomings. A popular example of this is Adam, derived from *Adaptive moment estimation* [53], which utilises the first and second moments of the loss function. Though technically different, it aims to combine the advantages of AdaGrad [54] and RMSProp (Tieleman & Hinton, unpublished). To date, Adam is still quite popular - it has been the most popular optimiser choice for several years [55].

Another optimiser following this evolution is used in this thesis, called Ranger [56]. It aims to combine RAdam (*Rectified Adam*, a version of Adam employing a rectifier to rule out excessively large variances) and LookAhead [57]. The LookAhead optimiser is designed to *look ahead* by simultaneously keeping two sets of weights, the *slow* and *fast weights*. The slow weights are updated in the direction of the fast weights, which propagate faster than the slow ones every iteration, and after k steps, the fast weights are reset to the value of the slow weights. The authors of LookAhead describe their approach as somewhat perpendicular to that of Adam [57], so combining them seems quite promising. As described in Ref. [58], the authors achieved better performance without much hyperparameter optimisation through the combined optimisers in Ranger than in each one individually.

An essential property of the optimiser is the *learning rate* (LR) – it is a free parameter chosen before training by the supervisor. The LR specifies how far each training step should move in the direction of the gradient.

3.4 Backpropagation

The algorithm through which all the nodes and weights get updated is called *Backpropagation*. It is essentially an excessive use of the chain rule in differentiation. One can find several independent sources of this algorithm; the term 'Backpropagation' was first coined in Ref. [59]. A full derivation from first principles can be found in various sources,

e.g. in Ref. [60]. As an example, we will have a look at the partial derivative of the loss function E with respect to a weight w_{ij}^k connecting the nodes i in layer $k-1$ and node j in layer k, while a_j^k denotes the node k in the output layer j

$$\frac{\partial E}{\partial w_{ij}^k} = \frac{\partial E}{\partial a_j^k} \frac{\partial a_j^k}{\partial w_{ij}^k}. \tag{5}$$

We can see that we can split up the calculation of the total gradient into a set of (computationally easier) partial derivatives at each node of the network. The first term of Eq. 5 is usually abbreviated to

$$\delta_j^k \equiv \frac{\partial E}{\partial a_j^k}.$$

Finally, if we include all nodes in all layers, we arrive at

$$\frac{\partial E}{\partial w_{ij}^k} = \delta_j^k o_i^{k-1} = g'(a_j^k) o_i^{k-1} \sum_{l=1}^{r^{k+1}} w_{jl}^{k+1} \delta_l^{k+1}, \tag{6}$$

where g is the activation function of the output layer a, and o_i^k is the output for node i in layer l_k.
In Eq. 6, we can finally see where the name comes from: δ_j^k at layer k is dependent on the errors δ_j^{k+1} at the next layer $k+1$ – errors *propagate backwards*, from the last layer to the first.

3.5 Layer Types

A NN is constructed by adding together multiple layers with specific functions, each layer consisting of several nodes – in general, this consists of an input layer containing the features of the input data, followed by the so-called hidden layers and the final output layer producing the human-readable output. A node in a given layer is connected to each node in the previous and the following layer. These connections are each assigned a weighting parameter - these so-called weights are the learnable parameters of the model, they get updated in each training iteration. The simplest method of adjusting the weights is gradient descent, an iterative minimisation algorithm. Usually, one training iteration is not done using the whole training data set but uses a smaller amount of data to minimise computational cost. The amount of training data used in one training iteration is referred to as *mini-batch*. An image of the ParticleNetLepton structure in this thesis is shown in Fig. 8; the specific layer types will be discussed in the following. All the layers are defined in the PyTorch package `torch.nn`. Below, the layer types used in the ParticleNetLepton architecture are described.

3.5.1 Linear

Layer that connects input and output by a simple linear transformation

$$y = xA^T + b, \tag{7}$$

where y denotes output features, x denotes input features, and A and b contain learnable parameters.

3.5.2 Batch Normalization

This layer performs a Batch normalization of the current mini-batch. It is defined by

$$y = \frac{x - \mathrm{E}[x]}{\sqrt{\mathrm{Var}[x] + \epsilon}} \cdot \gamma + \beta, \tag{8}$$

where again y denotes output features and x denotes input features. The variance and mean are calculated over each mini-batch, while γ and β are the learnable features of this layer.

According to Ref. [61], the Batch Normalization should reduce the internal covariance shift, which refers to the different distributions of the inputs to the layer. This is expected to make the training more robust, allowing for higher learning rates and less demand for careful parameter initialization. In the test NN used in Ref. [61], a significantly lower number of training steps were needed to reach the same level of performance as without Batch Normalization.

3.5.3 Rectified Linear Unit – ReLU

The Rectified Linear Unit (ReLU) is defined as

$$f(x) = x^+ \equiv \max(0, x). \tag{9}$$

It is an activation function, defined as the positive part of its input.

Activation functions perform an essential role in NNs: only non-linear activation functions allow NNs to perform non-trivial tasks - if only the identity activation function were used (equivalent to just using no activation function), the NN would just be one big series of vector-matrix multiplications. This would make the NN ‘collapse’ to just one big linear transformation [62]. Although many different hand-crafted activation functions have been proposed, ReLU is still the most widely used due to its simplicity and good performance [63].

3.5.4 Dropout

As already stated earlier, NNs are prone to overfitting, and due to this, the risk of impairing performance is quite high. A simple yet quite performant regularization technique to prevent this overfitting effect is shown in Ref. [64]: during training, random elements, according to a Bernoulli distribution, of the input tensor will be zeroed ('dropped'). This way, one is effectively training many slimmer networks and averaging each of those predictions. This technique was shown to improve performance but would come at a high computational cost if done manually.

3.5.5 Output layer: Softmax

The Softmax layer acts as the final output activation layer, assigning probabilities with which it assesses the test data to belong to one of the output classes. It was proposed in Refs. [65, 66] and is widely used in NNs today. It is a normalized exponential function defined as

$$\text{Softmax}(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}, \tag{10}$$

where x_i denotes a given output class, while x_j is summed over all output classes.
This convention ensures that the output values lie in the interval $[0, 1]$ as well as sum to 1 in all cases, allowing it to be interpreted as probabilities.

3.6 ParticleNet

3.6.1 The EdgeConv Operation

At the heart of the ParticleNet algorithm lies the EdgeConvolution operation. It was developed for exploiting spatial features of point clouds, nowadays used successfully, e.g., in 3D image recognition [52].

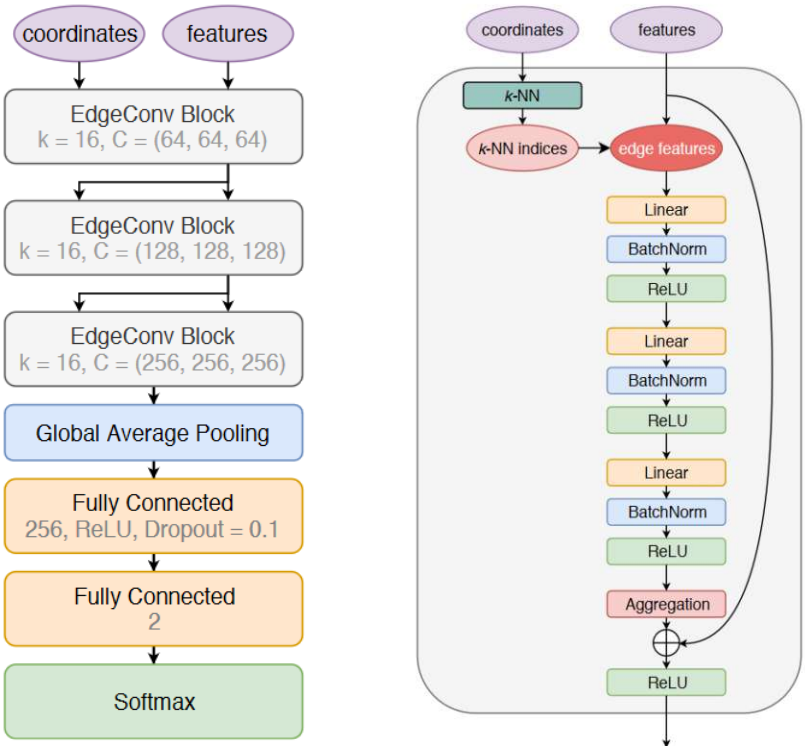
The EdgeConv operation in its general form is given by

$$\mathbf{x}'_i = \square_{j=1}^{k} \mathbf{h}_\theta(\mathbf{x}_i, \mathbf{x}_{ij}), \tag{11}$$

where $\mathbf{x}_i \in \mathbb{R}^F$ denotes the feature vector of the point x_i and $\{i_1, \ldots, i_k\}$ are the indices of the k nearest neighboring points of x_i. As described in Ref. [52], a special form of the edge function,

$$\mathbf{h}_\theta(\mathbf{x}_i, \mathbf{x}_{i_j}) = \overline{\mathbf{h}}_\theta(\mathbf{x}_i, \mathbf{x}_i - \mathbf{x}_{i_j}), \tag{12}$$

where the feature vectors $\mathbf{x}_{i_j}$ of the k nearest neighbours are replaced by their deviation from $\mathbf{x}_i$, is implemented. This choice should allow the EdgeConv to learn a well balanced fraction of local and global features. Ref. [52] also describes other choices, which would



(a) (b)

Figure 7: (a) The original ParticleNet network structure (b) Structure of the EdgeConv operation. Both taken from Ref. [2].

allow to achieve other proportions of local to global features. The edge function $\mathbf{h}_\theta$ is a function specifying the relationship between points and is parameterized by the learnable parameters θ – in this case, it is implemented as a multilayer perceptron (MLP). Finally, $\square$ is a channel-wise symmetric aggregation operation; for ParticleNet, the mean (i.e. $\frac{1}{k}\sum$) was chosen, as this choice achieved better performance than the max operation used in Ref. [52]. The parameters θ of the edge function are shared for all points in the point cloud, these are the learnable parameters of the network.

An important feature of the EdgeConv operation is that it can be stacked; it basically gives a mapping for points from one point cloud to another, where the number of points is constant. The only dimension that could be altered is the dimension of the feature vector of each point. This property allows using several consecutive EdgeConvs, allowing the model to extract features hierarchically – local features should be learned in the first instance of EdgeConv, while more global features should be learned in the later stages by moving to a higher dimension.

3.6.2 The ParticleNet architecture

The original ParticleNet architecture is shown in Fig. 7. It makes extensive use of the EdgeConv operation. The first EdgeConv block starts with determining the k nearest neighbours of an input particle using the 'coordinates' input. Afterwards, the 'features' inputs of these k neighbours are used to construct the 'edge features', which are the inputs of the EdgeConv operation. Each EdgeConv block needs to be defined by two hyperparameters: the number of nearest neighbours k and the number of channels C, which defines the number of nodes in each linear layer. One could also think of C as the dimension of the feature space in this block.

As shown in Fig. 7, k is set to 16 for all three blocks. Each EdgeConv block consists of three layers, which could have different sizes. Here a configuration where all the layers in one block have the same size is used: C is set to (64, 64, 64) for the first EdgeConv block, (128, 128, 128) for the second and (256, 256, 256) for the final third block. After the last EdgeConv block, a global pooling layer is applied, followed by a fully connected layer with a size of 256 nodes. Again, a ReLU activation is applied, while the dropout layer should inhibit overfitting. The final fully connected layer with two nodes and the Softmax activation delivers the final binary output.

3.7 ParticleNetLepton - employing ParticleNet for Lepton Identification

During this thesis, we altered the structure of ParticleNet to employ it to identify leptons in the CMS experiment. The basic idea behind this was to split the input data.

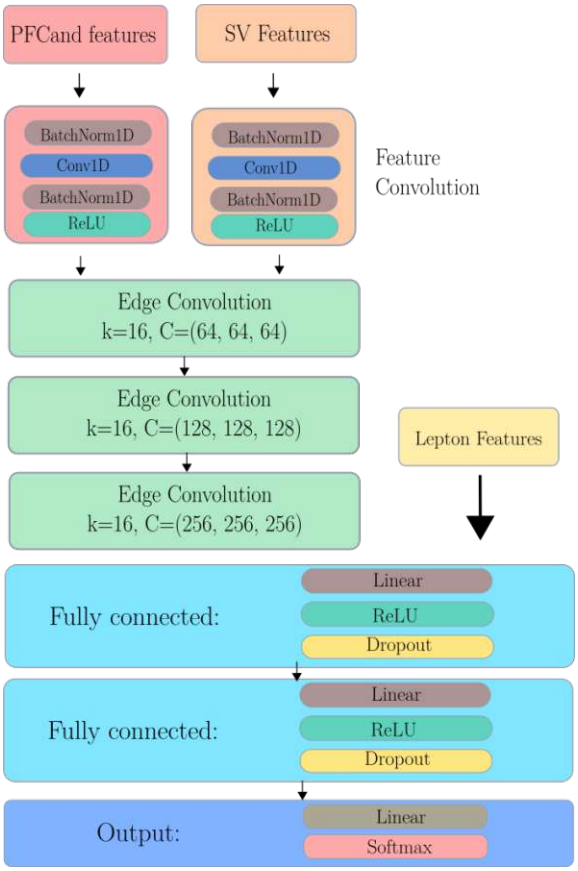


Figure 8: The structure of the ParticleNetLepton network.

While we feed the features of the PF candidates around the lepton (given as vectors) into the EdgeConv operations, the distinct lepton features (given as scalars) should be entered at the beginning of the fully connected layers, as can be seen in Fig. 8. The structure of the EdgeConv blocks was unchanged, but we added a second fully connected layer to ensure the network is able to learn all of the added lepton features

As already mentioned, this approach of splitting up the inputs to the NN is not new – it has already been employed successfully in DeepLepton [51]. The new approach in this thesis is the use of the DGCNN, therefore treating the input data as point clouds.

3.8 Training the Network

The training was performed using the *CLIP Batch Environment* (CBE) at the *Cloud Infrastructure Platform* (CLIP) of the Austrian Academy of Sciences (ÖAW). It is used by a broad range of disciplines at the ÖAW. It features several specific services, such as high-performance computing and high-memory nodes, and the most important ingredients for training NNs, nodes containing GPUs. Currently, there are four different nodes comprising high-performance GPUs by NVIDIA from the Pascal, Tesla, Volta and Ampere class. The most recent and performant Ampere class GPU nodes feature up to 320 GB of total RAM and up to 40 GB of GPU RAM.

