NIELS BOHR INSTITUTTET
UNIVERSITY OF COPENHAGEN

October 15, 2021

Convolutional neural networks for charged particle detection using Hough transform in the ATLAS detector at HL-LHC

Marcus M. Thomsen

Advisor: Stefania Xella

Contents

1 Introduction **4**

2 High Luminosity Large Hadron Collider (HL-LHC) **5**

2.1 The ATLAS Detector . . . 5
2.2 Data acquisition . . . 8
2.3 Collision simulations . . . 11
2.4 Hough Transform . . . 12

3 Artificial neural networks (ANN) **16**

3.1 Loss functions . . . 17
3.2 Convolutional neural networks (CNNs) . . . 18

4 HTT Simulation **21**

4.1 Data . . . 23

5 Results **25**

5.1 Current road finding optimization method . . . 25
5.2 Suggested road finding method . . . 25
5.3 Method evaluation . . . 26
5.4 Model training . . . 27
5.5 Target finding . . . 27
5.6 Model setup . . . 31
5.7 Single layer CNN . . . 32
5.8 Two layer CNN . . . 34
5.9 Duplicate removal . . . 38
5.10 Method analysis . . . 41
5.11 Network analysis . . . 44

6 Conclusions and outlook **49**

7 Appendix **53**

7.1 App1: Loss convergence plots . . . 53
7.2 App2: Network filters . . . 55

Abstract

Due to increased luminosity and bunch crossing rate at the HL-LHC an improvement to the trigger system at the ATLAS detector is needed. In this thesis convolutional neural networks are suggested to improve charged particle candidate finding in Hough transformed images of hits in the Inner tracker. The results presented are based on simulated proton-proton collisions and single muons within a region $0.3 < \phi_0 < 0.5$, $0.1 < \eta < 0.3$. As an example at 99% the average candidate count is found to decrease from 216 to 97 by using a two-layer CNN method compared to the current. This is a significant decrease that could have a significant influence on the viability of the Hough transform method when considering the future trigger system for ATLAS at the LHC.

1 Introduction

Due to an upgrade at the Large Hadron Collider (LHC) the mean number of interactions per bunch-crossing is expected to increase dramatically. Therefore new fast hardware based methods are needed in the Trigger Data Acquisition (TDAQ) process of the ATLAS detector to handle the high event frequency and pile up per event. For this a Hardware Tracker for the Trigger (HTT) has been suggested to implement fast tracking based on Inner Tracker (ITk) data. This is described in section 2.

Before fitting clusters of ITk hits to tracks in the HTT a fast algorithm is needed to narrow down what clusters to fit to or the computation time would explode. As an alternative to the currently intended application-specific integrated circuits (ASICS) matching clusters of ITk hits to pattern banks, a Hough transform based method described in section 2.4 has been suggested[1] for fast charged particle track identification. However, this Hough transform method still leaves a rather high amount of possible charged particle tracks to fit to. A reason for this might be that the information hidden in the Hough transformed images is not fully utilized in the current implementation of the HTT simulation described in section 4.

For this reason this thesis suggests using a convolutional neural network (CNN) for image recognition in section 3 to increase both precision and accuracy of charged particle track identification in Hough transformed images. By reducing the time spent on track fitting more of the relevant data can be saved in the DAQ-process. This will eventually increase the precision with which physics process of interest can be probed.

As the new High Luminosity-LHC and the upgraded ATLAS detector are not yet built this thesis is based on simulated data of muons and a mean number of $\mu_{pp} = 200$ proton-proton collisions per bunch crossing. Section 5 presents results for the current and suggested track finding methods. As an example is found that the number of charged particle track candidates can be reduced to less than half from 216 to just 97 within the investigated region $0.3 < \phi_0 < 0.5$, $0.1 < \eta < 0.3$ without reducing efficiency from 99% in finding muon tracks by using a 2-layer CNN method.

Finally in sections 5.10 and 5.11 the method is analyzed and is found not to have any significant biases at the measured precision. The performance has been studied in details and found to be stable in kinematic parameters.

2 High Luminosity Large Hadron Collider (HL-LHC)

Across the border between Switzerland and France lies the Large Hadron Collider at the CERN laboratory. The collider is so named because it is built to collide hadrons at very high energies (latest at $\sqrt{s} = 13$ TeV in run 2). In this thesis we shall focus on proton-proton collisions.

The probability of an inelastic event between the colliding particles is dependent both on the couplings between the colliding particles and, for bound objects like hadrons, the parton distribution functions. In a classical analogue this can be described as a tiny cross section for a beam to hit for the interaction to happen. The luminosity is the number of incoming particles per area per time interval and is thus directly proportional to the number of interactions. The total number of events over a time period can then be described as $n_{events} = \sigma \int Ldt$ were L is the luminosity and σ is the cross section. In collision experiments cross sections are often described in barn, $b = 10^{-28}m^2$.

After the end of the current run the LHC is going to be upgraded, the so called "Phase II upgrade". This upgrade will increase the collission energy to $14TeV$, but the primary upgrade is an increase in the beam intensity called luminosity. After the upgrade the LHC will be renamed the High Luminosity LHC (HL-LHC). This is because the luminosity will increase way beyond its current max of $L = 2 \cdot 10^{-5} \frac{1}{fb \cdot s} = 6 \cdot 10^2 \frac{1}{fb \cdot year}$ in the second half of 2017 to $L = 7.5 \cdot 10^{-5} \frac{1}{fb \cdot s} = 2.5 \cdot 10^3 \frac{1}{fb \cdot year}$ in the ultimate configuration, or about 4 times as much as the current max[2]. As the number of interactions depends on the luminosity the mean amount of interactions pr. crossing μ_{pp} will also increase. Specifically μ_{pp} will increase beyond its max of up to 60 in second half of 2017 to up to 200 in the ultimate configuration[2]. This due in part to an increased number of protons per bunch and in part an improved focusing of the beam[1].

The increase in luminosity is important as it greatly increases the amount of data gathered for studied processes. But it also puts even higher demands on the detectors and the data acquisition. Colliding hadrons allows for very high collision energies, but it comes at a price. Hadrons are affected by the strong force which has a very high coupling constant for low energies which creates showers of particles in the detector. All these events are called "minimum bias" and result in a huge "pile up" of data in the detectors per event. This along with a bunch crossing rate of 40 MHz gives an expected collision frequency of 8 GHz worth of proton-proton collision pile up for the ultimate configuration. This huge increase in data naturally puts very high requirements on the detectors precision to seperate objects and the data acquisition speed to be able to save events of interest. In the following chapter the ATLAS detector will be described followed by a chapter on how the data acquisition will be updated to cope with the increased data frequency.

2.1 The ATLAS Detector

The ATLAS Detector is one of the detectors at the LHC gathering data from the collisions. It is a 44 meters long cylinder with a diameter of 25 meters. At the HL-LHC it consists of the inner tacker (ITk) surrounded by two calorimeters

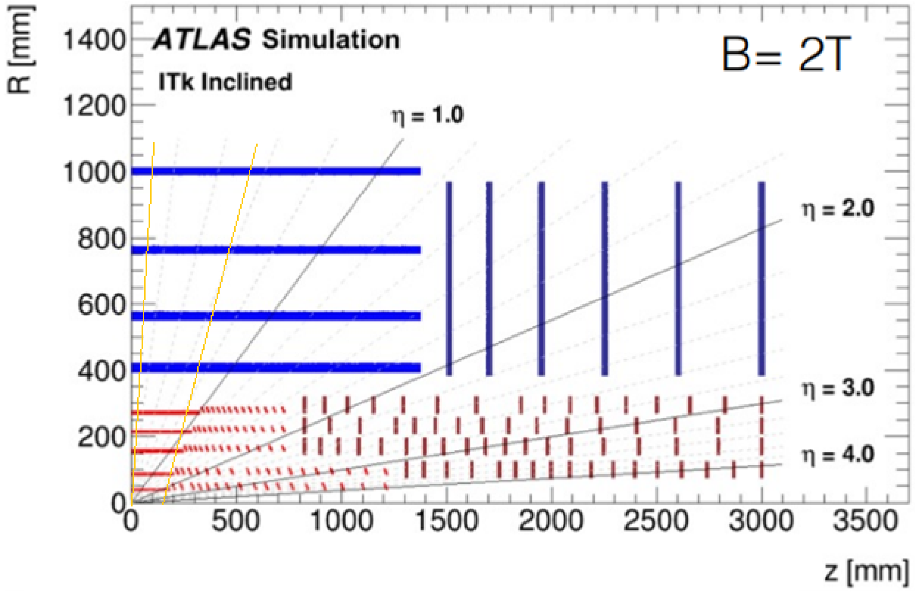

Figure 1: Drawing of the inner tracker, implied symmetric rotating around z and mirrored in $z = 0$. Red lines are pixel layers. Blue lines are double strip layers. The yellow lines show the area investigated, $|z| < 150mm$, $0.1 < \eta < 0.3$. B stands for the magnetic field in the ITk which is a uniform field of $2T$ along the beam axis. The magnetic field makes charged particles rotate in the plane transverse to the beam axis. Image taken from [3] and modified by me.

which are again surrounded by magnets[1]. In addition to these cylindrical subdetectors there are end caps, but as this paper will focus on charged particles in the $0.1 < \eta < 0.3$ region these are of no relevance. Here η is the pseudo rapidity, $\eta = -\ln\theta/2$ where θ is the angle down to the z-axis. As marked with yellow in figure 1 the end caps of the ITk are outside the area of study.

This project is focused on charged particle detection from clusters of hits in the ITk. This is especially important since one of the ways that the ATLAS collaboration intends to deal with this large increase in pileup is through tracking. This decision is based on two main arguments. The first being that a track fit gives parameters, especially the transverse momentum p_T, that can be used to characterize whether the found particle is of interest. Additionally though, tracks can be used to find primary vertices of interaction which is a robust way of grouping particles originating from the same event. As the current detector is fried by radiation it is therefore even more important to find a good replacement. The inner tracker used for Phase II will be an all new unit of silicon detectors. The new detector covers a larger pseudo-rapidity range than the previous (up to $\eta \sim 4$ as shown in figure 1). The ITk-barrel has five layers of pixel detectors closest to the beam surrounded by four double sided layers of silicon strip detectors outside these[2].

Figure 1 is an illustration of the ITk. The five red lines in the bottom left corner are the pixel layers and the four blue above are the silicon strip layers. The ITk is positioned in a uniform magnetic field of 2T pointing along the beam

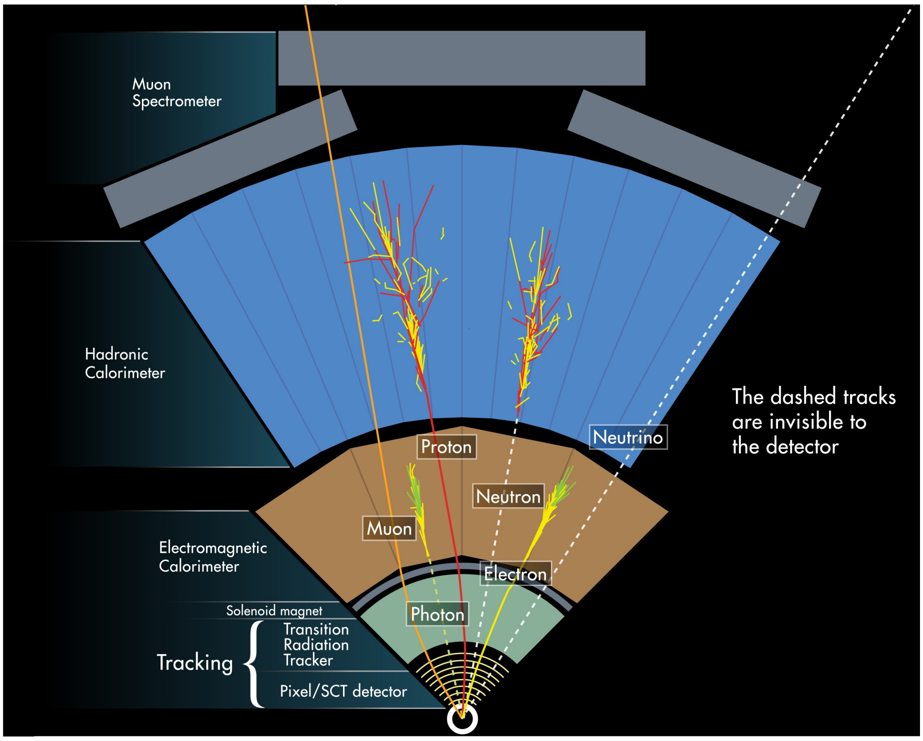

Figure 2: The image shows where different kinds of particles are detected in the ATLAS detector. This image describes the detector before the Phase II upgrade. Therefore all that is currently marked as ”tracking” should be interpreted as the Phase II ITk shown in figure 1. It shows that the charged tracks are detected and bent in the ITk. The proton should be interpreted as an example of long lived charged hadrons, i.e. protons, charged kaons an especially charged pions and corresponding anti particles. The image should be interpreted as showing the particles in the detector within some (z, η)-region all projected down on the transverse plane. Image taken from [6].

axis[3]. This is to help charged particle detection as moving charged particle tracks are bent in a magnetic field according to their charge and momentum.

The strips are down to 24mm long meaning that the barrel strips each have a rather low resolution along the z-axis. This is compensated for by adding them in layer pairs that are slightly tilted compared to each other (3^o) to also enable identification of the z-component[2]. They can also be used individually though when the exact z-position is not of importance. In the Hough transform described in chapter 2.4 the strip layers will be used as individual layers as the exact z-position is of lesser importance. This is because the movements of the charged particle in the transverse plane is independent on the movement in the z-direction when the magnetic field is uniform pointing along z. The pixels are so small that a single pixel hit is sufficient to give a good spatial bound on the hit point.

The pixels and strips detect electromagnetically charged particles passing, that is charged hadrons and charged leptons except for those that decay within

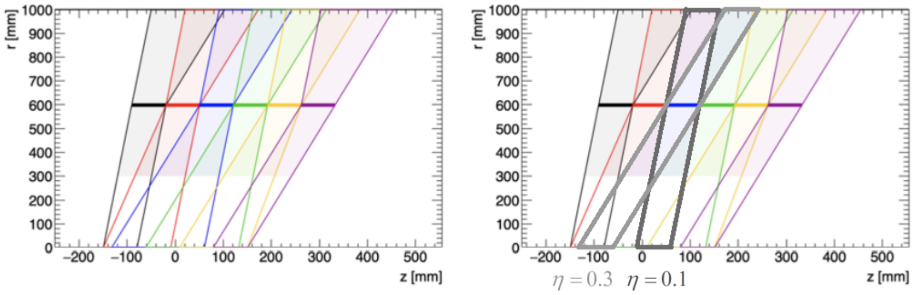

Figure 3: The left hand side of the figure shows so called ”key layer slicing”. Each colored region is a region with a starting point within $|z| \leq 150mm$ and η within 0.1 to 0.3. This key layer slicing is focused around the second pair of strip layers. The light grey area of the right hand figure corresponds to the $\eta = 0.3$ region covered by the blue slice of the left hand figure while the dark grey area shows the $\eta = 0.1$ region covered by the blue slice. Image taken from [5] and modified by me.

the beam pibe illustrated by a white circle of figure 2.

Looking further into the layers of figure 2 the innermost calorimeter is the electromagnetic calorimeter which stops and measures the energy of photons and electrons. The outermost calorimeter is the hadronic calorimeter stops and measures the energy of charged hadrons.[1]. The outermost magnets are for muon detection. The muon spectrometer hits along with energy deposits in the calorimeters is what is currently used to trigger on interesting events. Adding information from the ITk will therefore correspond to a whole new trigger level.

The problem remains however that analyzing the incoming data from each bunch crossing is made difficult due to the increased amount of minimum bias pile up. A way to deal with this problem is to split the data accumulated in the detector into multiple parts when analyzing. In this project six so called slices are used in z and η as shown in figure 3. Each of these coloured slices contains all particles with a certain η value going through some z-region of the so called ”key layer” chosen. The second pair of strip layers is used as this key layer. Focusing on the blue slice on the left hand side of figure 3, the right hand side of figure 3 shows that it covers $\eta = 0.3$ within the starting region $-130mm \lesssim z_0 \lesssim -60mm$ and $\eta = 0.1$ within the starting region $-10mm \lesssim z_0 \lesssim 60mm$ where the index 0 is to indicate that it describes z at $r = 0$.[5]

Originally slicing was based on the beam axis as the key layer but the new slicing has been introduced so as to reduce overlap between slices and thereby minimize the pile up for each slice. Considering that the amount of pile up scales with the size of the detector region, slicing clearly constitutes a significant decrease in the amount of pile up per investigated region at the cost of some overlap.

2.2 Data acquisition

As the mean number of interactions per crossing will increase from up to 60 to up to 200 at a frequency of 40 MHz this is a huge increase to the amount of data that has to be saved. Just continuing with the same trigger requirements would

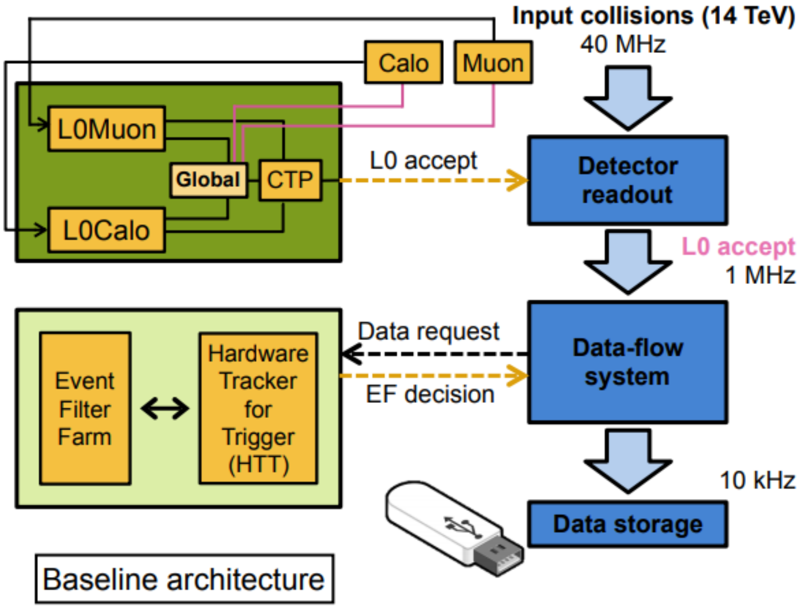

Figure 4: A simplified illustration of the TDAQ system. Events come in at a rate of 40 MHz and are reduced to 1MHz by the Level-0 trigger using information form the calorimeters and muon spectrometers. The event filter, now also drawing information form the ITk, does additional processing to reduce the event rate to 10kHz. The event rate is hereby reduced by a factor of 400 in total by the trigger system. Image taken from [7].

mean that one would have to make very large cuts on p_T to still be able to store data and thereby loose much of what is gained by the increase in luminosity. In fact ATLAS aims to improve the Trigger system so much that the p_T cuts can stay similar to the current or even decrease for some trigger parameters.[7]

In general hardware implementations run faster than software. Therefore when the input rate of events is at 40 MHz and you don't have infinite money for general purpose computing farms you have to run it in specialized hardware. The exact choice of design for the trigger system is not final, but the baseline trigger system of the Trigger and Data Acquisition (TDAQ) is two-leveled. It contains a first rough Level-0 trigger followed by the event filter as the second layer of the trigger system. A rough sketch of the TDAQ is drawn in figure 4.

