Technical University of Munich

School of Natural Sciences

Bachelor Thesis

Neural Networks for Photon Reconstruction in Electromagnetic Calorimeters for the COMPASS and AMBER Experiments

Aumiller, Sara

07.07.2023

Theme proposal/Reviewer: PD Dr. Jan Friedrich
Supervisors: Dominik Ecker, Dr. Stefan Wallner

Abstract

As traditional methods for photon reconstruction in electromagnetic calorimeters at the COMPASS and AMBER experiments at CERN are pushed to their limits, we consider an alternative approach using artificial neural networks. This thesis is a proof of principle study on whether neural networks can improve the reconstruction of photon positions and energies detected with the calorimeter on simulated data. The network can determine the position of a single photon with higher precision than the traditional fit method and shows roughly the same performance in energy reconstruction. Although the network was trained on simulated data, it succeeds in reconstructing real-data photons with the same accuracy as the traditional fit method. In the case of two overlapping photons, the network still performs better in position determination than the traditional method, however the energy prediction of the traditional method remains superior. This lack of performance might be due to the network's architecture being chosen too simply. Examination of the network's ability to count the amount of photons detected on the calorimeter showed that using neural networks might improve the separation of small-distance photon showers but should be trained sensibly to predict the correct number of photons in all cases. In short, artificial neural networks show potential in photon reconstruction in electromagnetic calorimeters but the network's complexity and training process must be chosen carefully to achieve better results than traditional methods.

Contents

1 Introduction

To prove new theories in high-energy physics, we need the ability to reconstruct particles produced in a high-energy particle collision. Therefore, precise particle detectors and detailed knowledge about advanced post-processing of data are needed. Modern high-energy physics often requires high precision experiments, such as the COMPASS[7] and AMBER[5] experiments at CERN. Interaction and decay photons are measured in electromagnetic calorimeters (ECAL). Reconstructing the energy and position of these photons with high accuracy is needed for conclusive physics results. Traditional reconstructing methods which rely on fitting photon-shower profiles to clusters measured with the ECAL show sufficient results and are thus eagerly developed further.

In recent years, a new method, machine learning, has been attracting more and more attention with performances showing potential to outperform traditional algorithms. Today, machine learning is easily accessible to anyone through free open-source software libraries like Tensorflow[2] that support python-interface implementations such as Keras[1]. Therefore, we consider machine learning as an alternative approach for photon reconstruction. We will use deep learning, more precisely artificial neural networks, to determine the correct amount of photons hitting the ECAL and their positions and energies from the ECAL signals. The method appears to be a reasonable approach as other experiments at CERN are also exploring reconstruction with the help of deep learning[8]. The main research question investigated in this thesis is as follows:

> **Research question**
>
> Can neural networks improve the reconstruction of photons measured in electromagnetic calorimeters with respect to the position and energy determination of one or more photons?

This is a proof-of-principle study. Therefore, the research question will be evaluated on simulated data using a realistic experimental setup from the COMPASS experiment.

2 COMPASS and AMBER experiments

2.1 The COMPASS experiment

COMPASS (COmmon Muon and Proton apparatus for Structure and Spectroscopy) is a fixed target experiment at the M2 beamline at CERN[7]. It took data between 2002 and 2022. It addresses a variety of open questions from Quantum Chromodynamics (QCD) by investigating the hadron structure and by looking for bound states of the strong interaction in spectroscopy experiments[9]. Depending on the physics program, it uses different targets. Also, the M2 beam line can provide a wide variety of different beams; hadrons, electrons, or muons with different energies and intensities. The spectrometer setup follows a two stage approach[7]: First, particles traveling in larger angles with respect to the beam axis are detected with the large-angle spectrometer setup containing a magnet (SM1), a ring imaging Cherenkov counter (RICH), an electromagnetic calorimeter (ECAL1), a hadronic calorimeter (HCAL1) and a muon filter. Then the set-up repeats similarly (SM2, ECAL2, HCAL2 and muon filter 2) in form of the small-angle spectrometer to detect particles with smaller angles as they are now, further away from the target, spread out wider. The experimental setup can be seen in figure 1.

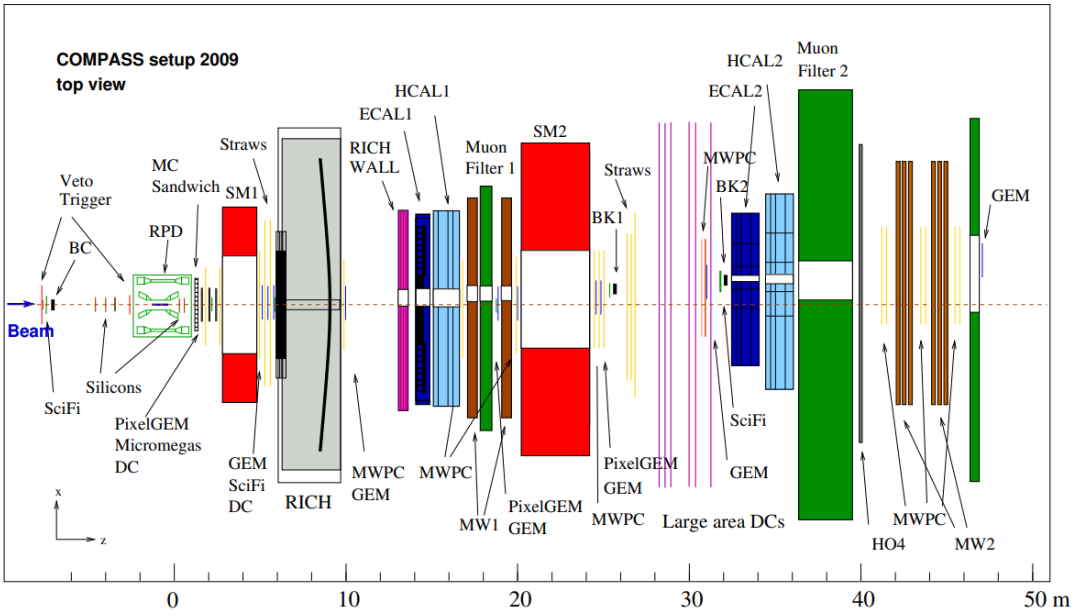


Figure 1: Sketch of the 60 m long COMPASS two-stage spectrometer. From[4]

Multiple of the COMPASS physics programs require good photon detection in the reconstructed final states[7]. Therefore, two electromagnetic spectrometers, ECAL1 and ECAL2, are implemented in the setup. An example of a physics program, which relies on high-resolution calorimetry, is the Primakoff program. The program addresses open questions in low-energy QCD in the non-perturbative regime which can be addressed in so-called Primakoff reactions. In these reactions ultra-relativistic particles scatter electromagnetically on heavy target nuclei. The COMPASS Primakoff program uses a negative pion beam which scatters on nuclear targets such as nickel[9]. There are different final states of interest for the program: $\pi^-\gamma$, $\pi^-\pi^0$, $\pi^-\pi^0\pi^0$ and $\pi^-\pi^+\pi^-$. Neutral pions decay almost instantly into two photons. This results in final states with one, two, or four photons which need to be reconstructed in the calorimeters. The physically interesting region for the physics program covers decay-photon energies within 2 to 180 GeV[9].

2.2 The AMBER experiment

The AMBER experiment is the successor experiment of COMPASS at the M2 beam line of the CERN SPS[5]. Three physics programs are envisaged in phase-1:

1) Proton charge-radius measurement using elastic muon-proton scattering
2) Drell-Yan and J/ψ production experiments using the conventional M2 hadron beam
3) Measurement of proton-induced antiproton production cross sections for dark matter searches

The initial data taking started in 2022. Again we give an example for the need of a precise electromagnetic calorimeter. Therefore, we will examine the proton charge-radius measurement in more detail. The measurement aims to contribute to a solution of the proton-radius puzzle. There are currently no measurements of the Q^2-dependence of the electric form factor using muon-proton scattering in high-energy regimes. AMBER aims to provide a proton charge-radius with a statistical accuracy of 0.01 fm and even smaller systematic uncertainty. This is a high precision experiment. AMBER can investigate the sources of the discrepancy too. The insufficient description of radiative effects might be one of them. The electromagnetic calorimeter will detect the radiative events by measuring the emitted soft photons with energies up to 2 GeV[5].

2.3 ECAL2 and its event reconstruction

We have seen that precise electromagnetic calorimeters are needed in the experimental setup of the COMPASS and AMBER experiments. Therefore, we have a closer look at the ECAL2, in the following referred to as ECAL, which is located at $z = 33.2$ m and is part of the small-angle spectrometer. It consists of 3068 modules with dimension of 3.83 x 3.83 cm^2[4]. The active area from the front (244 x 183 cm) is illustrated in figure 2. The ECAL has three different types of cells: lead glass (1332 modules) also called GAMS, hardened lead glass (848 modules) also called GAMS-R and shashlik (888 modules). Four modules are missing in the center part of the ECAL for beam particles to pass. The shashlik modules are 39 cm long and can detect particles with energies up to 200 GeV. If an electron, positron or photon hits the ECAL, it interacts via bremsstrahlung or pair-production. A cascade of secondary particles, a so-called shower, is produced which can be measured due to ionization and light signals[16]. From this signature the energy of the initial particle and its hit position on the ECAL can be determined.

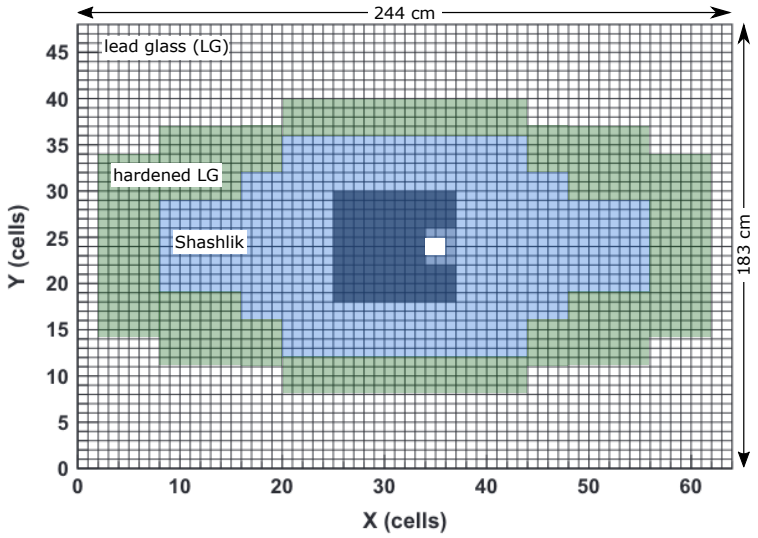


Figure 2: Active area of ECAL2. There are three different types of cells: lead glass (white), hardened lead glass (green) and Shashlik (blue) from the outer parts to the center. There is a hole of 2 x 2 modules in the middle-right part of the ECAL. Taken from[10]

The event reconstruction is performed using time and signal amplitude information[4]. All adjacent modules which detect a signal at the same time form a cluster. The energy deposition in a cluster needs to be associated with one or multiple incident particles. A single particle produces a shower in the ECAL. One cluster can contain multiple showers. We try to reconstruct the showers to determine the exact energy and hit position of the particle. In the following, the separation of clusters into showers and the determination of the energy and position of the initial particle will be referred to as "Lednev fit", or short "Lednev", after one of the creators of the ECAL reconstruction algorithm A. A. Lednev[14]. The algorithm has been further developed by Sebastian Uhl[19].

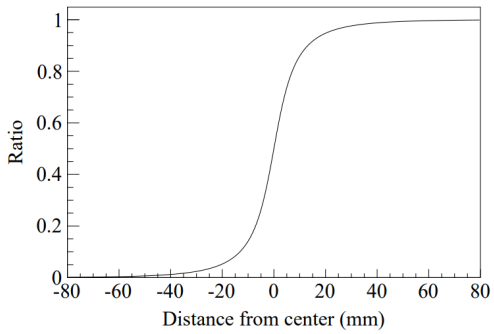


Figure 3: Cumulative function $F(x)$ that describes the energy deposition of a shower integrated along an axis x. Taken from[4].

The analysis starts by defining the clusters. Then lateral shower profiles need to be fitted into each cluster. Therefore, precise knowledge about the shower profile is needed. Empirical cumulative functions are used to describe shower profiles[13]. If the shower is projected onto an axis, we can integrate the energy from $-\infty$ to a point x from the shower center. The function converges to one when x reaches the end of the cluster. This gives an arcustangens-like function as seen in figure 3. To describe the different slopes, multiple arcustangens are added up so that the cumulative function in one dimension can be written as:

$$F(x) = \frac{1}{2} + \frac{1}{\pi} \sum a_i \arctan \frac{x}{b_i} \tag{1}$$

The shower profile can be parameterized by a_i and b_i which give the relative contribution and the width of contribution i correspondingly. The idea can be expanded to two dimensions. The parameters are once determined for real data with an electron calibration beam and single-photon events, and once for the simulation framework. Additionally, the ECAL cells are calibrated to the π^0 mass by using the decay channel of the neutral pion into two photons which are measured with the ECAL. The invariant mass $m_{\gamma\gamma}$ is calculated from a whole dataset in an iterative procedure for each cell individually so that each cell is calibrated with an energy-dependent constant to shift the $m_{\gamma\gamma}$ peak to the true pion mass.

Now that we have a precise knowledge of the shower profile, we can start fitting showers into clusters. We start with one shower in a cluster and place the middle of the shower profile onto the cell with the highest energy[4]. Then a log-likelihood fit is performed. Considering some constraints, we add a second shower to the cluster and perform the log-likelihood fit again. As we have twice as many free parameters, the cluster is fitted better. To judge whether it was physically sensible to describe the cluster with two showers instead of one, meaning to avoid artificial splitting, we compare the log-likelihood normalized to the number of degrees of freedom. If two showers describe the cluster better, the whole procedure is repeated with three showers. The procedure ends at a maximum of six showers in one cluster. The fit returns the energy E of a shower and the position x, y, z of a shower in the ECAL. If the fit already fails in the beginning, the energy will be returned as the sum of the energies deposited in all modules and the position will be given as the center of gravity of the deposited energy.

3 Neural networks

Deep Learning, which is a subdomain of Artificial Intelligence, being more precise a subdomain of Machine Learning, describes learning from data with computation through multi-layer neural networks and processing[17]. The word "deep" likes to emphasize the concept of multiple levels through which data is processed.

3.1 Supervised learning

In principle there are three different types of machine learning tasks[11]: Supervised, unsupervised and semi-supervised learning. We will focus on supervised training. This means the network can learn from a knowledge database with labeled data. Most commonly the dataset is split into three parts:

- Training set: Data used for training the network
- Validation set: Data used to evaluate the performance of the network during the training
- Test set: Only used after the training and validation phase to predict the final performance of the network.

The trainings set is normally chosen to be the largest dataset. A 80:20 split between trainings set and test is commonly used. The validation set can be taken as a subset of the training set. Supervised training is a two-phase algorithm: First, there is the training and validation phase, then the testing. Supervised learning can be split into two different families[11]:

- Classification: The labeled data is discrete. The aim is to learn the classification boundaries to predict the correct label.
- Regression: The target data is continuous. The aim is to predict a continuous value.

A common way to present a classifier's performance is the confusion matrix. Considering a binary classification problem, a confusion matrix shows the true positive and the true negatives values on the diagonal and the false positives and false negatives values on the anti-diagonal. The rows represent the actual class while columns show the predicted class.

Parametric models can be described using a function $f(\theta)$ with parameters θ[11]. Learning describes the procedure of adapting the parameters θ to a given input X and a desired output Y. A model used for fitting data must match the complexity of the problem: Trying to fit a parabola with a linear function represents the data insufficiently (underfitting) but an 8th order polynomial will fit through every point instead and looses generality (overfitting). To learn generality, neural networks need to be trained on a sufficiently large training dataset.