During this thesis, all of these separate GPU nodes were tested, but none had significantly better performances than the others. This implies that the bottleneck for training NNs on CBE is not the computational power of the GPU but rather the loading of data from the data storage.

3.8.1 Hyperparameter Optimization

An important aspect in training NNs is hyperparameter optimization - most notably optimization of the LR as discussed in Sec. 3.3. A vivid way of visualising the LR is how much the network should get updated during each training epoch. As in every epoch only a limited amount of data is fed into the network (not the whole dataset), choosing the LR too big implies that the network only 'remembers' information of the current training batch. Generally speaking, one has to perform a trade-off when setting the LR – choosing LR too small leads to slow convergence, choosing LR too big could lead to missing a global minimum and degrade performance.

The PyTorch software framework features a method called `lr-finder`, which performs a test run of the network calculating loss over a range of LR values with a set number of mini-batches. An image of that can be found in Fig. 9. An important note on the `lr-finder`: the best LR to start training is not found at the minimum loss, but at the

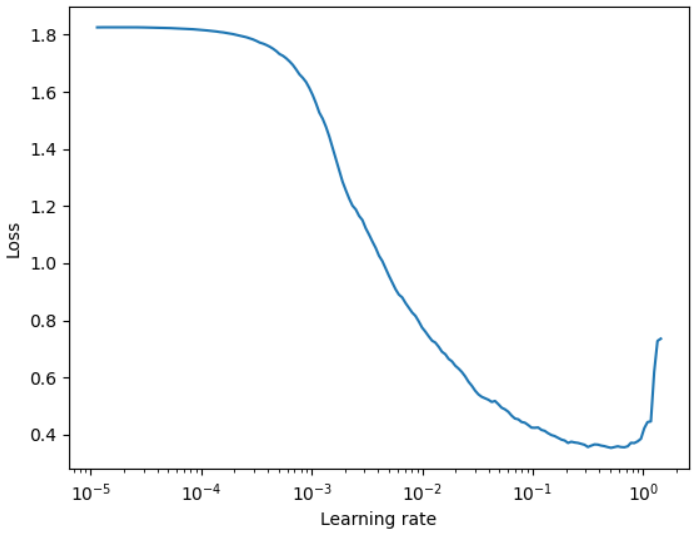


Figure 9: The `lr-finder` showing loss for a LR of 5×10^{-6} to 5×10^{0} using 200 steps.

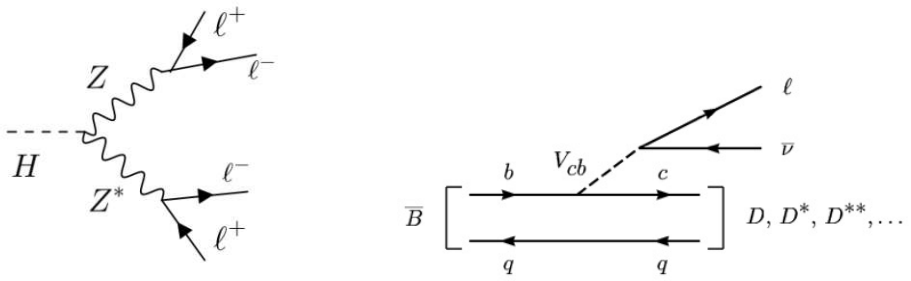
minimum of the gradient of the loss – that is the point of the steepest descent in this curve [67].

4 Using ParticleNetLepton for Lepton Identification

Precision measurements and searches for new physics at the CMS experiment of the LHC heavily rely on accurate particle identification and reconstruction. Prompt leptons emerge directly at the hard-scatter process or from the decay of very short-lived particles like $W^{\pm}$, Z, and Higgs bosons, and also possible new particles. An example of a prompt lepton can be found in Fig. 10a. The background to the prompt leptons can be divided into two categories:

- Non-prompt leptons: these are real leptons emerging from the decay of particles with a longer lifetime (later, we will further split up this category). The decay position is referred to as the Secondary Vertex (SV), which will differ from the primary vertex, the position of the proton-proton collision. An example of a non-prompt lepton is shown in Fig. 10b.
- Fake leptons: here, a non-lepton particle (e.g. a punch-through hadron from a high-energy jet) satisfies the lepton selection criteria.

Therefore, increasing discriminating performance between different categories of leptons can greatly enhance the sensitivity of measurements and reduce their statistical and systematic errors.



(a) Prompt lepton (b) Non-prompt lepton

Figure 10: (a) Example of a Higgs decay process with prompt leptons in the final states (b) A non-prompt lepton process.

As discussed in Sec. 3, ML algorithms and techniques have achieved tremendous performance gains in various fields of science and technology. Hence, we would also like to leverage ML techniques to boost the performance object identification in high-energy physics. Fig. 11 shows the increased performance of ParticleNet over older algorithms on a benchmark test of tagging top quarks decaying hadronically. The DeepAK8 algorithm is an older jet identification algorithm using recurrent neural networks (RNNs). It uses

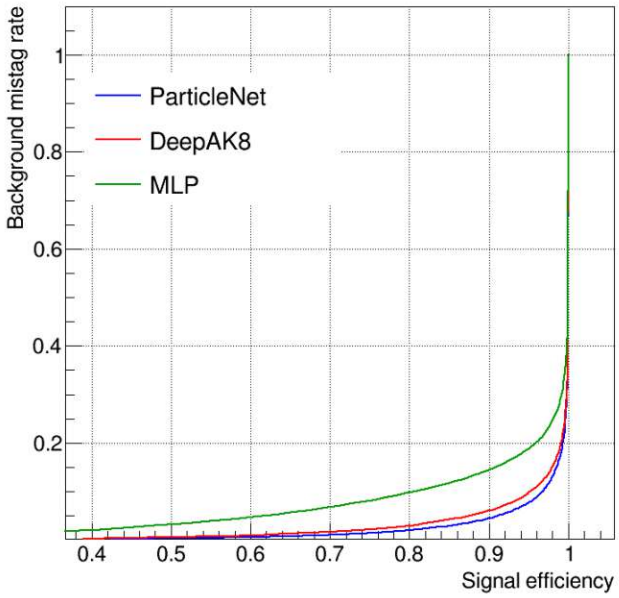


Figure 11: An example of the improvement in performance achieved by the ParticleNet algorithm, compared to the DeepAK8 algorithm and a simple MLP.

sequences of particles as candidates, therefore not using spatial relations of the particles. The MLP stands for multi-layer perceptron – a simple type of NN composed of several layers of perceptrons, a binary linear classifier.

4.1 Current State of the Art

4.1.1 Muons: Cut-based Identification

Muon selection is based on a series of conditions on different variables. Currently, several predefined identification requirements, commonly referred to as IDs, exist, which differ in signal efficiency. To compare the performance of ParticleNetLepton with the existing muon selection techniques, we use the Tight ID in this thesis. The thresholds for the variables used in the Tight ID criteria are described in Table 1.

The cut-based IDs are used in combination with the relative isolation. Relative isolation is a measure of the activity around the lepton. Its value will be small if it is a prompt lepton created in the decay of a short-lived particle like $W^{\pm}$ and Z and large if it is a non-prompt lepton, e.g., the one created in the leptonic decay of a B meson. Therefore, isolation provides a measure to separate leptons from different origins. In this thesis, the

Variable	Threshold value
Is a global muon?	> 0 (True)
Is a PF muon?	> 0 (True)
$\frac{\chi^2}{N_{dof}}$ of the global-muon track fit	< 10
At least one muon-chamber hit included in the global-muon track fit	> 0
Muon segments in at least two muon stations	> 1
transverse impact parameter d_{xy} w.r.t. the primary vertex	< 2 mm
longitudinal distance of the tracker track d_z w.r.t. the primary vertex	< 5 mm
# of pixel hits	> 0
# of tracker layers with hits	> 5

Table 1: Cuts on the variables used in the Tight ID. Taken from [68].

PF isolation and mini-isolation are used. The PF isolation variable is defined as the sum of the p_T of particles in a cone $\Delta R = \sqrt{\Delta\phi^2 + \Delta\eta^2} < 0.4$ around the lepton divided by the lepton p_T. Here $\Delta\phi$ is the difference in the azimuthal angle, and $\Delta\eta$ denotes the difference in pseudorapidity. The mini-isolation variable is optimised for identifying leptons in the decay chain of an object with large p_T. Here the cone size ΔR is not fixed, but varies with lepton p_T:

$$R_{mini-iso} = \begin{cases} 0.2, & p_T^l \leq 50\,\text{GeV} \\ \frac{10\,\text{GeV}}{p_T^l}, & p_T^l \in (50\,\text{GeV}, 200\,\text{GeV}) \\ 0.05, & p_T^l \geq 200\,\text{GeV} \end{cases}$$

The numerical value of the relative isolation in this thesis is set to be 0.15 for both PF and mini-isolation.

4.1.2 Electrons: Boosted Decision Tree based ID

Electron identification is currently based on boosted decision tree (BDT), an ML technique. It is a multivariate analysis (MVA) approach where each electron candidate gets assigned a score quantifying the probability it is classified into a specific category. A range of different variables are used in training, the complete list can be found in Ref. [69]. In this thesis, we use the Fallv2WP90_noIso ID: the v2 denotes the second training of the BDT with more samples and a faster algorithm. The WP90 indicates that it should deliver about 90% selection efficiency for prompt electrons. The noIso indicates that this ID was trained without using isolation variables.

4.2 New approach in this thesis

The current state-of-the-art approaches discussed in the previous section work quite well. However, there is still a scope to exploit more features, including the spatial correl-

ations of nearby particles, improving performance even further. Employing the EdgeConv operation at the heart of ParticleNet allows us to further enhance lepton identification performance by including features of those nearby particles. During training, ParticleNetLepton is expected to learn the characteristics of the neighbourhood of leptons from different origins and use that information to distinguish them.

4.3 Performance Quantification

Quantifying the performance of a classifier is done via statistical analysis of the predictions made by the classifier, which is, in this thesis, the ParticleNetLepton algorithm. A prediction made by a classifier can be classified into one of four groups:

- True positive: Prediction 'true' matches the correct label 'true'
- False positive: Prediction classifies event as 'true', while the correct label is 'false'
- True negative: Prediction 'false' matches the correct label 'false'
- False negative: Prediction classifies event as 'false', while the correct label is 'true'

These distinctions only differentiate between two labels 'true' and 'false'. In the case of more than two labels, which is true for most of the work performed in the context of this thesis, one has to adapt the data processing to circumvent this problem. Therefore, for each signal label, we add up all other labels to one background. To ensure the standardization to 1, we scale the score to the signal class by $\frac{\text{score}_{\text{signal}}}{\text{score}_{\text{signal}}+\text{score}_{\text{background}}}$. A typical example of the normalised output scores can be found in Fig. 12.

In this thesis, we quantify the performance of various classifiers by comparing *receiver operating characteristics* (ROC) graphs of the different discriminators. The ROC graphs are generated by considering *True Positive Rate* (TPR) and *False Positive Rate* (FPR) simultaneously.

$$TPR = \frac{N_{True\ positive}}{N_{True\ positive} + N_{False\ negative}} \tag{13}$$

$$FPR = \frac{N_{False\ positive}}{N_{True\ negative} + N_{False\ positive}} \tag{14}$$

To construct a ROC graph, one needs to take TPR and FPR as functions of the classifier score (a number between 0 and 1), then plot FPR over TPR for each point of the classifier score. For a high-performance classifier, this usually yields a somewhat hyperbolic-looking graph. This curve would be two straights forming a right angle for an ideal classifier. The ROC graph also allows for setting the desired working point (WP) of the classifier, which corresponds to a specific threshold on the classifier score - to use it for analysis, one has to define a threshold for either TPR or FPR and will then receive the corresponding value

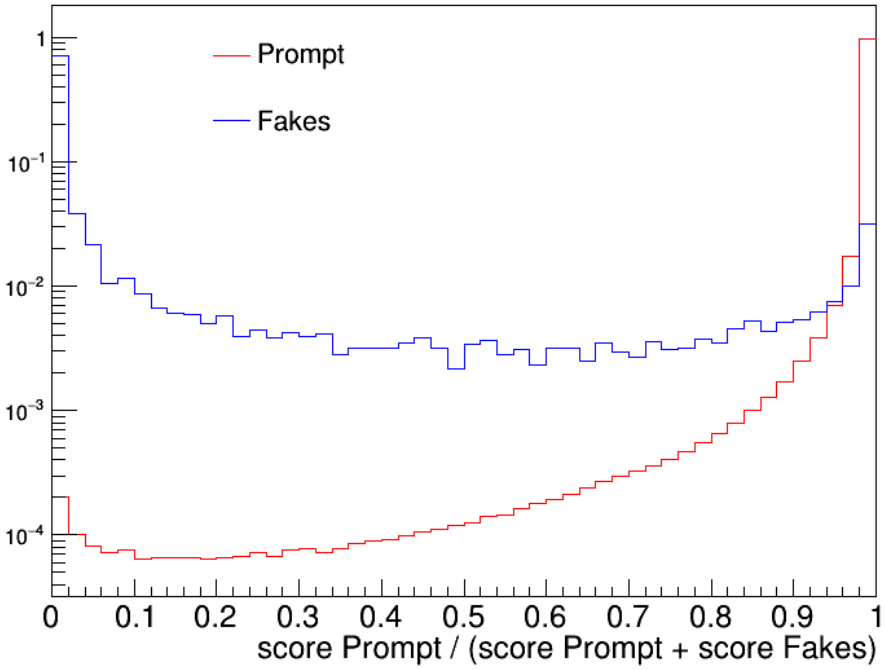


Figure 12: A typical output plot using the *prompt* label as signal and *fakes* as background.

of the other. In the following, we refer to the TPR as 'signal efficiency' and the FPR as 'background mistag rate'.