As the input is of rate 40 MHz the first trigger level-0 has to be extremely fast. It uses hits in the muon spectrometers and energy deposits in the calorimeters to do this first sorting of events. The event filter is a processor farm that is shown in the lower left hand corner of figure 4 to be assisted by the Hardware Tracking for the Trigger (HTT). The HTT consists of hardware processors such as field programmable gate arrays (FPGAs) and ASICs and uses hits from the ITk to perform quick charged particle track finding implemented in hardware to optimize the speed. ASICs especially are cost efficient when many are used, but unprogrammable. The design and implementation of the event filter is not yet final, but the algorithms studied in this thesis, described in sections 2.4 and

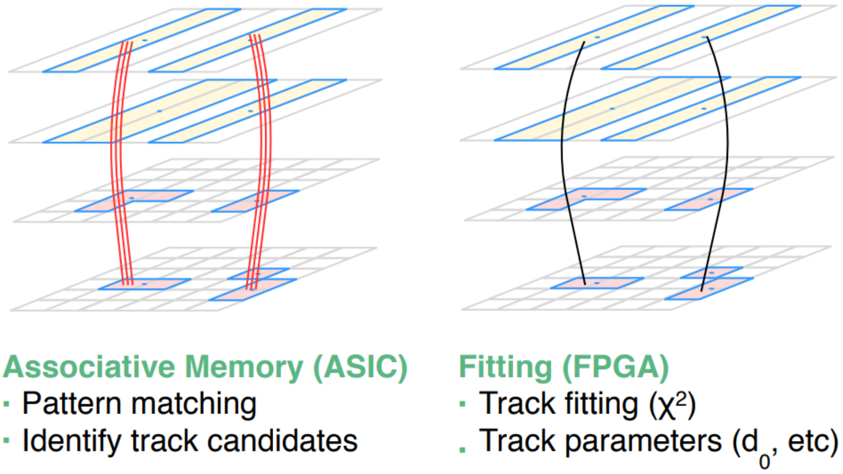
Associative Memory (ASIC)
· Pattern matching
· Identify track candidates
Fitting (FPGA)
· Track fitting (χ²)
. Track parameters (d0, etc)

Figure 5: An illustration of the two main steps of the HTT. In the ASICs the HTT performs pattern matching between the clusters in the ITk and the pattern bank. When track candidates are found the FPGAs perform track fitting. Image taken from [7].

5 can be used in any design.

The HTT consists of two parts, the regional (rHTT) and the global (gHTT). As the names suggest gHTT performs tracking in the whole detector while the rHTT performs tracking in specific regions specified by the L0-trigger. The gHTT and the rHTT can perform these tasks for $p_T > 1GeV$ at 100 kHz and $p_T > 2GeV$ at 1 MHz respectively[2].

As the rHTT is able to work at a higher rate than the gHTT there exists an evolved system where the rHTT is used as an individual trigger level (level 1) before the Event filter. This is intended for if the pile up of data proves too large. Then the data rate after the level 0 trigger of the ITk will be allowed to be up to 4 MHz, and the L1 trigger which shall then reduce the data rate to 6-800 kHz before the Event Filter[2].

The HTT is to consist of two main steps. After first combining hits to clusters these are matched to predefined patterns in associative memory (AM) ASICs. This is done using pattern banks of simulated muon hits in the ITk.[1] An illustration of the procedure is shown in the left hand side of figure 5. There the rectangles and squares represent clusters of hit strips and pixels respectively. The red lines represent the simulated muon match from the pattern bank using eight layers including the outer strip layers and at least 1 pixel layer. If a match is found the event passes as a candidate track for track fitting. Track fitting is done in FPGAs in the HTT.

In [1] using the Hough transform is suggested as an alternative to the pattern matching AM ASICs in case the ASICs become delayed or perform worse than expected.

Track fitting is a task whose time consumption scales very quickly with the amount of combinations of points to be fitted to. Therefore it is important to make a rough search for possible charged particle tracks to reduce the amount of possible hits used for track fitting. Instead of using a bank of patterns the Hough transform identifies charged particle tracks in a magnetic field as crossing lines in a transformed space described in section 2.4.

2.3 Collision simulations

To study the performance of the Hough transform algorithm described in section 2.4 on ITk input, test data is needed. Naturally as the HL-LHC is not yet built this will have to be simulated. The simulations consist of two parts. The first part is the collision simulation done in Pythia8, see documentation at [8]. This gives the outcomes of the $\mu_{pp} = 200, \sqrt{s} = 14$ TeV proton-proton collisions. The second part of the simulation is the particle behavior inside the detector (the ITk specifically) including subsequent decays. This is done in Geant4, see documentation at [9].

In this project we use two sets of input files. The first is a signal file. As just explained we intend to find charged particle tracks in the detector and thus the signal file should consists of long lived charged particles. This includes leptons (electrons and muons), mesons (charged pions and kaons primarily) and baryons (protons) along with all their corresponding antiparticles. A file of muons is chosen as these behave rather nicely in general, but a file of any of the previous particles had been valid. These simulated muons are made using Geant4 only. They are made to be flat in the region $0.3 < \phi_0 < 0.5$, $0.1 < \eta < 0.3$ (where ϕ_0 is the starting angle in the transverse plane) and $1/p_T$, where $1GeV \leq p_T \leq 800GeV$. The files include the true muon track parameters along with the corresponding detector hits.

The second is a minimum bias file. This is to include the result of 200 proton proton collisions at $\sqrt{s} = 14$ TeV to be used as background. A low percentage of the elements in these collisions is charged particles of the sort mentioned earlier. An even lower percentage actually fall into the η, z region within which we search. Those that do not will leave seemingly random tracks in the lower layers or traverse multiple slices. However, the vast majority of ITk hits come from so called ”soft QCD” particles, i.e. low p_T hadrons, as the strong force is extremely strong at low energies. These bend so much in the electromagnetic field of the ITk that they will often only hit few of the ITk layers or hit them at very different angles in the transverse plane.

Such interactions are the result of strong force interactions and parton distributions that are not possible to calculate directly from the standard model. Therefore there is naturally some difference between a simulated and a real dataset, but as seen in [8] the error on the differential cross sections are rather small at $\sqrt{s} = 13$ TeV and we shall therefore assume that these errors are also reasonably small at $\sqrt{s} = 14$ TeV. After creation in Pythia8 these particles are also simulated in the detector using Geant4 returning them as hits and corresponding tracks.

To these files are also added muons for all η and all ϕ and has $p_T = 10GeV$. As this muon is useless for testing because it rarely falls within the area of investigation $(0.1 \leq \eta \leq 0.3, 0.3 \leq \phi_0 \leq 0.5$) and only has one p_T value this is manually removed by removing clusters in which this is the major p_T contributor. When other particles occasionally made hits contributing to the same cluster this might add a tiny error, but in combination with the rarity that the muon even falls within the region of interest this is completely irrelevant.

Now talking about the use of particle hits in the ITk, clusters are combinations of hits that are identified as most likely originating from the same object. An example can be seen in figure 5. Focusing on the bottom left layer one can see 5 hit pixels, but only two matching track patterns. That is because the five hits are identified as only two individual clusters.

Having now described the data acquisition, the signal and the background we are ready to present the Hough transform used for charged track identification using input clusters in the ITk.

2.4 Hough Transform

As mentioned this thesis will focus on the Hough transform as an alternative to the AM ASICS for first sorting of hits when searching for charged particle tracks in the ITk. The Hough Transform is a reparameterization of the problem. For our problem at hand we shall focus on the circular Hough transform which is used to detect circles from points. We shall let the points be the collision point paired with the detector clusters, and the circles are the circular paths that the charged particles traverse in a uniform magnetic field inside the ITk transverse to the beam axis. If multiple detector clusters correspond to a similar circular path, they might originate from a charged particle traveling through the ITk! This way long lived charged particles of sufficient transverse momentum can be detected.

Assuming a particle to be a primary particle, it must be created on the beam axis, that is at the origin of the transverse plane. If the particle makes a cluster — ? in the ITk we will then have two points of the particle track, the origin and the cluster. Using the equation of the circle:

$$r^2 = (x-a)^2 + (y-b)^2 \tag{1}$$

We can use the equation of the circle for these two points to find the parameters of the circular motion:

$$\begin{aligned}&x^2 + y^2 = (x-x_i)^2 + (y-y_i)^2 \\ &\Updownarrow \\ &2r = \tfrac{r_i}{\sin\phi_i \sin\theta + \cos\phi_i \cos\theta}\end{aligned} \tag{2}$$

Where we used the polar coordinates $(x, y) = (r\cos\theta, r\sin\theta)$. In this equation (x_i, y_i) or (r_i, ϕ_i) are the coordinates of the cluster and (r, θ) are the polar coordinates for the center of circular motion all shown in figure 6.

It can be seen from figure 6 that the starting angle ϕ_0 of the particle track can be found from θ: $\theta = \phi_0 - \pi/2$. Inserting this into equation 2 and using some trigonometric identities we get:

$$r(\phi_0) = \frac{r_i}{\sin(\phi_0 - \phi_i)} \tag{3}$$

Equation 3 is the Hough transform of the problem. However, we'd like to also relate the radius of the circular motion to a kinetic variable of the particle. For this we will use that the ITk is positioned in a uniform magnetic field along the beam-axis (z).

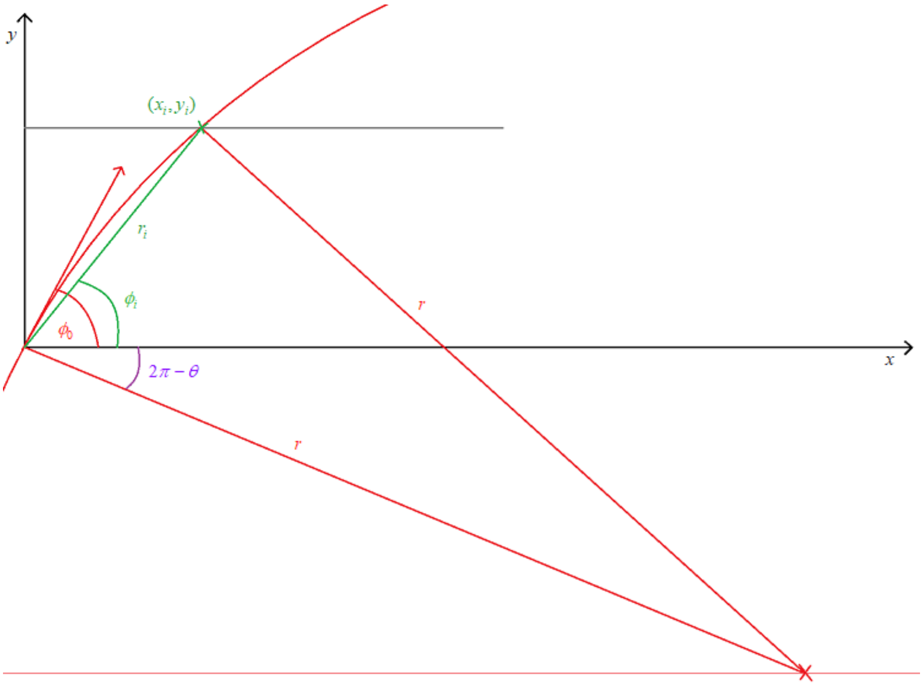

Figure 6: The red is the circle of motion for the particle. (r, θ) is the coordinates of the center of circular motion. The green line refers to a hit in a detector layer and (x_i, y_i) and r_i, ϕ_i describes its position. ϕ_0 is the starting angle of the particle moving from the origin. It is seen that $\theta = \phi_0 - \pi/2$

.

In a magnetic field with negligible electrical field the Lorentz force on a charged particle is described by:

$$\bar{F}_{EM} = q\bar{v} \times \bar{B} \tag{4}$$

where q is the particle charge, $\bar{v}$ is the particle velocity and $\bar{B}$ is the magnetic field of the detector.

Assuming that no other force affects the particle, it will move in spirals perpendicularly to the direction of the uniform magnetic field. As the cross product between two parallel vectors is 0 we can allow ourselves to extract and ignore the z-component of the particles spiraling movement as $\bar{B} = B\hat{z}$ and focus on the movement in the transverse plane.

$$\bar{v}_T = v_T\hat{\phi} \tag{5}$$

Inserting again into equation 4:

$$\bar{F}_{EM} = qv_T B\hat{r} \tag{6}$$

As this is the force creating the circular motion it must be equal to the centripetal force in the plane of circular motion:

$$\begin{aligned} &\bar{F}_C = \tfrac{p_T v_T}{r}\hat{r} \\ &\Updownarrow \\ &qBr = p_T \end{aligned} \tag{7}$$

We have thus established the relationship between the transverse momentum and charge of the particle and the radius of its circular movement. This can now be combined with the Hough transform of equation 3:

$$B\frac{q}{p_T} = \frac{\sin(\phi_0 - \phi_i)}{r_i} \tag{8}$$

Which is the Hough transform in kinematic variables $(\phi_0, q/p_T)$ for a detector hit (ϕ_i, r_i) from a primary charged particle in a uniform magnetic field. As the particle momenta that we are looking for are quite high $(p_T >= 1GeV)$ $\phi_0 - \phi_i$ is in general rather small which means that the lines of the Hough transformed image are very close to straight lines:

$$B\frac{q}{p_T} \approx \frac{\phi_0 - \phi_i}{r_i} \tag{9}$$

From which it is also seen that q/p_T always grows with ϕ_0 for high p_T. Also the constant r_i specifying the radius from the origin of the hit means that for high radii q/p_T grows more slowly with respect to ϕ_0 than at low radii. Focusing on hits in ITk layers this means that clusters in the innermost layers of the tracker will create steeper lines. Therefore clusters from the same charged particle hitting different layers will make lines of different slope coefficient crossing in the same point corresponding to the true $(\phi_0, q/p_T)$-value of the track.

A made up example of use is shown in figure 7. The figure illustrates a charged particle track producing five clusters in different layers of the ITk. The blue area shows the bins corresponding to 4-5 hit layers while the green area shows the bins corresponding to 3 hit layers. The fact that multiple lines cross the same $(\phi_0, q/p_t)$ bin indicates that these clusters originate from a charged particle track following a circular path in the uniform magnetic field. In general the higher bin count the more likely it is that they originate from a charged particle traveling through the ITk with the corresponding $(\phi_0, q/p_T)$.

Data is accumulated within some region in η, z. If one uses the entire range $|z| \leq 150mm$, $0.1 < \eta < 0.3$ this corresponds to a lot of accumulated background noise. Therefore slicing in z and η is used as described in 2.1.

An example of a muon and the same muon inside minimum bias can be seen in figure 8. The images are represent Hough transformed images of clusters in the ITk. The bin count is the amount of lines from different hit layers crossing that bin. While identifying the charged particle track seems like a very straight forward task on the left hand image of figure 8, it gets a bit more complicated in the right hand image.

One can go straight by each bin count, but as seen in the single muon image it is not just the central bin, but an entire area that has high intensity. Multiple machine learning methods have been developed for pattern recognition, but especially convolutional neural networks (CNNs, see [10] for a general introduction) have become almost synonymous with image recognition as they are excellent at combining and interpreting information of a spatial nature. This will therefore be introduced in the next section as a way to improve charged particle finding in Hough transformed images of ITk clusters.

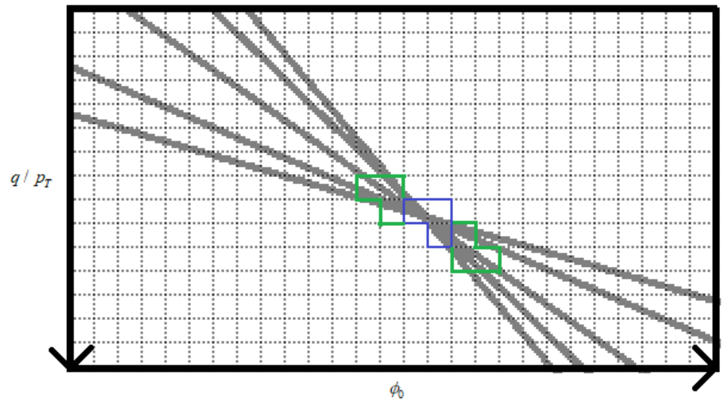

Figure 7: The figure shows an example Hough Transform image of a charged particle in a detector leaving 5 clusters of hits in different detector layers. The grid is the binning in $(\phi_0, q/p_T)$ and the arrows point towards higher q/p_T and ϕ_0. Each hit corresponds to a relation between charge and momentum and starting angle described by equation 8. When the lines have different slope in the Hough Transformed image it means that they originate from different ITk layers. Each bin can be assigned a count equal to the amount of clusters from different layers corresponding to that bin. In this figure the blue area marks all bins corresponds to 4-5 hit layers and the green area marks all bins with 3.

.

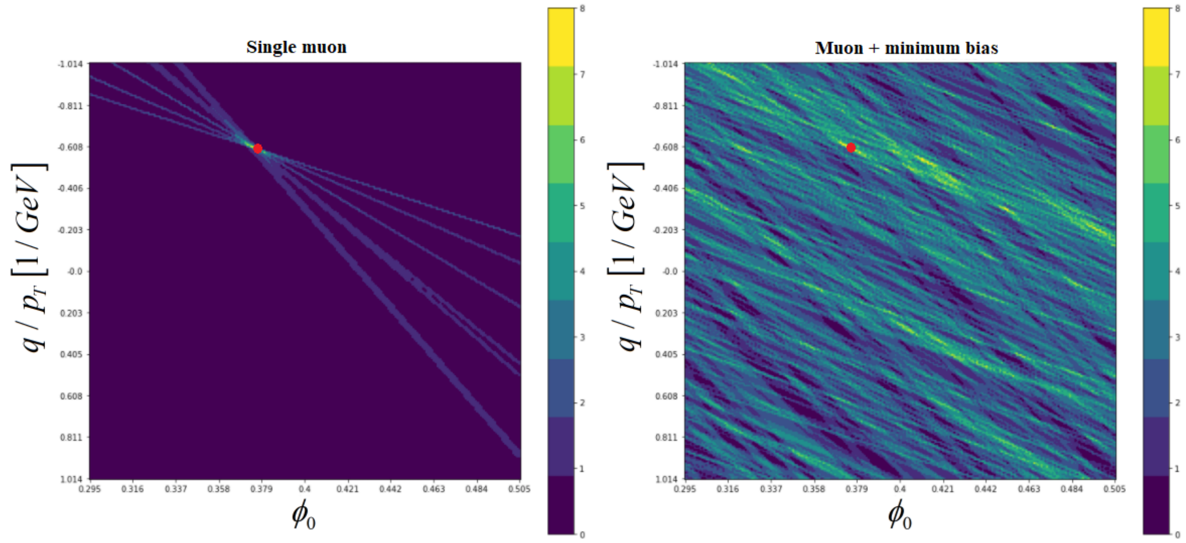

Figure 8: The left hand image shows a lone muon in a Hough transformed image, summed so that the bin counts represent the number of lines from different hit layers crossing that bin. The red point is the target used for training, see 3.2. The right hand image is a Hough transformed image of the same muon in minimum bias. It is seen from this image that more high intensity regions exist than that of the added muon.