3.2 Network architectures

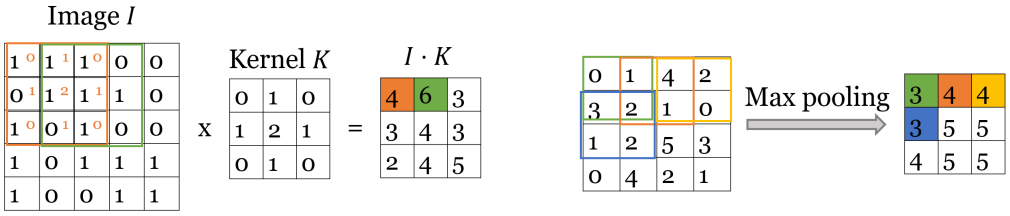
A network is build by connecting many layers of nodes[11]. How the nodes and layers are connected is defined by the network architecture. Networks with the most-simple architectures are:

- Fully connected networks (FCN): Nodes between two adjacent layers are all connected to each other, but neurons within a layer have no connection.
- Convolutional neural networks (CNN): Local connections of the nodes, meaning they are only connected to their local neighbors in the subsequent layer, and parameter sharing are used. CNN are very popular as they can extract relevant features e.g. from pictures efficiently[6]. They use fewer parameters which simplifies the training and increases the speed.

The concept of CNN in 2D will be explained in the following. One convolutional layer consists of a collection of convolutional filters called kernels. In our case they are two-dimensional. A kernel K slides over the whole image I while the dot product is calculated at each point. Mathematically, this can be written as[11]:

$$(K \cdot I)_{i,j} = \sum_m \sum_n K(m,n) I(i-m, j-n) \tag{2}$$

The procedure is illustrated for an easy example image and kernel in figure 4a. This operation generates a feature map. Normally a pooling layer is added after a convolutional layer. This is done to downsize the feature maps. A pooling layer also slides kernels over the image and returns a single value for each kernel position, e.g. the mean or the maximum value. Max pooling is graphically illustrated in figure 4b. In these examples shown in figure 4 the kernels always moves by one. This is not mandatory, e.g. they can also slide by two each time. How much a kernel moves is referred to as a stride.



(a) Example of a convolutional operation in 2D. The kernel K slides over the image I and performs the dot product at each point.

(b) Max pooling operation. A kernel slides over the image and for each position the maximum value is returned.

Figure 4: Graphical explanation of common CNN features: The convolutional operation (left) and pooling operation (right).

Let us come back to nodes. We will focus on feed-forward neural networks, meaning nodes are not connected in cycles. Nodes which are inspired by biological neurons take input data, process it and decide whether to fire a signal. The mathematical model[11] of a node can be written as

$$\sigma(b + \sum w_i x_i) = y \tag{3}$$

where σ is the non-linear activation function, b the bias and w_i the weights which are applied to each input x_i. The weights and the bias of each node will be adapted during training. A node is graphically illustrated in figure 5.

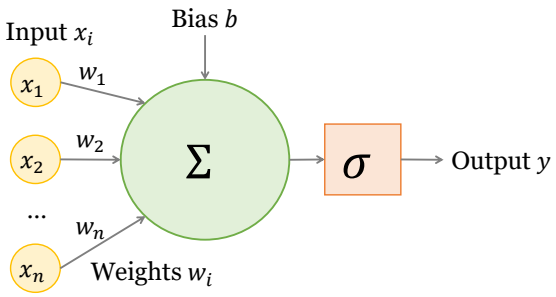


Figure 5: Illustration of the mathematical description of a node

Activation functions need to be non-linear so that the network is able to approximate any function. If otherwise, a multi-layer network could always be reduced to a network with a single hidden layer which could only separate the input data linearly. The most commonly used activation function is ReLU (Rectified Linear Unit)[11].

$$\sigma(x)_{\text{ReLU}} = \max(0, x) \tag{4}$$

In classification problems the N outputs should sum up to 1 as probabilities of the input being classified into the different categories are returned. This can be achieved using the softmax activation function[6].

$$\sigma(x)_{\text{softmax},i} = \frac{\exp x_i}{\sum_{j=1}^{N} \exp x_j} \tag{5}$$

3.3 Training the network

After defining our model, we can discuss how to train it. The relationship between the output of a model $\hat{y}$ and the desired output y can be measured using a loss function $\mathcal{L}$. This can be the mean-squared error[11]:

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \tag{6}$$

In the training phase we aim to minimize the loss by small iterative adjustments of the model's parameter[11]. The parameters are usually initialized randomly and follow a normal distribution. We search for the global minimum by guiding the searching process towards the direction of the maximum descent of the loss function. This method is called gradient descent. The model parameters W are updated in one step like[11]:

$$W_s = W_{s-1} - \eta \Delta \mathcal{L}(\text{dataset}; W_{s-1}) \tag{7}$$

η is the learning rate and one of most important hyperparameters for the learning phase[11]. Choosing it too high will make the training unstable while choosing it too small may not lead into the global minimum. Nowadays adaptive learning rate optimization methods are used. The most common one is the Adaptive Moment Estimation (ADAM)[11]. To apply the adaption of parameters to all layers and nodes backpropagation is used. If a network has processed all samples of the training set, we call it one trained epoch. We train the network over many epochs until it reaches the global minimum. To decide whether a network is fully trained, we look at the loss of the validation and training data over the epochs. If the validation loss rises again and overfitting happens then the best trained network is the one at the epoch of the validation loss minimum. This is illustrated in figure 6.

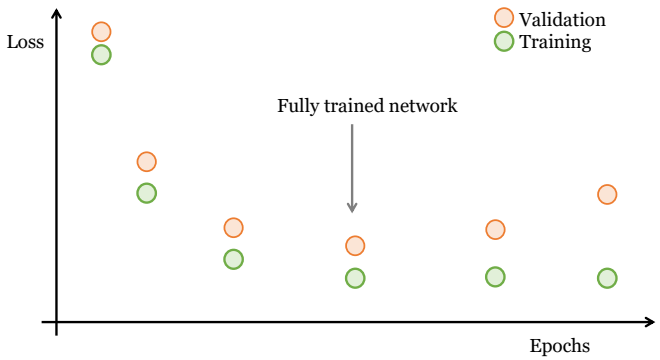


Figure 6: Illustration of training and validation loss over the epochs. A network shows overfitting if the validation loss rises again. The best and fully trained model is given at the epoch of the validation loss minimum.

Overfitting is a common problem when using neural networks. This means the network only adapts explicitly to the given training data but shows poor performance on the unseen test sample. To overcome this problem, regularization is used. The most common regularization is known as dropout[11]. New network architectures are build by dropping neurons, so setting their output to zero, with the probability p. Which neurons are dropped out is chosen randomly at each new training iteration. A single node can therefore not rely on the others and co-adaptation between them, which hinders generalization, is reduced[11].

3.4 Slot Attention: A modern approach to object-centric representations

After introducing the basics about neural networks, a modern and more advanced network architecture, Slot Attention[15], will be explored in the following. The Slot Attention module is an architectural component of a neural network. It produces a set of abstract representations of the objects detected in an image. We call them slots. One slot can bind to any object in the input. The binding of the slots to the objects happens through a competitive procedure over multiple rounds of attention[15]. The Slot Attention method produces a set of output vectors that hold permutation symmetry. Slot Attention is highly competitive with other related approaches but is more efficient in memory consumption and faster to train[15]. Figure 7 illustrates the working principle of a Slot Attention module. Three objects can be seen in the initial picture. The slots in green, blue and red bind to the three objects over three rounds of attention. The fourth orange slot describes the background.

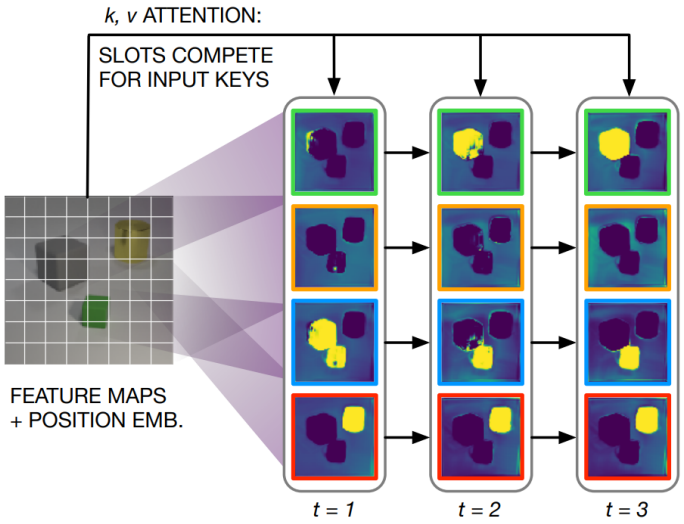


Figure 7: Illustration of a Slot Attention module. Taken from[15]. Four slots (green, orange, blue and red) compete over three round of attention ($t = 1, 2, 3$) for the three objects in the picture. At round three $t = 3$, three of the slots (green, blue, red) hold one object each. The other slot (orange) describes the background.

There are multiple ways to implement Slot Attention. It can either be used for object discovery or for set predictions[15]. In object discovery we consider unsupervised learning. The input image gets encoded into a set-structured hidden representation of the objects in the image. This can be done using CNN followed by the Slot Attention module. Then each slot will be decoded into an image which shows the object represented by this slot. Adding up all the decoded slots should return the initial image. Set prediction uses supervised learning. The input image gets encoded again by a CNN and the Slot Attention module. Then a multi-layer perceptron (MLP), which is similar to a small FCN, is applied to each slot. The outcome can then be compared to the labeled data using a permutation invariant loss function as the order of predictions and labels is arbitrary. The state-of-the-art permutation invariant loss function uses the Hungarian algorithm[12].

Furthermore, Slot Attention holds the potential to detect a varying number of objects. Figure 8 shows the performance of a Slot Attention model on a modified version of a dataset called CLEVR, provided by the Multi-Object Datasets library[15]. The dataset holds images with up to six objects per image. Figure 8 illustrates how an image gets separated into different pieces by the slots. If there are more slots than objects, for example in the first image, then the background is spread uniformly across the empty slots, in this case slot 3 and 5.

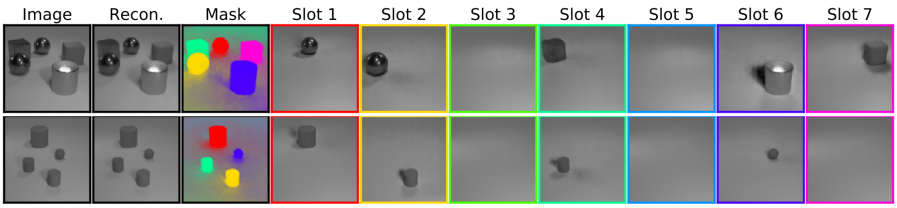


Figure 8: Visualization of per-slot reconstruction of a Slot Attention model on a modified dataset of CLEVR. Taken from[15].

4 Photon reconstruction with neural networks

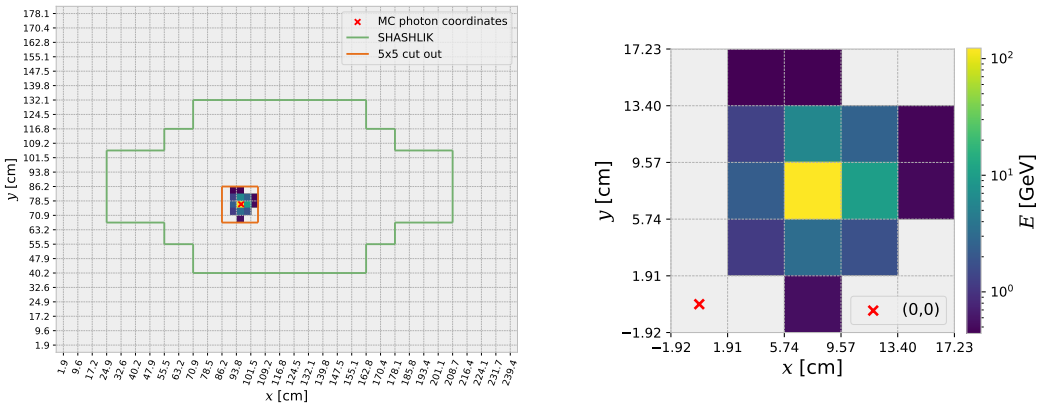
We will explore photon reconstruction in the ECAL2, from now on only referred to as ECAL, with neural networks as an alternative approach to the analytic Lednev fit. We use simulated data from the Monte Carlo (MC) simulation framework COMGEANT based on GEANT3[4]. The data is then processed using the COMPASS reconstruction framework called CORAL. The Lednev fit is a part of the CORAL framework[3]. The simulated data is generated using the same simulation and reconstruction settings as the Primakoff run in 2009. The MC simulation is performed by Dominik Ecker. Simulated data offers information about the true position, energy, and number of photons. It is hence a convenient starting point for this proof of principle study because we can use supervised learning. We will also evaluate whether a network trained on simulated data gives reasonable results on real data, see section 4.1.7.

4.1 Reconstruction of one photon

We start with attempting to reconstruct single photons hitting the ECAL one at a time. The aim is to correctly determine the energy E of the entering photon and its position (x, y) on the ECAL surface at $z = 3302.7$ cm.

4.1.1 The simulated MC data

In supervised learning, neural networks will learn any existing correlation in the given data. So the used dataset is extremely important and will be presented in the following. The ECAL consists of different cell types as explained in section 2.3. In this study only events in the shashlik region are considered to avoid different ECAL cell responses. In addition, there must be at least one cell with no energy response in between a cluster and the edge of shashlik-GAMSR modules as well as to the hole in the middle so that no signal gets lost. How do we give the event as an input to the neural network? One can consider every cell output in the shashlik region as one input parameter of the network. For the entire shashlik area, this corresponds to 888 input nodes[4]. However, a single photon event typically deposits energy in less than 25 cells while the rest of the cells has a response of zero energy. By cutting out the region where the photon hits the ECAL, we reduce the complexity of the problem without loss of information. As even high-energy photons form clusters that fit into 5x5 cells, we take a 5x5 grid with the measured energy of each cell as the network's input. If a cluster is smaller than 5x5 cells, it is randomly placed in the grid. Cells that are not hit have a value of zero. An example is shown in figure 9. In addition to the photon energy, the network is supposed to learn the position of the incoming photon. However, by only considering the cut-out region of the ECAL the information about the global position of the photon in the ECAL is lost. Therefore, we introduce a new coordinate system which has its origin in the middle of the lower left cell of the 5x5 grid. The network then learns the (x, y) position of the photon with respect to the new coordinate system.

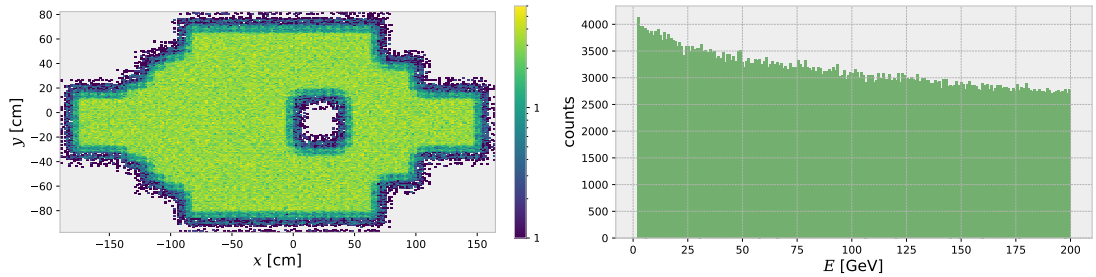


(a) ECAL2 with one cluster in the shashlik region

(b) The same cluster in the 5x5 grid with a new coordinate system (red)

Figure 9: Example event with cut out cluster.

The first dataset (dataset 1) contains photons travelling parallel to the beam axis z. They cover an energy range of $E_\gamma \in [2, 200]$ GeV. This energy range covers the sensitive and most precise range of the shashlik ECAL2 modules[4]. They illuminate the whole shashlik region evenly. Dataset 1 has 753 956 suitable clusters after removing problematic events which are too close to the hole or to the GAMS-R area. The second dataset (dataset 2) contains photons which reach the calorimeter not perpendicular to the z-axis but at an angle $\alpha_z \in [0, 0.027]$ rad so that the photons point back to the target. Since the target dimension is small compared to the calorimeter, every point on the ECAL surface can be correlated to a narrow angle range. The photons in dataset 2 have the same energy range as dataset 1. Dataset 2 contains 628 548 suitable clusters. The distributions of photon energies and positions in this dataset are shown in figure 10. The events are again distributed equally over the whole shashlik part and there are slightly more events at low energies than at high energies.



(a) Positions of all photons which illuminate the shashlik cells. The region of the hole is cut.

(b) Energy distribution of all used clusters.