For the model trained for the purpose of this thesis, we used at most six lepton classes as training labels. Therefore, every input particle will belong to one of these six labels. They are:

- `prompt`: lepton coming from W±, Z, or Higgs
- `HF hadron`: lepton from the decay of a B- or D-hadron
- `LF hadron`: lepton from the decay of hadrons comprised of only u-, d- and s-quarks
- `tau`: light lepton from decay of a τ lepton
- `photon`: lepton from conversion of a high energy photon
- `fakes`: lepton without a matching to any lepton at the particle level

4.4 Input data and training variables

As a general rule of thumb, one can choose 'good' variables, implying that they have a lot of discrimination power, by the naked eye, looking at the distributions of the variables for each label. Variables with different distribution shapes for different labels will have good discriminating power, while those with similar distributions will not. This is how one chooses most training variables, while other variables are considered while fine-tuning the training iterations. Examples of variables with good discrimination power can be found in Fig. 13 - these variables are also included in all training iterations done in the context of this thesis. The full lists of training variables are given in Tables 2 for the PF candidate features, 3 for the SV input features, 4 for the muon features and 5 for the electron features.

Feature	Explanation
$\Delta\eta$	Pseudorapidity difference between lepton and PF Candidate
$\Delta\phi$	azimuthal angle difference between lepton and PF Candidate
ΔR	$\Delta R = \sqrt{\Delta\phi^2 + \Delta\eta^2}$ of PF Candidate w.r.t. lepton
PUPPI Weight	PF Candidate pileup per particle identification weight excl. lepton
d_z^{Sig}	PF Candidate d_z (longitudinal impact parameter) significance
d_{xy}^{Sig}	PF Candidate d_{xy} (perpendicular impact parameter) significance
`hcalFractionCalib`	Fraction of PF candidate energy in HCAL after calibration
pixelhits	# of hits in the pixel detector matched to the particle track
nTrackerLayers	# of hits in pixel and strip tracker
isElectron	Classified as e by PF (Boolean)
isMuon	Classified as μ by PF (Boolean)
isChargedHadron	Classified as charged hadron by PF (Boolean)
fromPV	particle from primary vertex (Boolean)

Table 2: Input features of the PF candidates.

Feature	Explanation
$\Delta\eta$	Pseudorapidity difference between lepton and secondary vertex (SV)
$\Delta\phi$	azimuthal angle difference between lepton and SV
ΔR	ΔR of SV w.r.t. lepton
$N_{d.o.f.}$	# of degrees of freedom in vertex reconstruction
Δp_T	p_T ratio between lepton and SV (GeV)
mass	mass (GeV)
d_{xy}	longitudinal impact parameter (cm)
ntracks	# of tracks associated with SV
d_{3D}^{Sig}	three-dimensional impact parameter significance
cospAngle	cosine of the angle between the line joining SV to the primary vertex and the beam axis

Table 3: Features of SV used as inputs.

Feature	Explanation
d_{xy}	Impact parameter of the lepton in transverse (x-y) plane (cm)
χ^2	χ^2 of the track fit
d_{xy}^{Err}	d_{xy} error
d_{xy}^{Sig}	d_{xy} significance
d_z	Longitudinal impact parameter (cm)
d_z^{Err}	d_z error
d_z^{Sig}	d_z significance
E_{ECAL}	Energy fraction of lepton in ECAL
`hit`	# of hits in tracker matched to the lepton track
`hcaloverecal`	Ratio of the hadronic energy within a cone of $\Delta R = 0.15$ behind the supercluster to the energy in 5x5 crystal array around the centre of SC
IP_{3d}	3D impact parameter (cm)
`lostHits`	# of tracking layers which have no hits matched to lepton track
$N_{d.o.f.}$	# of degrees of freedom for lepton track fit
`pixhit`	# of hits in pixel detector matched to the lepton track
`nTrackerLayers`	# of hits in pixel and strip tracker matched to the lepton track
`posmatch`	Position match
`segmentComp`	Segment compatibility
$S_{IP_{3D}}$	3-D impact parameter significance
`minisoch`	Mini-isolation variable computed using charged hadrons only
`minisonh`	Mini-isolation variable computed using neutral hadrons only
`pfRelIso03_drcor`	PF relative isolation with $\Delta R < 0.4$
`ecloverpout`	e supercluster (SC) energy / track momentum at calorimeter extrapolated from the outermost track state
`supcl_preshvsrawe`	ratio of e energy in the preshower detector and energy ofSC
`dr03HcalDepth1-TowerSumEt_Rel`	Relative energy in the first segment of HCAL in $\Delta R < 0.3$

`deltaetacltrkcalo`	η difference between SC and electron track
`r9full`	Energy of 3x3 crystal array around the center of SC divided by SC energy
`e1x5bye5x5`	Energy of 1x5 crystal array divided by Energy of 5x5 crystal array of the SC
`sigmaietaieta`	Energy weighted spread of SC in η
`sigmaiphiiphi`	Energy weighted spread of SC in ϕ
`supcl_phiWidth`	Width of SC in ϕ
`supcl_etaWidth`	Width of SC in η
`fbrem`	Energy fraction from matched bremsstrahlung photons
`tightcharge`	Quality of electron track fit
`eInvMinusPInv`	Inverse ECAL Energy minus inverse tracker momentum $(\frac{1}{E_{ECAL}} - \frac{1}{p_{Tracker}})$
`closeTrackNLayers`	# of tracking layers with hits used by e reconstruction
`eoverp`	ECAL Energy divided by inverse tracker momentum $(\frac{E_{ECAL}}{p_{Tracker}})$
`mvaFall17V2noIso`	Classified as e by MVA ID of Fall 2017
The following features of the AK4 jet nearest to the e (within $\Delta R < 0.4$) are used	
`jetbtag`	DeepJet b-tagging score of the jet
`jetPtRelv2`	p_T component perpendicular to jet direction
`jetRelIso`	Jet relative isolation for nearest jet
`jetNDauCharged`	# of charged constituents
`jetPtRelv2_abs`	Absolute value of p_T w.r.t the axis of the jet
`jetPtRatio`	ratio of p_T^{jet} to p_T^{lepton}

Table 5: Electron training input features.

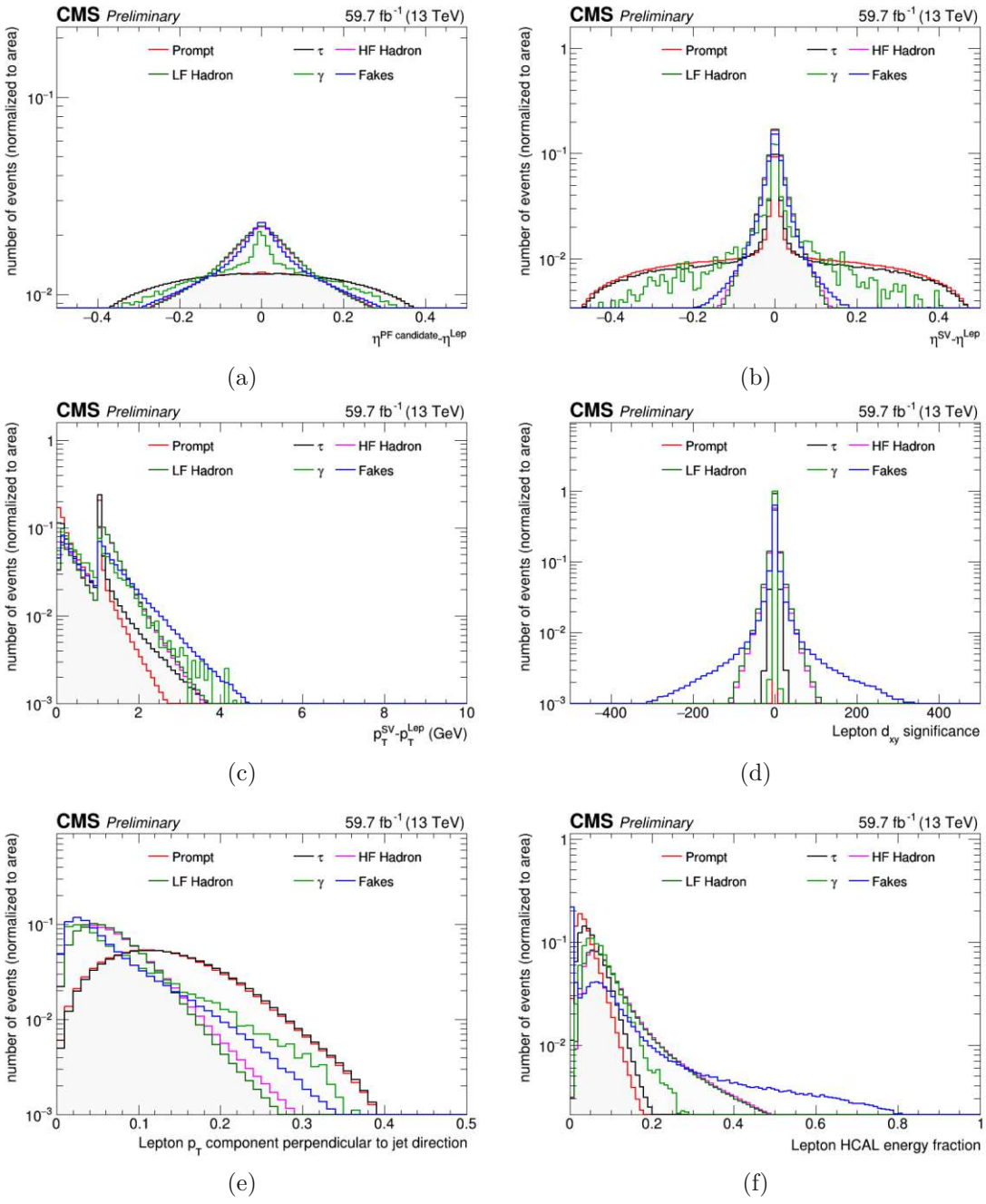


Figure 13: Distributions of some input variables split up according to lepton classes.

Feature	Explanation
d_{xy}	Impact parameter of the lepton in transverse (x-y) plane (cm)
χ^2	χ^2 of the track fit
d_{xy}^{Err}	d_{xy} error
d_{xy}^{Sig}	d_{xy} significance
d_z	Longitudinal impact parameter (cm)
d_z^{Err}	d_z error
d_z^{Sig}	d_z significance
E_{ECAL}	Energy fraction of lepton in ECAL
E_{HCAL}	Energy fraction of lepton in HCAL
`hit`	# of hits in tracker matched to the lepton track
`hcaloverecal`	Ratio of the hadronic energy within a cone of $\Delta R = 0.15$ behind the supercluster to the energy in 5x5 crystal array around the centre of SC
IP_{3d}	3D impact parameter (cm)
`lostHits`	# of tracking layers which have no hits matched to lepton track
$N_{d.o.f.}$	# of degrees of freedom for lepton track fit
`pixhit`	# of hits in pixel detector matched to the lepton track
`nTrackerLayers`	# of hits in pixel and strip tracker matched to the lepton track
`posmatch`	Position match
`segmentComp`	Segment compatibility
$S_{IP_{3D}}$	3-D impact parameter significance
`pfRelIso03_drcor`	PF relative isolation with $\Delta R < 0.4$
`minisoch`	Mini-isolation variable computed using charged hadrons only
`minisonh`	Mini-isolation variable computed using neutral hadrons only
The following features of the AK4 jet nearest to the e (within $\Delta R < 0.4$) are used	
`jetRelIso`	Jet relative isolation for nearest jet
`jetNDauCharged`	# of charged constituents
`jetbtag`	DeepJet b-tagging score of jet
`jetPtRelv2`	p_T component perpendicular to jet direction
`jetPtRelv2_abs`	Absolute value of p_T w.r.t the axis of the jet
`jetPtRatio`	ratio of p_T^{jet} to p_T^{lepton}

Table 4: Muon input features used in the final training

4.5 Muons

As the CMS is built especially to measure muon properties, most of the research in this thesis was done on samples of muons. As training data, we use custom-built data samples (called *ntuples*) generated from information available from the worldwide server network of the CMS collaboration. Those are produced from samples of simulated events using

MADGRAPH5 aMC@NLO [35] and POWHEG [70] generators and PYTHIA8 [71] parton showers followed by detector simulation using GEANT4 [38]. Specifications of the event samples used for training can be found in Table 6. We have included leptons with $p_T > 20\,\text{GeV}$ and $|\eta| < 2.5$. The training variables we used can be found in Tables 2, 3 and 4.

Training sample	Generator used
QCD multijet events generated in different ranges of H_T, which is the sum of jet p_T, from 300 GeV to 2000 GeV	MADGRAPH5 aMC@NLO
Top quark pair production where a W boson from one of the top quarks decays leptonically	POWHEG

Table 6: Generation of the training events.

4.5.1 First training: using six classes

We first started with using all six lepton classes included in our ntuples for training. As Fig. 14 indicates, this configuration is suboptimal for the `tau` and `photon` classes. There are two reasons for this: Firstly, the numbers of leptons in `tau` and `photon` classes are very small compared to those in other classes. To ensure unbiased statistics, ParticleNetLepton uses approximately the same input numbers for every label. This leads to the fact that the network ends up using quite a small number of entries for training. Secondly, the leptons from `tau` class behave very similarly to the `prompt` ones, so it will be tough for the NN to discern these.

However, as can be seen in Fig. 14, even here, we can outperform the existing cut-based identification methods of the Tight ID (with mini-isolation and relative isolation conditions) by almost an order of magnitude.

4.5.2 Improving performance: using four classes

Better performance can be achieved using only four lepton classes as training labels – these are `prompt`, `fakes`, `HF hadron` and `LF hadron`. As the statistics for these classes are large, the network can be trained far more efficiently. Fig. 15 shows the ROC graphs for this training – as one can see clearly, the performance is improved further by quite a margin. Most optimisation and analysis work was done on this configuration; this will be described more thoroughly in the following sections.