.

3 Artificial neural networks (ANN)

To efficiently recognize charged particle-like patterns from the Hough Transformed image an artificial neural network is applied. An artificial neural network (ANN) or just neural network for short (NN) is a very diverse category of machine learning methods, but we shall here focus on the simplest types called feed forward neural networks (FFNN).

Put generally a NN is any algorithm that maps $\bar{x} \in R^n \rightarrow \bar{y} \in R^m$ through a series of linear transformations, often separated by nonlinear functions. Iteratively this can be written as:

$$x_j^{l+1} = f^{l+1}\left(W_{j,i}^{l+1} x_i^l\right) \tag{10}$$

In which x_i^l is the neurons(the vector) of layer l and $W_{j,i}^{l+1}$ and f^{l+1} are the linear transformation and the activation function between layers l and $l+1$ respectively. $x_i^{l=0}$ is the input and $y_j = x_j^{l=L}$ is the output of the network. In figure 9 is illustrated a simple FFNN of three dense layers. A dense layer is a layer where each element of the layer is allowed to depend on all elements of the previous layer. I omitted the bias parameter which we absorbed into $W_{j,i}^{l+1}$ for notational convenience. The bias is simply another model parameter that is added to every neuron of the new layer independently of the previous layer. Comparing figure 9 to our problem x_i^0 corresponds to the Hough transformed image and y_m corresponds to some output determining which bins correspond to muons and which do not.

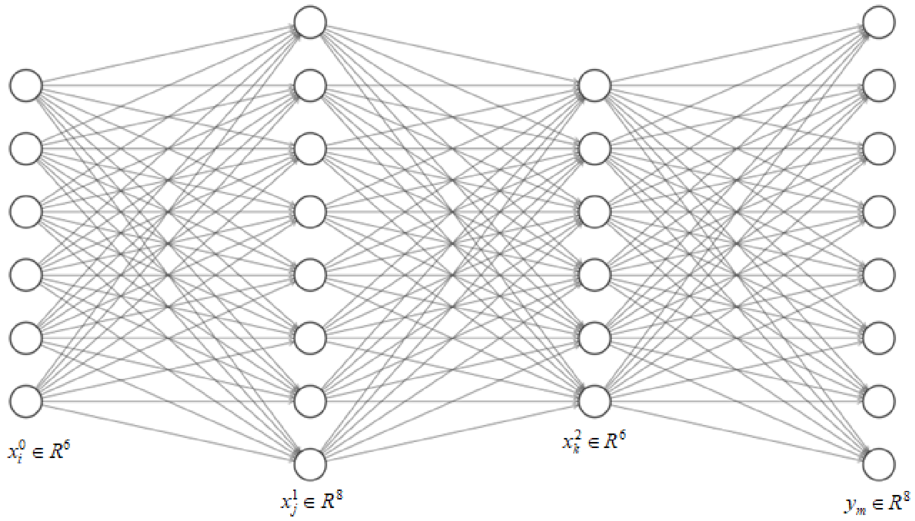

Figure 9: This example FFNN maps $x_i^0 \in R^6 \rightarrow y_m \in R^8$.

.

As any sequence of linear transformations is also a linear transformation an n-layered network without any nonlinear transformations can be described by a single layer transformation. The only way to model non-linear dependencies is thereby to use nonlinear activation functions. I shall call the number of layers in the model the ”depth” of the neural network and the combination of layer count, neuron count prer layer and nonlinear activation functions ”model complexity”.

The goal of a neural network is in general to “best” produce a target from the input. The target is completely dependent on the problem. The purpose at hand is to identify a charged particle track and this physics signal is to be represented as a target. This can be seen as a classification problem, which would in the simplest case correspond to mapping the entire input onto a vector with length equal to the number of classes. For instance, a vector of input parameters could either originate from a charged particle, or not originate from a charged particle giving two output classes in all. “Is” and “is not”. What exactly to choose as target proved to be non-trivial and is further discussed in 5.

Having defined now a problem of identification the meaning of “best” producing a target becomes more clear. The best model is the one that minimizes some error between the output of the model y_i and the target truth t_i.

3.1 Loss functions

There are multiple words used for these functions describing the error between output and target, but most often “loss function” is used. The purpose however is to describe how large of an error we make by trusting the model output (or the ”loss” of information).

The loss function used depends on the problem and it is not necessarily trivial to choose a suitable loss function. However for a classification problem it is well established to use the negative cross entropy:

$$Er\left(y,t\right) = -\sum_i t_i \log y_i \tag{11}$$

Which can be interpreted as “the measure of surprise”. That is, if we interpret y_i as the probability given the model that the event is of the i’th class then the cross entropy describes the overlap between the two distributions and thereby how “surprised” (or not) the model is by the truth.

For the model output y_i to be properly interpreted as a probability it ought naturally to be normalized such that $0 \leq y_i \leq 1$ and $\sum\limits_i y_i = 1$. One could choose any normalization satisfying this requirement, but it is conventional to use the softmax function:

$$y_i = \frac{e^{a_i}}{\sum\limits_j e^{a_j}} \tag{12}$$

Where a_i should be interpreted as the output of the last layer, just before activation function. As there is, in general, no analytical solution to finding the lowest loss a gradient descent based method is used for the minimum search. A gradient descent uses the negative gradient for the current model to move towards lower values and eventually a minimum. If we list all the model parameters into one vector $\theta = \begin{bmatrix} W^0_{0,0} \\ W^0_{0,1} \\ \vdots \\ W^L_{I,J} \end{bmatrix}$ we can in the simplest form write the stochastic gradient descent method:

$$\theta_{s+1} = \theta_s - \eta \nabla_\theta Er(\theta_s, x, t) \tag{13}$$

Where θ are the model parameters, x is the current input vector, t is the corresponding target and s is the iteration index of the model and η is an adjustable learning rate used to avoid overshooting. Here x, t are understood to be a subsample from the entire set. Many momentum based methods have been evolved (adding information from previous steps) but we shall not dwell more on this than the basic principle here. The gradient descent used in this project is the one called ADAM as implemented in Pytorch, see Pytorch documentation[11]. For the original paper on ADAM see[12].

With the gradient decent and a loss function the model can now be optimized. However, it is not always given that NN model will find the best possible model parameters. The more parameters a model has, the more prone it will be to over fitting. That is a very deep (many layered) NN model with many neurons per layer will tend to find a solution that is ”too good to be true” if the high degree of freedom allows it to account for all spuriousities of the training data. Also, even if the model doesn't over fit, it may take very long to converge, and may never, if the signal gets overshadowed by useless information. A way to restrict the flow of information in the model for problems with a local nature is through convolutions.

3.2 Convolutional neural networks (CNNs)

The idea of using a convolutional neural network is that some problems are of a spatial nature. That is the interpretation of one point is way more dependent on its neighbours than on far away points. In the Hough transformed image of a muon as shown in the left hand image of figure 8 one sees that it creates a high intensity pattern in the region $q/p_T \sim -0.7$ to -0.5 and $\phi_0 \sim 0.36$ to 0.39. It is the focal point of the lines in this region that makes it possible to identify the charged particle track. However it is clear, that apart from this small region, the image is quite low on information about this muon. Adding now that the right hand image of figure 8 in which minimum bias is in included representing the real world scenario is full of noise, using all the neurons at the same time will be of way more confusion than use.

If instead of fully connected layers one guides the network by using convolutions in the 2D image only local patterns will be identified. The formula for a discrete 2D-convolution is given by:

$$y_{m,n} = (g * f)_{m,n} = \sum_{i,j} g_{i,j} \cdot f_{m-i,n-j} \tag{14}$$

Where g is a matrix called a convolutional kernel. But what would give a better intuition of its use however would be a simple example showing that it can be used to identify patterns in an image. Let us take as an example the noisy image of figure 10 in which a square is hidden.

A square is characterized by having four straight lines in two parallel pairs and we can define two convolutional kernels that will find these lines. These are shown in figure 11. Here g_1 is a horizontal line identifier and g_2 is a vertical line identifier. Using each of these two kernels on the noisy image of figure 10

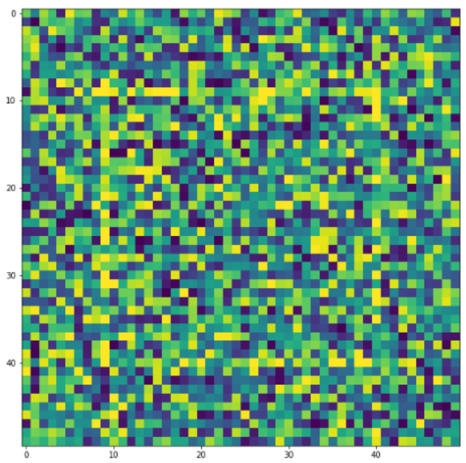

Figure 10: A square hidden in random noise.

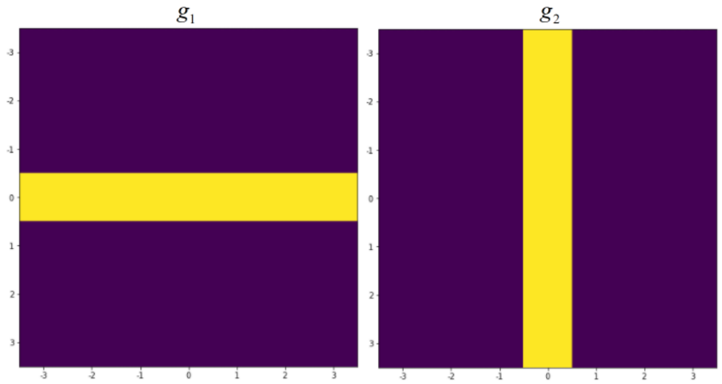

Figure 11: Two convolutional kernels represented as 2D images. Yellow corresponds to a value of 1 while dark blue corresponds to a value of 0. g_1 is a horizonatal line identifier, while g_2 is a vertical line identifier.

.

we acquire the images shown in figure 12. We can see that g_1 and g_2 identifies the horizontal and vertical lines respectively suggesting that there is a square in the image. This example shows how simple convolutions can be used to identify spatial patterns. More sophisticated networks with more layers of convolutions can learn more sophisticated patterns building on the patterns identified in the previous layer. In the end a dog can be distinguished from a cat!

The shape of the convolutional kernels is a so called hyper parameter to tweak before training the network. The choice of kernel for this problem is discussed in 5.5. The main parameters to tweak for a CNN model are:

- The convolutional kernel shapes
- The amount of layers in the model and how they are connected
- Which activation functions to use

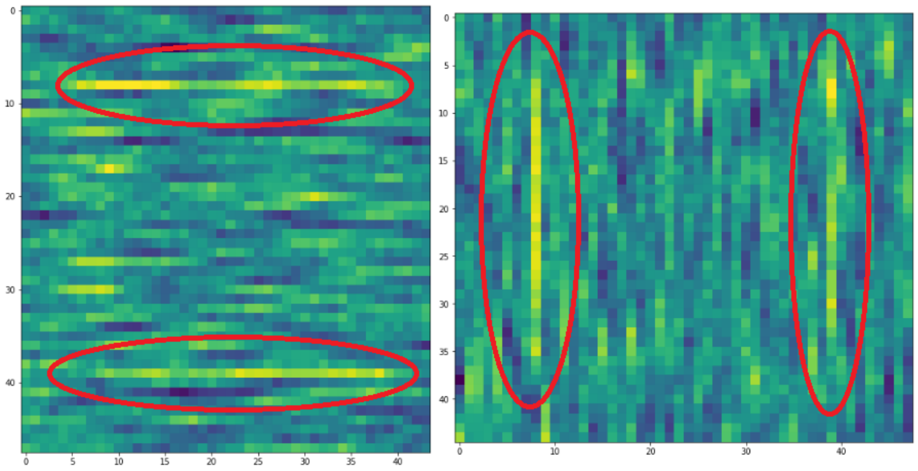

Figure 12: The left hand image is the result of convoluting figure 10 with the horizontal line identifier g_1 of figure 11. The right hand image is the result of convoluting figure 10 with the vertical line identifier g_2 of figure 11. The areas marked with red show the lines fromt the original square found by convoluting with g_1 and g_2.

- Which loss function to use
- Whether to use stride and/or max pooling
- How many channels to use

Where channels means using multiple kernels on the same image. This way more features of the image can be extracted as in figure 12 where the convolution with g_1 and g_2 give two different images containing different information also called channels. Stride means to reduce the image size by "jumping" over some integer number of bins while convoluting. Max pooling is in general used along with stride and is simply replacing each $n \times m$ area by its maximal value. This way the most important information can be saved for most problems while the complexity is reduced. Activation functions are in general meant to apply non-linear properties or add certain characteristics to the output of a layer, like the softmax function shown in 12 which is non-linear and maps outputs to a value between 0 and 1 summing to 1. The most used function between layers in neural networks is the rectified linear unit (ReLU) function:

$$f(x) = \begin{cases} x, x \geq 0 \\ 0, x < 0 \end{cases} \tag{15}$$

Which allows for sophisticated patterns while still being cheap in computation time.

The problem at hand is not the most classical classification problem. The problem is not just to identify whether there is any charged particle track within the Hough transformed image, but rather where it is. So to maintain it as a classification problem the output of the model will have to be two values pr. image bin. These will represent the probability that this bin is a muon-target and the probability that it is not respectively. I.e. we map $\mathbf{x} \in R^{n_{q/p_t} \times n_{\phi_0}} \rightarrow \mathbf{y} \in R^{n_{q/p_t} \times n_{\phi_0} \times 2}$. The target will then for each bin be a vector $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$ if the bin is a muon-target and a $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$ if it is not. For this reason using stride and max pooling is problematic for the problem at hand as the information of the exact target position will be lost when the granularity is reduced.

It should be noted, that the most muon-like objects in the minimum bias data actually are charged particle tracks and they are therefore valid to identify. It was however not possible to get the exact target position from the current data extract for these objects and as a result anything but added muons is characterized as background in both training and testing.

4 HTT Simulation

To test and optimize the performance of the HTT system a simulation of it has been made called HTTSim. The goal of HTTSim is to take input hits left by particles in the ITk layers and use these for track finding and fitting to identify charge particle tracks for later use in the Event Filter. Below the current HTT simulation is described.

The current HTT simulation takes as input a number of input hits. These are then combined into clusters of hits and these clusters are then sent to the road finding tool. In this thesis we shall focus on the Hough transform as the charged particle road finding tool as described in section2.4. In the study presented in this thesis clusters from all the three outermost strip layer pairs, from one of the innermost strip layer pair and the outermost pixel layer are used. See figure 13.

The Hough transform algorithm maps clusters from within a given slice in a given η-region onto a 2D-histogram in the (ϕ_0,q/pT) space. This can be done for multiple slices in z, η individually as described in section 2.1. As muons with $p_T \geq 1GeV$will in general hit the same layer only once we are for the purpose of charged particle track finding not interested in how many clusters correspond to a specific bin in the layer but only if it has any at all. Therefore the aggregated images over all 8 layers have bin counts equal to the number of layers hit corresponding to that bin, not the total amount of clusters. The higher the bin count, the more layers have a cluster corresponding to that $(\phi_0, q/p_T)$-value, and the more likely it is that it comes from a charged particle traveling through the ITk. To illustrate this procedure equation 16 is shown as a made up example of only two 3×3 layers (meaning a Hough transformed image with 3 bins along each axis for only 2 ITk layers) where the bin count in each layer should be interpreted as the number of clusters corresponding to that bin in that layer. I use the function I call "bool" where

$$bool(i) = \begin{cases} 1, i > 0 \\ 0, i = 0 \end{cases}, i \in N$$

The result is then the aggregated image of the two layers:

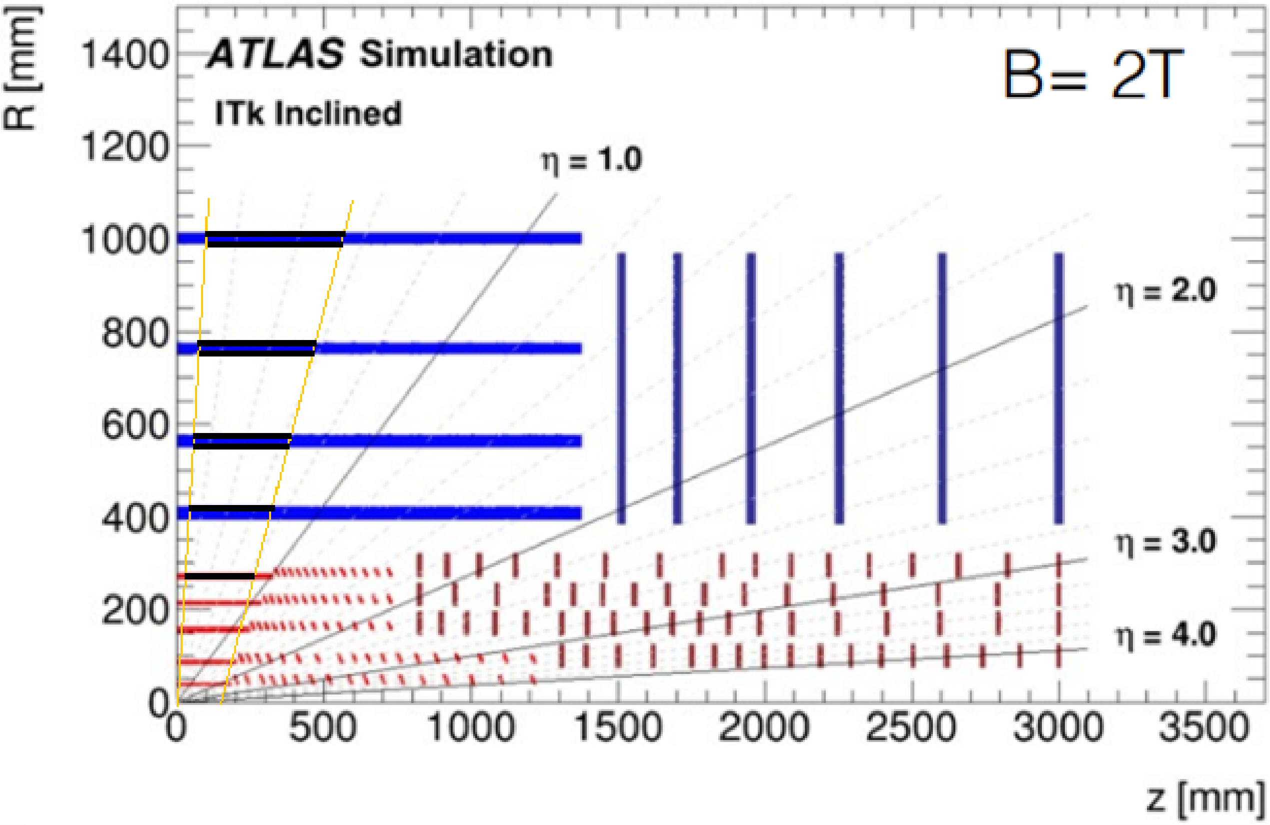

Figure 13: Drawing of the inner tracker with the 8 layers used in the Hough transform marked with black lines. One strip layer from the innermost strip layer pair is not used why there is only one black line at radius 400 mm, but two at the others.

$$
\begin{aligned}
bool(layer_1) + bool(layer_2) &= bool\begin{pmatrix} 0 & 0 & 1 \\ 0 & 2 & 1 \\ 2 & 0 & 0 \end{pmatrix} + bool\begin{pmatrix} 0 & 1 & 3 \\ 0 & 2 & 0 \\ 2 & 1 & 0 \end{pmatrix} \\
&= \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 1 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 2 \\ 0 & 2 & 1 \\ 2 & 1 & 0 \end{pmatrix}
\end{aligned} \tag{16}
$$

Every bin of the aggregated image with a hit layer count above a given threshold is then characterized as a possible road a charged particle might have followed. This is done for every slice in the event. From now on each aggregated image created for each of the 6 slices in each event will be simply called an ”image”.