Figure 10: Distributions of dataset 2.

We use 80% of the data for training the network and save 20% as the test sample. 10% of the training set is used as validation data.

4.1.2 Architecture of the neural network

As the single-photon case can be reduced to 25 input parameters corresponding to the deposited energy in each pixel of the 5x5 grid, we use a FCN to determine the energy and position of the photon. This is a regression problem. The conceptional architecture can be seen in figure 11. The output of the network has the form $[x,y,E]$ with the units of cm and GeV. The network has 86 339 parameters, uses the ReLu activation function in all layers but one as the last layer has no activation function. It consists of seven layers with different amount of nodes. We use the Adam optimizer with a learning rate of $\alpha = 0.00001$ and the mean squared error (MSE) as a loss function. The batch size is chosen to be 64. We use the same network architecture for dataset 1 and 2 only that we add a normalization layer for dataset 2. Therefore, we calculate the mean of 3.9 GeV over all cell energy values of dataset 2 and a variance of 16.2 GeV.

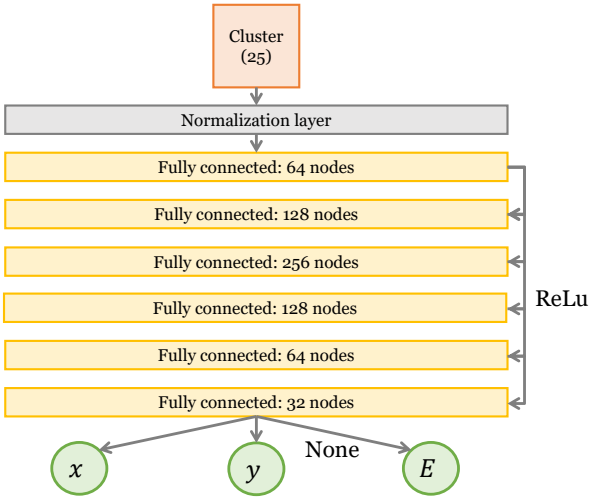


Figure 11: Illustrated architecture of the neural network.

4.1.3 Results and comparison to Lednev

To begin with, the networks are trained for 200 epochs on each dataset to get a first impression of the performance. The fully trained network is shown in chapter 4.1.6. The networks are evaluated on the test samples. The outputs of the neural networks can now be compared to the simulated quantities x_{MC}, y_{MC} and E_{MC}. To estimate the performances on the two datasets the difference between the positions $x_{\mathrm{NN}} - x_{\mathrm{MC}}$, $y_{\mathrm{NN}} - y_{\mathrm{MC}}$ and the relative difference of the energies $\frac{E_{\mathrm{NN}} - E_{\mathrm{MC}}}{E_{\mathrm{MC}}}$ are shown in the histograms in figure 12 and 13. For comparison, the performance of the Lednev fit is shown too.

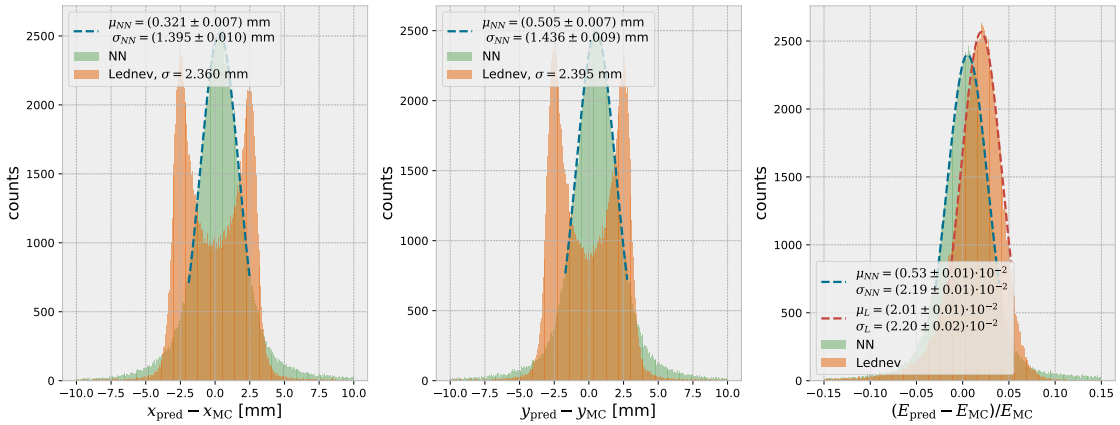


Figure 12: Performance of the neural network ("NN", green) and Lednev (orange) on dataset 1 ($\alpha_z = 0$). Differences between the predicted and true values of (left) the x-position, (middle) the y-position and (right) the energy E. While the Lednev fit shows two peaks in the position determination, the network is centered with one peak closely around zero. The network and Lednev fit show similar performance on the energy E.

Figure 12 shows the differences between the predicted and true values for x, y, and E for both, the Lednev fit and the network. The variance can be interpreted as the spatial and energy resolution of the ECAL achieved with the different methods. The network performs better than the Lednev fit on dataset 1 with $\alpha_z = 0$ in the determination of photon position (x, y) as the variance is 1.7 times smaller, and it shows no double peak. The double-peak structure from Lednev indicates one of the drawbacks of this algorithm: Showers hitting the left part of the cell have their positions pushed further to the right, and showers hitting the right part of the cells are moved to the left by the algorithm. The performance on the energies of the neural network and the Lednev fit have the same variance of $\sigma_E = 2.2\%$ but the Lednev fit shows a stronger shift in the mean value μ. This is a known feature of the used COMGEANT simulation framework[10]. The network has a longer tail compared to Lednev.

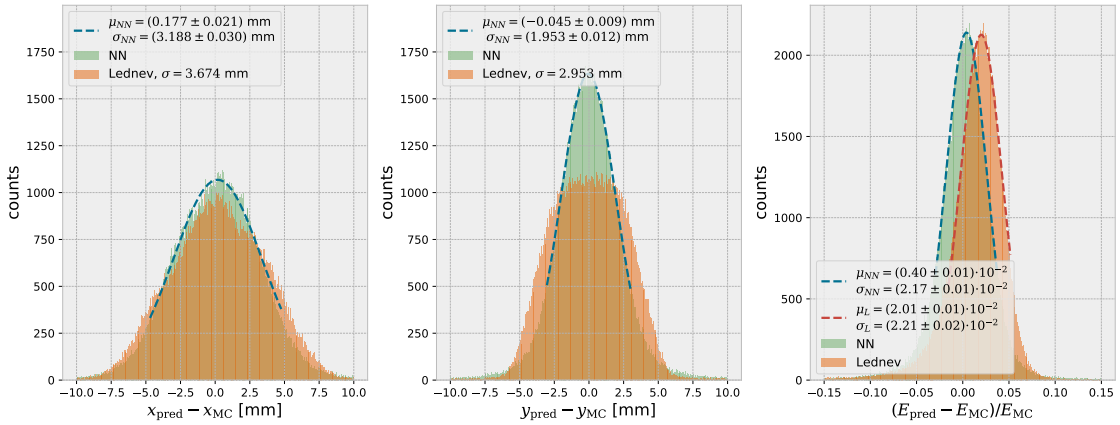


Figure 13: Performance of the network (green) and Lednev (orange) on dataset 2 ($\alpha_z \neq 0$). Differences between the predicted and true values of (left) the x-position, (middle) the y-position and (right) the energy E. The performance of the network and Lednev on x is poorer than on y.

Looking at the performance on dataset 2 with $\alpha_z \neq 0$ in figure 13, the performance on the energies stays roughly the same for both with $\sigma_E = 0.022$ when comparing to the performance

on dataset 1. The performance on x drops drastically for both. The variance σ_x of the network doubles. The performance on y also drops but only by 23% for the Lednev fit and 38% for the network. As the only difference between dataset 1 and 2 are the angles of the photons towards the beam axis, this must be the source for the poorer performance on (x, y). If a photon enters the ECAL with an angle, the particle shower happening in the cells will propagate with this angle. This means that the position with the highest-deposited energy in the ECAL does not equal the inertial position of the photon on the ECAL surface. It is plausible that the variance of the neural network for x is larger than for y as the angles towards the x-axis can be larger than towards the y-axis due to the shape of the shashlik region. The 2D-histogram in figure 14 proves the missing angle correction of the network. There is a linear shift of the positions x and y as a function of the angles.

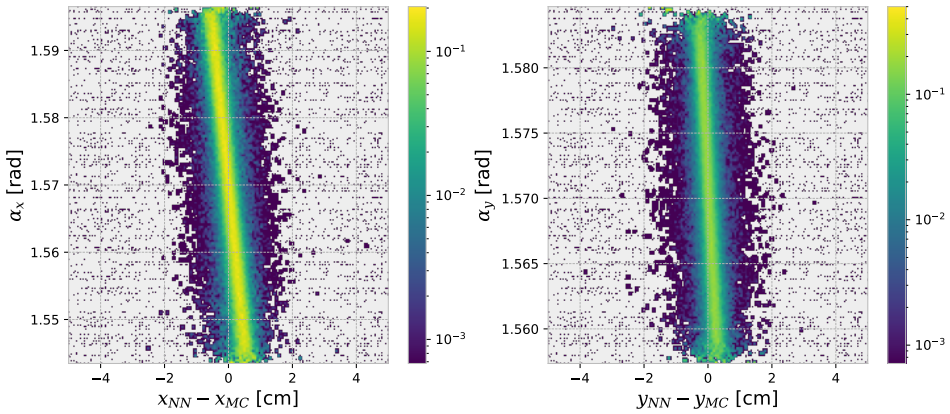


Figure 14: Normalized 2D-histogram of (left) $x_{\mathrm{NN}} - x_{\mathrm{MC}}$ and (right) $y_{\mathrm{NN}} - y_{\mathrm{MC}}$ against the angle towards the corresponding axis. The linear shift shows that the network does not correct for the angle of the entering photon

We have to conclude that the distortion of the cluster shapes due to the angles of the photons is too subtle compared to the relatively rough cell structure. The network is hence not able to reconstruct small angles from the cluster shapes and the information is lost.

4.1.4 Position correction due to angles

To confirm the hypothesis and check for plausibility, we check whether the network learns the correlation between the cluster position and the angle: We give the global position $(x_{\mathrm{coord}}, y_{\mathrm{coord}})$ of the new coordinate system of the 5x5 grid as an additional input to the network. The network can then technically learn the correlation between the position of the cluster in the ECAL and the angle it needs to correct for. If the network learns the correction with this additional information, we can confirm that the shower shapes of the clusters in the shashlik region change too subtle with varying angles for the network to detect.

We use the same network as before for dataset 2 but change the input size from 25 to 27 as the global position of the coordinate system in the 5x5 grid is added. The performance of this network on x, y and E can be seen in figure 15 in orange. The network with local information only is shown in green as comparison.

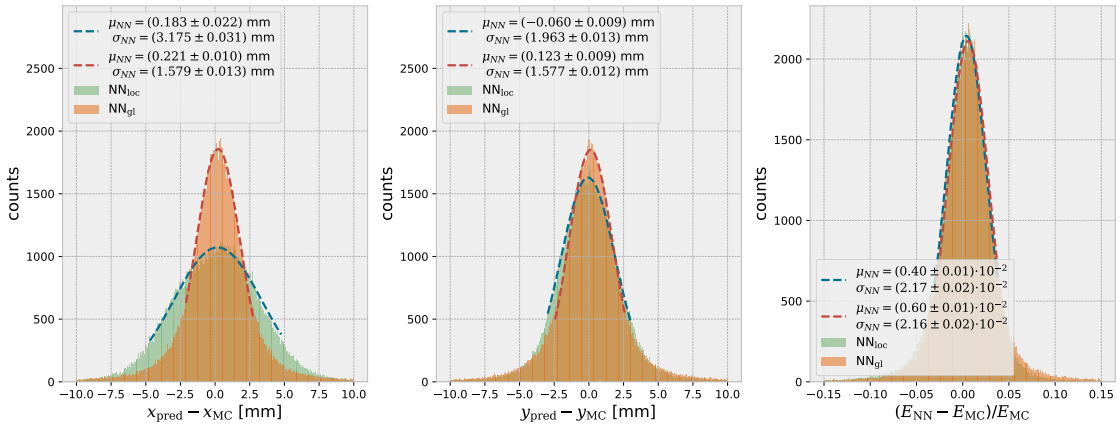


Figure 15: Performance of the network with added information about the global position of the cluster (orange) and for comparison the network with local information only (green) on dataset 2. Differences between the predicted and true values of (left) the x-position, (middle) the y-position and (right) the energy E. The network with global information shows an increase of performance on the position determination compared to the network with local information.

The performance on (x, y) of the network with global information about the cluster position increases significantly compared to the network with local information. The variance drops from $\sigma_x = 3.175$ mm to 1.579 mm and $\sigma_y = 1.963$ mm to 1.577 mm and is comparable to the performance on dataset 1 with $\sigma_x = 1.395$ mm and $\sigma_y = 1.436$ mm. The performance on the energy determination stays the same. The 2D-histograms 16 of the position deviation against the angles α_i show that the network with global information about the cluster position learns the linear shift with respect to the angle.

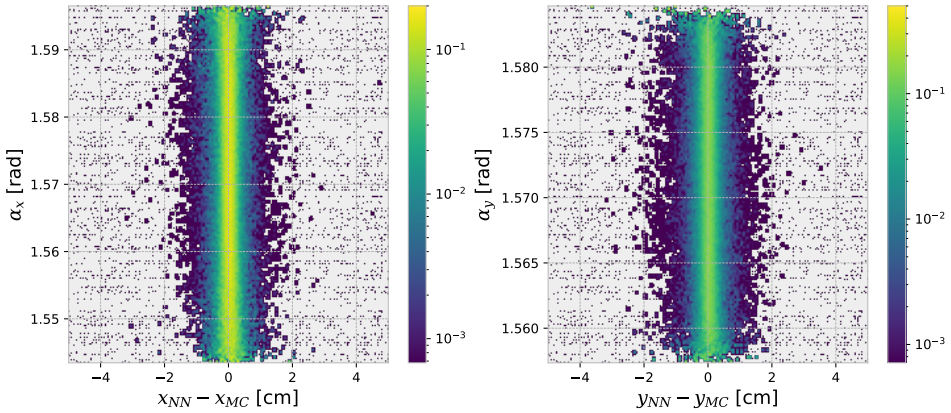


Figure 16: Normalized 2D-histogram of (left) $x_{\mathrm{NN}} - x_{\mathrm{MC}}$ and (right) $y_{\mathrm{NN}} - y_{\mathrm{MC}}$ against the angle towards the corresponding axis of the network with the global information about the cluster position. The straight line shows that the network performs for every angle with the same precision.

Further, the shift of the reconstructed position with respect to the position of the photon on the calorimeter surface depends on the penetration depth. The deeper a photon deposits its energy, the more its reconstructed position is shifted. The higher the energy of a photon, the larger the penetration depth into the ECAL. This means that the correction in x or y due to angles is smaller for low-energy photons than for high-energy photons with same angles. To check

whether the network corrected better than a linear shift and therefore also learned about the energy dependence of the correction, the shift on the performance with local information was done manually for comparison. We fit the linear function $\alpha(x) = a \cdot x + b$ to the data in figure 14 with $a = -0.050$ rad/cm and $b = 1.570$ rad so that every x can be corrected: $x_{\text{corr}} = x - \frac{\alpha - b}{a}$. Now the corrected x-histogram can be compared to the x-histogram from the network with global information about the cluster. The comparison is shown in figure 17.