4.5.3 Performance stabilisation: using two classes

Following the trend seen in the transition from six to four classes, the next step to enhance performance further would be to reduce the training classes further. We accomplish

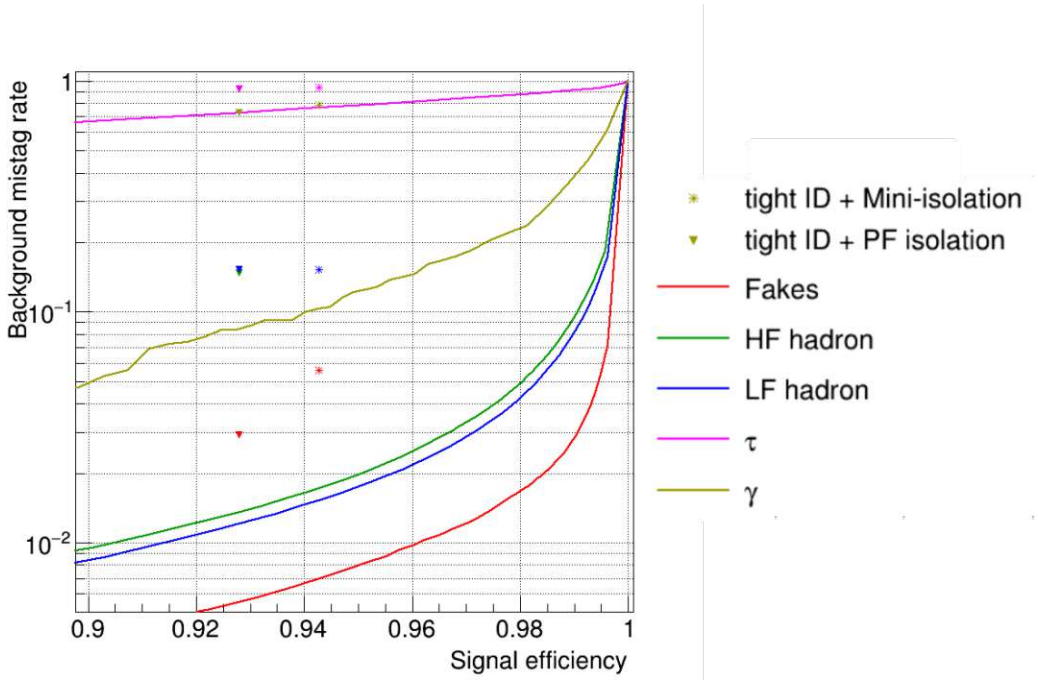


Figure 14: Performance of the ParticleNetLepton algorithm shown by ROC curves of the model using all six lepton classes as training labels. For comparison, the Tight IDs with mini- and relative isolation < 0.15 are added.

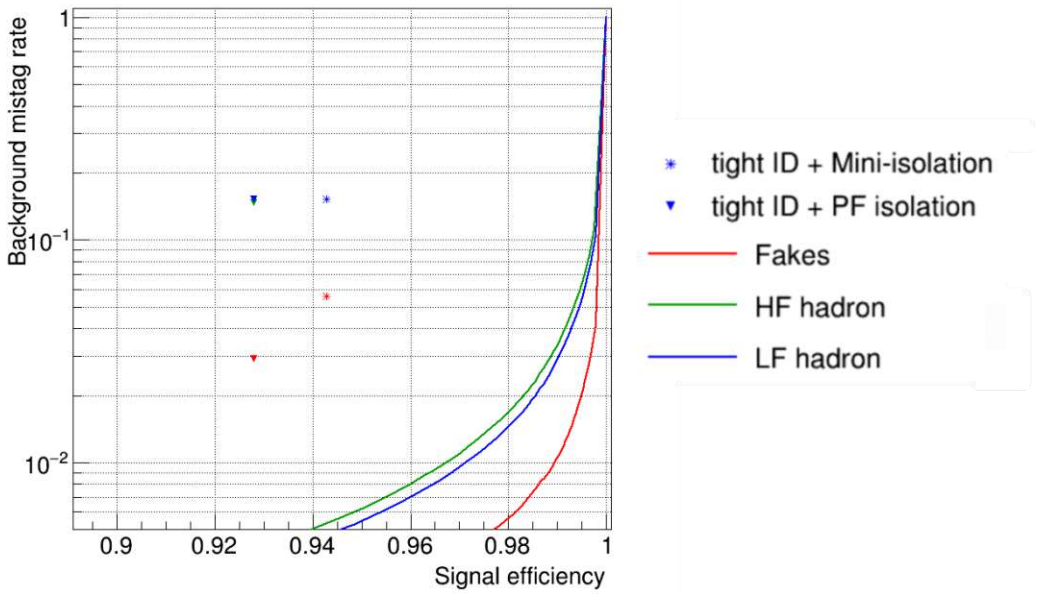


Figure 15: Performance of the ParticleNetLepton algorithm shown using ROC curves of the model trained on four labels.

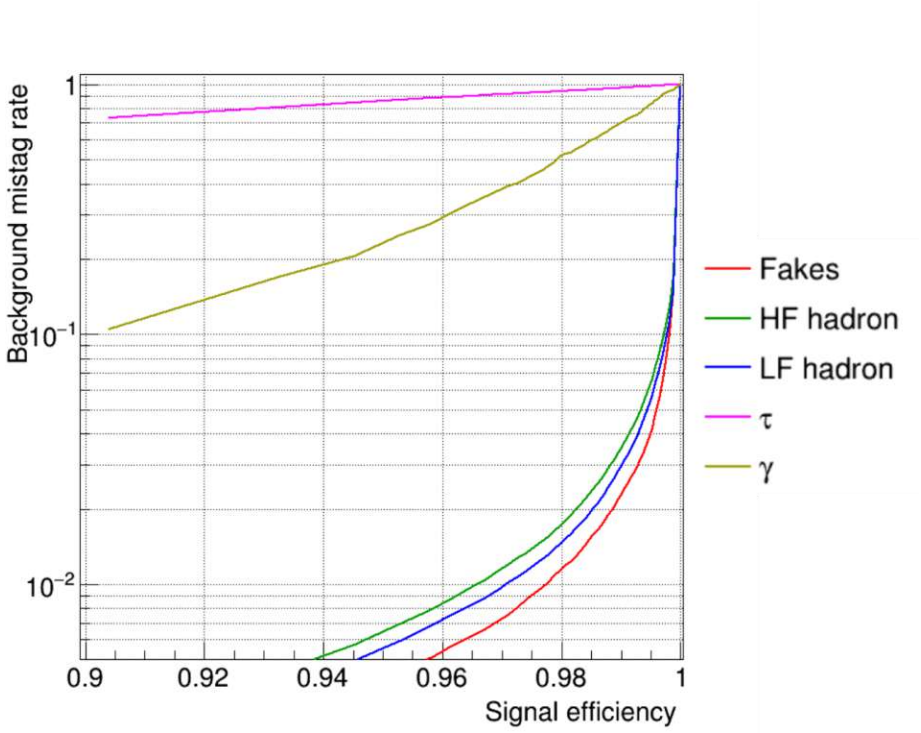


Figure 16: The ROC curves showing the performance of the ParticleNetLepton algorithm trained on only two labels. The performance is quite similar to the four-label model, indicating stability of the model.

this by merging `fakes`, `HF hadron` and `LF hadron` and `photon` labels to one `background` label and `prompt` and `tau` to one `signal` label. In this case, the model is trained only to separate `signal` from `background` classes but ignores the other lepton labels. However, this information still is stored for each event and can be used in plotting to break up the `signal` and `background` labels to make the results comparable.

The performance of this model is shown in Fig. 16; it is pretty similar to that of the model trained on four classes. We took this as a sign of good stability and an indication that we have already reached the network's full potential with training on four classes.

4.5.4 Top Lepton MVA

A more challenging test for the ParticleNetLepton algorithm than the Tight ID would be the TopLepton MVA ID developed by scientists at Ghent University, Belgium. The TopLepton ID is constructed using BDTs and is specially tailored for detecting leptons in the semileptonic decay of the W boson from top quark decays. In this use case, it is the best-performing identification method used in CMS data analyses. As shown in Fig. 17, the ParticleNetLepton algorithm offers better performance than the TopLepton MVA ID. For a signal efficiency of 94%, we can reduce the background mistag rate by about 30% for the `HF hadron` and `LF hadron` classes on hadronic backgrounds.

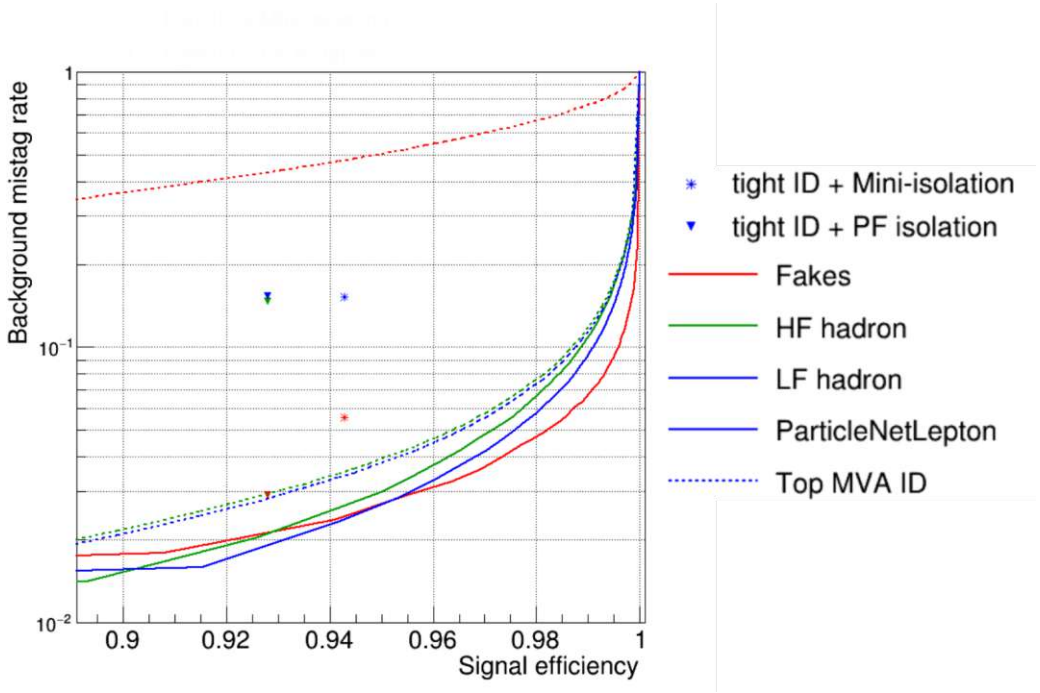


Figure 17: Performance comparison of the ParticleNetLepton network trained using four lepton classes with the TopLepton MVA ID. The ParticleNetLepton algorithm can still outperform this specialised ID by a few per cent.

It is important to mention that so far, we have not set any selection conditions prior to training or while quoting the performance. However, in many measurements, it is common to apply selection conditions on the data used, so also for the TopLepton ID, some preselection conditions are in place – they can be found in Table 7. To ensure a fair comparison, we have also set these preselection conditions while quantifying the performance of ParticleNetLepton compared to the TopLepton MVA ID. This leads to the fact that

on first look, performance seems to be worse – but this can be entirely attributed to the different selection conditions, and the network gets to classify a 'more difficult' testing set.

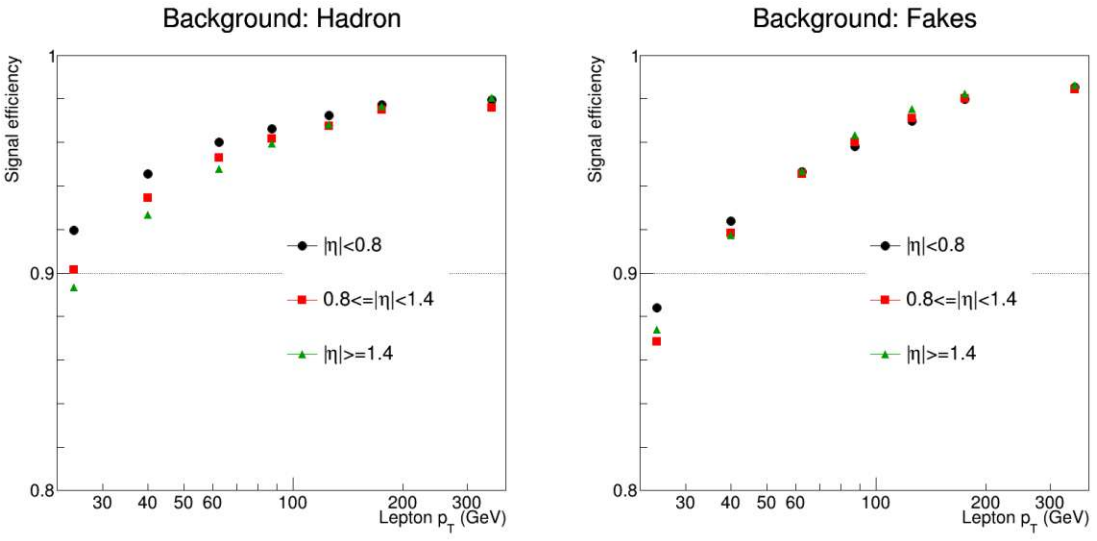
Variable	Threshold value
is Loose?	> 0 (True)
lepton $\lvert d_{xy}\rvert$	< 0.05
lepton $\lvert d_z\rvert$	< 0.1
3D impact parameter significance $sip3d$	< 8.0
mini-isolation	< 0.40

Table 7: Precondition selections for the TopLepton MVA ID for muons.

As the TopLepton ID is suited specifically for the phase space defined by the conditions in Table 7, one could suspect that imposing these criteria also on the training set – therefore moving the training into the same phase space as the testing set – could further improve performance. However, we found that this leads to worse performance, likely because the training data set will be quite small.

4.5.5 Performance details – kinematic dependencies

To understand the performance in more depth, we split up the prediction data in various ways – into different regions of lepton p_T and η, the dependency of the number of PF candidates within $\Delta R < 0.4$ around the lepton, or the number of primary vertices in the event the lepton has originated from.