After this process the found roads are either used directly for track fitting or they undergo so called “duplicate removal”. The purpose is to eliminate found roads likely to originate from the same object. This is done within an $n \times n$-bin area of the image. The idea is, that close to the focal point of multiple lines there will in general be multiple high count bins in the image as illustrated in figure 7. Therefore possible charged particle roads are in the duplicate removal of the current HTTSim required to fulfill at least 1 of the following conditions withing an $n \times n$-area around the bin:

- It has more hit layers than any other nearby bin.
- It is tied for the most hit layers but has more total clusters than any other nearby bin.
- It is tied for the most hit layers and the most total clusters but has more clusters than the bottom left bin.

The first criterion makes sense as the higher the bin count is the more likely the bin is to contain a muon. The second criterion is likely because the probability that a particle of interest hit a bin is in general larger if many particles hit the bin than if only few did, there are simply more ”lottery tickets”. The last condition is an arbitrary choice made for consistency.

For the duplicate removal 3×3, 5×5 and 7×7-bin areas are used and results using own implementation of the current algorithm are shown in section 5.9.

4.1 Data

The data used in this project for the presented results is unaggregated images for each layer extracted from the Hough transform algorithm. The images are made from single muon events and minimum bias events created as described in section 2.4 and extracted before aggregating the layers. This is because it increases the amount of ways in which data can be used, but in general the images will be aggregated as described in 4 before use. To fully utilize the extracted data I perform data augmentation. Data augmentation is to use data in different ways to fully utilize the information in the data in the training. In classification of real life images this is in general done by turning the image in different ways and mirroring it as a dog upside down or a mirrored dog is still a dog. For the same purpose single muon images are mixed with different minimum bias images while both training and testing. This is done for the Hough Transformed images of the layers before creating the aggregated images. This way the same muon will have to be found in different backgrounds altering the difficulty of the exercise significantly depending on the density of the background in the region of the muon focal point making the algorithm much less prone to overfitting. It should be noted here, that as the suggested method involves convolusions drawing information from the local area around each bin this method will be much more affected by this mixing than a simple bin count threshold.

In the current Hough Transform algorithm the following parameters can be adjusted:

- Whether to use slicing in z and η as shown in figure 3.
- Number of q/p_T- and ϕ_0-bins of the Hough transform image.
- The q/p_T and ϕ_0-range.
- Number of extra bins (called padding) in q/p_T and ϕ_0 in both ends. This extends the range of the Hough transformed image beyond the chosen q/p_T and ϕ_0-range by adding more bins at the edges.
- Whether to use an extra layer (e.g. use the currently unused strip layer, see figure 13).
- Hit extension (Whether to extend clusters to adjacent bins in ϕ_0 for each layer). If a layer has a hit extend of 1 then not just the bin corresponding to a cluster gets a count, but also the bins adjacent in ϕ_0 at either side. This is represented as a vector representing the hit extend in that layer going from inner to outer.
- Whether to use a convolution before applying threshold and if so, which.

I have opted to use the most used setup. I have done this to maximize comparability with the current results. The adjuster parameters are listed in table 1.

Number of slices	n_{sl}	6
Number of q/p_T and ϕ_0-bins	$n_{q/p_T}, n_{\phi_0}$	216, 216
Padding bins	$n_{q/p_T-pad}, n_{\varphi_0-pad}$	2, 6
q/p_T and ϕ_0-range	$q/p_T, \varphi_0$	-1 to 1 GeV, 0.3 to 0.5 rad
Whether to use extra layer	extra layer?	No
Hit extension (for each layer)	n_{extend}	2, 1, 0, 0, 0, 0, 0, 0
Convolution	k_{conv}	Unused

Table 1: Adjustable parameters in the currently implemented Hough transform algorithm.

Table 2 shows the amount of images used for training and testing.

The muons and minimum bias events are then mixed in different ways while training and testing in mini-batches of 8 (a few are smaller) creating the combination numbers listed when mixing. The amount of data is relatively low

	Training	Test
Minimum bias	1.842	464
Muons	1.703	1.500
Combinations	392.114	73.080

Table 2: Data used for training and testing. The combinations number is created by combining muons with different minimum bias files. Not all combinations are used in the results shown.

because data extraction takes a long time, but the data augmentation allows for much more efficient use of the saved data.

For each event there are 6 slices. Each are by the current and the suggested method treated as were they individual events, except when investigating whether the added muon was truly found in efficiency calculations.

5 Results

This section is first devoted to describing how the suggested CNN-based method for road finding should improve the results compared to the current method and how the methods will be evaluated. Next the algorithm used for target finding for the model training will be described before single and two layer CNN methods are presented with results. Finally the method dependence on different parameters will be analyzed along with the channels of the neural network.

5.1 Current road finding optimization method

Currently the method for optimizing the road finding algorithm using Hough transform wrt. efficiency and roads found (see 5.3) consists of manually optimizing each parameter and putting a threshold on the bin count. This includes every parameter mentioned in section 4.1

It is clear that these variables are so many that it will be hard to test a sufficient part of the solution space. Most likely the reason that the use of convolutions has, to some extent, been abandoned though suggested in [1] is, that it is too difficult to find a good solution solely by trial and error when both the kernel size and values in the convolution used are to be determined along with every other parameter at the same time. Yet as argued in 3.2 convolutions will likely be of use to improve muon detection compared to simple thresholds on bin counts.

If a CNN was used however this would eliminate a factor of trial and error by introducing gradient descent to determine the parameters of the convolution. Additionally it would add new options such as the use of multiple layers of convolution and non-linear activation functions enabling non-linear dependencies.

5.2 Suggested road finding method

Compared to just putting a threshold on the bin count a CNN should always be able to do as good (if no interesting information is found in nearby bins) or better. More importantly though, convolutional neural networks are very good at identifying sophisticated spatial patterns. A CNN should both be able to identify patterns that are “muon-like” and patterns that are “non-muon-like” and combine this information to most efficiently sort the muon-like patterns from the background.

In general by using scores acquired through a CNN you also expand the spectrum of possible scores to be continuous[1]. This allows you to fine tune for the desired efficiency and set a threshold accordingly instead of being restricted to an integer number of hit layers between 0 and 8. This can also be the explanation as to why it has not yet been of any help to increase the number of layers used. With only 8 (or 9 when adding an extra layer) hit layers as different possible thresholds you have to be lucky to find that a threshold exists that gives a good balance between efficiency and road count. Adding an extra layer might push this balance.

Adding the extra layer has not been tried in this thesis, but it would be interesting to see if the model would benefit form this information.

5.3 Method evaluation

When comparing methods it is necessary to be clear as to what characterizes a good one. The purpose of the method is to find as many of the added muons as possible while finding the least amount of roads possible. A road is a bin in the Hough transformed image that by a given method is characterized as a muon road candidate for later track fitting. I define efficiency as how large a portion of the true muons are found, that is:

$$Eff_{muon} = \frac{n_{found}}{n_{total}} \tag{17}$$

What muons are characterized as roads is decided by the method used on a mixed single muon and minimum bias file. For the current method this means a threshold on the bin count and on the suggested method this means a threshold on an output score of the CNN. For a muon to be characterized as found I require that any bin with at least six hit layers in the single muon file within a slice is selected as a road. This is because the HTT track fitter requires at least six hit layers to fit a track.[1] For the same reason the given efficiencies are only for muons that make at least 6 hits within a single slice as the rest are deemed unfindable by this definition. Roads are found independently for every slice of the event. However, the muon needs only be found inside a single slice to be characterized as found even if it leaves a sufficient amount of hits in more slices.

The efficiency alone is not sufficient to evaluate the quality of the model, reducing computation time is the main goal of using the Hough Transform before track fitting. The computation time scales with the number of combinations of clusters that are to be fitted to. This is calculated by multiplying the numbers of clusters in each layer of each found road and adding for all the roads. As the total combination number per event hereby naturally scales with the number of found roads, the road count is used as measure of computation time.

[1] Limited only by model complexity (number of layers, neurons and activation functions) and computer precision.

5.4 Model training

Convolutional neural networks can in general be arbitrarily complex models and optimizations should thus start from a simple model and then gradually evolve into a more complex model with complexity limited by the complexity of the data, the signal strength, the amount of data and computational power.

In terms of computational power the suggested models are limited to be able to train on an 8 GB GPU (GeforceRTX 2060 Super) as this is what was accessible.

5.5 Target finding

For a supervised network to be able to train, a target t_i is needed to compare the output of the network y_i to. As a reminder I intend to minimize the negative cross entropy to optimize model:

$$Er\,(y,t) = -\sum_i t_i \log y_i$$

In which t_i contains the information of whether the bin is a target bin or not and y_i is how likely the network predicts that this bin is to be a target bin.

However as mentioned in section 3 target finding proved non trivial. The obvious target would be to have $t_{i_1} = \begin{bmatrix}1\\0\end{bmatrix}$ in the bin corresponding to the true $(\phi_0, q/p_T)$-parameters of the muon and $t_i = \begin{bmatrix}0\\1\end{bmatrix}$ for $i \neq i_1$. However there is in general a displacement between the true parameters and the focal point of the lines in the image. The reason for this might be deviations in the magnetic field for which corrections are not yet added to the Hough Transform algorithm. But for now, as the goal is to find the actual detector clusters for track fitting and not an ideal set of parameters, the truth parameters will not be a ”good” target.

Instead we should start by defining what is meant by a ”good” target for this problem:

1. A good target is a target, that if found, will be good for track fitting.
2. A good target is recognizable by the chosen model and distinguishable from non-muon-like patterns.

Point 1 is obvious. There is no reason to train the network to find something that we cannot use for track fitting. Point 2 is about being able to train the network and depends on the chosen model type. We have chosen a CNN, which means that a local area around the target-bin should be recognizable and distinguishable from other patterns in the image. How this is done will be discussed later in this section. If the target is not highly recognizable or distinguishable, then the network will have a hard time finding the target, thus one will get a high road count for a given efficiency, when training it to find this target.

The two criteria are ambiguous and highly problem dependent and finding an algorithm for optimal target finding is thus also. Below I describe how I choose a target that fulfills both of these requirements.

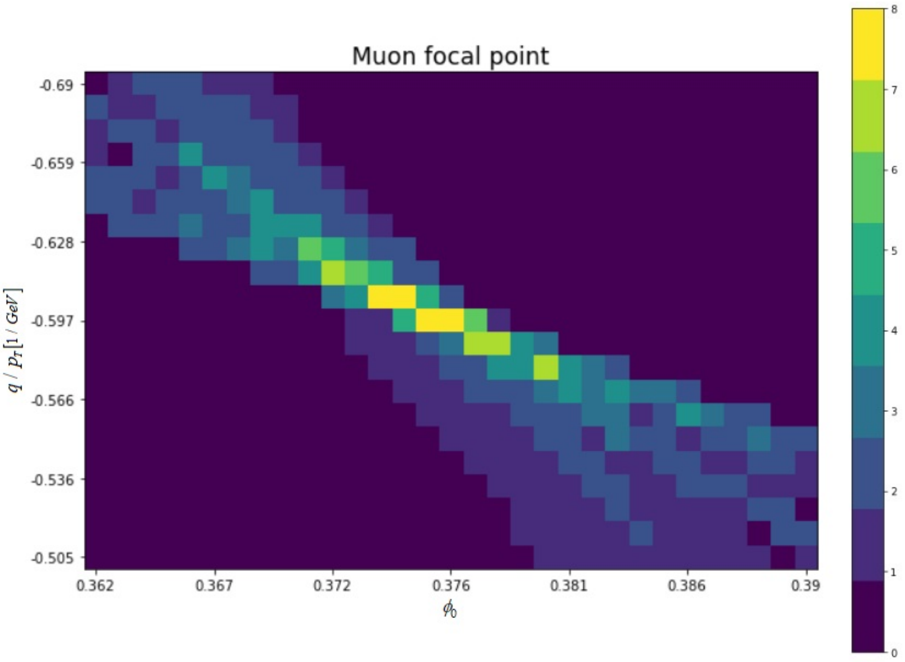

Figure 14: A zoom in on the focal point of the lines at in the left hand image of figure 8. It is seen that the focal point apart from the high intensity is characterized by being very thin transverse to the lines.

.

Point 1 means that the muon should have multiple hit layers corresponding to this bin so that a good track fitting can take place. This requirement is already somewhat satisfied by requiring at least 6 hit layers by the muon. Further, I require that the number of layers hit by the muon for the chosen bin is the maximal for the entire slice. The currently implemented algorithm requires at least 7 hit layers in a bin to identify it as a road. A requirement of only 6 hit layers should therefore allow for finding more muons that are currently undetected by the current method.

In general, there will be more than 1 bin with the maximal number of layers hit by the muon for the given slice. One could choose to use all such points as targets, but that would interfere with the second point; the target should be ”recognizable”. The local information for each of these points will in general be of a different nature and the algorithm would therefore be looking for multiple distinct patterns simultaneously making the target difficult to recognize. This suggest that only 1 bin per slice should be used as target.

I therefore use maximal recognizability as criterion to identify which of the remaining bins should be chosen as target. In figure 14 I have plotted the focal point of the lines for the muon shown in the left hand side of figure 8.

The chosen method (a CNN) predicts how likely a bin is to be a muon-target based on local information of the Hough transformed image. By using nearby bins to decide which bin should be the target we can thus make sure that the target is recognizable by the model. One should now notice that the center area

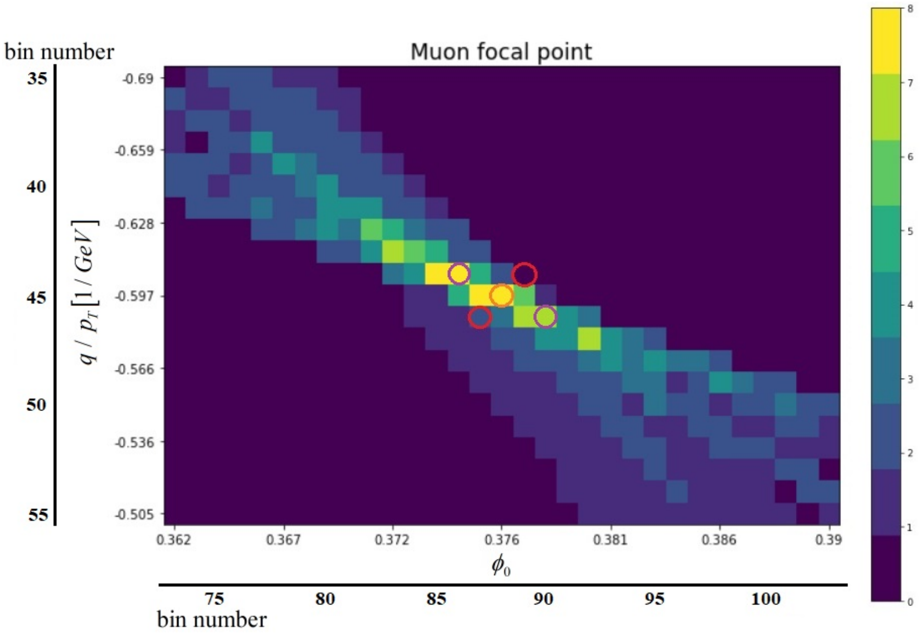

Figure 15: The same muon focal point as in 14, but with an orange circle marking the bin in which steepness is calculated and two red and two purple circles marking the two different directions in which steepness is measured. The total steepness in the direction across the lines is given by equation 18 in combination with each bin count to be $2 \cdot 8 - 2 - 0 = 14$ and similarly for the direction along the lines given by equation 19 is $2 \cdot 8 - 8 - 7 = 1$.

of figure 14 is characterized by being very thin orthogonal to the lines, it is very "focused". Should multiple lines meet by chance, it is unlikely that they should meet exactly as for the true muon.

To describe the shape in this high intensity area I define steepnesses for every point in some given direction. For the bin marked by an orange circle in figure 15 the steepness "across" the lines is calculated by subtracting the bins marked by red circles from 2 times the orange circle bin.

$$st^{across}_{88,45} = 2 \cdot image_{88,45} - image_{87,46} - image_{89,44} = 2 \cdot 8 - 2 - 0 = 14 \quad (18)$$

Similarly I define the steepness "along" the lines by subtracting the purple circle bins from two times the orange circle bin.

$$st^{along}_{88,45} = 2 \cdot image_{88,45} - image_{87,43} - image_{89,47} = 2 \cdot 8 - 8 - 7 = 1 \quad (19)$$

These steepnesses can be calculated for any bin however. Among bins in single muon images that have at least 6 hit layers, that is maximal within the slice, these parameters can be used to identify which bin to choose as target. Choosing the bin with the lowest steepness along the lines and resolving any further ties by choosing the bin with the highest steepness across the lines a target can be chosen.