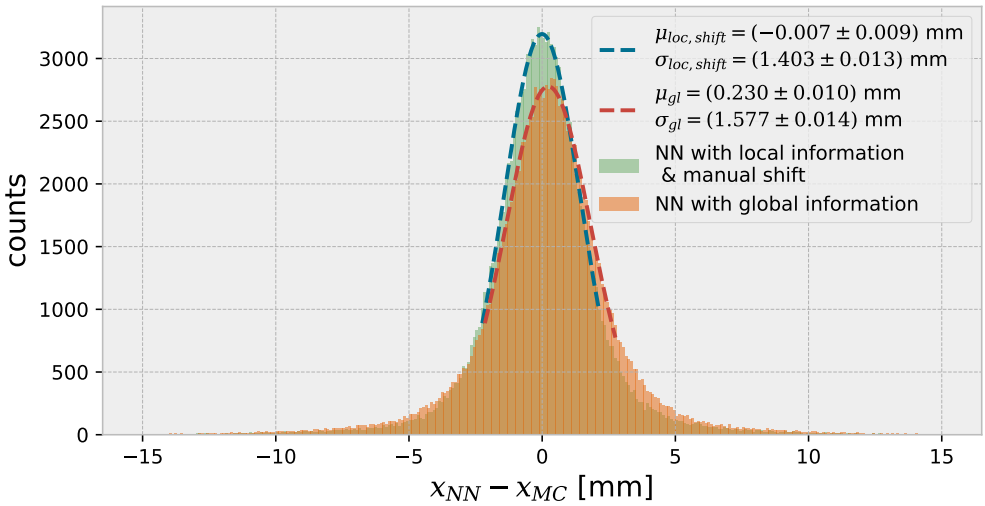


Figure 17: $x_{\text{NN}} - x_{\text{MC}}$ histogram of the network with information about the global cluster position (orange) and manually-corrected $x_{\text{NN}} - x_{\text{MC}}$ histogram of the network with no information about the cluster position on the ECAL (green). The variance is smaller when doing the shift manually.

The manually shifted result has an 11% lower variance than the neural network with information about the global cluster position. This means that the network did not learn the energy dependence of the shift as otherwise it would have outperformed the manual analytic correction. The greater variance shows that the network did not learn the linear shift of the position with the same precision as the manual shift. The energy dependence of the angle correction might be a small correction compared to other effects. For example one point of the ECAL does not correspond to a single angle but to a narrow angular distribution. This is because the place of origin (x_0, y_0, z_0) of the photons is not point-like at (0,0, z_{target}) but expands over the target in x, y and z. By performing the manual shift, every coordinate gets shifted using the angle and not the position on the ECAL which leads to a more precise result.

To summarize: Giving the global position of the cluster on the ECAL to the network improves its performance on the photon position significantly. The learned correction is the linear relationship between the angle and position shift with no energy dependence. This correction could only be learned as the position on the ECAL and the angles are correlated in dataset 2 as all photons start at the target. This will not be the case in real data where photons and other particles, which generate an electromagnetic shower in the ECAL, can also be created in secondary vertices. Therefore, a position on the ECAL can not be correlated with an angle anymore. The performance of the network will drop in this case, although the cluster shape might deform into an ellipse for larger angles which might then be traceable for the network, so that it may learn the angle correction from the cluster shapes.

To reduce the effect of the photon angles on the reconstructed position, we can shift the z-position of our reconstruction. By placing z to the average penetration depth of the shower profile in the ECAL instead of on the ECAL surface, the angle effect will reduce significantly. We

calculate the average penetration depth s as $s = \frac{2}{a_x + a_y} = -20.07$ cm where $a_{x,y}$ are the fitted parameters of the manual shift in figure 14. We will apply this z shift from now on, meaning that the reconstructed position will now be at $z = 3322.77$ cm instead of at the ECAL surface.

4.1.5 Placing the clusters in the grid

Coming back to the basics, one might be wondering whether placing the clusters randomly in the grid is the best method. To study this, we look at the network with global information about the grid position. Of particular interest are the clusters which the network has had the most difficulties to learn. Therefore, we will analyze all values that lie more than one σ away from the average μ when looking at the plots in figure 15. The histograms of this study can be seen in figure 18.

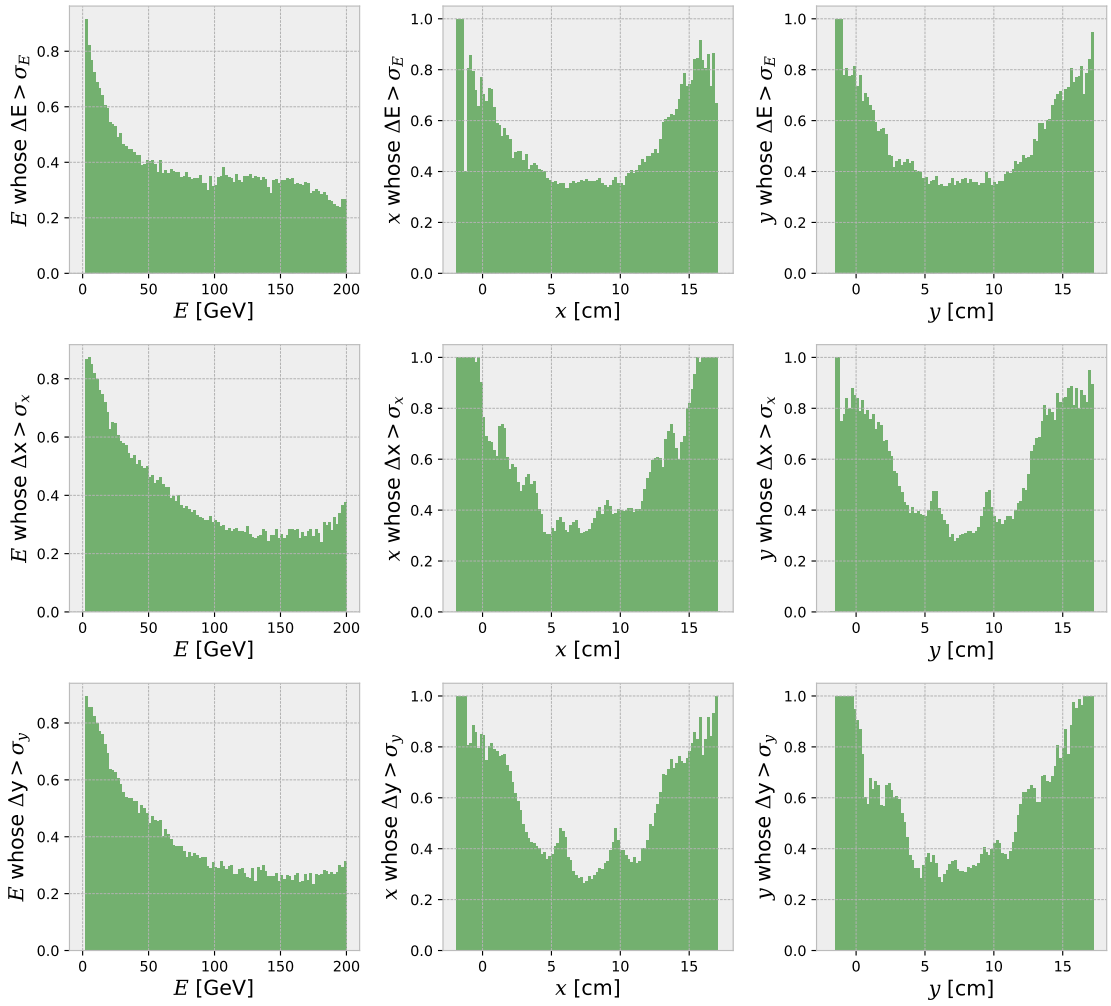
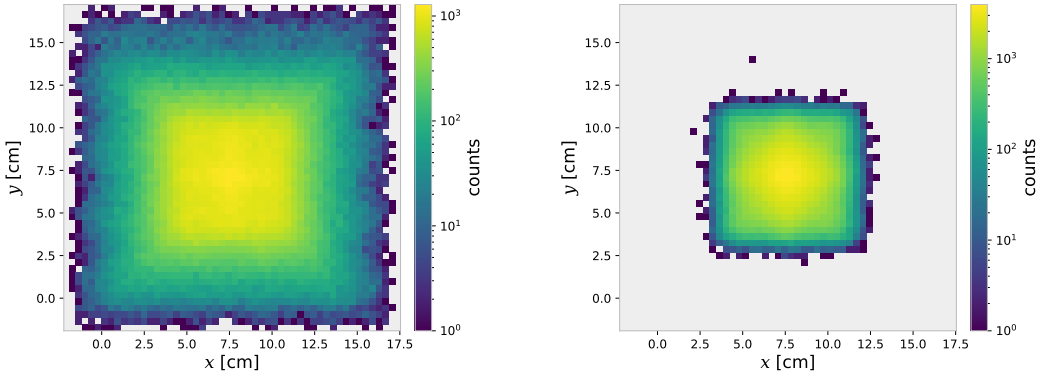


Figure 18: Histogrammed values of quantities x, y, E that lie more than one σ away from the mean μ of figure 15. The histograms are normalized to the amount of all events per bin. E.g. the first (second/third) row shows events whose energy values E (x/y) are badly learned. For these badly learned events all quantities x, y, E are histogrammed as a fraction of all events. The symbol Δ indicates the difference to the MC truth value so $\Delta x = x_{\mathrm{NN}} - x_{\mathrm{MC}}$, $\Delta y = y_{\mathrm{NN}} - y_{\mathrm{MC}}$ and the relative difference of the energy $\Delta E = \frac{E_{\mathrm{NN}} - E_{\mathrm{MC}}}{E_{\mathrm{MC}}}$. Events with low energy or events hitting the grid close to the edge are learned worst.

The first row in figure 18 shows that if the energy of an event is badly learned the energy values

are probably low. This is reasonable. Each cell has an energy threshold before firing a signal. So part of the photon energy will always stay undetected below this threshold. For smaller energies this fraction is larger compared to the total energy of the photon. The position of low-energy events is also detected worse (first plots in the second and third row in figure 18) as low-energy photons stimulate fewer cells. The extreme example is the one-cell event where only one cell fires. The uncertainty of this event in the position must be half the cell size ± 1.92 cm and can not be specified with any more precision. If more cells are hit, the center of gravity can be determined with more precision. Events with badly learned energy often lie more to the outside of the 5x5 grid as can be seen in the second and third plot of the first row. There is no physical explanation for this behavior, so it must be a learned bias by the network. Looking at the photon distribution over the 5x5 grid of the training data in figure 19a, the amount of photons hitting the grid closer to the edge is drastically lower than the amount of photons hitting the grid in the center. Although the photons are placed randomly in the 5x5 grid the shape of the cluster is limiting the uniform illumination of the grid. This bias in the training data leads to the bias in the neural network. This effect can also be seen when looking at the second and third plots in the second (third) row of figure 18 that show the position of events with badly learned x (y) values. Nearly 100% of the events hitting the grid on the edge lie more than one σ away from the distribution. The spikes in these plots may come from the periodic structure on the cells. Events that hit closer to the cell edge will probably also stimulate a neighboring cell so that the center of gravity can be determined with more precision compared to a photon hitting a cell center. To reduce the effect of badly learned events close to the grid edge, a more uniform distribution of the photons in the grid is desirable. This can be achieved by placing the clusters as centered as possible as seen in figure 19b. Therefore, the clusters will be placed in the middle of the grid from now on.



(a) Clusters placed randomly in the grid. The distribution peaks in the middle and drops to the edges

(b) Clusters placed in the middle of the grid which makes the distribution more uniform in the hit area

Figure 19: Photon distribution within the 5x5 grid

4.1.6 The fully trained model

To finish the single-photon study, we will present a fully trained network using a dataset similar to dataset 2 but with twice the amount of clusters (1 377 741). We will call it dataset 3. The network used is the same as in the last chapters which is graphically illustrated in figure 11. As mentioned in the last chapters 4.1.4 and 4.1.5, we let the network determine the position of the clusters 20.07 cm behind the ECAL surface to reduce effects due to angles and place the clusters centered in the grid to get a more uniform distribution. The same hyperparameters are used for the training as before (MSE loss function and Adam optimizer with a learning rate of $\alpha = 0.00001$, batch size of 64). How to decide whether a model is fully trained is discussed in chapter 3.3.

We increase the number of epochs from 200 to 3000 epochs. But the network still shows no overfitting, so no increase of the validation loss over 3000 epochs, which can happen if the network is too small to learn the full complexity of a problem. Therefore, we add another 256 nodes fully-connected layer after the already existing one. The network has now 152 131 parameters instead of 86 339. We will refer to this model as "final model" or "final network" from now on.

The loss function of the final model can be seen in figure 20 and shows overfitting. We save the network at the minimum validation loss after 1849 epochs. The training to this epoch took around 9h on one GPU.

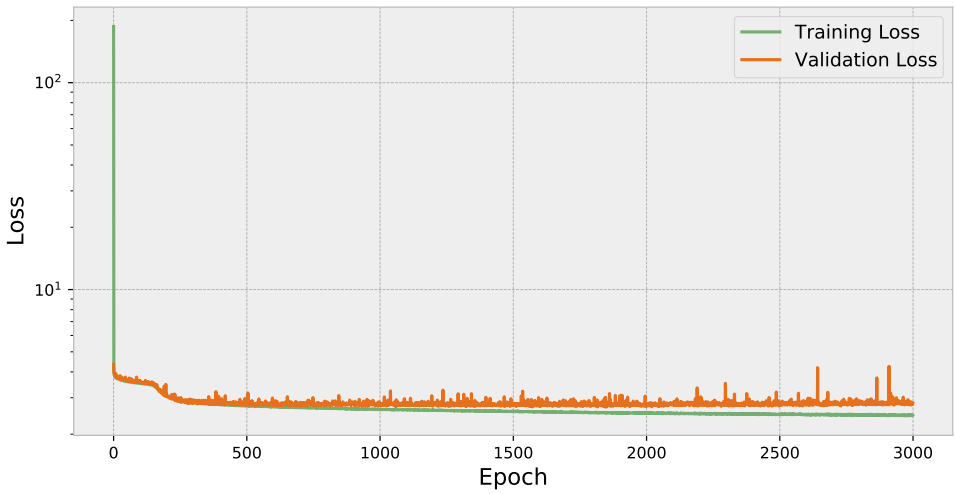


Figure 20: Loss function of the final model. Overfitting occurs after 1849 epochs as the validation loss rises again. This shows that the network is fully trained after this point.

The performance of the final model is shown in figure 21. It shows a clear sharpening on the predicted position compared to the previous networks. The variances are $\sigma_{x,\mathrm{NN}} = 0.909 \pm 0.005$ mm, $\sigma_{y,\mathrm{NN}} = 0.888 \pm 0.005$ mm and $\sigma_{E,\mathrm{NN}} = (2.01 \pm 0.01) \cdot 10^{-2}$. The network improves the spatial resolution by a factor of 2.6 for x and 2.7 for y and the energy resolution by a factor of 1.1 for E compared to the Lednev resolution.

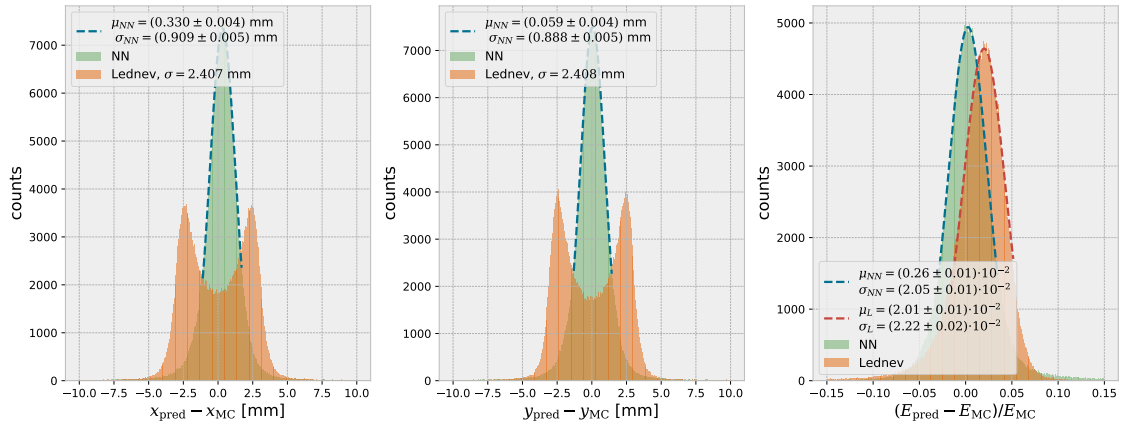


Figure 21: Performance of the final model network on the parameters x, y, E from left to right. The network outperforms Lednev in the position parameters but the performance on the energy is similar.