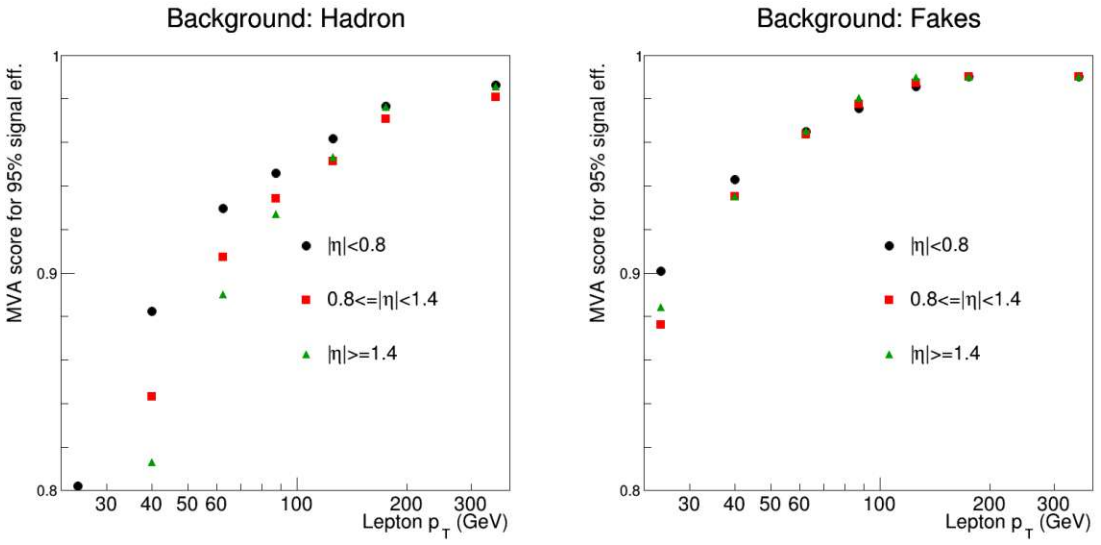


(a) (b)

Figure 18: The efficiency for the selection of prompt muons as a function of lepton p_T, (a) against HF Hadron and LF Hadron as backgrounds, (b) against `fakes` background.

Fig. 18 shows the dependency of the signal efficiency with p_T and η. For both plots, the signal label is `prompt`, the background labels are `HF hadron` and `LF hadron` summed up for the plot titled 'Hadron' and `fakes` for the figure titled 'Fakes'. We can see that performance improves with p_T. However, there is little dependency on η.

To use the ParticleNetLepton classifier in analysis, it will be important to avert dependencies on kinematic variables. To achieve this, we calibrate the MVA scores in different p_T ranges. This is done by requiring the signal efficiency to be 95% and determining the threshold achieving this in different p_T and η regions. This yields the classifier threshold for each p_T and η bin to have the same signal efficiency. The plots showing these thresholds are shown in Fig. 19.

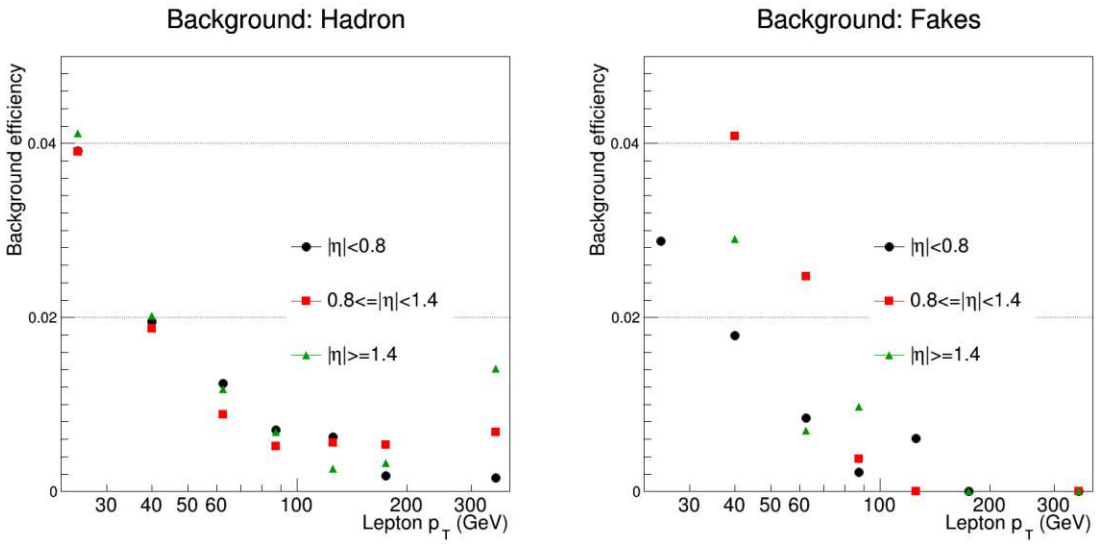


(a) (b)

Figure 19: Calibration at 95% signal efficiency (a) for `hadronic` background, (b) for `fakes` background.

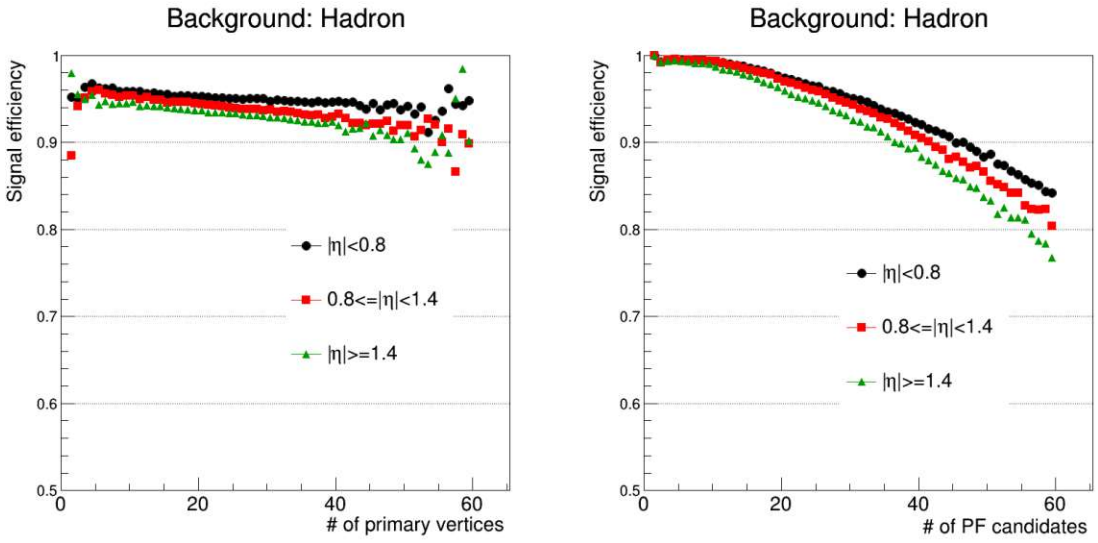
Fig. 20 shows the corresponding background efficiencies, plotted in the same p_T and η ranges. We can see that also the background efficiency worsens in the low p_T bins, which indicates increasing performance for high p_T. Again, there is almost no correlation regarding η.

Fig. 21 shows the dependency of the signal efficiency on the number of primary vertices and the number of PF candidates. Interestingly, the number of primary vertices does not influence the performance significantly, implying that the algorithm is expected to work well in a high pileup environment. However, there is a clear correlation between performance and the number of PF candidates – performance tends to worsen with a rising number of PF candidates.



(a) (b)

Figure 20: The background efficiency for the thresholds corresponding to 95% prompt-muon selection efficiency shown in the same p_T and η bins for (a) `hadronic` and (b) `fakes` background.



(a) (b)

Figure 21: Signal efficiency for prompt-muon selection as a function of (a) the number of primary vertices, (b) the number of PF candidates.

Fig. 22 gives further details about the performance. The ROC graphs are shown split up into seven different p_T and three η bins for both ParticleNetLepton and the TopLepton MVA ID. Again, the increase in performance is true mainly for the high p_T regions, while in low p_T regions we perform worse than the TopLepton MVA. Interestingly, here we see a dependency on η for the first time – ParticleNetLepton does not significantly depend on η, while the TopLepton ID performs worse in higher η regions. Therefore, we conclude that we outperform the TopLepton ID everywhere except for $p_T < 30\,\text{GeV}$ and $|\eta| < 0.8$.

Fig. 23 shows the comparison between ParticleNetLepton and the TopLepton ID split into different p_T and η bins for `fakes` background. For the `fakes` background, we outperform the TopLepton ID by over a magnitude – consistent with the inclusive performance plots shown before.

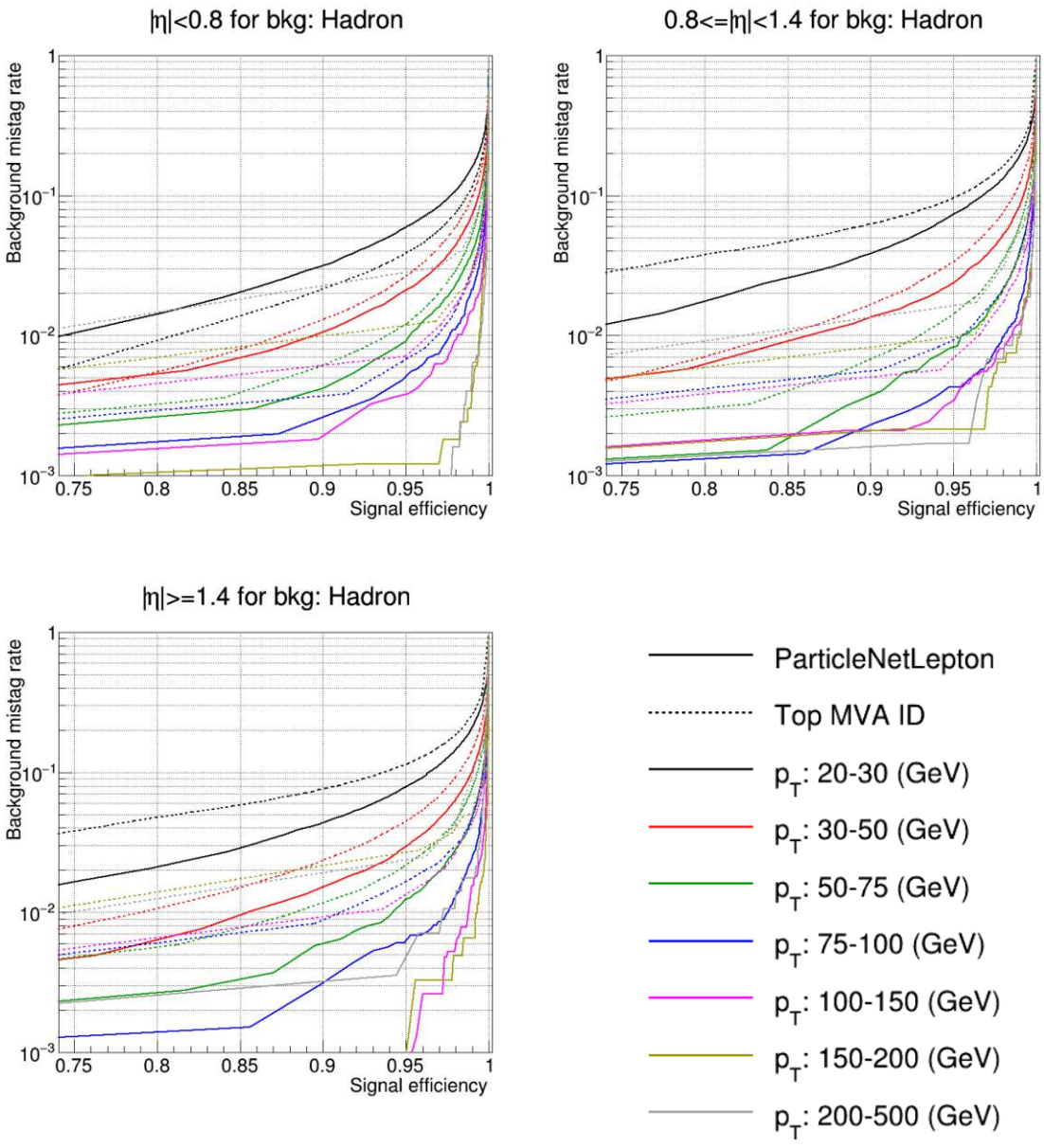


Figure 22: Performance comparison between ParticleNetLepton and the TopLepton MVA algorithms shown using ROC graphs split up into different p_T and η bins for `hadronic` background.

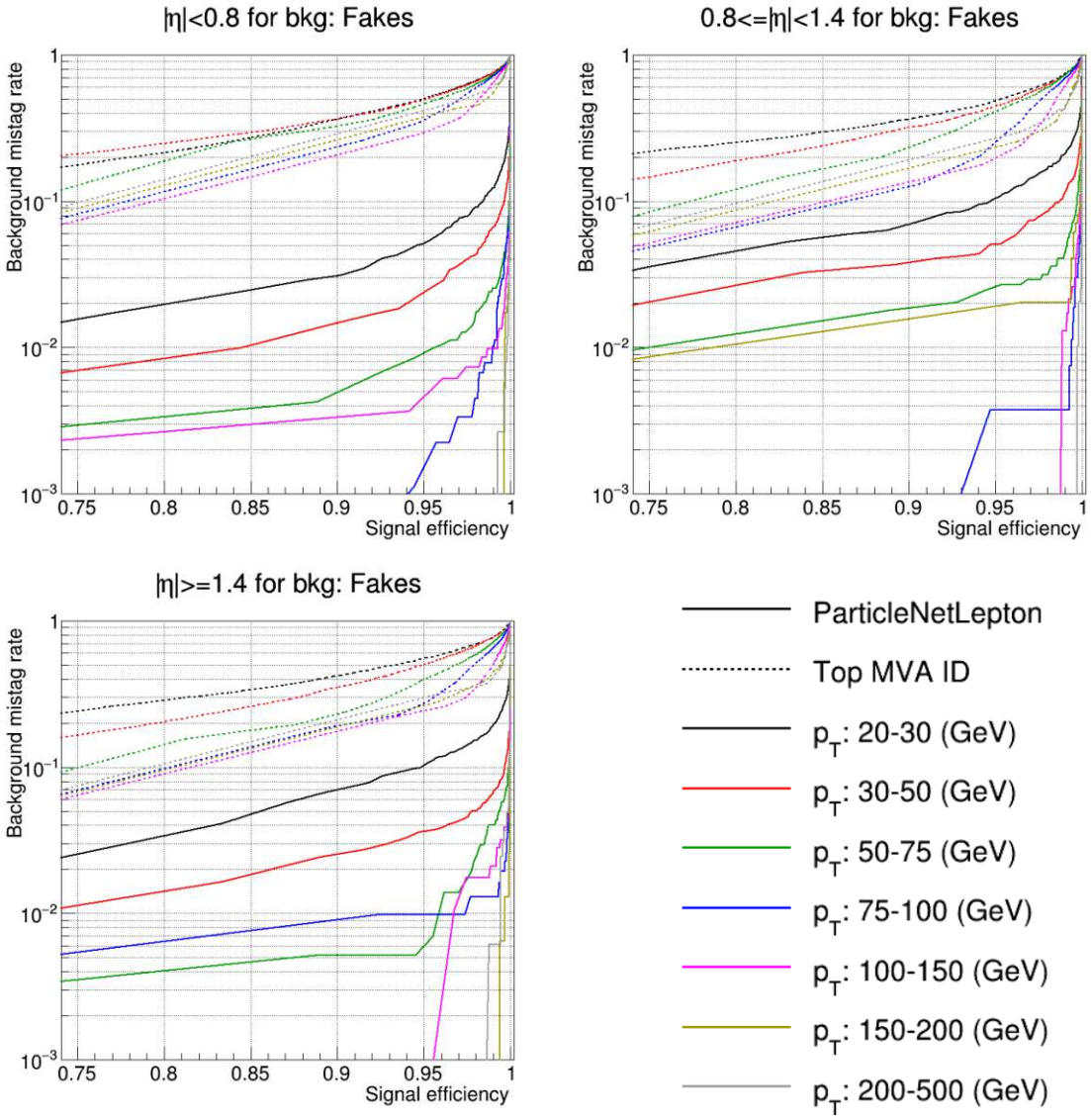


Figure 23: Performance comparison between ParticleNetLepton and the TopLepton MVA algorithms shown using ROC graphs split up into different p_T and η bins for `fakes` background.