Plotting the distribution of these steepness parameters for the found target but now in muon + minimum bias against other points of sufficient intensity in

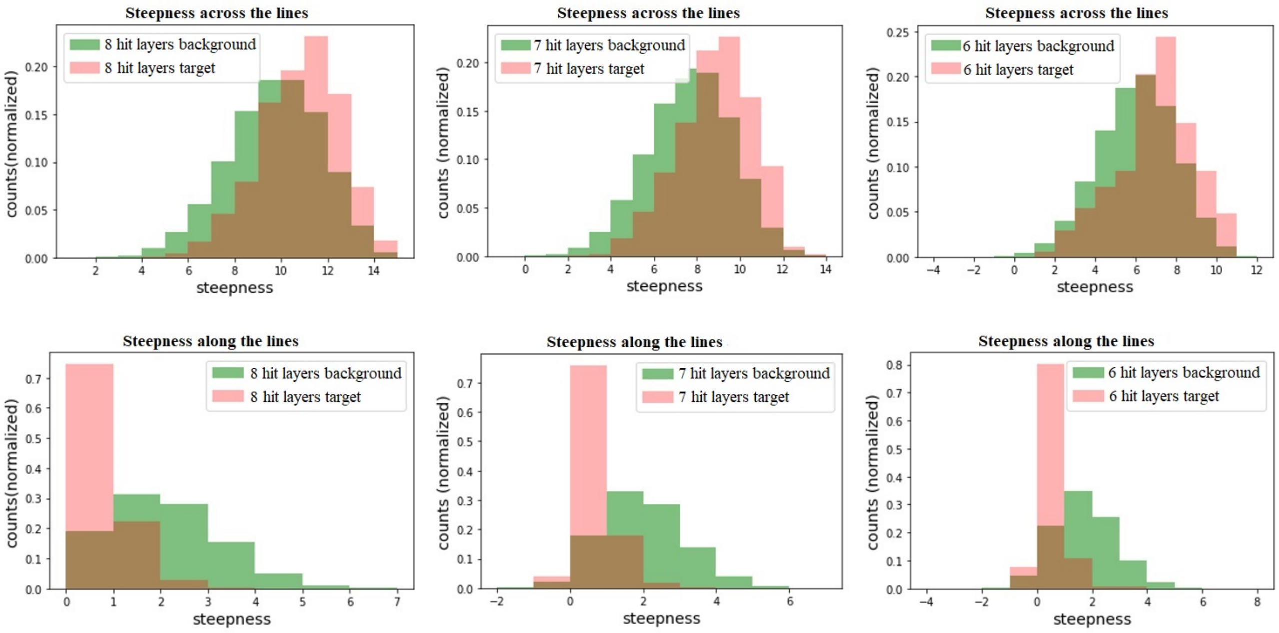

Figure 16: The plots show distributions for the steepness parameter for the direction across the lines and along the lines. The plots are subdivided into the amount of hit layers are in each bin in minimum bias (or minimum bias + muon for the target) as steepness naturally scales with the value in the measured bin. The green graphs show all bins with the corresponding value in minimum bias. The red graph is specifically for the points specified as targets. The brown area is where the targets and the background overlaps. It is seen that there is a slight distinction between the target chosen to represent the signal and the other bins along the ”across”-steepness direction. Along the lines however there is a very high distinction between the target chosen to represent the signal and the background.

minimum bias (background) one can see that these target points are indeed to some degree distinguished from the rest, see figure 16.

It is clear from these distributions that there is some possibility to distinguish between the targets and minimum bias other than just the number of hit layers. Especially the steepness along the lines differs highly from the rest of the bins as seen in the distributions at the bottom row of 16, but also the steepness across shown at the top row provides slight distinction.

As the central target-bin will in general have a high value (at least 6 of 8) having low steepness in the steepness along the lines requires high values in the bins subtracted also. Indeed looking at the bottom left distribution of figure 16 you see that more than 70 % of the target 8 hit layers bins have steepness 0 along the lines, meaning that the bins along the lines at either side also have 8 hit layers. This pattern can therefore be rewarded by a convolutional kernel with high values in these points, for instance:

$$conv_{along} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{20}$$

Where the relative positions of the "1"'s in the convolution are the same as the relative positions of the orange and purple circle bins of figure 15. Similarly the high steepness in the direction across the lines can be rewarded by this convolutional kernel:

$$conv_{across} = \begin{bmatrix} 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 \end{bmatrix}$$

as the negative values in the convolution will give high negative contributions to bins that also have high bin counts in the direction across the lines.

As both of these steepness parameters, can be rewarded by convolutions, I have thus found targets that are useful for track fitting and that are recognizable by the chosen method, a CNN. We are therefore ready to efficiently train the network to find a target. In addition it is clear that at least a 3×5 kernel is needed so as to utilize the low steepness along the lines as in equation 20 that was found to have high distinctive power in figure 16. Indeed models using 3×5 kernels proved to perform better than square kernel ones (like 3×3 and 5×5) in tried models not presented here.

5.6 Model setup

In general, the count in a bin scales with nearby bins. This makes sense as for any bin where many lines cross we will expect to find nearby bins in which all or some of the same lines also meet. If we focus on the orange mark bin in figure 15 with 8 hit layers we will notice that directly to the left we have another bin of 8 hit layers. And down and to the right we also find a bin with 7 hit layers. But to treat all 6,7 and 8 hit layer bins evenly this is a problem as the 8 hit layers bins will in all simple (few layered) CNNs tend to map to more extreme values as convolutions are simply multiplications within a local area. A way to deal with this problem is to divide the bin count after convolution with the bin count before convolution. This is shown below using equation 14 with an added bias term k.

$$\begin{aligned} {f^{l+1}}_{m,n} &= \left(g * f^l\right)_{m,n} + k = k + \sum_{i,j} g_{i,j} \cdot {f^l}_{m-i,n-j} \\ \tilde{f}^{l+1}_{m,n} &= \left(g * f^l\right)_{m,n} / {f^l}_{m,n} = k / {f^l}_{m,n} + \sum_{i,j} g_{i,j} \cdot {f^l}_{m-i,n-j} / {f^l}_{m,n} \end{aligned}$$

Assuming now that we are in a target bin indexed (m_1, n_1) and the surrounding bins are somewhat proportional to the central bin we can extract the central bin count giving ${f^l}_{m_1-i,n_1-j} \approx {f^l}_{m_1,n_1} \cdot s_{i,j}$, where I will call $s_{i,j}$ the "texture" and $s_{0,0} = 1$. This allows me to reduce the equation slightly:

$$\begin{aligned} \tilde{f}^{l+1}_{m_1,n_1} &\approx k / {f^l}_{m_1,n_1} + {f^l}_{m_1,n_1} / {f^l}_{m_1,n_1} \cdot \sum_{i,j} g_{i,j} \cdot s_{i,j} \\ &= k / {f^l}_{m_1,n_1} + \sum_{i,j} g_{i,j} \cdot s_{i,j} \end{aligned}$$

The bias term which now gives a $k/{f^l}_{m_1,n_1}$-term could be removed, but it appears that the model benefits from this non-linearity. It is worth noticing that k is in itself an independent model parameter allowing the model to scale this non-linear term to the need. The other term is now assumed independent of the central bin value and could thus be interpreted as a measure of ”texture” comparable between 6, 7 and 8 hit layer target bins. To enable the model to still use the bin value directly $f^l_{m_1,n_1}$ this can be added through a separate parallel layer. Hereby we have avoided the forced bias of the network between different count bins yet allowed an intentional bias as a separate parameter. We thus arrive at the formula for the first convolutional layer used:

$$\tilde{f}^1_{m,n} = \sum_{i,j} g_{i,j} \cdot {f^0}_{m-i,n-j} / {f^0}_{m,n} + k/{f^0}_{m,n} + {f^0}_{m,n} \tag{21}$$

[2]

5.7 Single layer CNN

Figure 17 shows a single layer CNN intended to find the bins most likely to contain a muon-target. The red rectangle shows a two channel 3 × 5 kernel convolution mapping to the topmost image. Two channels are here used because two are needed to have an ”is” and ”is not” a target for each bin of the input image as described in section 3.2. ”Normalization” is noted referring to the division by the central bin as in $\tilde{f}$ by equation 21. To still allow a direct dependence on the central bin value this is added as a separate 1×1 convolution marked with blue. These are then simply added together making a 1-layer network with non-linear dependence on the image as described in equation 21. The outermost edge of the image is cut to reduce edge effects caused by lack of information outside the image range. As the last step the softmax-function, see equation 12, is used on the two channels mapping each bin pair to values between 0 and 1 summing up to 1.

If the mathematical model used to describe a problem, here a CNN, is very complex compared to the complexity of the input data and the amount of input data, over fitting might occur. Over fitting means that the complex mathematical model starts describing behaviour that is due to random variations, not inherent structures, in the training data. A way to test whether over fitting occurs is to see whether the loss measured on an alternative data set (drawn from the same problem) converges as low as the loss on the training data set. If it does not, it must mean that the model is accounting for random variations in the training data which is naturally useless for solving the general problem. The CNNs presented in this thesis are however very simple in comparison to the amount of data used, and the complexity of the data, so over fitting has not been a problem. Converging loss functions can be found in the appendix however, section 7.1 in which the training and validation loss follow nicely.

The method is, as explained, intended to find the muon, while finding the least amount of total roads. As an example of the CNN performance we can use the muon of figure 18 shown in and outside minimum bias for a single slice. On the minimum bias + muon image (the right hand image) we can use the current

[2] Divison by 0 never happens as this is only evaluated for bins of at least 6 hit layers, as all other are all ready known to have 0 probability of being a muon target by definition.

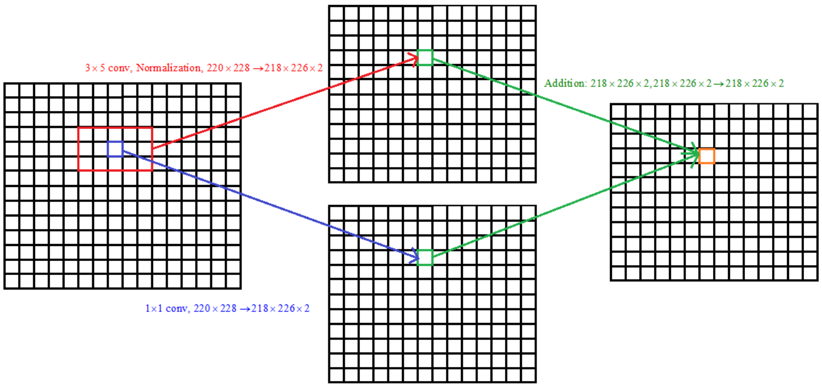
3 × 5 conv, Normalization, 220 × 228 → 218 × 226 × 2
Addition: 218 × 226 × 2, 218 × 226 × 2 → 218 × 226 × 2
1×1 conv, 220 × 228 → 218 × 226 × 2

Figure 17: The figure is an illustration of the 1-layer CNN. The red rectangle illustrates the use of two channels of 3×5 kernels with subsequent normalization according to equation 21. The image below shows the result of the two channels of 1×1 convolutions without this normalization. The images are then simply added together giving the full equation 21 for two channels. The resulting image is subject to the softmax-function of equation 12 and evaluated using the cross-entropy loss as described in section 3.2. This means that the output can be interpreted as the probability that the given bin corresponds to a target, given the model. The outermost edge of the image is not used so as to reduce edge effects. While it is only a single layer model, it still has a non-linear dependence on the input from the normalization.

method and the suggested single layer CNN-method (evaluated at a threshold corresponding to 99% efficiency). The resulting images are shown in figure 19. These images mark all bins yellow that have a score above the chosen threshold and are therefore counted as roads by the current and the suggested method respectively. As the reds dot shows the bins containing at least 6 layers hit by the muon these are what the methods are intended to find.

It is clear that while both methods find the muon, the suggested CNN model produces much less roads. The methods are evaluated as described in section 5.3 and averaged over all test images. For comparison my implementation of the current method with a threshold of 7 hit layers, as is most used, has an efficiency of 99.2 % and finds 349 roads.

The results for the 1 layer model is presented for multiple efficiencies in table 3. The first row is the result for the current method and the following gives the road count for the suggested method evaluated at different efficiencies. It is clear from the table that the results can be improved significantly by this 1 layer CNN (with a nonlinear dependence on the input).

Focusing on 99 % efficiency we get only 143 roads against the 349 for the current method or a reduction by almost 3/5 with a slight loss of efficiency. Alternatively the efficiency can be increased to 99.5 % finding now 257 corresponding to a reduction of road counts by about 1/4. This is just for a 1-layer CNN. By adding additional layers non-linearities and bins from further away can be added which increases the allowed complexity of the model and thereby possibly the performance.

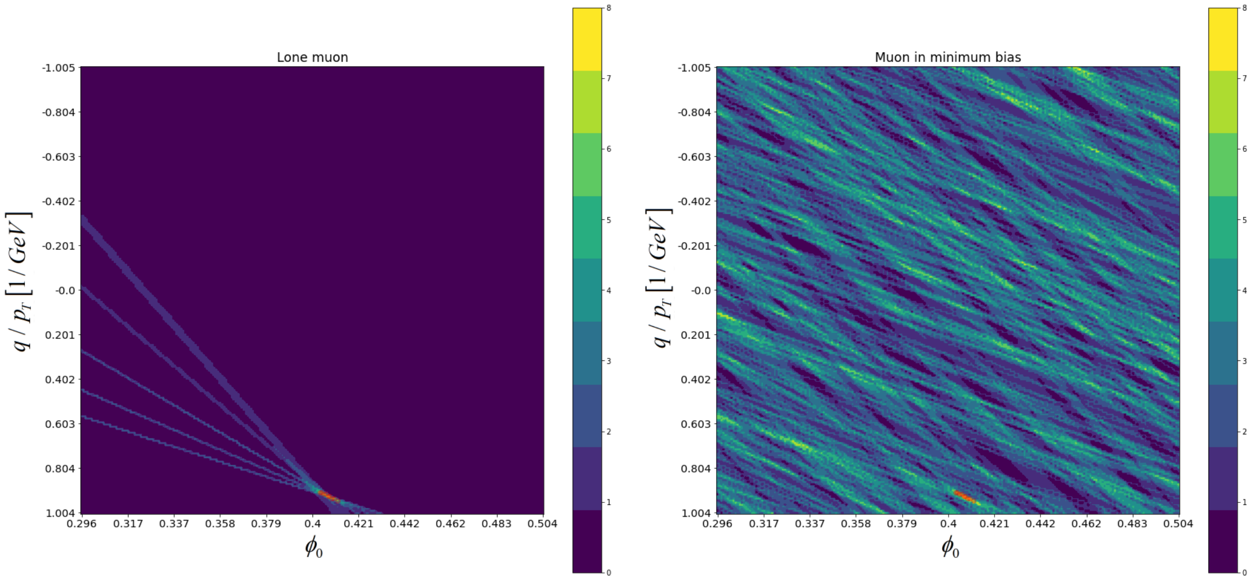

Figure 18: The left hand image is the Hough transformed image of a single muon. The right hand image is the same muon in minimum bias. The red dots are the bins that have at least 6 layers hit by the muon. That means that if a track finding method finds any of these the track finding method has found the muon.

Method	Efficiency	Number of roads
Current method, threshold 7	99.2 %	349
Single layer CNN	98.0 %	90
Single layer CNN	98.5 %	119
Single layer CNN	99.0 %	161
Single layer CNN	99.5 %	229

Table 3: Efficiency and road counts for the current model which is a simple threshold of 7 hit layers and the results for the single layer CNN.

5.8 Two layer CNN

In figure 20 two 3×5 convolutional layers are used separated by a ReLU-activation function, see equation 15. The first layer maps each slice onto 5 channels (of which 4 are from the red 3×5 convolution and the last is by the blue 1×1 convolution in figure 20. This is similar to the 1-layer model, yet the 1×1 convolution is here interpreted as an independent channel, not added to the result of the other convolution). This is followed by a ReLU-function and another 3×5 convolution on to 2 channels. Compared to the 1 layer network this is allowed to use information from a larger area, as two consecutive 3×5 convolutions collectively use a 5×9 area. Additionally it is allowed to have more complex dependencies on the input, by the increase in channels used before the added ReLU-activation function.

Converging loss functions can be found in the appendix, subsection 7.1, but

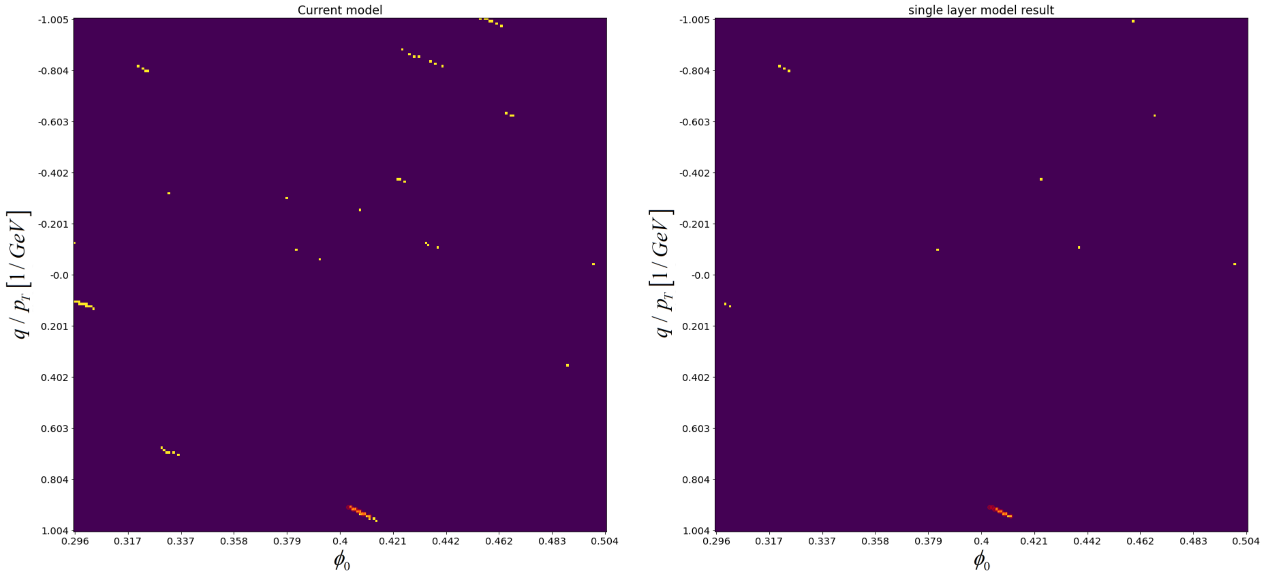

Figure 19: The leftmost image shows the possible charged particle roads marked as yellow dots found in figure 18 with the current model. The rightmost image are the found roads for the suggested single layer CNN-model at a threshold corresponding to 99% efficiency. The red dots mark the bins that have 6 or more layers hit by the muon. The single layer CNN method reduces the amount of roads found in this image significantly while both models are able to find the true muon as both have found roads in the red dot area.

this network does not shown signs of over fitting either.

Focusing once again on the image on the right hand side of figure 21, the output of the single layer network and the suggested two layer CNN model for a threshold corresponding to 99% efficiency is shown in figure 21.

Comparing the two images of figure 21 there does not seem to be much of a difference in the road counts. The results averaged over the test set is shown in table 4. Here one sees however, that the two layer CNN constitutes an improvement on all evaluated efficiencies compared to the single layer CNN.