One-cell events are good candidates to check the final model for plausibility and validity. Events where only one cell is hit have low energy and will probably hit the cell closer to its center as otherwise a second cell will fire. If the network has no trained bias one would expect a sharp peak in the middle of this cell. The outcome of this study is shown in figure 22. The network shows indeed a centered peak with only few outliers. The mean of the true photon positions is $x_{\text{MC}} = 1.935$ cm and $y_{\text{MC}} = 1.917$ cm. The mean of the predicted photon positions is $x_{\text{NN}} = 1.930$ cm and $y_{\text{NN}} = 1.812$ cm. The predicted mean positions match the mean MC position well. The y position shows a slight shift compared to the MC position. The total deviations between true and predicted means are much smaller than the variances of the true photon distribution which are primarily determined by the cell size. All-in-all, we can conclude that the network is reasonably trained on one-cell events.

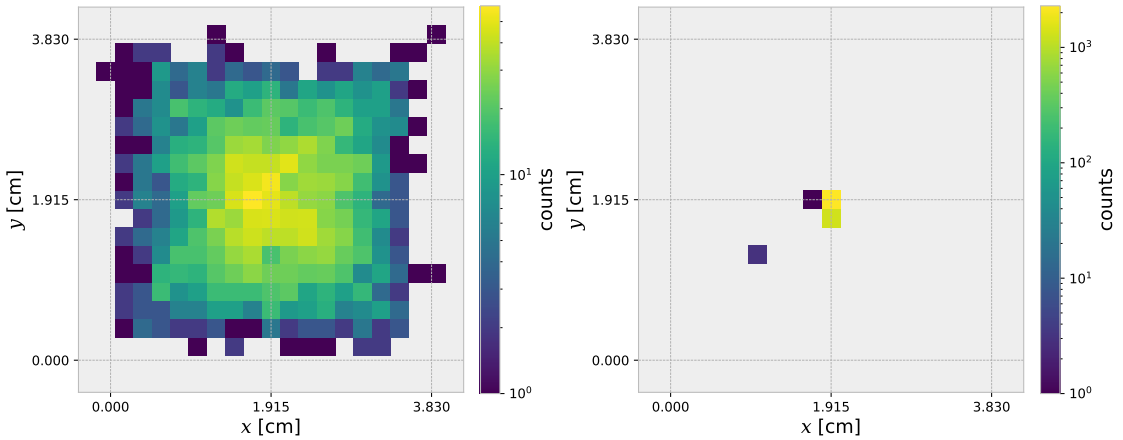


Figure 22: Left: MC truth distribution of photon positions (x, y) of one-cell events. The photons illuminate the cell completely but not uniformly. The cell is more likely hit near the center. Right: Performance of the final network on one-cell events. The network shows a centered peak in the middle of the cell with few outliers and a small shift in y.

4.1.7 Evaluation on real data

The final model in the previous section showed improvement on the photon position reconstruction compared to Lednev. One might hope that the photon simulation reflects reality to a sufficient level so that a network trained on simulated data may be applied on real data. This will now be investigated by attempting to reconstruct the pion and eta masses using their decay into two photons.

$$\pi^0 \to \gamma\gamma, \quad \eta \to \gamma\gamma \tag{8}$$

We take the data of weeks 45, 46, and 47 of the COMPASS Primakoff run 2009 and apply a basic event selection: Only events with two separated photon clusters which were selected with the Lednev prediction were used. The showers must be separated as otherwise the neural network can not be used. We apply more constraints: The minimum energy of one cluster must be at least 2 GeV and is determined via the Lednev prediction. This is important as the network is not trained on photon showers below 2 GeV. Additionally, there must be one reconstructed primary vertex at the target area between -72.3 $< z <$ -72.9 cm. The coordinates of the primary vertex can be interpreted as the point of production of the two photons. A primary vertex is determined by matching a beam trajectory to at least one outgoing tack. 392 930 events fulfill all demands.

To determine the invariant mass of the two photons, we use four-momentum conservation:

$$m^2_{\gamma\gamma} = (p^\mu_{\gamma_1} + p^\mu_{\gamma_2})^2 = 2E_{\gamma_1}E_{\gamma_2}(1 - \hat{p_{\gamma_1}} \cdot \hat{p_{\gamma_2}}) \tag{9}$$

The energies E_{γ_i} are simply the measured energy on the ECAL. The unit vectors $\hat{p_{\gamma_i}}$ can be determined by knowing the coordinates of production and the position (x, y, z) on the ECAL. We choose the coordinates of production to be the ones of the reconstructed primary vertices. Now the invariant photon mass can be calculated using the Lednev prediction of x, y and E on the ECAL or using the prediction of the final model neural network. The result can be seen in figure 23.

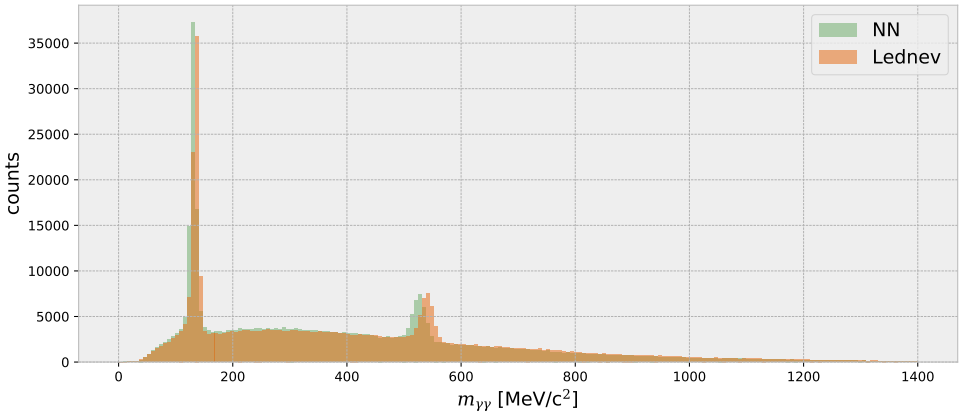


Figure 23: Reconstructed invariant two-photon masses with help of the network (green) or Lednev (orange).

The mass $m_{\gamma\gamma}$ is predicted to be between 0 and 1400 MeV/c^2 and shows two peaks at the pion and eta masses. The background is low compared to the peaks. The two mass peaks can be investigated by fitting background and signal over the peaks.

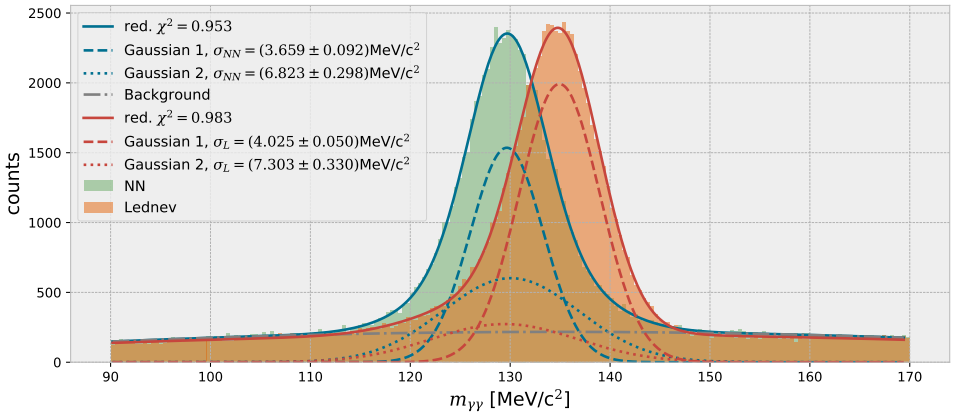


Figure 24: Invariant mass prediction with fitted π peak of neural network and Lednev. The variance of the network $\sigma_{\mathrm{NN}} = 5.365$ MeV/c^2 is by 3.0% smaller than the Lednev variance $\sigma_{\mathrm{L}} = 5.531$ MeV/c^2 while the signal strength of Lednev $A_{\mathrm{L}} = (2.51 \pm 0.08) \cdot 10^4$ c^2/MeV is by 2.8% higher than the networks signal strength $A_{\mathrm{NN}} = (2.44 \pm 0.14) \cdot 10^4$ c^2/MeV.

The signal of the π peak is fitted with two Gaussian functions $\frac{A_i}{\sqrt{2\pi}\cdot\sigma_i} \exp\left(\frac{-(x-\mu_i)^2}{2\sigma_i^2}\right)$ and a second order polynomial $ax^2 + bx + c$ for the background as this gives a good fit with a reduced χ^2 around 1. Lednev is closer to the true pion mass of $m_\pi = 134.9768 \pm 0.0005$ MeV/c^2[20] than the network although Lednev shows a shift when evaluating on the simulated data. This is due to that fact that, unlike the neural network, the Lednev fit is calibrated on real data. In order to compare Lednev and the network the variances and the strength of the signal corresponding to the amplitude of each peak are the important variables. As the mean of the two Gaussians are not identical and the Gaussians are therefore difficult to combine, we subtract the fitted polynomial background of each bin and calculate the mean of the distribution $\mu = \frac{\sum m_i \cdot n_i}{\sum n_i}$ with n_i being the amount of counts in bin i of mass m_i. The variance is then calculated by $\sigma = \sqrt{\frac{\sum (m_i-\mu)^2 \cdot n_i}{(\sum n_i)-1}}$. This gives $\sigma_{\mathrm{NN}} = 5.365$ MeV/c^2 and $\sigma_{\mathrm{L}} = 5.531$ MeV/c^2. So the network shows a small improvement of 3.0%. The signal strength of the two Gaussian fits can be calculated by adding the two amplitudes. One obtains $A_{\mathrm{NN}} = (2.44 \pm 0.14) \cdot 10^4$ c^2/MeV and $A_{\mathrm{L}} = (2.51 \pm 0.08) \cdot 10^4$ c^2/MeV. The signal strength of Lednev is 2.8% stronger. The two methods show similar results for the π mass peak.

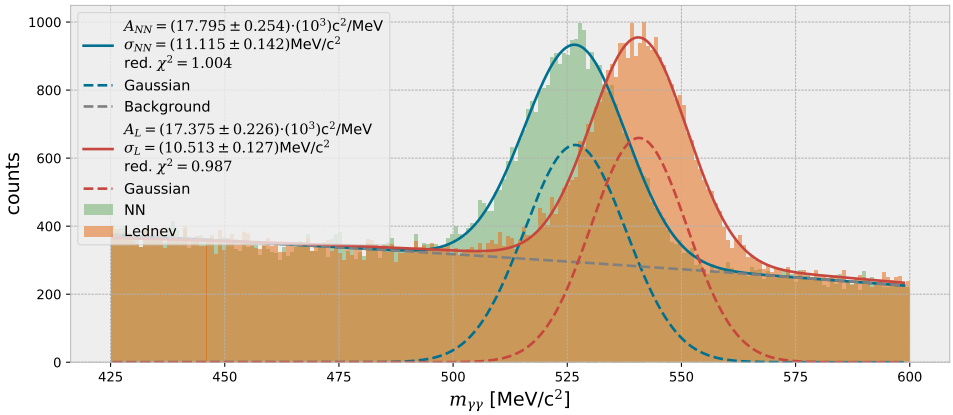


Figure 25: Invariant mass prediction with fitted η peak of neural network and Lednev. The variance is by 5.4% smaller using Lednev while the signal strength is 2.4% higher using the neural network

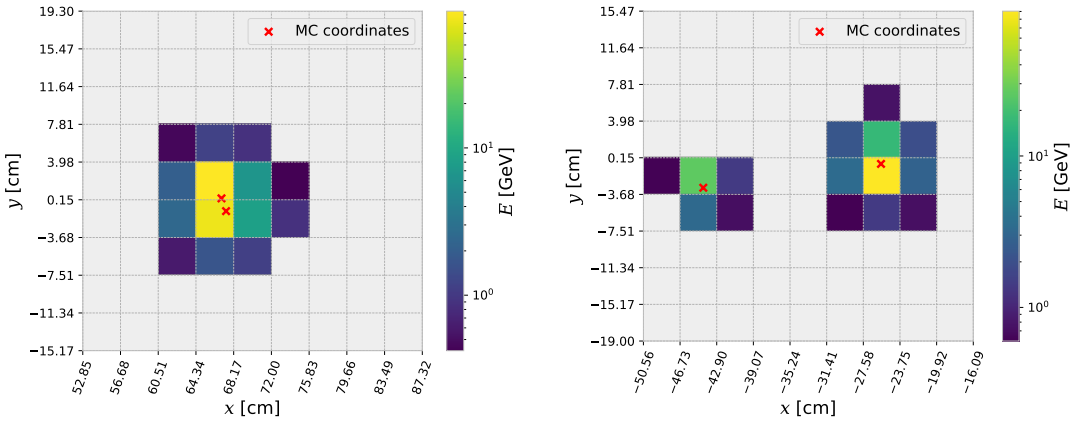
The η peak with one Gaussian as signal and a second order polynomial is shown in figure 25. Here the variance σ and amplitude A can be compared directly. While the variance of the Lednev fit is slightly smaller (5.4%) than the network's, the signal strength of the network is a bit higher (2.4%).

To summarize the study on real data: With a suitable event selection of the real data the neural network can be trained on simulated data and still give reasonable results on real data. We conclude this due to the results of the reconstruction of the invariant mass of two photons and comparison of the signals for pion and eta masses between the network and Lednev. One should highlight that the Lednev fit was calibrated on real data using an electron beam[4] and is hence expected to give good results on real data while the neural network can only be as good as the simulation it was trained on. To improve the performance of the network on real data in the future, one might think about training the network on electron beam data.

4.2 Reconstruction of two photons

As the neural network succeeded to reconstruct the energy and position of one entering photon, we increase the complexity of the problem to two photons hitting the ECAL at the same time. If separated, this can be seen as two single-photon events but if the showers of two photons overlap and form one cluster, the problem gets more complicated. Again the network should reconstruct the position (x, y) and energy E of the photons hitting the ECAL.

4.2.1 The simulated MC data

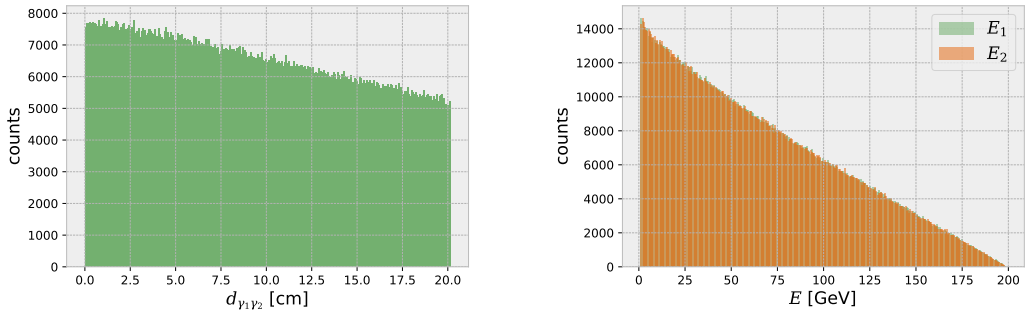


(a) Overlapping showers, one cluster

(b) Non overlapping showers, two clusters

Figure 26: Examples of two photons in a 9x9 grid.

The simulated data used for this problem is equally obtained as for the one photon case. But instead of a 5x5 grid a 9x9 grid is used. The clusters are placed as centered as possible in the grid. This gives a circular distribution of all showers in the grid. Showers that do not fit into the 9x9 grid are not interesting as they have distances large enough to be easily treated as two single-photon events. The dataset contains 1 961 747 events (dataset 4). Two example events are shown in figure 26. The whole shashlik region gets illuminated but the beam-line hole is cut. The distribution of energies and the distance between the two photons can be seen in figure 27.



(a) Distribution of the distance $d_{\gamma_1\gamma_2}$ between photon 1 and photon 2.

(b) Distribution of the energy E for both photons 1 and 2. They equal each other and overlap completely. The sum of both photon energies does not exceed 200 GeV.

Figure 27: Distributions of the two-photon dataset 4

All events with a photon distance smaller than 4 cm are cut. Otherwise, the photons will hit the same cell as one cell has the dimension of 3.83 cm x 3.83 cm. It is very hard to separate two showers of photons hitting the same cell. This modified dataset contains 1 056 051 events (dataset 5).

As for the single-photon case 80% of the data is used for training the network and 20% is saved as the test sample. 10% of the training set is used as validation data.