4.6 Electrons

As already stated, a lot of the development and analysis work was done using muon samples, and many of these findings can be directly used for the electron samples. For example, the four-label configuration also proved as the most performant model configuration for the electron samples. Thus, the electron chapter will be shorter than the muon chapter. The variables used to train the electron models can be found in Table 5.

Variable	Threshold value
pass conversion veto	> 0
lepton $\lvert d_{xy}\rvert$	< 0.05
lepton $\lvert d_z\rvert$	< 0.1
3D impact parameter significance $sip3d$	< 8.0
lepton mini-isolation	< 0.40
# of lost Hits	< 2

Table 8: Precondition selections for the TopLepton MVA ID for electrons.

4.6.1 Using four training classes – compared to TopLepton ID

Fig. 24 shows the performance of the ParticleNetLepton algorithm using electron samples, comparing it to the TopLepton MVA ID. For a fair comparison, the preselection conditions for electron samples are again taken into account; these can be found in Table 8. Again, we can outperform the TopLepton ID – we can reduce the background mistag rate by a factor of 2 for `HF hadron` and a factor of 4 for `LF hadron`. This poses a clear improvement in performance for electron identification techniques. For the `fakes` background, the ParticleNetLepton algorithm can reduce background suppression by an order of magnitude.

4.6.2 Performance details

Here we can make the same considerations as in chapter 4.5.5. Figure 25 again shows the dependency of the signal efficiency with p_T and η. Here we can see some deviations from the muon samples: We can recognize some decline with sinking p_T again, but as well with very high p_T. Moreover, we see a clear correlation with η – performance degrades significantly in more forward regions.

Fig. 26 shows a similar trend for electrons with `hadron` background as seen already in the model trained to identify muons: In the low p_T and $|\eta|$ region, we perform worse than the TopLepton MVA ID, in all other regions we can outperform it. Again, we see a dependency regarding η, as the TopLepton ID performance also shows an η dependency.

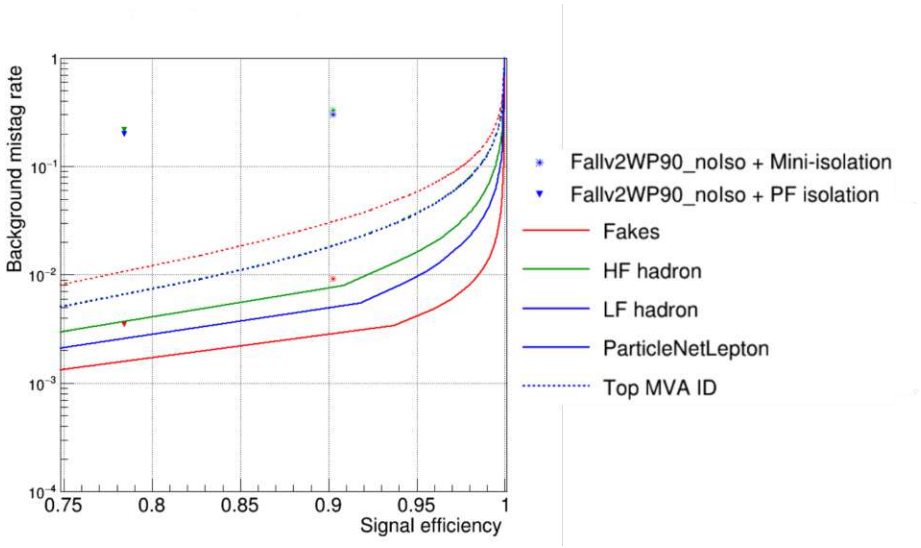
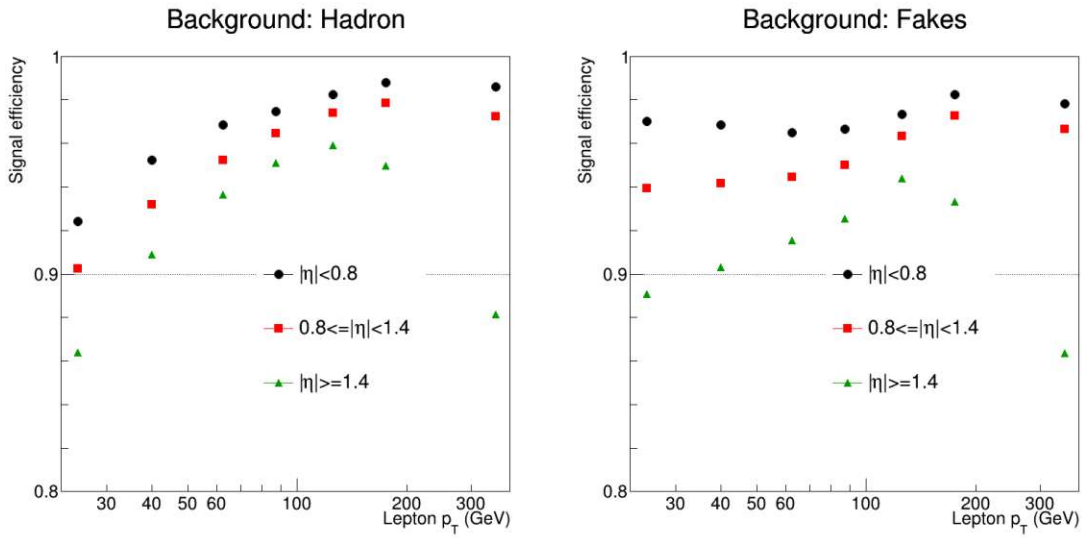


Figure 24: ROC graph showing the performance of ParticleNetLepton on electron samples, compared to the TopLepton MVA ID.



(a) (b)

Figure 25: The efficiency for the selection of prompt electrons as a function of lepton p_T, (a) against `hadronic` background, (b) against `fakes` background.

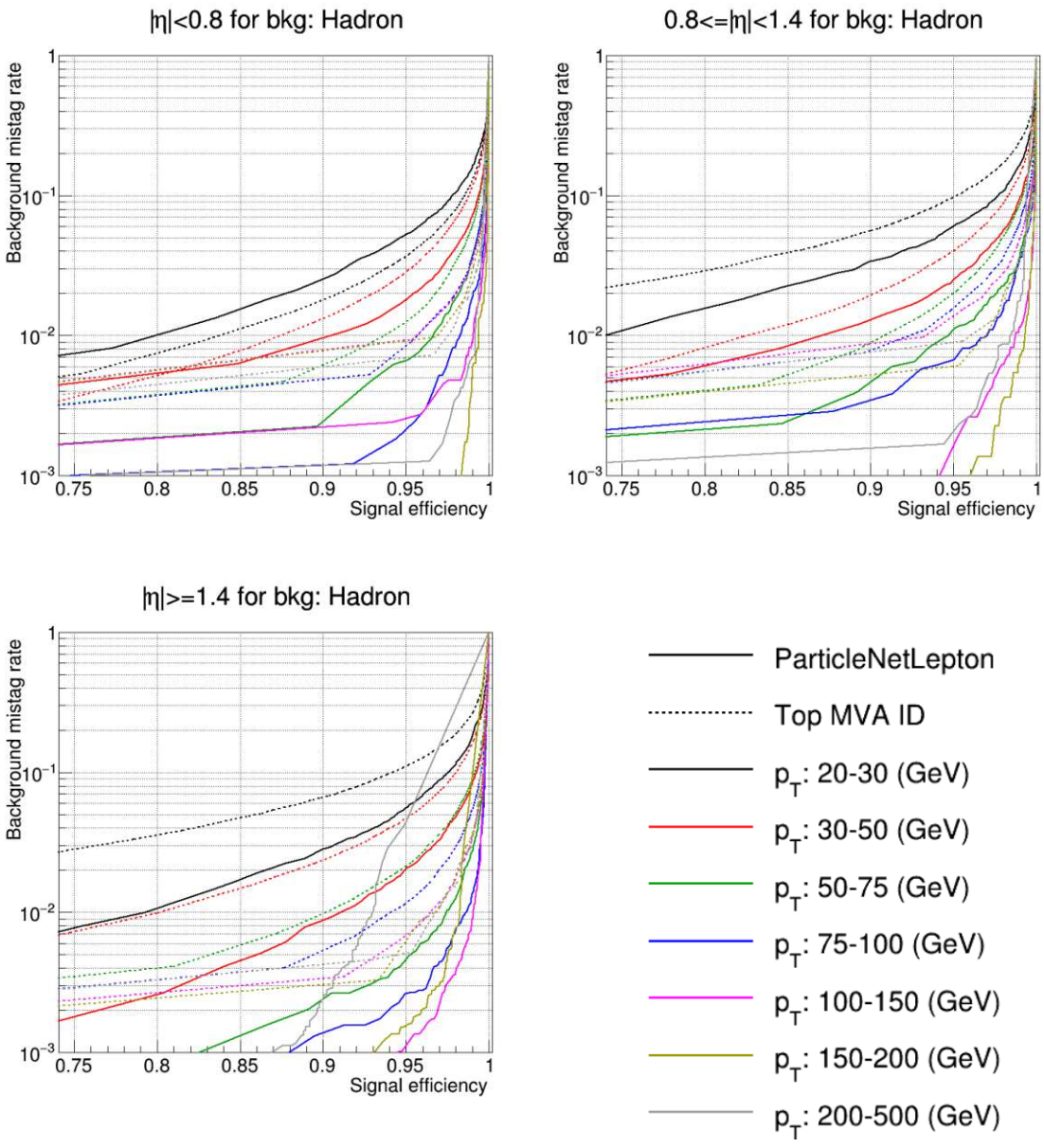


Figure 26: Performance comparison between ParticleNetLepton and the TopLepton MVA algorithms shown using ROC graphs of the electron training split up into different p_T and η bins for `hadronic` background.

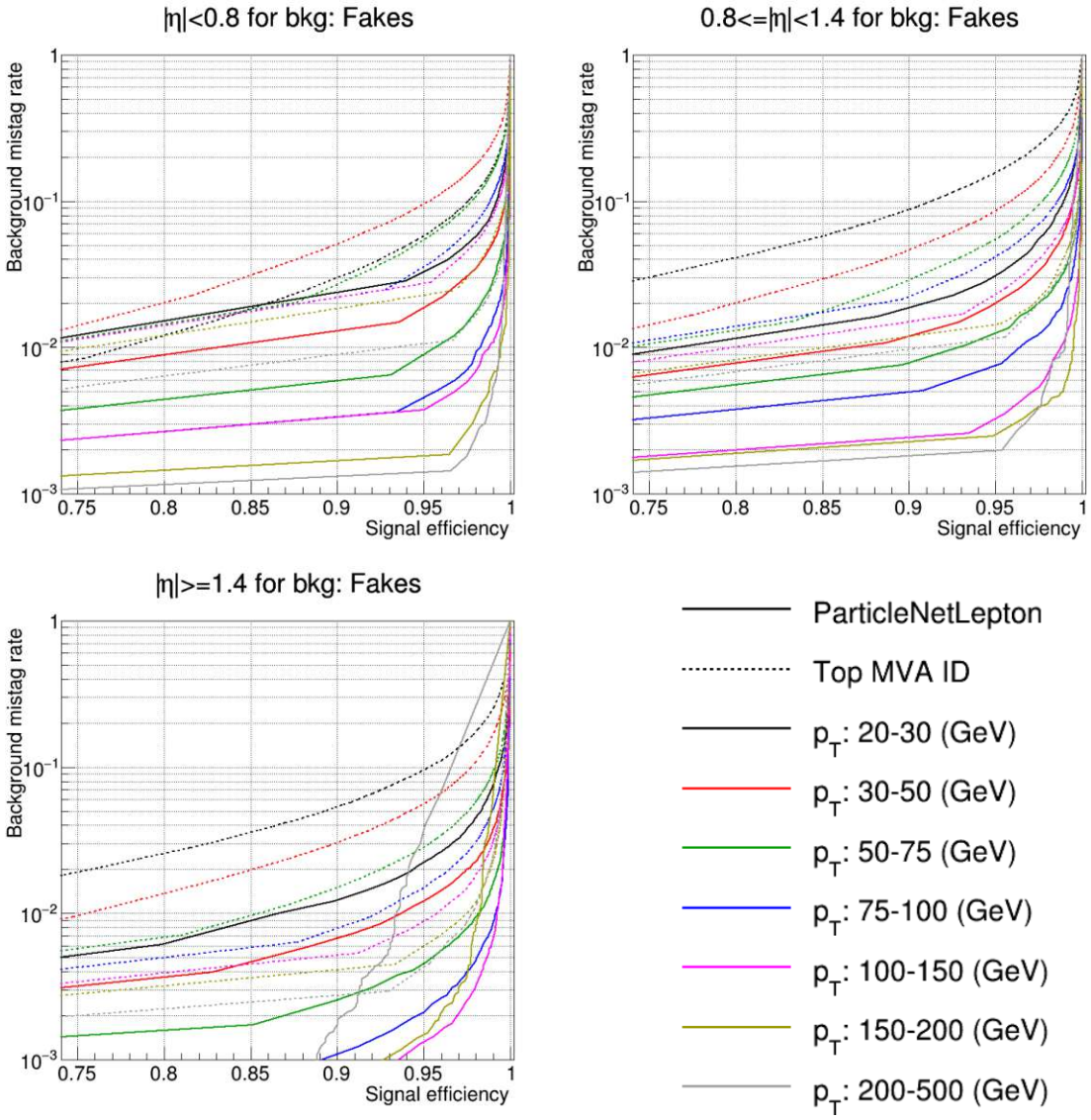
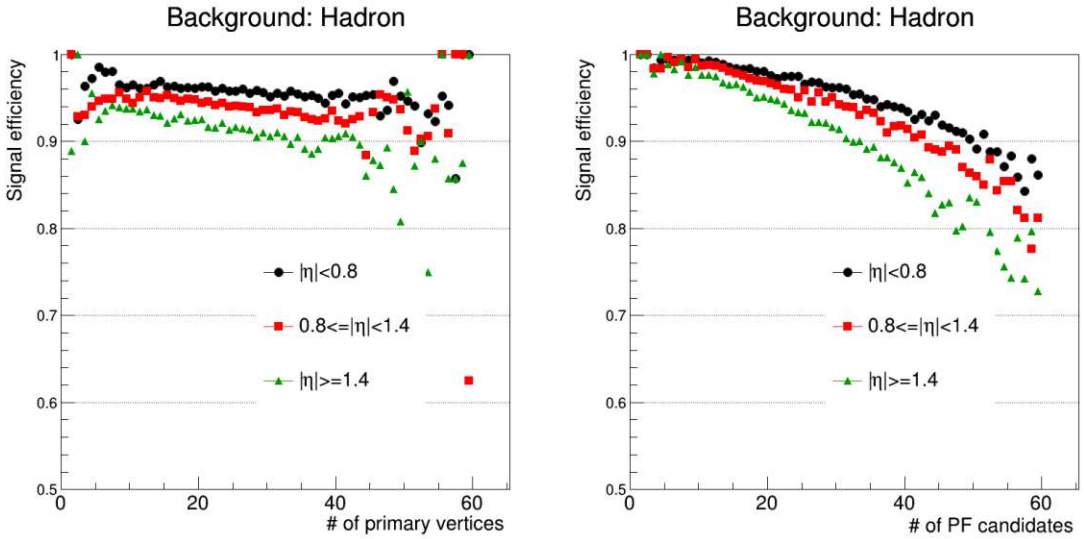