Method	Efficiency	Number of roads
Current method, threshold 7	99.2 %	349
Two layer CNN	98.0 %	70
Two layer CNN	98.5 %	86
Two layer CNN	99.0 %	127
Two layer CNN	99.5 %	209

Table 4: Efficiency and road counts for the current method which is a simple threshold of 7 hit layers and the results for the two layer CNN.

At 99% efficiency this method is able to bring down the number of roads by almost 2/3 compared to the original which is quite some improvement. Increasing the efficiency to 99.5% instead the road count falls from 349 to 209 or

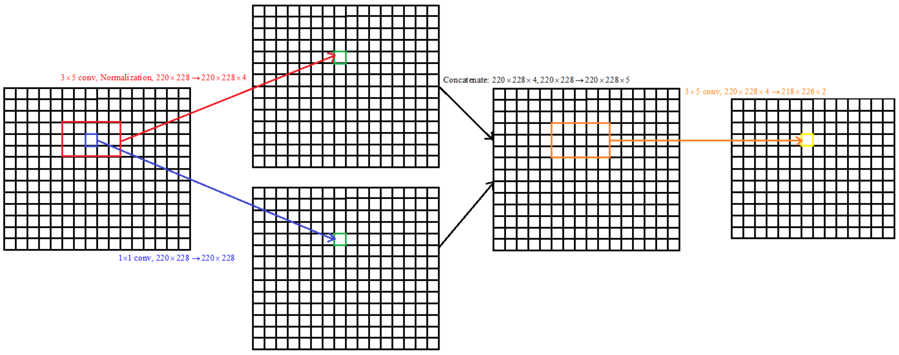
3×5 conv, Normalization, 220×228 → 220×228×4
1×1 conv, 220×228 → 220×228
Concatenate: 220×228×4, 220×228 → 220×228×5
3×5 conv, 220×228×4 → 218×226×2

Figure 20: The figure is an illustration of the 2-layer CNN. The red rectangle illustrates the use of 4 channels of 3 × 5 kernels with subsequent normalization according to equation 21. The image below shows the result of the single channel of 1 × 1 convolutions marked with blue without this normalization. The images are then concatenated into 5 channels (in contrast to addition in the single layer CNN). This image is then subject to the ReLU function followed by another 3×5 convolution onto 2 channels. The resulting image is subject to the softmax-function and evaluated using the cross-entropy loss as described in section 3.2 meaning that the output can be interpreted as the probability that the given bin corresponds to a muon, given the model. The outermost edge of the image is not used to reduce edge effects.

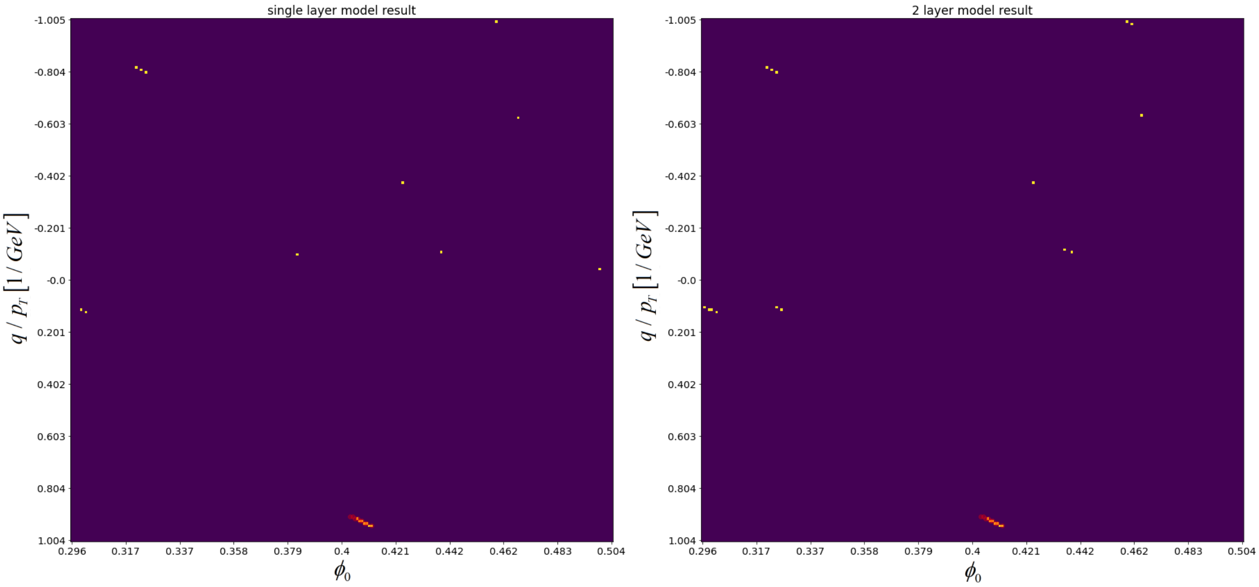

Figure 21: The leftmost image shows the possible charged particle roads found in figure 18 with the single layer network at a threshold corresponding to 99% efficiency. The rightmost image are the found roads for the suggested two layer CNN at the same efficiency. There does not seem to be much of a difference between the two images.

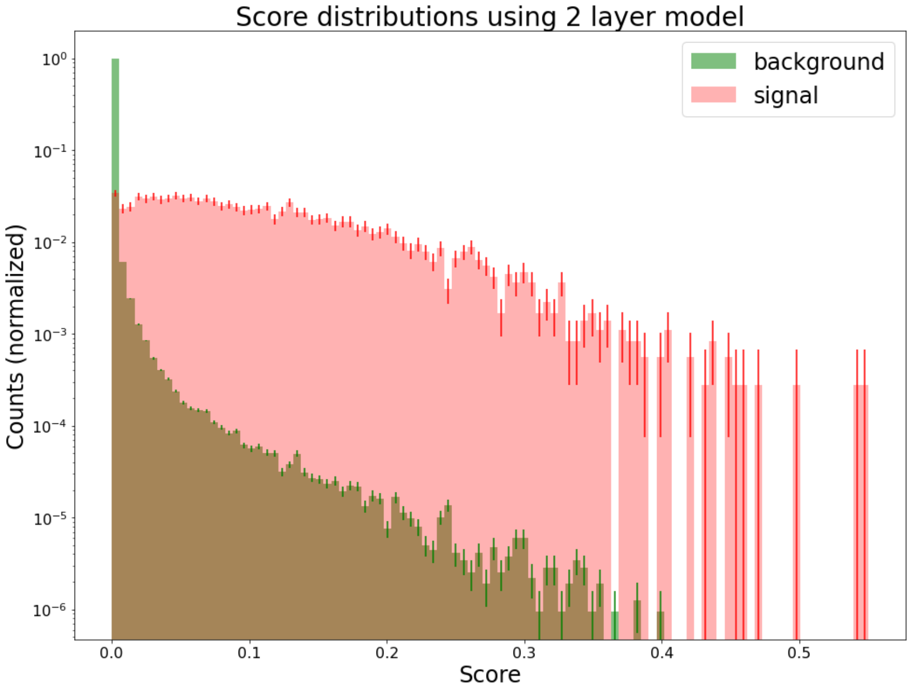

Figure 22: The figure shows score distributions for background and signal. The background is evaluated in minimum bias only on bins with at least 6 hit layers as all others are set to 0 probability all ready and are therefore sorted out automatically. Even then, far most of the background scores almost 0 probability of being a target. The signal here is the best scoring sufficient bin for each event. "Sufficient" here means that it has at least 6 true muon hits. This is chosen as they represent the minimal threshold needed to find a true muon road in this event. The errors shown is the standard deviation of a binomial with a uniform prior distribution as described in [13]. The brown is where the two distributions overlap

by about 2/5 compared to the current method. It is worth noting that a two layer CNN without ReLU between the layers, thus corresponding to one big 5×9 convolution has been tried, but yielded higher road counts at measured efficiencies. Apparently the model benefits from this added nonlinearity.

Figure 22 shows the distribution of background and signal scores assigned by the two layer CNN. The signal scores are the highest scoring bin in each event with a sufficient amount of layers hit by the muon. This is chosen because they represent the maximal threshold that can be put on the score and still find the muon and we gain nothing by finding the same muon more than once. It is clear that the two layer CNN is rather efficient in distinguishing signal from background assigning probabilities of being a target very close to 0 for far most of the 6 or more hit layers background. Most likely the highest scoring background bins are charged particles from the origin. The distribution for the signal only is shown in figure 23

In figure 24 the lowest scoring bins have been taken out to show more clearly

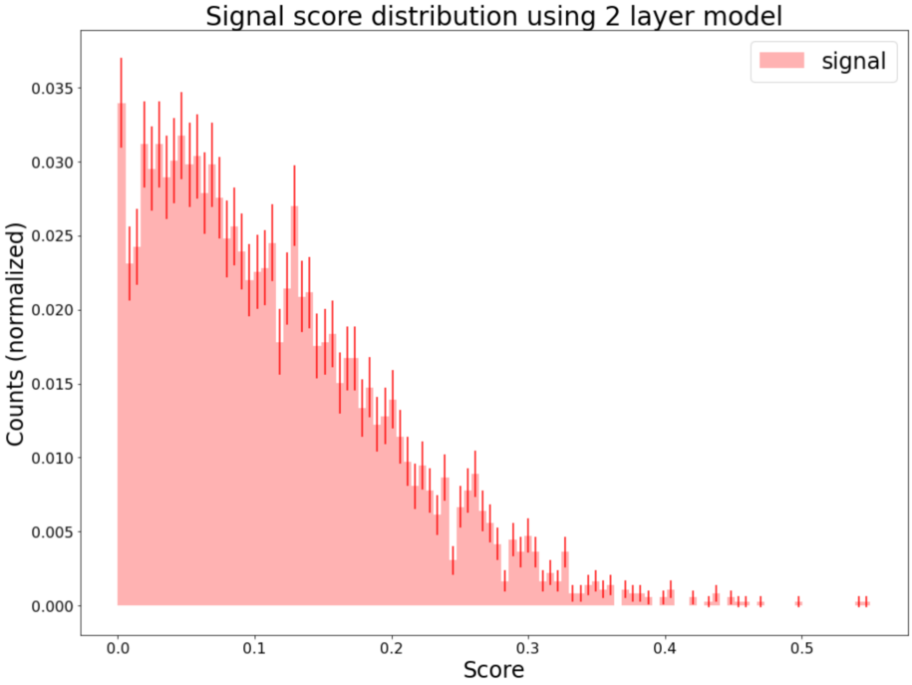

Figure 23: The figure shows the score distribution for the signal only. A peak in the distribution is seen around the score 0.05.

the effect of the 99% efficiency threshold. While naturally removing about 1% of the signal the cut removes 95% of the 6 or more hit layers background.

The result can be represented by a roc curve as shown in figure 25 describing the false positive rate acquired for a given efficiency. The roc curve shows a high distinction between signal and background. The high slope of the roc curve at high efficiency says that significant road reductions could be acquired by small reductions in efficiency if desired.

I have tried to add additional layers of convolutions, but they always acquire higher road counts for the same efficiency than the two layer model. The reason might be that adding information further from the muon center allows for more noise relative to the signal, see also section 5.11.

5.9 Duplicate removal

To reduce the number of found roads after the currently implemented algorithm "duplicate removal" can be performed as described in section 4. The purpose is to eliminate excess roads that originate from the same object. Reducing road counts within a 3×3 and a 5×5 area on the current model leads to the efficiencies and road counts showed in table 5.

It is seen that the efficiency is only slightly affected by the 3x3 reduction. On the other hand the number of found roads is reduced by almost 2/5. It is here worth noting though that this is still higher than even the result for the single layer model at 99% efficiency. Using the 5x5 reduction an even bigger road

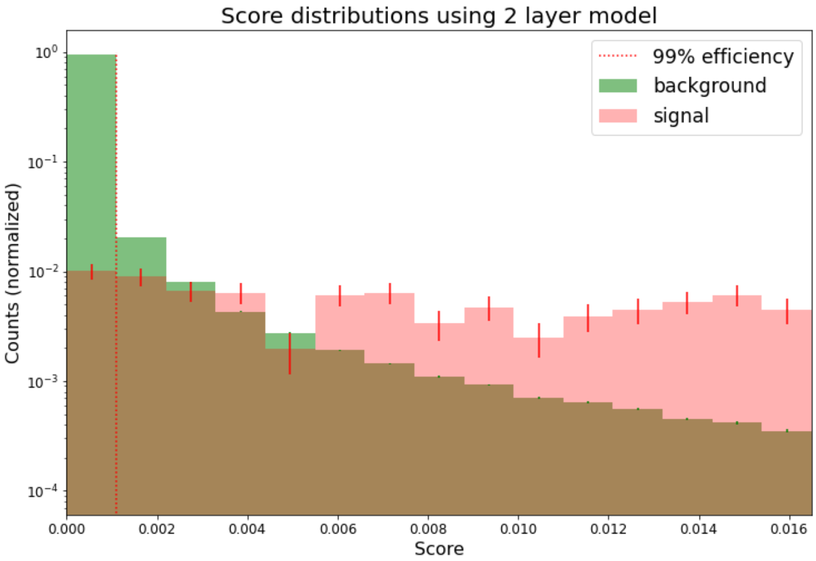

Figure 24: The figure shows a zoom in on the lowest scores of figure 22. The red area is the signal, the green is the background. The brown area is overlap between signal and background. The figure shows that more than 90% of the 6 or more hit layers background is removed when choosing a threshold corresponding to an efficiency of 99%.

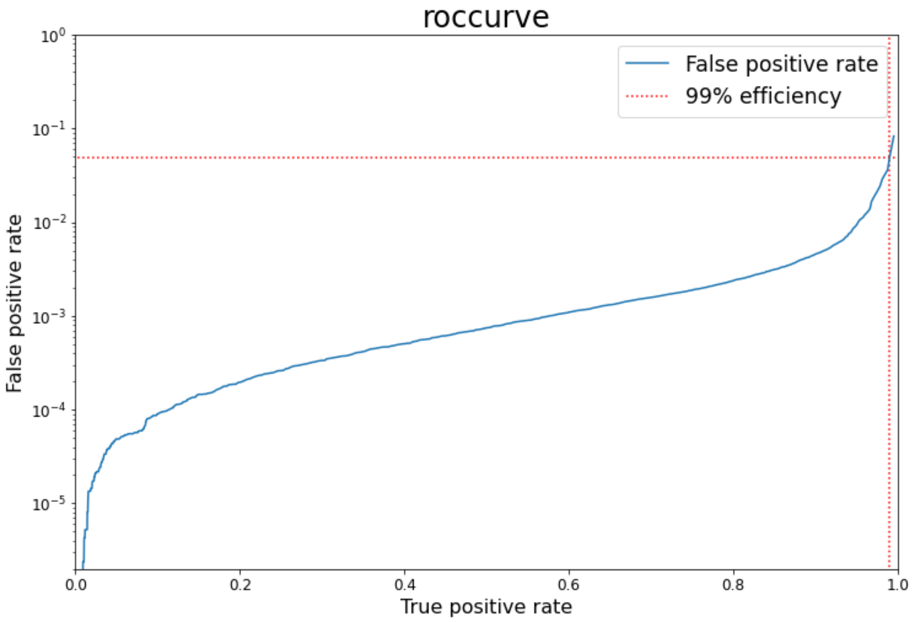

Figure 25: The figure shows the false positive rate acquired for a given true positive rate (efficiency). As an example is drawn the 99% efficiency yielding a false positive rate of about 5% (not counting bins with less than 6 hit layers). It is of course important to notice while only a small fraction of the background is accepted, there is still many more background than signal bins.

Method	Efficiency	Number of roads
Current method, threshold 7, 3×3 removal	99.0 %	216
Current method, threshold 7, 5×5 removal	98.1 %	154

Table 5: The resulting road counts and efficiency by using duplicate removal within an area of 3×3 and 5×5.

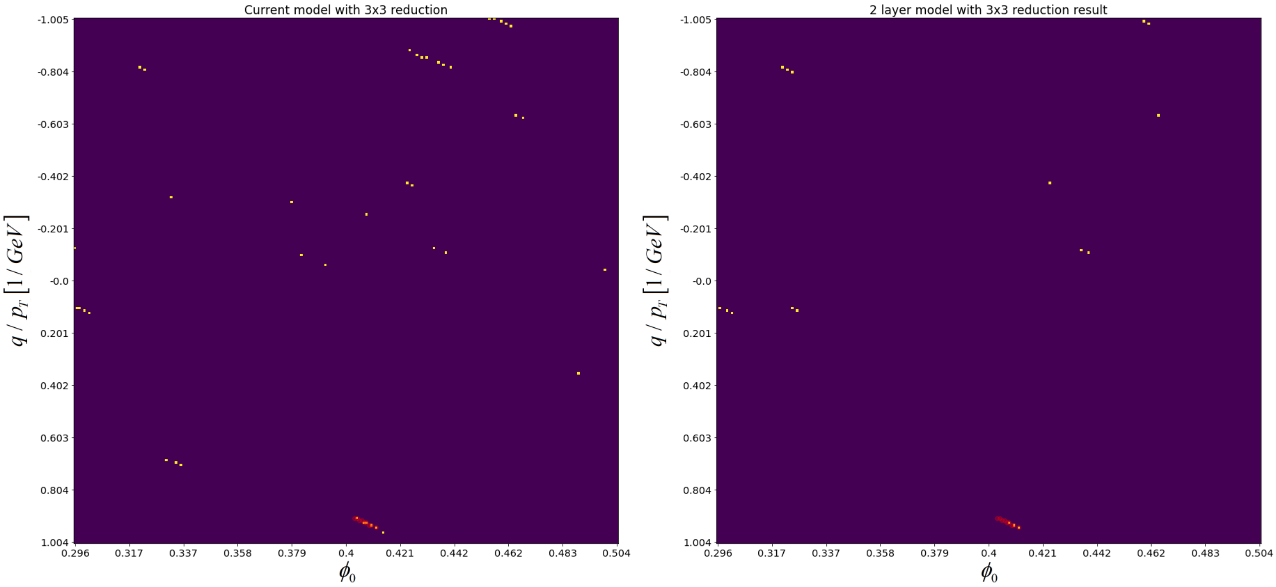

Figure 26: The leftmost image shows the possible charged particle roads found in figure 18 with the current model with reduction within a 3×3 area. The right hand image shows the found roads for the suggested two layer CNN with reduction within a 3×3 area at 99% efficiency. While the amount of roads found in the current method image is significantly lower than what is found without the 3×3 reduction, the suggested two layer method still reduces the amount of found roads significantly more.

reduction of almost 4/7 is acquired but the efficiency also drops significantly to only 98.0%. Due to this high drop in efficiency for the 5×5 reduction we shall focus on the result for the 3×3 reduction.

One can in a similar way reduce the number of found roads for the suggested algorithm. When looking at an $n \times n$ area in the image the number of roads can be reduced by using only bins if they have the highest score within the area utilizing the continuous scoring. We are here similarly assuming that the high scoring bins inside a local area in general come from the same signal. The resulting images for the current method and the two layer method on the muon + minimum bias image of figure 18 is shown in figure 26

In the left hand image of figure 26 you clearly see that the current method reduces the amount of roads found significantly in the red muon-area. You see the same effect in the right hand image of figure 26 however and still the current method finds roads in many areas that are ignored by the two-layer CNN. For the CNN methods we get the road counts of table 6 for maximizing within a 3×3-area.