4.2.2 Permutation-invariant loss function

When approaching the problem, one needs to think about labeling the photons as the solution to the two-photon problem is ambiguous. The output of the network has six parameters x_1, y_1, E_1 and x_2, y_2, E_2. Which photon corresponds to the index 1 and which to index 2? First, we tried to sort the photons after energy. The photon with higher energy corresponds to photon 1 and the other one to photon 2. This training did not lead to good results as the network struggled with the assignment. It is more promising to make the loss function invariant under permutation. It should not matter whether the network is labeling a photon as 1 or 2 as long as it can determine the position and energy correctly. Such a loss function with true values Y and predicted values $\hat{Y}$ can be defined as:

$$\begin{aligned} \text{loss} = \min((x_1 - \hat{x_1})^2 + (y_1 - \hat{y_1})^2 + (E_1 - \hat{E_1})^2 + (x_2 - \hat{x_2})^2 + (y_2 - \hat{y_2})^2 + (E_2 - \hat{E_2})^2, \\ (x_1 - \hat{x_2})^2 + (y_1 - \hat{y_2})^2 + (E_1 - \hat{E_2})^2 + (x_2 - \hat{x_1})^2 + (y_2 - \hat{y_1})^2 + (E_2 - \hat{E_1})^2) \end{aligned} \tag{10}$$

Two units, GeV and cm, get mixed in the loss function. In order to penalize both quantities equally we introduce weights. The position gets divided by the desired resolution of 0.9 mm which was achieved in the single-photon case. The energy gets divided by 2% of the true energy value as a relative resolution of 2% was achieved in the single-photon case.

4.2.3 Architecture of the neural network

As the fully-connected network is sufficient in the single-photon case, this is a reasonable approach for the two photon case too. A FCN network learned the positions and energies of the two photons to some extent. Convolutional networks are commonly used for image recognition[6], and it appears to be the leading solution for the two-photon reconstruction too. At first a zero-padding is added around the 9x9 grid so that the used kernel of the CNN can fully stride over the clusters. We choose a kernel size of 4x4 as all single-photon events fit into 5x5 cells. A 4x4 kernel should therefore be able to learn the shower shape of one photon. After that we apply a max-pooling layer to reduce the dimensionality and extract the most important features. A fully connected network is attached afterwards. The network has 537 686 parameters in total. This is 3.5 times more than for the single-photon case. A graphical illustration of the network can be seen in figure 28.

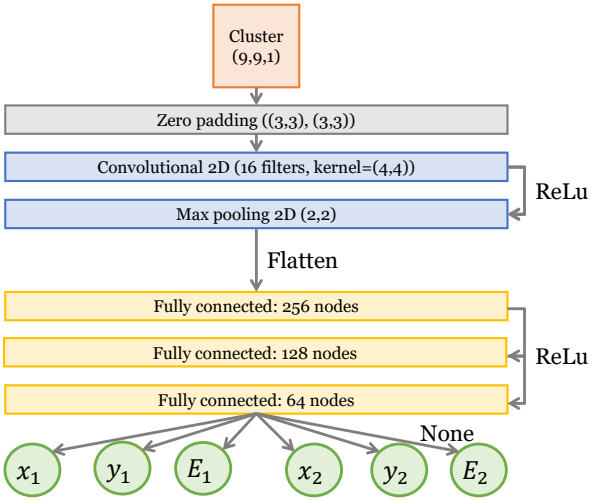


Figure 28: Illustrated architecture of the neural network used for reconstructing two photons.

4.2.4 Results and comparison to Lednev

The network which is illustrated in figure 28 was trained on dataset 5 with the permutation-invariant loss function with weights described in chapter 4.2.2. The Adam optimizer with a learning rate of $\alpha = 0.00001$ and a batch size of 64 was used. The loss over the epochs can be seen in figure 29. There is no clear overfitting after 3000 epochs, although the minimum validation loss appears at epoch 1614. Training 3000 epochs took approximately 14:30 h on one GPU. This network will be referred to as "basic network" or "basic model" from now on.

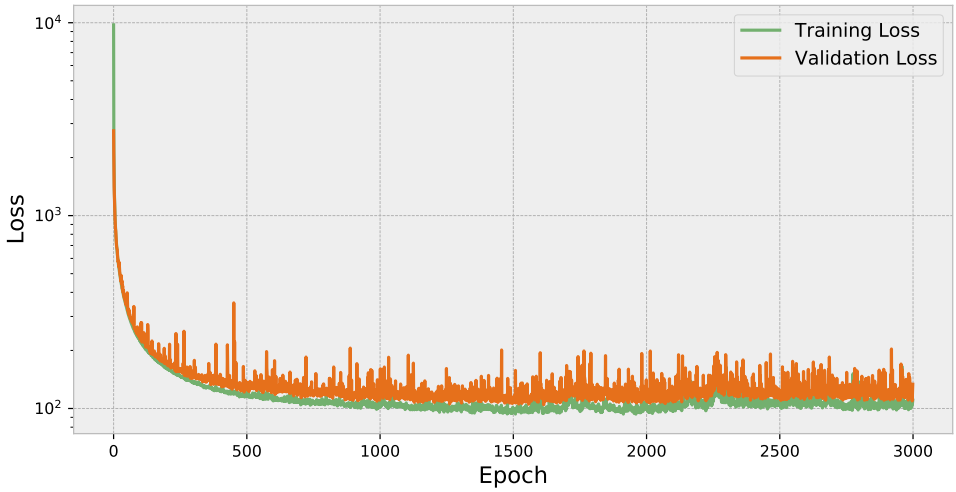


Figure 29: Loss over the epochs of the two-photon basic network. The smallest validation loss is at epoch 1614.

Figure 30 shows the results of photon 1 and photon 2 merged. This is reasonable as the loss function is permutation invariant and labeling the photons as 1 and 2 has no meaning. The histogram showing Lednev has fewer entries as we only consider events of which Lednev was able to identify the correct amount of photons, in this case two. The network performs better than Lednev by a factor of 1.7 for the variances in x and y and therefore improves the spatial resolution compared to the traditional Lednev fit. Comparing this to the final model of the single-photon case shown in figure 21, the performance of the basic network drops by a factor of 1.8 for x and y. We need to be careful with this comparison as the problem also got more difficult by overlapping clusters. The performance of Lednev drops too when comparing single-photon and two-photon events. Evaluating the relative performance between the network and Lednev, the network performs better than Lednev on the position determination in both cases, by a factor of ~ 2.6 for one photon and a factor of 1.7 for two photons. Looking at the determination of energy in the two-photon case in figure 30, Lednev shows a 1.4 times smaller variance than the network. In the single-photon case the performances of the network and Lednev were roughly the same. So the relative performance of the network to Lednev dropped in all quantities too. This is a hint that the basic network shows a worse performance not only due to the more complex task.

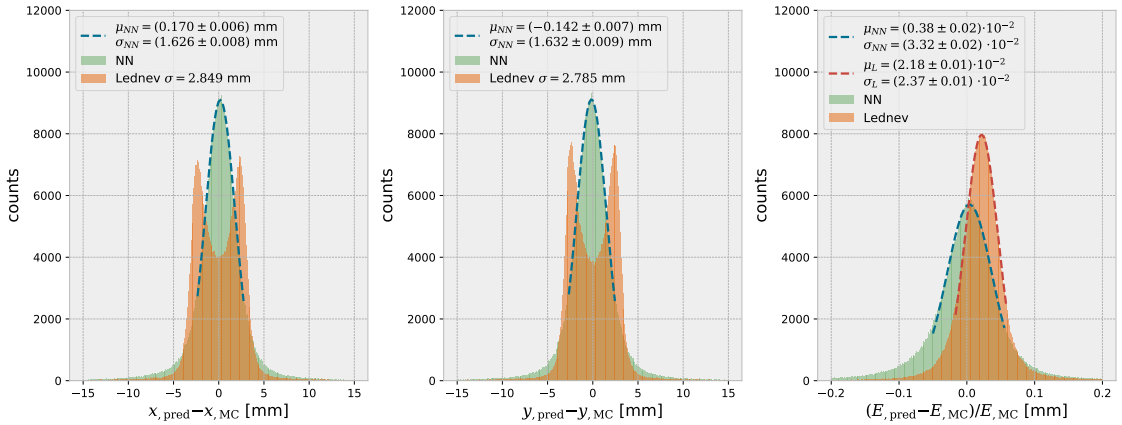


Figure 30: Performance of the basic network (green) and Lednev (orange) on dataset 5 with two photons on the parameters x, y, E from left to right. While the network has a better spatial resolution than Lednev, Lednev can determine the energy better when comparing σ.

4.2.5 Analysis of the network's performance

Before the examination of the worse performance of the two-photon basic network compared to the single-photon final model, it is interesting to investigate what exactly the basic network has learned and what it is struggling with. Some selected 2D-relation plots can be seen in figure 31. Subfigure 1 and 2 give information about confusing photon 1 with photon 2. Other approaches than the permutation-invariant loss function have shown an anti-diagonal (diagonal) correlation between the distance of the photons in x against the deviation of x_1 (x_2). These relations gave a hint that the network confused the x-positions of the two photons. These plots looked identical for y. This effect vanished nearly completely now, only some very few outliers can be seen on the anti-diagonal (diagonal) in subfigure 1 (2) of figure 31. Subfigure 3 shows that there is no correlation between learning the x and y position. That events with low energy have a worse position resolution can be seen in subfigure 4. This is due to the earlier discussed fact that fewer cells are hit and therefore the center of gravity can be determined with less accuracy. Subfigure 5 shows that energies are learned worse if the distance between the clusters is small. This is reasonable as the clusters start to overlap more. It is harder to assign every photon the correct energy. Subfigure 6 illustrates that there is no correlation between the relative energies. This is expected as the weight applied on the deviation of the energy in the loss function is determined by 2% of the true energy value. Therefore, the relative deviations are equally penalized over the whole range of energy. But as a consequence the deviation of energy 1 and energy 2 to the true values is biased as shown in subfigure 7. Even if one energy is learned well the other one can still be far off. Subfigure 8 illustrates that the sum of the energies is mostly learned correctly, but the energy of some events is estimated too low. A possible solution to this problem might be to implement the condition of the sum of the energies into the loss function. Subfigure 9 shows that the relative energies are nicely learned over all ranges of energies as expected due to the loss function. Only lower energies are learned a bit worse as discussed before. The last subfigure 10 shows similar to subfigure 8 that badly learned energies are mostly too low.

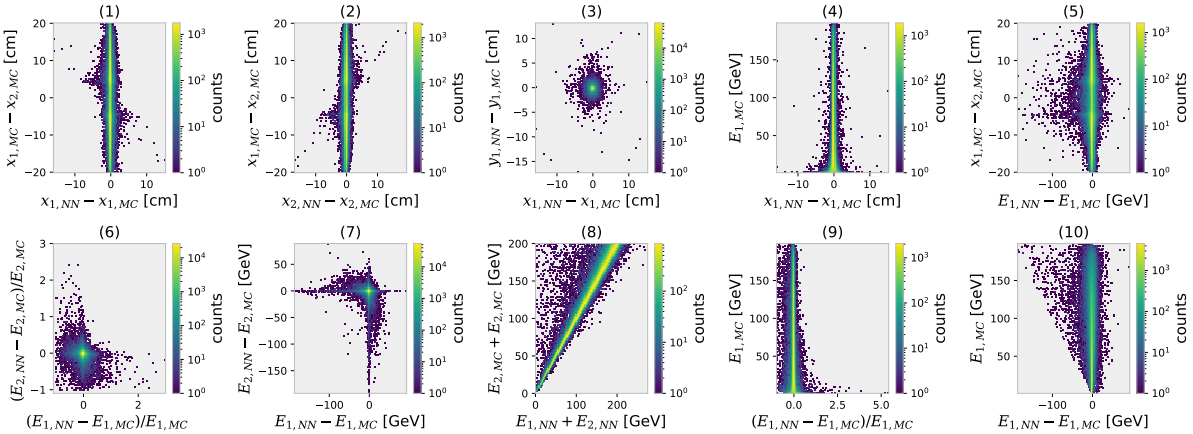


Figure 31: Correlation plots between the quantities x, y and E for the performance of the basic network. The network does not confuse photon 1 with photon 2. The correlation between position arguments and energy are as expected. It struggles to learn the correct energies which are sometimes too low.

In analogy to the one-photon case, we look at events for which the basic network performed badly so the prediction deviates from the truth more than one σ. The histograms for the two-photon network can be seen in figure 32. As the plots for badly learned x values look similar to the ones for y, only the plots for y are shown. The first subplot illustrates that events with badly learned energies have either low or high energies. The low energies were already discussed in the single-photon case. That high energies are badly learned might be due to the data distribution shown in figure 27b. There are fewer events with high energies so that the network has fewer examples to learn on. Besides that, a reason might be that high energy clusters have more overlap and the energies are not correctly assigned to the individual showers. Subplots 2 and 3 are expected to be flat or slightly higher to the edges of the grid. Instead, they are a bit unsteady. The few peaks below $x = 0$ cm and $y = 0$ cm can be explained. We set the coordinate system to be in the center of the lower left cell. As we place the clusters as centered as possible, we neglected that there is still a possibility that showers have negative coordinates. In fact 112 do, which is 1.0% of the whole dataset. The network uses the ReLu activation function which only fires for positive values. So negative coordinates will not be learned. This could be easily fixed by setting the coordinate system to the lower left corner of the lower left cell in the grid. The second row in figure 32 shows badly learned y values. These events are more likely to have low energy or high energy. We have discussed this earlier. One would expect the distribution of x values to be flat. As for the third plot in the first row it increases for lower values. We can not explain this bias. When looking at the y values with badly learned y positions, the cell structure of the grid can be seen as indicated in red. If a photon hits a cell more to the edge it will most likely also hit the neighboring cell. Therefore, the center of gravity of the energy deposit can be determined with more precision. It follows that photons hitting the ECAL in a cell center have worse resolution. This can be seen by the small increase of events at these positions indicated in red. As said before, we can not explain why events closer to the left edge of the grid are learned worse than on the right. The bad performance of events with positions below zero are again due to combination of using the ReLu activation function and the choice of coordinate system.

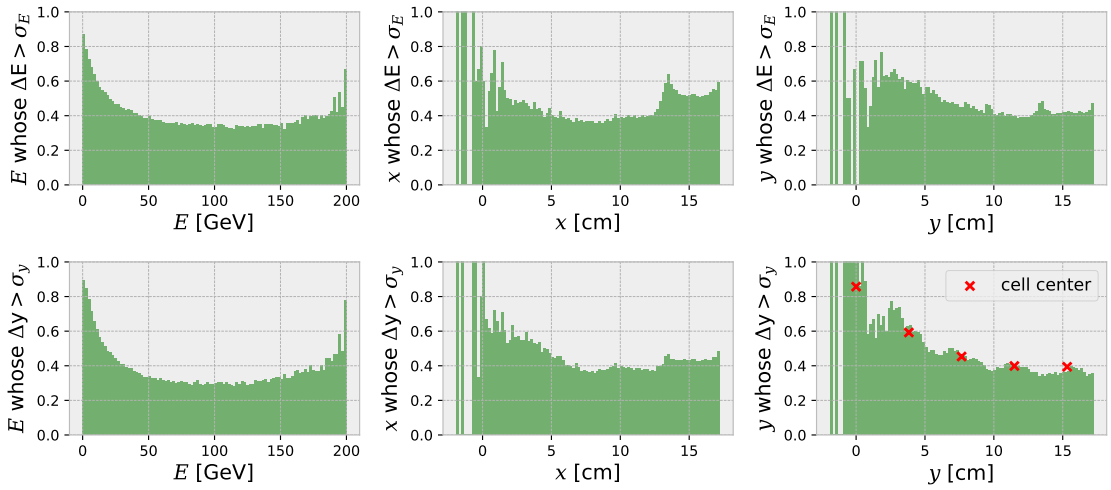
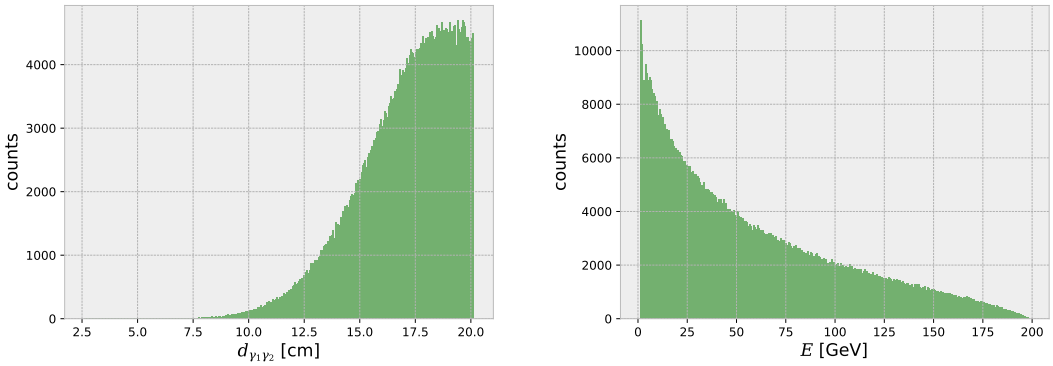


Figure 32: Histogrammed values of quantities x, y, E that lie more than one σ away from the mean μ, see figure 30. The histograms are normalized to the amount of all events per bin. The first (second) row shows events whose energy values E (y) are badly learned. For these badly learned events all quantities x, y, E are histogrammed as a fraction of all events per bin. The symbol Δ indicates the difference to the MC true value so $\Delta y = y_{NN} - y_{MC}$ and the relative difference of the energy $\Delta E = \frac{E_{NN} - E_{MC}}{E_{MC}}$. Events with very low and very high energies are learned badly. Due to the ReLu activation function and the choice of the coordinate system position values below zero are not learned. The cell structure of the grid can be seen due to relatively higher counts in the cell centers indicated in red.