Figure 27: Performance comparison between ParticleNetLepton and the TopLepton MVA algorithms shown using ROC graphs of the electron training split up into different p_T and η bins for `fakes` background.

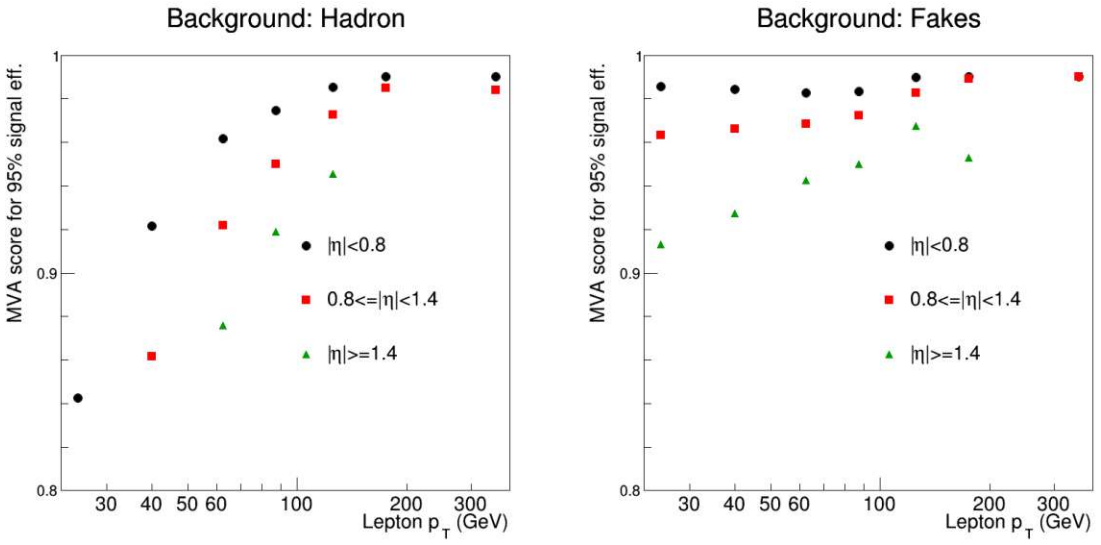
The comparison of ParticleNetLepton with the TopLepton ID regarding electrons with `fakes` background can be found in Fig. 27. These validate the previous findings: for `fakes` background, we can outperform the TopLepton ID by almost an order of magnitude.

The behaviour shown in Fig. 28 matches the observations made previously very closely. The number of primary vertices does not influence performance significantly, but we can observe performance degrading with a higher number of PF candidates. This matches the observations in Sec. 4.5.5. Moreover, we also can validate the observation that performance seems to worsen in higher η bins – this can be seen in both Figs. 25 and 28.



(a) (b)

Figure 28: Signal efficiency for prompt-electron selection as a function of (a) the number of primary vertices, (b) the number of PF candidates.

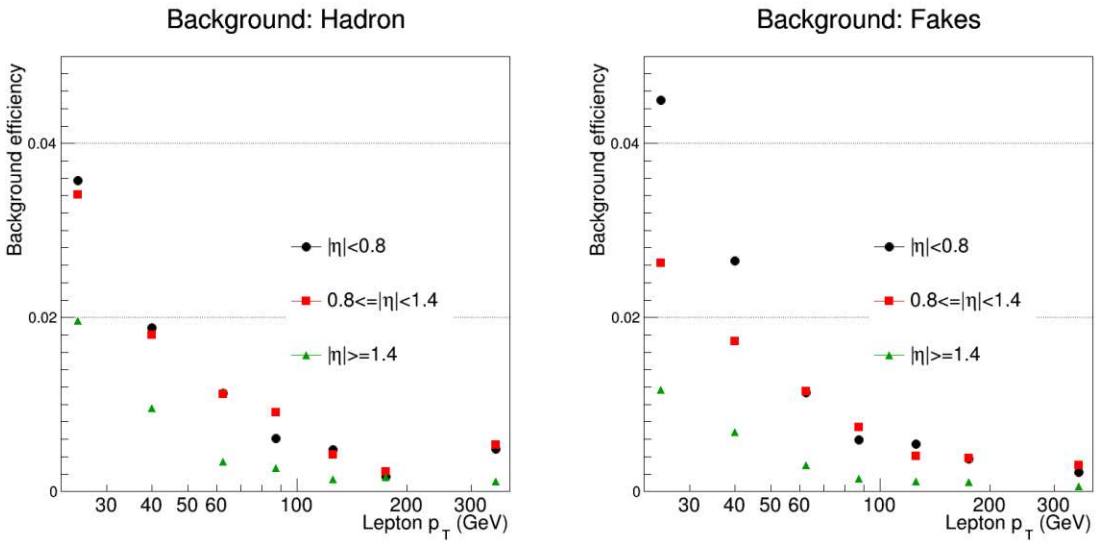


(a) (b)

Figure 29: Calibration at 95% signal efficiency (a) for `hadronic` background, (b) for `fakes` background for the electron training.

As was already the case in the muon model, because ParticleNetLepton shows a significant p_T dependency, there is a need to calibrate the ROC graphs. Again, we calibrate on 95% signal efficiency. The according plots can be found in Fig. 29. Interestingly, for electrons, we can also see a clear correlation with η, which was not the case for muons.

Fig. 30 shows the corresponding background efficiencies. We can see again that the background efficiency deteriorates in the low p_T bins – this indicates the decreasing performance in low p_T regions. The correlation regarding η is also clearly visible.



(a) (b)

Figure 30: The background efficiency for the thresholds corresponding to 95% prompt-electron selection efficiency shown in the same p_T and η bins for (a) `hadronic` and (b) `fakes` background.

5 Conclusion

In this thesis, we adapted ParticleNet, a graph neural network-based algorithm, to identify leptons in the CMS experiment – this solidified into the ParticleNetLepton algorithm. We have exploited the EdgeConv operation employed by ParticleNet to utilize the correlations of particles in the vicinity of the leptons for their identification. We modified the architecture of the ParticleNet network by injecting the lepton features identified by the particle-flow (PF) reconstruction into the fully connected layer of ParticleNet and inserting a second fully connected layer. To train the model, we used simulated events of top quark pair production, where a W boson from one of the top quarks decays leptonically and multijet production via strong interaction. Finally, we demonstrated a calibration on 95% signal efficiency, as we find that the ParticleNetLepton performance is dependent on kinematic variables. We show the calibration for the transverse momentum p_T and pseudorapidity η.

We compared the performance of ParticleNetLepton to the standard criteria used in the CMS measurements – we can outperform those by an order of magnitude for both electrons and muons. We also compared the performance with the so-called TopLepton MVA ID, which is a boosted decision tree based technique specialised to identify leptons in the top quark decay chain. In the case of muons, we can outperform this special identification technique in most parts of the phase space, except for very-low p_T in the central regions of the detector. Summing over all kinematic regions, we can reduce the background mistag rate by about 30% for a signal efficiency of 94% for muons originating from hadrons, as shown in Fig. 17 of Sec. 4.5.4. For electrons, we outperform it in a large phase-space region as well, with the same limitation as for muons. In Fig. 24 of Sec. 4.6.1 we show that we achieve a factor of two in improved background suppression for electrons originating from the decay of heavy flavoured hadrons and a factor of 4 for those originating from the decay of light hadrons, both at 95% signal efficiency. We also show that the performance is independent of the number of simultaneous collisions, indicating good performance in a high-pileup scenario.

This thesis focuses on implementing the ParticleNetLepton algorithm and proving its performance capabilities – future works could involve a complete hyperparameter optimisation of the model. Furthermore, this thesis only used simulated data – future analysis on real collision data has to be done to calibrate the performance of the ParticleNetLepton algorithm.

6 Acknowledgements

This thesis would not have been possible without the help of a lot of people. First, I would like to thank my supervisor Robert Schöfbeck, who always encourages me to give my best. Whenever I encountered a problem and needed advice, Robert had an open ear for me.

I also want to thank Suman Chatterjee, who answered all of my questions with great patience, helped me whenever I was stuck somewhere and did a lot of work which he was in no way obliged to do. Likewise, I want to thank the rest of the CMS analysis group at PSK for making my master's thesis quite a pleasant time.

Finally, I would like to thank my parents and family for their support throughout my studies and for keeping me grounded.

References

[1] Lyndon Evans and Philip Bryant. 'LHC Machine'. In: *Journal of Instrumentation* 3.08 (Aug. 2008), S08001. DOI: 10.1088/1748-0221/3/08/S08001. URL: https://dx.doi.org/10.1088/1748-0221/3/08/S08001.

[2] Huilin Qu and Loukas Gouskos. 'Jet tagging via particle clouds'. In: *Physical Review D* 101.5 (Mar. 2020). DOI: 10.1103/physrevd.101.056019. URL: https://doi.org/10.1103%2Fphysrevd.101.056019.

[3] The CMS Collaboration. 'Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC'. In: *Physics Letters B* 716.1 (Sept. 2012), pp. 30–61. DOI: 10.1016/j.physletb.2012.08.021. URL: https://doi.org/10.1016%2Fj.physletb.2012.08.021.

[4] The CMS Collaboration. 'Observation of a new boson with mass near 125 GeV in pp collisions at $\sqrt{\{s\}}$=7 and 8 TeV'. In: *Journal of High Energy Physics* 2013.6 (June 2013). DOI: 10.1007/jhep06(2013)081. URL: https://doi.org/10.1007%2Fjhep06%282013%29081.

[5] The ATLAS Collaboration. '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 (2012), pp. 1–29. ISSN: 0370-2693. DOI: https://doi.org/10.1016/j.physletb.2012.08.020. URL: https://www.sciencedirect.com/science/article/pii/S037026931200857X.

[6] V. Fanti et al. 'A new measurement of direct CP violation in two pion decays of the neutral kaon'. In: *Physics Letters B* 465.1-4 (Oct. 1999), pp. 335–348. DOI: 10.1016/s0370-2693(99)01030-8. URL: https://doi.org/10.1016%2Fs0370-2693%2899%2901030-8.

[7] The UA1 Collaboration. 'Experimental observation of isolated large transverse energy electrons with associated missing energy at s=540 GeV'. In: *Physics Letters B* 122.1 (1983), pp. 103–116. ISSN: 0370-2693. DOI: https://doi.org/10.1016/0370-2693(83)91177-2. URL: https://www.sciencedirect.com/science/article/pii/0370269383911772.

[8] The UA2 Collaboration. 'Observation of single isolated electrons of high transverse momentum in events with missing transverse energy at the CERN pp collider'. In: *Physics Letters B* 122.5 (1983), pp. 476–485. ISSN: 0370-2693. DOI: https://doi.org/10.1016/0370-2693(83)91605-2. URL: https://www.sciencedirect.com/science/article/pii/0370269383916052.

[9] CERN. *The birth of the Web*. URL: https://home.web.cern.ch/science/computing/birth-web (visited on 27/10/2022).

[10] The CMS Collaboration, Thomas Mc Cauley and Lucas Taylor. *CMS Higgs Search in 2011 and 2012 data: candidate photon-photon event (8 TeV): 3D, r-phi and r-z transverse views*. CMS Collection. 2013. URL: http://cds.cern.ch/record/1606503.

[11] CERN. *CERN's accelerator complex*. URL: https://home.cern/science/accelerators/accelerator-complex (visited on 27/10/2022).

[12] Maurizio Vretenar et al. *Linac4 design report*. Vol. 6. CERN Yellow Reports: Monographs. Geneva: CERN, 2020. DOI: 10.23731/CYRM-2020-006. URL: https://cds.cern.ch/record/2736208.

[13] Lyndon Evans and Philip Bryant. 'LHC Machine'. In: *Journal of Instrumentation* 3.08 (Aug. 2008), S08001. DOI: 10.1088/1748-0221/3/08/S08001. URL: https://dx.doi.org/10.1088/1748-0221/3/08/S08001.

[14] The ATLAS Collaboration. 'The ATLAS Experiment at the CERN Large Hadron Collider'. In: *JINST* 3 (2008). Also published by CERN Geneva in 2010, S08003. DOI: 10.1088/1748-0221/3/08/S08003. URL: https://cds.cern.ch/record/1129811.