3×3-maxing	98.0 %	98.5 %	99.0 %	99.5 %
Single layer CNN	71	100	136	271
Two layer CNN	55	70	97	192

No 3×3-maxing	98.0 %	98.5 %	99.0 %	99.5 %
Single layer CNN	90	119	161	229
Two layer CNN	70	86	127	209

Table 6: The table shows the results acquired by choosing only bins that are maximal within a 3×3 area. The previous results without this 3×3 reduction is included in the rows below.

For the single layer CNN we find improvements in road counts all the way up to and including 99% efficiency. For the two layered network however there seems to be improvement all the way up to and with 99.5% efficiency, though especially for lower efficiencies. Comparing this to the reduced road count of the current algorithm the 2 layer model is able to reduce the total road count from 216 to 97 which is less than half while maintaining the same efficiency.

5.10 Method analysis

In this section the two layer method will be analyzed. The figure 26 clearly shows that while both the current and the suggested two layer method are able to find the muon the total amount of roads is significantly reduced by using the CNN model with and without 3×3 reduction. A hint of what happens can be seen by focusing on the region $0.41 \leq \phi_0 \leq 0.44, -0.9GeV \leq q/p_T \leq -0.7GeV$ in figure 18. This is a general high intensity region and is thus by the current method identified as possible muon roads when a bin has 7 hit layers or more. On the other hand, the suggested method does not identify any possible muon roads in this region, likely because true muons do not generate larger areas of high intensity. They make sharp high intensity peaks as can be seen in the muon image in the left hand side of figure 18. So while the crossing of multiple lines does indicate the presence of a muon, the CNN apparently identifies that the general behavior in the local area is not very ”muon-like”.

To be able to account for possible biases of the method, its dependency on different parameters is presented below. As the two layer method with 3×3 reduction produced the lowest road count at 99% efficiency the plots will be made for this algorithm.

Most of the simulated muons are in the central slices as shown in figure 27, where slices represent a region in (z, η). Additionally the amount of test muons is relatively low (1500). Therefore the threshold on the score is set to acquire a 98% total efficiency when plotting the distribution of muons by slice to better see any potential differences. It does not seem however that there is a significant slice dependency, but there could likely be if the amount of data was larger as there is a significant difference in the amount of background between the innermost and the outermost slices. This could count in both ways. More background makes higher amounts of hit layers per bin on average which is highly correlated with the probability that there is a muon in the specified bin

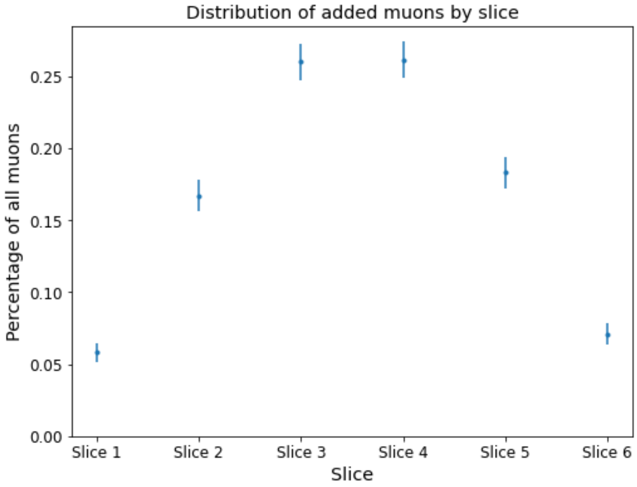

Figure 27: The distribution of added muons by slice. See figure 3 for an illustration of the slice-regions.

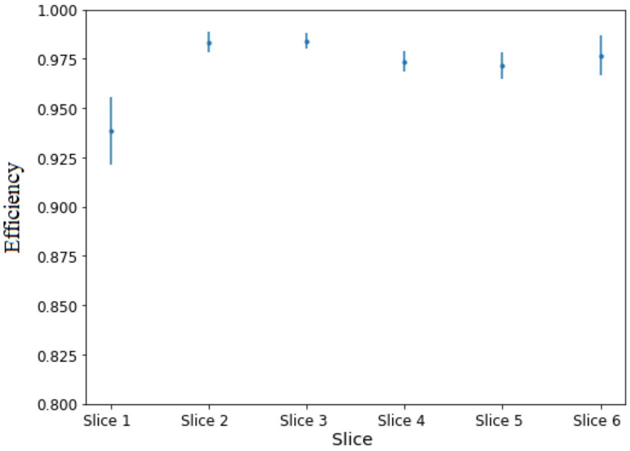

Figure 28: The efficiency on muon identification on the test set for 98% total efficiency by slice. 98% is chosen to increase the visibility of eventual deviations.

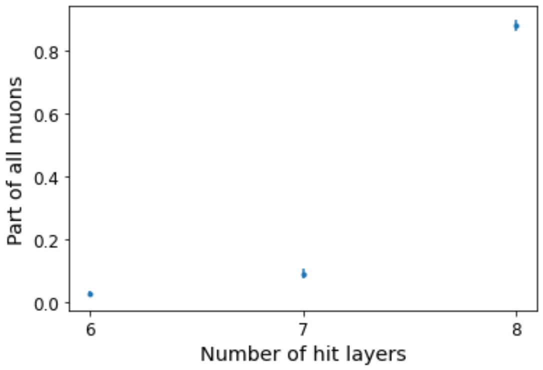

Figure 29: The amount of muons distributed by number of hit layers within a single slice.

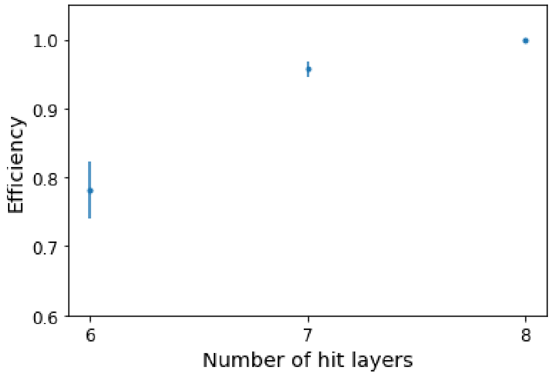

Figure 30: The amount of muons found by number of hit layers within a single slice with a cut on score so that 99% total efficiency is acquired. Unsurprisingly the efficiency increases with the amount of layers the muon hits.

why the muons more often will be found in central slices when putting a simple threshold on the bin count. On the other hand the general noise in an area will distort the charged particle pattern possibly reducing the probability assigned by the model identifying the pattern as ”non-muon like” and the correlation is therefore not that trivial for a CNN method.

A parameter that will definitely affect the efficiency is the amount of layers hit by the muons. Most of the added muons in the test set hit all 8 layers. But 33 and 113 muons hit only 6 and 7 different layers respectively within a single slice as shown in figure 29.

The currently implemented algorithm will find all muons that hit at least 7 layers inside a single bin, but only few that hit 6. The proposed algorithm on the other hand utilizes not just the central bin value but also the patterns surrounding it enabling it to find a larger portion of the 6 hit layers muons. In figure 30 the efficiency is put to 99% and shows how many 6,7 and 8 hit layer muons are found respectively.

As not all the 7 hit layer muons are found, the image shows that once in a while a 6 hit layer muon will be easier to find, either by own pattern or by the

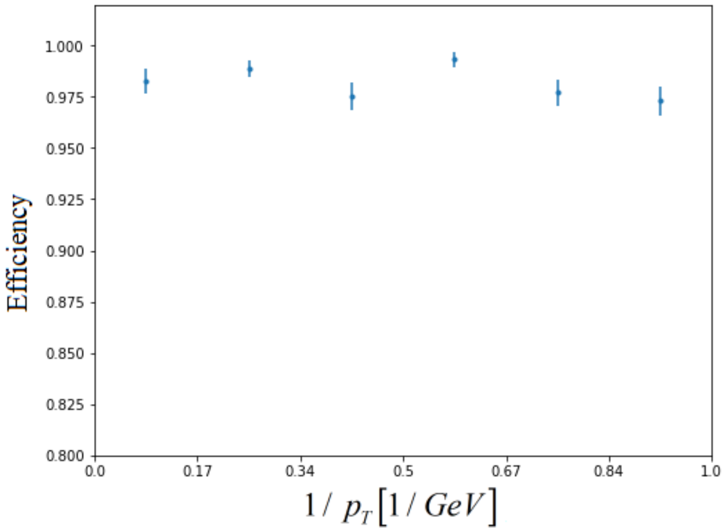

Figure 31: The amount of muons found by $1/p_T$ with a cut on score so that 98% total efficiency is acquired.

minimum bias clouding the image of 7 hit layers muons. The 8 hit layer muons are apparently so clear though that all in the test set are found no matter what minimum bias file they are mixed with.

Similarly the ϕ_0 and p_T distributions can be plotted. The distribution of muons found over all p_T (when choosing 98% efficiency) is shown in figure 31 and the same for ϕ_0 in figure 32.

There is no clear tendency in the distributions for p_T for ϕ_0 though there could be with a larger data sample. The model is a “local model”. That means, it takes information from the surroundings, not the values of p_T and ϕ_0. The only situations in which the model could depend on these is thus if the muon patterns are themselves dependent on these parameters or when found close to the edge of the image where the lag of information could lead to the model having a hard time to identify the signal. The former is unlikely as the lines are almost straight and the pattern near the crossing should therefore be almost the same for all $(q/p_T, \phi_0)$ within the image. The latter is partially corrected for by not evaluating on the out most edge of the image. However 2 consecutive 3×5 convolutions means that any point can be affected by the 5×9 nearest bins meaning that there is definitely some room for unfortunate edge effects, especially in ϕ_0. This would be something that could be corrected for however by not evaluating on the $(2, 4)$ outer most bins in q/p_T and ϕ_0 respectively.

5.11 Network analysis

The following section will attempt to identify what information the 2-layer CNN bases its decision making on. In general neural networks are difficult to interpret as they in include many variables and often are non-linear functions. This is also the case for the two layer network, even though it is rather simple. However there are certain techniques to visualize the information used by the network.

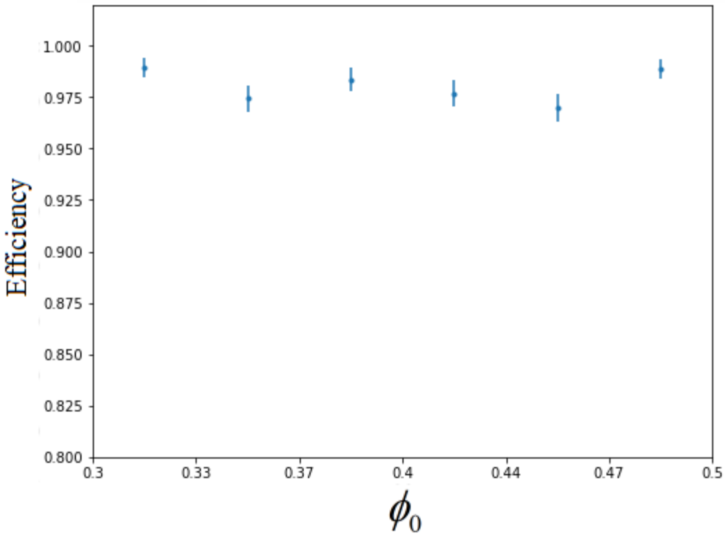

Figure 32: The amount of muons found by ϕ_0 with a cut on score so that 98% total efficiency is acquired.

By looking at the outputs of the last layer (inspired by the CAM-method, see [14]) of the network you can get an idea of what the network is looking for. In figure 33 is shown an image of a single muon and the same muon in minimum bias respectively.

The final convolutional layer of the two-layer cross entropy model has 10 filters (5 × 2 for mapping 5 channels to 2 as seen in figure 20). In figure 34 are shown two image outputs of the last layer that each contribute to the probability that there is a muon in each given point of the right hand image of figure 33. For an example of all filters contributing to the value representing the probability that there is a target in each bin see the appendix, section 7.2

The leftmost filter that I have named “Muon finding filter” seems to find the general position of the muon. The second image is a bit harder to interpret. But it seems that it finds thin lines of high intensity, along with valleys of low intensity adjacent to each other. In other words it seems to be focusing the probabilities. The problem of the focusing filter seems to be that it finds two hotspots in the image: around ($\phi_0 = 0.345, q/p_T = 0.52$) and around ($\phi_0 = 0.34, q/p_T = 0.485$). But combining with the information from the muon finding filter, there is no doubt which of these regions is the correct one!

Adding the result of these two filters we already see quite some increase in the intensity around the muon compared to the background from the original image of figure 33 to the combined image in figure 35.

Using the full model, that is, adding all 10 filters, allowing only bins with at least 6 hit layers and using the softmax function the final result of the model on this image is shown in figure 36.

In figure 36 it can be seen that the model has assigned 22 % for the central bin to be the target it is trained to look for and only assigning probabilities above 2.5 % to two other bins (which also have 8 layers hit by the muon). It should be noted that 22 % is a rather high number as even true charged particles from

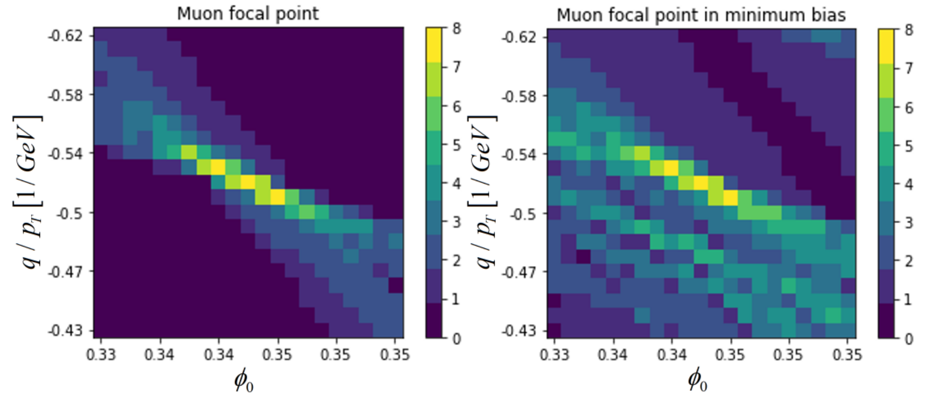

Figure 33: A muon focal point to the left, and the same muon in minimum bias to the right.

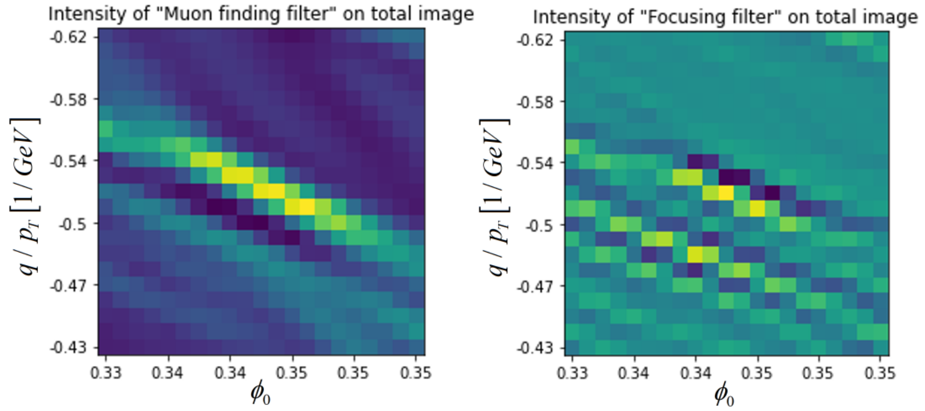

Figure 34: Two filters contributing to the evaluated probability that there is a muon in each bin of the right hand image in figure 33. The leftmost I have labeled ”Muon finding filter” as it intensifies the overall probability in the area where the true muon is found relatively to the other high intensity area bottom left of the muon. The rightmost I have labeled ”Focusing” filter as it creates thin lines of high intensity. On the other hand it seems to have assigned high values to the background to the bottom left of the muon. One could say that the Muon finding filter is very accurate, but not very precise. On the other hand the focusing filter is rather precise, but not very accurate.