4.2.6 Non-overlapping clusters

The basic network shows a drop of performance compared to the single-photon network. The reason for the worse performance might be the more complex task. If photon showers overlap, the positions and energies are harder to separate. To exclude this possibility, we check whether the performance increases and meets the performance on the single-photon network if only two separated showers are used. We use a dataset of 417 438 two-photon events with no overlap between the photon showers. The distributions of this dataset can be seen in figure 33. The energy distribution shows an increase towards small energies as it is more likely that photons with less hit cells, so low energies, do not overlap. Almost all photons have a distance greater than 10 cm. We will refer to this dataset as dataset 6.



(a) Distribution of the distance $d_{\gamma_1\gamma_2}$ between photon 1 and photon 2. Only 2.8% of the events have a photon-distance smaller than 12 cm due to the no-overlap condition.

(b) Distribution of the energy E of both photons. There are many events with low energies.

Figure 33: Distributions of dataset 6 with non-overlapping clusters.

We train the network for 3000 epochs. The lowest validation loss occurs at epoch 1970. The results of the study can be seen in figure 34. The network improves the spatial resolution by a factor of 2.2 on x and 2.1 on y compared to Lednev. Lednev predicts the energy better by a factor of 1.1 when comparing the variances σ. Comparing the results to the basic network in figure 30, the performance of the network increases by a factor of 1.1 for x and y and by 1.2 for the energy. When comparing the relative differences to Lednev the performance of the network increases from a factor of 1.7 to ~ 2.1 on the positions. The network can still not predict the energies better than Lednev, although the factor between them decreases from 1.4 to 1.1. The overall performance of the two-photon network increased when using separated clusters but does it match the performance of the single-photon network?

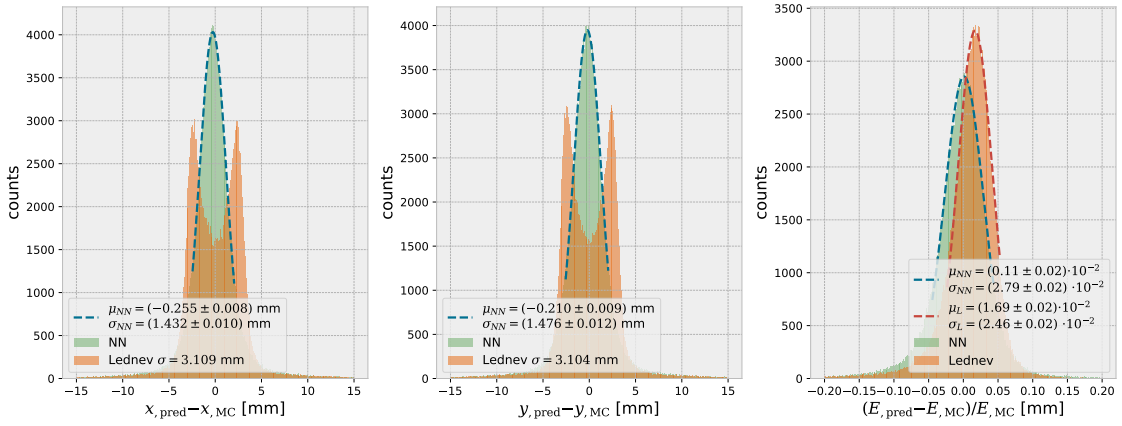


Figure 34: Performance of the network (green) and Lednev (orange) on dataset 6 with two non-overlapping photon showers on the parameters x, y, E from left to right. The network has a better the spatial resolution compared to Lednev but Lednev predicts the energy better when comparing the variances σ.

4.2.7 Comparison with the single-photon network

To get a clear comparison of the two-photon network's performance on non-overlapping cluster to the performance of the final model network of the single-photon case, we take the same network architecture and hyperparameters of the final model network but train and evaluate it on the non-overlapping dataset 6 by separating the two clusters manually. To get an equal comparison we train the network on the same amount of events (417 438) with single photons and for the same amount of epochs. The lowest validation loss appears at epoch 2687. We evaluate it on the same test sample as the two-photon-non-overlapping network in section 4.2.6. The result can be seen in figure 35 together with the performance of the two-photon network.

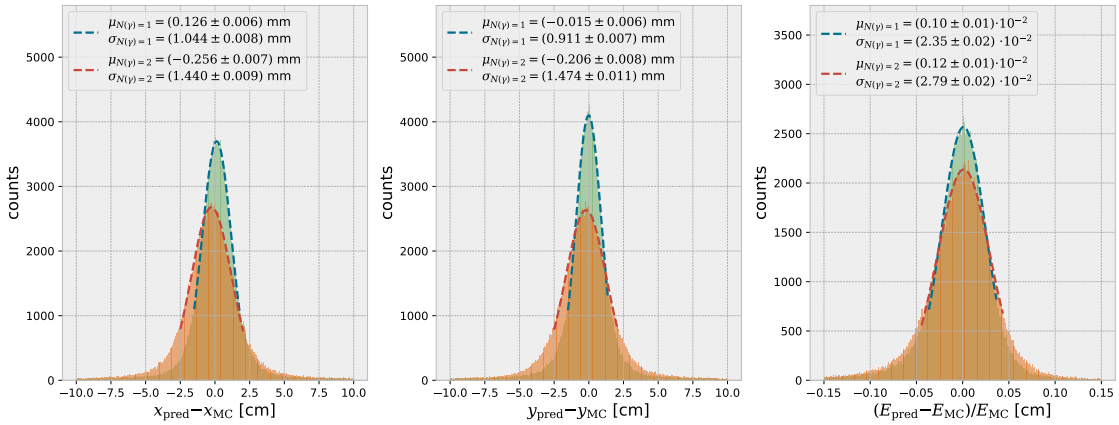
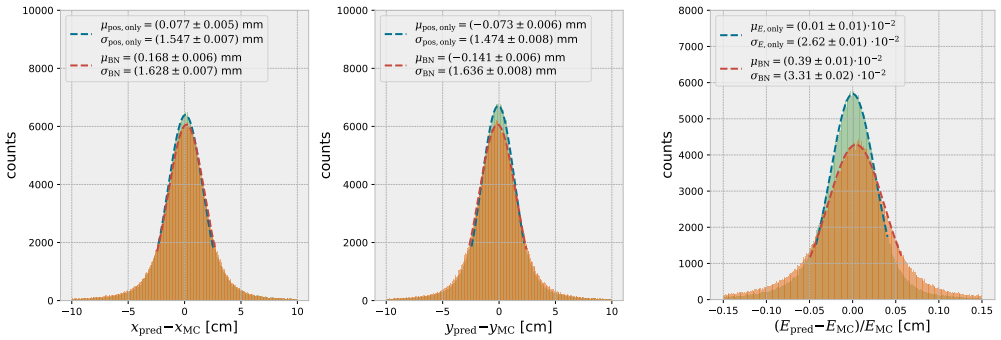


Figure 35: Performance of the single-photon network (green) and two-photon network (orange) on dataset 6 with non-overlapping photon showers on the parameters x, y, E from left to right. The single-photon network outperforms the two-photon network on all quantities.

Comparing the two performances of the single-photon and two-photon network, the position determination is 1.4 times more precise for x and 1.6 times more precision for y if the single-photon network is used. The energy resolution of the single-photon network is better by a factor of 1.2. It gets clear that evaluating two separated showers on the single-photon network gives a better result than evaluating them together as a two-photon event with the two-photon network. We can conclude that the two-photon basic network presented in 4.2.4 does not show worse performance because of the more complex task but shows a lack of precision in general.

4.2.8 Separate determination of position and energy

There is the possibility that the basic network is showing bad performance because the weights in the loss function do not compensate enough for the mix of units (cm and GeV). Therefore, we use the same network as the basic model in section 4.2.4 but only require to a) learn the positions (x_i, y_i) and b) learn the energies E_i. We train for 3000 epochs and the lowest validation loss appears at epoch 2543 for a) and epoch 766 for b). The results can be seen in figure 36a for a) and 36b for b).



(a) Performance of the network that only learns the positions x and y (green)

(b) Performance of the network that only learns the energy E (green)

Figure 36: Performance of the network with separately learned and evaluated quantities (green) on dataset 5 on the parameters x, y, E from left to right. The performance of the basic network (BN) is shown in orange for comparison.

The networks with separately learned position and energy show a better performance as the variances sank from $\sigma_x = 1.628$ to 1.547 mm, $\sigma_y = 1.636$ to 1.474 mm and $\sigma_E = 3.31$ to 2.62. Therefore, the position determination improved by 5.0 % on x, 9.9 % on y and 20.8 % on E. The stronger improvement on the energy shows that learning the energy gets suppressed by learning the positions in the merged network and the deviation of the energy has less impact on the loss. This is probably the result of a not optimal chosen weight. The weights could either be adapted or data preprocessing could be used. By applying the standard score to the data, the imbalance of the units might be reduced. Applying the logarithm to the energies might also help to deal with the large range of energies of three orders of magnitude (2-200 GeV). All quantities show an improvement when being learned separately. This is a hint that the complexity of the basic network is too small to learn all quantities at once. We have seen in the single-photon case that mixing units in a loss function even without weights leads to reasonable results. If one quantity gets suppressed, the network must be in a situation that requires to decide which quantity to learn as the capacity is not sufficient to learn all. This "decision" is taken via the impact of the quantities in the loss.

Even though the performance increases when separating the determination of the position and energy, the network still shows a worse performance compared to the single-photon final network shown in figure 21. The performance dropped by a factor of 1.7 for x and y and 1.3 for E when comparing variances. We can conclude that the mixing of the units in the loss function can not be the predominate reason for the worse performance of the basic network.

4.2.9 Mono-energetic dataset

The generally worse performance of the two-photon basic network raises the question if the network is even able to determine quantities of two photons at the same time. To check this, we simplify the task drastically by using a mono-energetic dataset of $E = 80$ GeV and investigate how well the network is able to learn the positions. This task might be easier for the network to learn as all showers should have a very similar shape due to the same energy. We will use two modifications: One dataset (290 968 events) will still have the condition of at least 4 cm between the photons, the other one will not have any constraints on the distance between the photons (513 439 events). The network stays the same as illustrated in figure 28, only the last output layer has four instead of six nodes as we do not require to learn the energy due to the constant value.

We train the 4-cm-network for 3000 epochs with the same hyperparameters as on the other dataset described in chapter 4.2.4. The lowest validation loss is at epoch 2920 which indicates that the network is not fully trained. If we spent time fine-tuning the network, we would also decrease the learning rate. The results of the 4-cm-network can be seen in figure 37. The results are similar

to the performance of the single-photon final model network in figure 21 as the standard deviations are nearly the same. Comparing the network's performance on the mono-energetic dataset to the performance of the basic network discussed in section 4.2.4, the spatial resolution increased by a factor of 1.8 for x and 1.9 for y. Why did the network perform so much better on this dataset than on dataset 5? The performance of Lednev improved too. This is an indicator that the whole problem is easier. When using an energy of $E = 80$ GeV, on average 18.2 cells are hit per event. This means 9.1 cells per shower. These are huge showers and the center of gravity can be determined with large accuracy. Additionally, the network only has to learn a single shower shape of clusters with $E = 80$ GeV. Once it learned the shower shape the position of the photon is easy to determine. The network shows a great performance with a standard deviation 2.7 times smaller than Lednev.

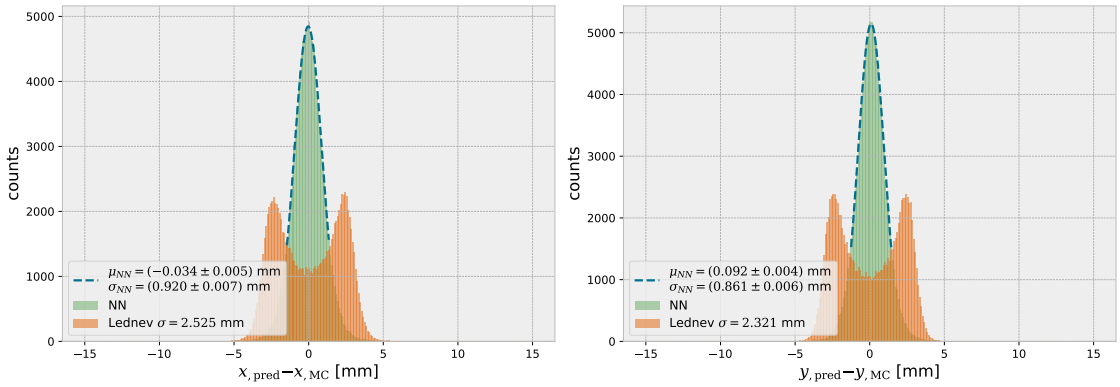


Figure 37: Performance on the $E = 80$ GeV dataset with a minimum distance of 4 cm on the parameters x, y from left to right. The network outperforms Lednev.

We drop the 4 cm condition to test the limits of the network. The smallest distance between the photons is 5 mm. The distribution of $d_{\gamma_1\gamma_2}$ increases for small distances. The lowest validation loss appears at epoch 795. The results of the study can be seen in tabular 1. The performance of the network compared to the 4 cm dataset drops by 7.9% on x and by 10.0% on y. Still, compared to Lednev, the spatial resolution is better by a factor of 2.9 in x and a factor of 3.0 in y. One needs to keep in mind that the histogram of Lednev holds fewer entries as we always only consider events with correctly identified number of photons. In this dataset Lednev miss-identified 22.7% of the events. The network has the advantage of having no choice than to predict two clusters.

	σ_x [mm]	σ_y [mm]
NN	0.983 ± 0.007	0.950 ± 0.006
Lednev	2.868	2.816

Table 1: Performance on the $E = 80$ GeV two-photon dataset with at least 5 mm between photon positions.

This study has shown that the network can easily detect two photons in one picture and determine its position with high precision compared to Lednev. We need to highlight that detecting mono-energetic photons is a very simple task. Still, this shows that a network is generally able to learn quantities of two-photon events, even if the photon clusters overlap, with a similar precision as the single-photon network.

We have not yet answered the question of why the basic network shows no optimal and even bad performance in the general two-photon case explained in section 4.2.4. The studies on non-overlapping clusters in section 4.2.6, the separate prediction of positions and energies in section

4.2.8 and the mono-energetic study in this section have three things in common: All these tasks are easier, the architecture of the network is the same and the network's performance improved compared to the basic model. This is a hint that the basic network might not be complex enough to be able to learn the full extent of the two-photon detection problem, and we did not find the optimal network architecture matching our problem's complexity.

4.3 Counting photons

The amount of photons hitting the ECAL at the same time is a desired variable as the networks we have presented until now only work for a fixed number of photons. Therefore, it is interesting to see if a neural network can determine the number of photons on the ECAL and whether it can do this better than Lednev.