[15] The CMS Collaboration. 'The CMS experiment at the CERN LHC'. In: *Journal of Instrumentation* 3.08 (Aug. 2008), S08004. DOI: 10.1088/1748-0221/3/08/S08004. URL: https://dx.doi.org/10.1088/1748-0221/3/08/S08004.

[16] The ALICE Collaboration. 'The alice experiment at the CERN LHC'. In: *Journal of Instrumentation* 3 (Aug. 2008), S08002. DOI: 10.1088/1748-0221/3/08/S08002.

[17] The LHCb Collaboration. 'The LHCb Detector at the LHC'. In: *Journal of Instrumentation* 3 (Aug. 2008), S08005.

[18] CERN. *CERN Annual report 2021*. Tech. rep. Geneva: CERN, 2022. DOI: 10.17181/AnnualReport2021. URL: https://cds.cern.ch/record/2807619.

[19] Roderik Bruce et al. 'Reaching record-low β^* at the CERN Large Hadron Collider using a novel scheme of collimator settings and optics'. In: *Nuclear Instruments & Methods in Physics Research Section A-accelerators Spectrometers Detectors and Associated Equipment* 848 (2017), pp. 19–30.

[20] Maximilien Brice, Michael Hoch and Joseph Gobin. 'View of the CMS Detector before closure'. 2008. URL: http://cds.cern.ch/record/1133594 (visited on 01/02/2023).

[21] Suman Chatterjee. 'Jets as probes for precision measurement and candles for physics beyond standard model'. PhD thesis. Tata Institute of Fundamental Research Mumbai, 2020.

[22] The CMS Collaboration. *CMS Physics: Technical Design Report Volume 1: Detector Performance and Software*. Technical design report. CMS. There is an error on cover due to a technical problem for some items. Geneva: CERN, 2006. URL: https://cds.cern.ch/record/922757.

[23] David Barney. 'CMS Detector Slice'. CMS Collection. 2016. URL: https://cds.cern.ch/record/2120661.

[24] *The Phase-2 Upgrade of the CMS Tracker*. Tech. rep. Geneva: CERN, 2017. DOI: 10.17181/CERN.QZ28.FLHW. URL: http://cds.cern.ch/record/2272264.

[25] University of Zurich. *Simple example of 3d axes with spherical coordinates*. URL: https://wiki.physik.uzh.ch/cms/latex:example_spherical_coordinates (visited on 15/11/2022).

[26] Sergio Cittolin, Attila Rácz and Paris Sphicas. *CMS The TriDAS Project: Technical Design Report, Volume 2: Data Acquisition and High-Level Trigger. CMS trigger and data-acquisition project*. Tech. rep. Geneva, 2002. URL: http://cds.cern.ch/record/578006.

[27] CERN. *Triggering and Data Acquisition | CMS Experiment*. URL: https://cms.cern/detector/triggering-and-data-acquisition (visited on 15/11/2022).

[28] The CMS Collaboration. *CMS TriDAS project: Technical Design Report, Volume 1: The Trigger Systems*. Tech. rep. URL: http://cds.cern.ch/record/706847.

[29] The CMS Collaboration. 'Particle-flow reconstruction and global event description with the CMS detector'. In: *Journal of Instrumentation* 12.10 (Oct. 2017), P10003. DOI: 10.1088/1748-0221/12/10/P10003. URL: https://dx.doi.org/10.1088/1748-0221/12/10/P10003.

[30] The CMS Collaboration. 'Performance of the CMS muon detector and muon reconstruction with proton-proton collisions at $\sqrt{s}$=13 TeV'. In: *Journal of Instrumentation* 13.06 (June 2018), P06015. DOI: 10.1088/1748-0221/13/06/P06015. URL: https://dx.doi.org/10.1088/1748-0221/13/06/P06015.

[31] The CMS Collaboration. 'Performance of electron reconstruction and selection with the CMS detector in proton-proton collisions at $\sqrt{s} = 8$ TeV'. In: *Journal of Instrumentation* 10.06 (June 2015), P06005. DOI: 10.1088/1748-0221/10/06/P06005. URL: https://dx.doi.org/10.1088/1748-0221/10/06/P06005.

[32] W Adam et al. 'Reconstruction of electrons with the Gaussian-sum filter in the CMS tracker at the LHC'. In: *Journal of Physics G: Nuclear and Particle Physics* 31.9 (July 2005), N9. DOI: 10.1088/0954-3899/31/9/N01. URL: https://dx.doi.org/10.1088/0954-3899/31/9/N01.

[33] Andy Buckley et al. 'General-purpose event generators for LHC physics'. In: *Physics Reports* 504.5 (2011), pp. 145–233. ISSN: 0370-1573. DOI: https://doi.org/10.1016/j.physrep.2011.03.005. URL: https://www.sciencedirect.com/science/article/pii/S0370157311000846.

[34] Michael H. Seymour and Marilyn Marx. 'Monte Carlo Event Generators'. In: (2013). DOI: 10.48550/ARXIV.1304.6677. URL: https://arxiv.org/abs/1304.6677.

[35] J. Alwall et al. 'The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations'. In: *Journal of High Energy Physics* 2014.7 (July 2014). DOI: 10.1007/jhep07(2014)079. URL: https://doi.org/10.1007%2Fjhep07%282014%29079.

[36] C. Oleari. 'The POWHEG BOX'. In: *Nuclear Physics B - Proceedings Supplements* 205-206 (Aug. 2010), pp. 36–41. DOI: 10.1016/j.nuclphysbps.2010.08.016. URL: https://doi.org/10.1016%2Fj.nuclphysbps.2010.08.016.

[37] Christian Bierlich et al. 'A comprehensive guide to the physics and usage of PYTHIA 8.3'. In: (2022). DOI: 10.48550/ARXIV.2203.11601. URL: https://arxiv.org/abs/2203.11601.

[38] S. Agostinelli et al. 'Geant4—a simulation toolkit'. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 506.3 (2003), pp. 250–303. ISSN: 0168-9002. DOI: https://doi.org/10.1016/S0168-9002(03)01368-8. URL: https://www.sciencedirect.com/science/article/pii/S0168900203013688.

[39] Alexander L. Fradkov. 'Early History of Machine Learning'. In: *IFAC-PapersOnLine* 53.2 (2020). 21st IFAC World Congress, pp. 1385–1390. ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2020.12.1888. URL: https://www.sciencedirect.com/science/article/pii/S2405896320325027.

[40] David Silver et al. 'Mastering the Game of Go with Deep Neural Networks and Tree Search'. In: *Nature* 529.7587 (2016), pp. 484–489. ISSN: 0028-0836. DOI: 10.1038/nature16961.

[41] Matthew Johnston. *Biggest companies in the world by Market Cap*. 2022. URL: https://www.investopedia.com/biggest-companies-in-the-world-by-market-cap-5212784.

[42] Bernhard E. Boser, Isabelle M. Guyon and Vladimir N. Vapnik. 'A Training Algorithm for Optimal Margin Classifiers'. In: *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*. COLT '92. Pittsburgh, Pennsylvania, USA: Association for Computing Machinery, 1992, pp. 144–152. ISBN: 089791497X. DOI: 10.1145/130385.130401. URL: https://doi.org/10.1145/130385.130401.

[43] Frank Rosenblatt. 'The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain'. In: *Ideas That Created the Future: Classic Papers of Computer Science* (2021), pp. 183–190. DOI: 10.7551/mitpress/12274.003.0020.

[44] Xin Jin and Jiawei Han. 'K-Means Clustering'. In: *Encyclopedia of Machine Learning*. Ed. by Claude Sammut and Geoffrey I. Webb. Boston, MA: Springer US, 2010, pp. 563–564. ISBN: 978-0-387-30164-8. DOI: 10.1007/978-0-387-30164-8_425. URL: https://doi.org/10.1007/978-0-387-30164-8_425.

[45] Salim Dridi. *Unsupervised Learning - A Systematic Literature Review*. Dec. 2021. DOI: 10.13140/RG.2.2.16963.12323.

[46] Dor Bank, Noam Koenigstein and Raja Giryes. 'Autoencoders'. In: *CoRR* abs/2003.05991 (2020). arXiv: 2003.05991. URL: https://arxiv.org/abs/2003.05991.

[47] Adam Paszke et al. *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. 2019. DOI: 10.48550/ARXIV.1912.01703. URL: https://arxiv.org/abs/1912.01703.

[48] Gregor Kasieczka et al. 'Deep-learning top taggers or the end of QCD?' In: *Journal of High Energy Physics* 2017.5 (May 2017). DOI: 10.1007/jhep05(2017)006. URL: https://doi.org/10.1007%2Fjhep05%282017%29006.

[49] Josh Cogan et al. 'Jet-images: computer vision inspired techniques for jet tagging'. In: *Journal of High Energy Physics* 2015.2 (Feb. 2015). DOI: 10.1007/jhep02(2015)118. URL: https://doi.org/10.1007%2Fjhep02%282015%29118.

[50] E. Bols et al. 'Jet flavour classification using DeepJet'. In: *Journal of Instrumentation* 15.12 (Dec. 2020), P12012–P12012. DOI: 10.1088/1748-0221/15/12/p12012. URL: https://doi.org/10.1088%2F1748-0221%2F15%2F12%2Fp12012.

[51] G. Mörtl. 'DeepLepton: Muon identification in 13 TeV pp collisions at the CMS experiment using deep learning techniques'. MA thesis. Technische Universität Wien, 2019.

[52] Yue Wang et al. *Dynamic Graph CNN for Learning on Point Clouds*. 2018. DOI: 10.48550/ARXIV.1801.07829. URL: https://arxiv.org/abs/1801.07829.

[53] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2014. DOI: 10.48550/ARXIV.1412.6980. URL: https://arxiv.org/abs/1412.6980.

[54] John Duchi, Elad Hazan and Yoram Singer. 'Adaptive subgradient methods for online learning and stochastic optimization'. In: *Journal of Machine Learning Research* 12.Jul (2011), pp. 2121–2159.

[55] Eustace M. Dogo, Oluwatobi J. Afolabi and Bhekisipho Twala. 'On the Relative Impact of Optimizers on Convolutional Neural Networks with Varying Depth and Width for Image Classification'. In: *Applied Sciences* 12.23 (2022). ISSN: 2076-3417. DOI: 10.3390/app122311976. URL: https://www.mdpi.com/2076-3417/12/23/11976.

[56] Less Wright. *Ranger - a synergistic optimizer*. https://github.com/lessw2020/Ranger-Deep-Learning-Optimizer. 2019.

[57] Michael R. Zhang et al. *Lookahead Optimizer: k steps forward, 1 step back*. 2019. DOI: 10.48550/ARXIV.1907.08610. URL: https://arxiv.org/abs/1907.08610.

[58] Less Wright. Sept. 2019. URL: https://lessw.medium.com/new-deep-learning-optimizer-ranger-synergistic-combination-of-radam-lookahead-for-the-best-of-2dc83f79a48d (visited on 12/01/2023).

[59] David E. Rumelhart, Geoffrey E. Hinton and Ronald J. Williams. 'Learning Internal Representations by Error Propagation'. In: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*. Ed. by David E. Rumelhart and James L. Mcclelland. Cambridge, MA: MIT Press, 1986, pp. 318–362.

[60] *Backpropagation*. URL: https://brilliant.org/wiki/backpropagation/#deriving-the-gradients (visited on 13/01/2023).

[61] Sergey Ioffe and Christian Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. 2015. DOI: 10.48550/ARXIV.1502.03167. URL: https://arxiv.org/abs/1502.03167.

[62] Siddharth Sharma, Simone Sharma and Anidhya Athaiya. 'ACTIVATION FUNCTIONS IN NEURAL NETWORKS'. In: *International Journal of Engineering Applied Sciences and Technology* 04 (May 2020), pp. 310–316. DOI: 10.33564/IJEAST.2020.v04i12.054.

[63] Prajit Ramachandran, Barret Zoph and Quoc V. Le. *Searching for Activation Functions*. 2017. DOI: 10.48550/ARXIV.1710.05941. URL: https://arxiv.org/abs/1710.05941.

[64] Geoffrey E. Hinton et al. *Improving neural networks by preventing co-adaptation of feature detectors*. 2012. DOI: 10.48550/ARXIV.1207.0580. URL: https://arxiv.org/abs/1207.0580.

[65] John S. Bridle. 'Probabilistic Interpretation of Feedforward Classification Network Outputs, with Relationships to Statistical Pattern Recognition'. In: *Neurocomputing*. Ed. by Françoise Fogelman Soulié and Jeanny Hérault. Berlin, Heidelberg: Springer Berlin Heidelberg, 1990, pp. 227–236. ISBN: 978-3-642-76153-9.

[66] John Bridle. 'Training Stochastic Model Recognition Algorithms as Networks can Lead to Maximum Mutual Information Estimation of Parameters'. In: *Advances in Neural Information Processing Systems*. Ed. by D. Touretzky. Vol. 2. Morgan-Kaufmann, 1989. URL: https://proceedings.neurips.cc/paper/1989/file/0336dcbab05b9d5ad24f4333c7658a0e-Paper.pdf.

[67] Sylvain Gugger. *How do you find a good learning rate*. Mar. 2018. URL: https://sgugger.github.io/how-do-you-find-a-good-learning-rate.html (visited on 13/01/2023).

[68] *Baseline muon selections for Run-II*. URL: https://twiki.cern.ch/twiki/bin/viewauth/CMS/SWGuideMuonIdRun2#Tight_Muon (visited on 24/01/2023).

[69] *Multivariate Electron Identification for Run2*. URL: https://twiki.cern.ch/twiki/bin/viewauth/CMS/MultivariateElectronIdentificationRun2#General_information (visited on 24/01/2023).

[70] Simone Alioli et al. 'A general framework for implementing NLO calculations in shower Monte Carlo programs: the POWHEG BOX'. In: *Journal of High Energy Physics* 2010.6 (June 2010). DOI: 10.1007/jhep06(2010)043. URL: https://doi.org/10.1007%2Fjhep06%282010%29043.

[71] Torbjörn Sjöstrand et al. 'An introduction to PYTHIA 8.2'. In: *Computer Physics Communications* 191 (June 2015), pp. 159–177. DOI: 10.1016/j.cpc.2015.01.024. URL: https://doi.org/10.1016%2Fj.cpc.2015.01.024.