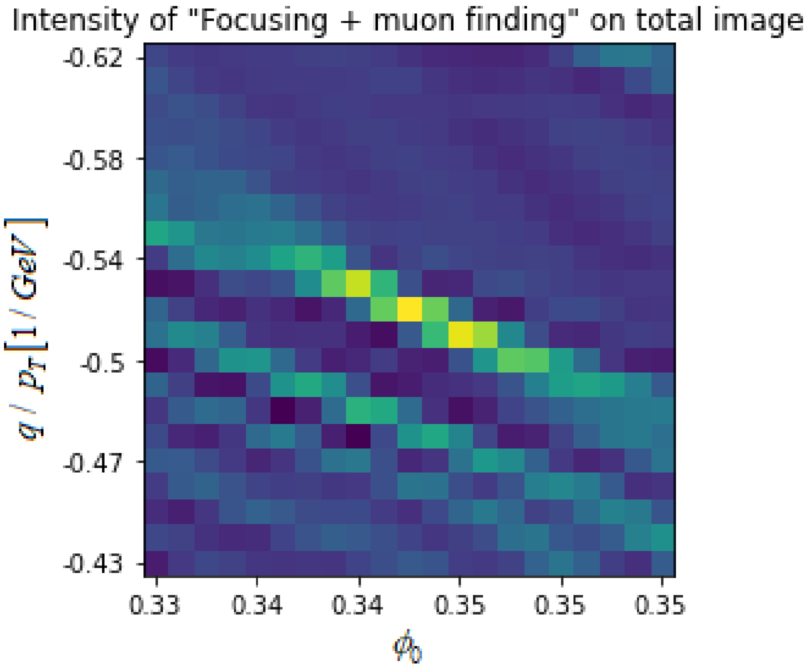

Figure 35: This image is the result of adding the two filters of figure 34. It appears more focused than the rightmost image of figure 33

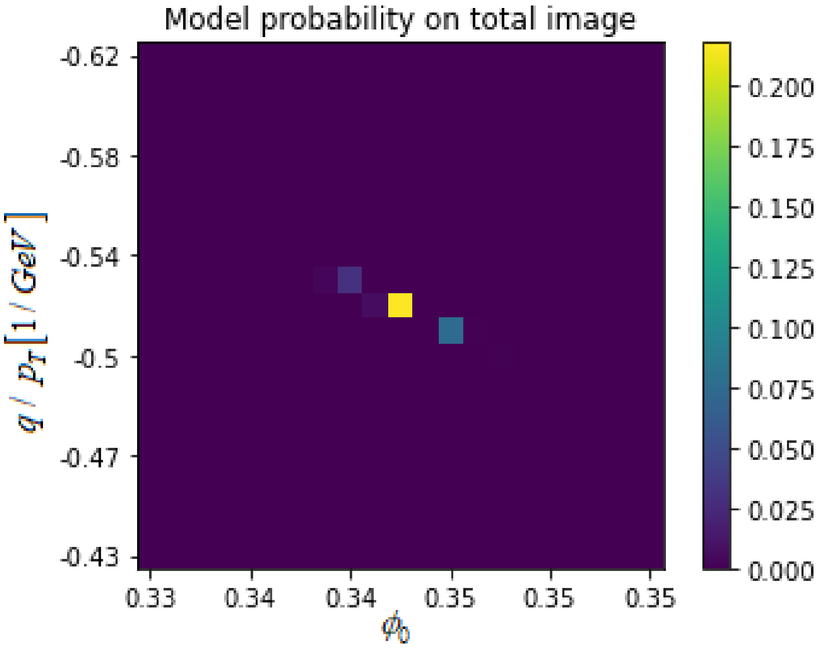

Figure 36: Probability scale

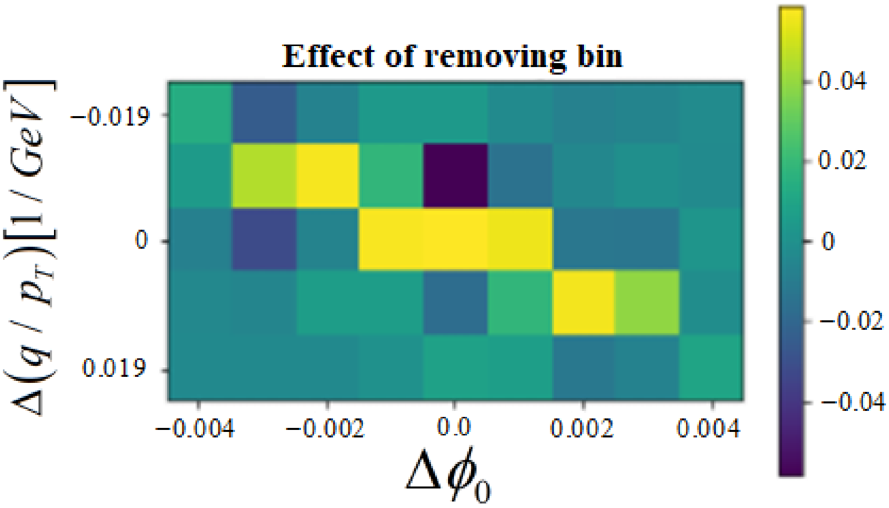

Figure 37: This image shows the change in resulting probability in the central bin caused by setting each bin around a muon target to 0 one at a time averaged over multiple images. $\Delta\phi_0$ and $\Delta\left(q/p_T\right)$ denote the shift in ϕ_0 and q/p_T with respect to the target bin. The shape of the image is 5×9 as this is the maximal range that can affect the bin probability after two layers of 3×5 convolutions.

the minimum bias are considered false while training and the prior probability being only 0.06 %. By prior probability I mean the probability that we should by chance pick the bin with the muon target out of all the 6 or more hit layer bins.

The two layer CNN model cannot be represented by a single convolutional kernel. One can try and estimate this however by, for multiple targets inside minimum bias, finding all bins that can affect it. For the two layer model we then have a 5×9 area for each image centered at the target point. Then one can try dropping them out (setting them to 0) 1 at a time to get a linear interpretation of how each bin affects the output result. Figure 37 shows how the value of each of these bins on average over many targets in minimum bias affect the score assigned to the target bin.

The yellow areas are where there is a loss of 5.5 % to 6 % probability on average that there is a muon, given the CNN, in the central point. Curiously it shows that 5 other points than the central bin itself (the 4 other yellow bins, and the dark blue) are almost as important for the presence of a target in the central point. In words the model states, that very important information is lost by using only the central bin value as is done for the current method. Also quite interestingly it shows that the presence of high values just above and below the center is a very bad sign. In other words a high steepness (see section 5.5) is required along the vertical axis . This suggests also that high steepness along the vertical axis had probably been an even more important factor in target finding than along the diagonal (called ”across” in section 5.5) which was used. This should probably be investigated in future target finding for this problem. Also it seems that low steepness on the vertical axis would be a strong indicator for a good target. The fact that these high importance bins are within 3×5 around the center also tells us why the single layer 3×5 kernel is able to acquire relatively good results. While missing some bins, and some nonlinear properties, it still has information from the most important ones.

Furthermore the low values along the edges of figure 37 can also be a clue as to why attempts to add extra layers have been unsuccessful. Information at this distance from the center is down prioritized, indicating that the model thinks the signal is feeble compared to the noise in this region. Considering that the strength of the signal decreases going further from the target point, adding extra long distance points would thus further increase the amount of noise compared to signal possibly disturbing rather than helping the training.

6 Conclusions and outlook

The increase in luminosity of the HL-LHC increases the demands for the data acquisition speed. Therefore the trigger system will be upgraded to better choose interesting events rather than put high cuts on p_T. For this the HTT is proposed as a hardware based method for fast triggering based on tracking that is stable in high minimum bias.

As an alternative to the pattern matching ASICs of the HTT the Hough transform can be used to identify charged particle tracks for later track fitting. However, the current method for charged particle road finding through Hough transform is missing important information. By using a CNN an optimal function based on local information can be found through gradient descent to find optimal solutions. It is therefore expected to decrease road counts compared to the current method significantly.

Picking out examples of the improvement in performance, the efficiency and road count is found to go from 99.2% efficiency and 349 roads for the current to 99% efficiency and 161 roads for the single layer CNN or just 127 for the two layer CNN. Reducing the number of roads within a 3×3 area decreases the efficiency and road count of the current method to 99% and 216, but similarly the road count for the two layer method decreases to just 97. A CNN based method is thus able to reduce the road count per image to less than half of the current method at 99% efficiency. This is such a decrease in road counts that it could have a significant influence on the viability of the Hough transform method when considering the exact final setup of the TDAQ. The decreased road counts for track fitting comes at a prize of increased computations in this step, why it is relevant to notice that even the single layer CNN without 3×3 reduction constitutes an improvement.

The two layer model with 3×3 reduction is not found to have any bias on investigated particle parameters within the measured precision. The two layer model uses multiple filters, but it is found that one of them tends to find the general position of the muon, while another focuses the high probabilities into small regions in figure 34. In figure 37 it is found that the most important bins for the probability given the two layer CNN that there is a muon in the central bin of a true target are also contained by the single layer model. Interestingly also, it seems that many bins are assigned almost as much importance as the central bin value, which is all that is used by the current method, showing that the CNN thinks that there is much more information to be gained than the central bin.

For further work on CNNs for road finding in Hough transformed images it remains to be evaluated how the CNN would in practice be implemented in the TDAQ event filter more than just show what it can achieve as is the work presented here. Further the performance needs to be studied in other regions of the detector, in particular other regions of η where the amount of background noise will be higher.

In relation to computational power this study presents road counts that are proportional to the computations needed for track fitting. The computation time saved in track fitting is not measured compared to the increase in the track finding process and this is naturally essential to fully evaluate which road finding method to use.

Currently some of the background is interesting objects, in the sense that they are charged particles of sufficient momentum and within the area of interest. The training could possibly be helped therefore by a data extract in which it would be possible to assign the same target finding as for the single muons. In relation to this it is naturally necessary also to evaluate the model performance on other charged particles than muons.

To improve the CNNs an important consideration is the quality of the target finding as this is essential for the quality of the training. A possible way to optimize this is to try other steepness parameters like the horizontal and vertical that are shown in figure 37 to be of importance.

The remaining strip layer could be added to the Hough transform. Likely this would increase the performance of the CNN as more layers means higher certainty that a high count bin originates from a charged particle and not random hits.

A stronger GPU could be useful as the size of the current has been some limit. While indications from trial and error in this study are that there it is no benefit by using information from further away, the amount of channels could be increased to possibly search for more patterns and would ease training and evaluation for larger statistics.

References

[1] Mårtensson, M., *search for leptoquarks with the ATLAS detector and hardware tracking at the High-Luminosity LHC*, (PhD dissertation, Acta Universitatis Upsaliensis), 2019, page 67, 83 [`http://urn.kb.se/resolve?urn=urn:nbn:se:uu:diva-390352`]

[2] *Technical Design Report for the Phase-II Upgrade of the ATLAS Trigger and Data Acquisition System*, The ATLAS Collaboration, CERN, June 2018, pages 1, 8-9, 22, 44, 92-93, 110, 117-118, [`https://cds.cern.ch/record/2285584/files/ATLAS-TDR-029.pdf`]

[3] Hsu, Shih-Chien, *ATLAS Inner Tracker (ITk) Upgrade*, The ATLAS Collaboration, CERN, Jan 2018, page 4, [`https://cds.cern.ch/record/2302625/files/ATL-ITK-SLIDE-2018-073.pdf`]

[4] Brenner, Richard, *Hardware Tracking for the Trigger (HTT) in ATLAS*, 2019, slides 5,7, [`https://indico.cern.ch/event/742793/contributions/3298729/attachments/1821634/2979760/ATLAS_HTT_CTD-WIT2019.pdf`]

[5] Xu, Riley, *Hough Configurations*, HTTSim team, The ATLAS Collaboration, March 2021, slide 2, [`https://indico.cern.ch/event/1017912/contributions/4271819/attachments/2208459/3737104/HTTSim%2021-03-16.pdf`]

[6] Pequenao, Joao and Schaffner, Paul, *How ATLAS detects particles: diagram of particle paths in the detector*, The ATLAS Collaboration, CERN, Jan 2013, [`https://cds.cern.ch/record/1505342?ln=no`]

[7] Valente, Marco *The ATLAS Trigger and Data Acquisition Upgrades for the High-Luminosity LHC (HL-LHC)*, The ATLAS Collaboration, CERN, July 2019, [`https://cds.cern.ch/record/2692161/files/ATL-DAQ-PROC-2019-020.pdf`]

[8] *The Pythia 8 A3 tune description of ATLAS minimum bias and inelastic measurements incorporating the Donnachie-Landshoff diffractive model*, The ATLAS Collaboration, CERN, August 2016 , [`https://cds.cern.ch/record/2206965/files/ATL-PHYS-PUB-2016-017.pdf?version=1`]

[9] *Geant4 developments and applications*, J. Allison Dubois, P. Arce, Araujo H., Apostolakis J., Amako K., Northeastern University, February 2006, [`https://cds.cern.ch/record/2206965/files/ATL-PHYS-PUB-2016-017.pdf?version=1`]

[10] *Convolutional neural networks: an overview and application in radiology*, Rikiya Yamashita, Mizuho Nishio, Richard Kinh Gian Do and Kaori Togashi, August 2018, [`https://insightsimaging.springeropen.com/articles/10.1007/s13244-018-0639-9`]

[11] *ADAM*, Pytorch documentation, Pytorch, 2019 [`https://pytorch.org/docs/stable/generated/torch.optim.Adam.html#torch.optim.Adam`]

[12] *Adam: A Method for Stochastic Optimization*, Kingma, Diderik P. and Ba, Jimmy, 2017, [`https://arxiv.org/abs/1412.6980`]

[13] *Treatment of Errors in Efficiency Calculations*, Ullrich, T. and Xu, Z., Brookhaven National Laboratory, October 2018, [`https://arxiv.org/pdf/physics/0701199v1.pdf`]

[14] *Learning Deep Features for Discriminative Localization*Kingma, Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva and Antonio Torralba, Computer Science and Artificial Intelligence Laboratory, MIT, [`http://cnnlocalization.csail.mit.edu/Zhou_Learning_Deep_Features_CVPR_2016_paper.pdf`]

7 Appendix

7.1 App1: Loss convergence plots

Here is presented plots of converging loss functions for the two suggested CNNs in figures 38 and 39. The loss shown is the cross entropy loss divided by the number of bins with counts of 6 or more as the rest are ignored. The losses start relatively low because a prior probability is calculated for the model based on how many targets there are compared to bins of at least 6 counts. This is added as a starting value for the final bias term so that image independent models will map each bin to this prior probability. In both plots the loss of the train and validation set seem to follow each other nicely indicating that the networks are not over fitted.

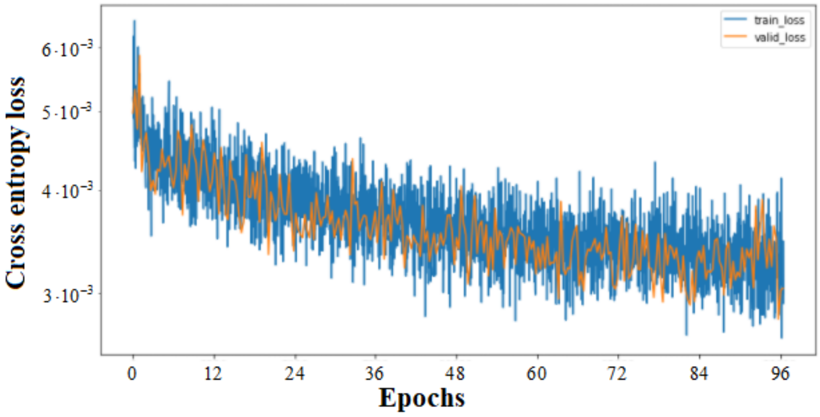

Figure 38: The plots shows the converging loss for the single layer model. The training loss does not seem to divergence from the validation loss, indicating that the model is not over fitting.

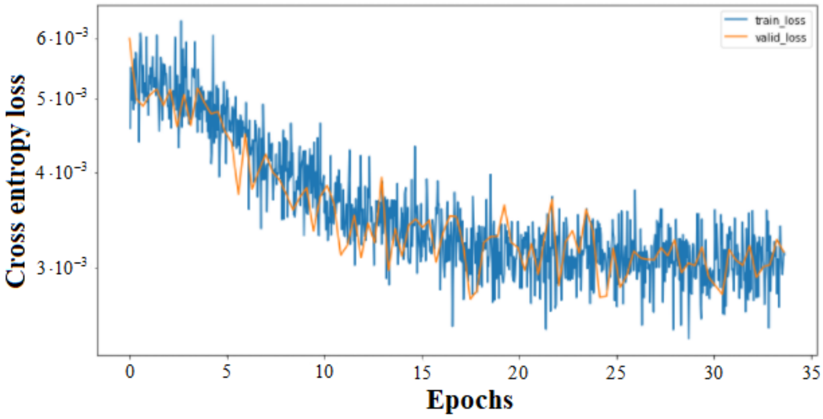

Figure 39: The plots shows the converging loss for the 2 layer model. The training loss does not seem to divergence from the validation loss, indicating that the model is not over fitting. It converges much quicker than the single layer model.

7.2 App2: Network filters

All five filters of the last layer of the two layer CNN contributing to the bin representing the probability that there is a muon are shown in figure 40. Apparently filter 3 does nothing, indicating that one less channel would be sufficient. Filter 0 finds two highest intensity positions and filter 1 and 4 help indicating the correct of these. Filter 2 seems to have been miss led by the background but the resulting image after soft max and removing low hit layer bins looks nice.

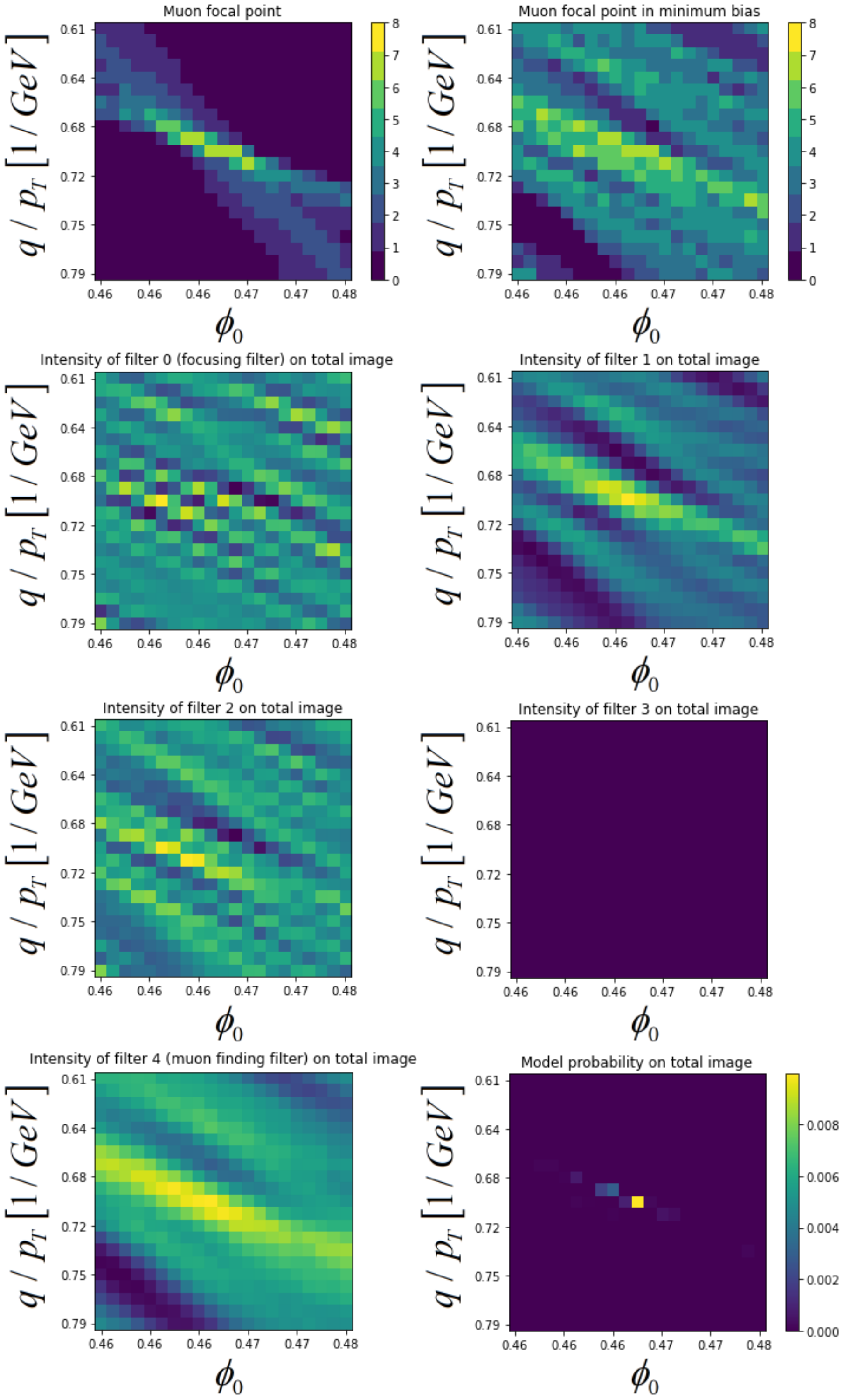

Figure 40: A muon focal point and the same muon in minimum bias are presented on the first row. The next are all filters of the two layer CNN contributing to the probability that there is a muon in each bin. The last image is the output probabilities. Apparently filter 3 does nothing.