4.3.1 The simulated MC data

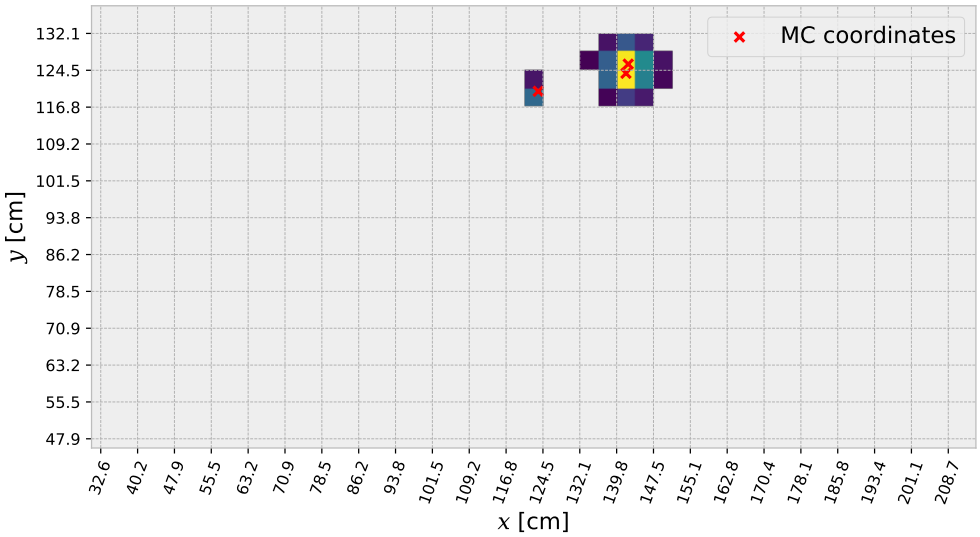
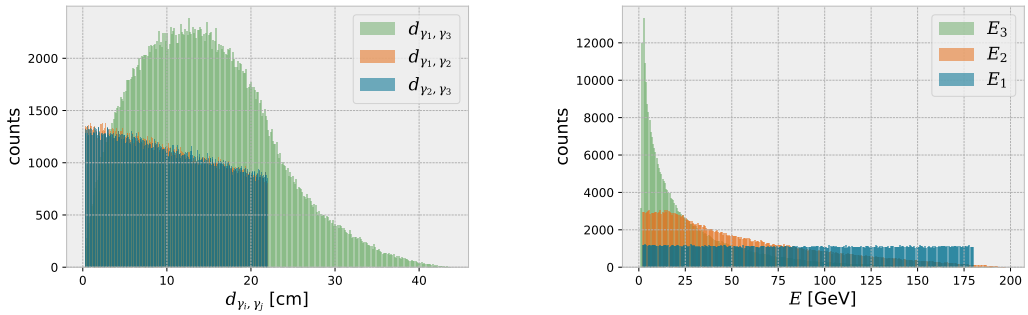


Figure 38: Example of the network input. Three showers that build two clusters on the shashlik part of the ECAL.

As this is a feasibility study we will restrict the number of photons to be one, two or three. We use dataset 3 for one photon and dataset 4 with no minimum photon distance for two photons. Therefore, the smallest distance between photons is 2.5 mm. The distribution of the dataset used for three photons which we will call dataset 7 is shown in figure 39. It contains 329 523 events. The way photon 2 with respect to photon 1 is created is the same as for photon 3 with respect to photon 2 so the two distance distributions match. No constraint on a minimum distance is applied and the smallest distance between them is 2.5 mm. The sum of all three photon energies is a maximum of 202 GeV. Therefore, the third photon normally has a very small energy.



(a) Distribution of the distance $d_{\gamma_i\gamma_j}$ between the three photons. The distribution of $d_{\gamma_1\gamma_2}$ equals $d_{\gamma_2\gamma_3}$.

(b) Distribution of the energy E for photons 1, 2 and 3.

Figure 39: Distributions of the three-photon dataset 7

To make sure every case gets the same attention in the training, we use an equal amount of events (329 000) for each category 1, 2 and 3 photons. The whole mixed dataset contains 987 000 events. Instead of cutting out the hit area, we give the whole shashlik part of the ECAL as an input. It has the size 25x49 cells. An example of an input can be seen in figure 38.

As previously we use 80% of the data for training the network and 20% is saved as the test sample. 10% of the training set is used as validation data.

4.3.2 Architecture of the neural network

As counting how many photons appear on the ECAL is basically image recognition, we apply a convolutional network. The main difference to the determination of the position and energy of a photon is that this is a classification problem, not a regression problem. This means that the output has 3 bins of which each represents one class, in this case the classes "one photon", "two photons" and "three photons". The value of each bin represents the probability of an event being classified as this bin. So the values of all bins sum up to 1. This is implemented via the softmax activation function. Dropout layers are used to prevent early overfitting. The network used for this problem is illustrated in figure 40.

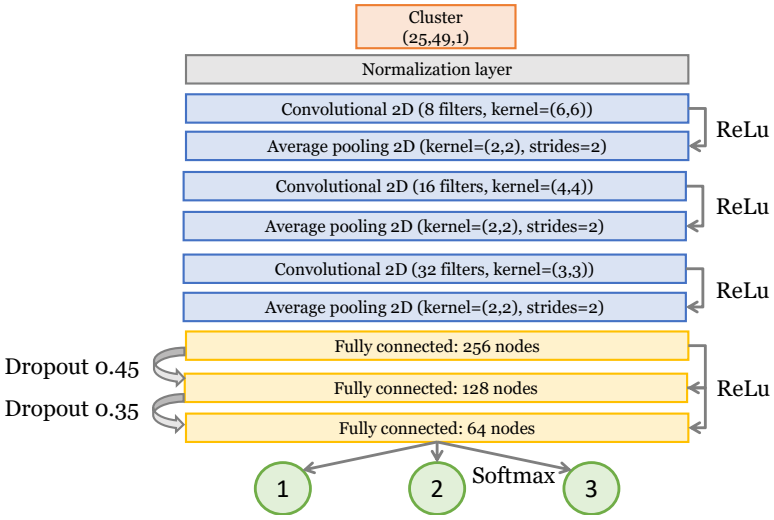


Figure 40: Graphical illustration of used convolutional neural network for counting photons on the ECAL. It has 196 059 parameters.

We used the common loss function of classification problems, the categorical cross-entropy loss and the Adam optimizer with a learning rate of $\alpha = 0.003$ and a batch size of 64 for the training.

4.3.3 Results and comparison with Lednev

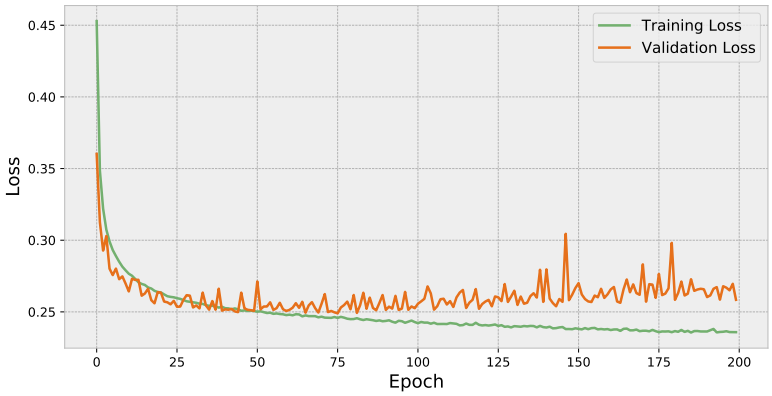
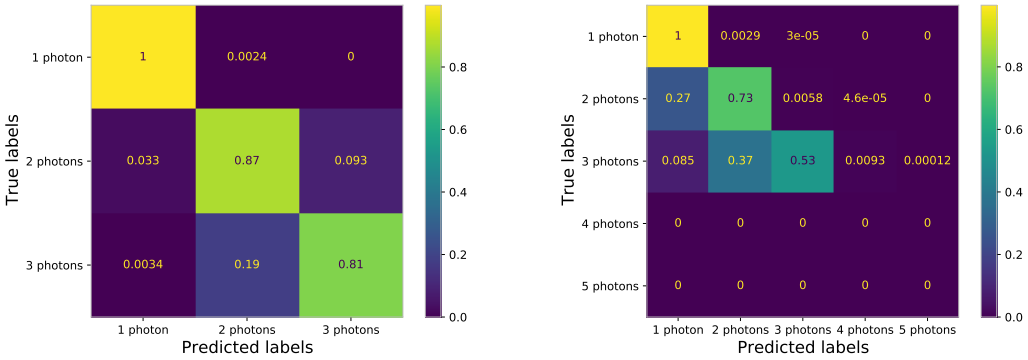


Figure 41: Loss over the epochs. The lowest validation loss is at epoch 73.

The loss over the epochs can be seen in figure 41. If one wanted to fine tune the network, increasing the drop out would probably delay the overfitting a bit more. The lowest validation loss already occurs at epoch 73. To evaluate the performance of the network, the confusion matrix is a useful tool. It can be seen in figure 42a. The network has no difficulties to detect one photon as it determines all one photon cases correctly and also does not falsely predict one photon as a two or three-photon event. It has some struggles to differentiate between two and three photons. 9% of the two photon events were predicted to be three photons and 19% of the three photon events as two photons. Three overlapping cluster with nearly no distance between them look similar to two photons being very close. This gets clear when looking at the energy distribution of dataset 7 in figure 39b. It is very likely that one of the three photons has a very low energy value and will get swallowed by the bigger ones if the showers overlap and build one cluster.

Figure 42b shows the performance of Lednev on the same data. It can determine all one photon cases correctly. Lednev already struggles with the separation of one and two photons as it predicts 27% of the two-photon cases as one photon. It has even more difficulties to distinguish two and three photons as is predicts 37% of the three-photon events to be two photons. It even predicts one photon in 8.5% of the cases. Lednev is tuned so that it will rarely overestimate the amount of photons but decide for the lower amount of photons if unclear to avoid artificial splitting. The network outperforms Lednev in counting up to three photons on dataset 7.



(a) Confusion matrix of the neural network

(b) Confusion matrix of Lednev

Figure 42: Performance on counting up to three photons on the ECAL

4.3.4 Analysis of the network's performance

The success of separating showers is strongly connected to their distance. Therefore, we investigate the performances of the described network and Lednev on data with different minimum distances. For this study we define the minimum distance of three photons by the smallest distance between all.

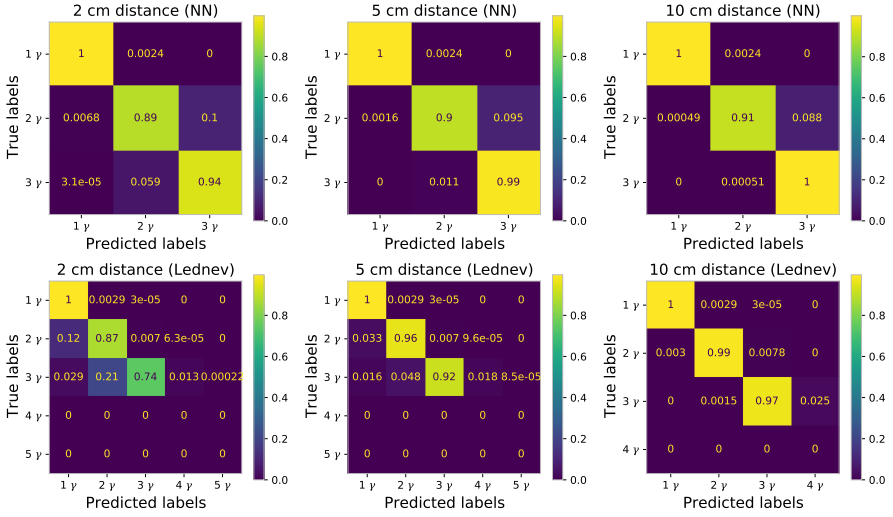


Figure 43: Confusion matrices of the network (first row) and Lednev (second row) for different minimum distances between the photons: 2 cm, 5 cm, 10 cm from left to right.

The confusion matrices for different distances can be seen in figure 43. At a minimum distance of 2 cm the network can nearly identify all cases by 90%. Lednev succeeds above 90% after 5 cm. When looking at 10 cm, the prediction of Lednev is a lot more accurate on two photons. It can detect 99 % correctly while the network still only predicts 91% correctly. The network still classifies 9% of the two-photon events as three photons. Where does this bias come from? When looking at the data distribution of the energies in figure 39b, it gets clear that one of the three photons has a very low energy, e.g. 35% of them have energies below 10 GeV. Approximately 2 cells per event get hit if photons have energies below 10 GeV. This is not much compared to a 80 GeV nine-cell cluster event. The network learned that even if it detects two photons, there

might be a third photon hidden within a bigger cluster. Even if we increase the distance between the photons even more, the network will still falsely predict three photons in two-photon cases. This can only be fixed by training on a different dataset.

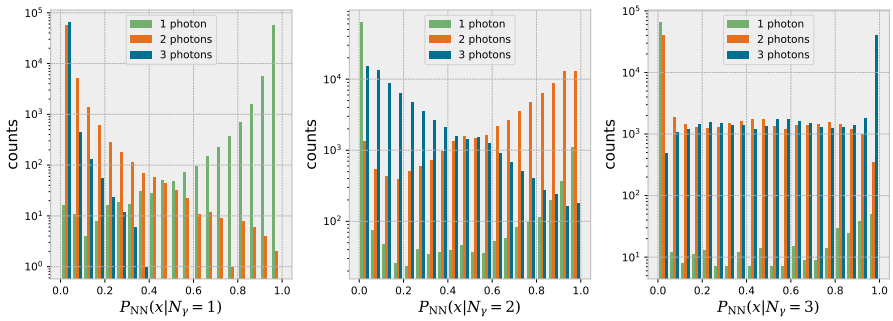


Figure 44: Probability for one (green), two (orange) and three (blue) photons given by the neural network for one, two and three photons ($N_\gamma = 1, 2, 3$) from left to right

We can further evaluate how certain the network is with its decision. Therefore, the histogrammed values for the returned probability of one (green), two (orange) and three (blue) photons for one two and three photons ($N_\gamma = 1, 2, 3$) are shown in figure 44. The first plot shows the returned probabilities for one photon events ($N_\gamma = 1$). We can see that the network is mostly very certain about detecting one photon. It is also very certain not to see three photons. In very few cases it detects two instead of one photon. This might be the case if photons have very small distances between each other so that the showers overlap strongly and can not be distinguished form a single-photon cluster. The second plot shows two photon events $N_\gamma = 2$. We see that the network normally does not predict one photon but is sometimes very sure (100%) to see one photon. As one can see a linear decrease of three-photon counts over the probability, we can say that the probability for three photons drops exponentially over the probability. The third plot shows events with three photons $N_\gamma = 3$. One can see that two and three photons are nearly predicted the same amount of times between $x = 0.1$ and $x = 0.9$. But the network is very often close to 100 % certain to detect three photons. This is probably the case if they are disjoint. Most of the time the network is very certain not to see one photon. If it detects one photon instead of three then it does so with large confidence. This probably happens if all three photons hit the ECAL very close to each other.

To summarize the chapter about counting photons on the ECAL: The network is able to give reasonable predictions for one, two and three photons hitting the ECAL and even performs better than Ledenv on this task. The presented network works well for small distances but as trained on low-energetic photons in the three-photon case, it may confuse two and three photons. For large distances Lednev is therefore better to identify the right amount of photons. Changing the data used for training and fine-tuning the network will very likely improve the performance. The problem may be extended for more than three photons hitting the ECAL.

5 Discussion and outlook

This proof of principle study has shown that neural networks can reconstruct simulated photons in an electromagnetic calorimeter. The method has potential for determining the correct number of photons at smaller distances than the traditional Lednev fit. While the network can determine up to three photons in at least 81% of the cases correctly, Lednev detects at least 53% of them correctly if there is no minimum distance between the photons. Although the network seems to show better results, its performance is low for photons separated by at least 10 cm, as only 91% or more of the events are determined correctly. This bias might be the result of the training-dataset

distributions. It is highly likely that one photon of a three-photon event has low energy and can not be distinguished from two photons if the low-energy photon is covered by a high-energy shower. To achieve better results than the traditional Lednev fit in all cases, the training dataset needs to be chosen differently.

The network can predict the position of simulated single photons more precisely. The spatial resolution of the ECAL could be reduced by a factor of 2.7 using the neural network. It shows similar performance on the determination of photon energies compared to Lednev. The most important adjustments explained in this thesis for the good results on the position determination (x, y) are: Setting the z-position to the average penetration depth of a shower in the ECAL to avoid angle effects; placing the clusters centered into the cut-out grid to have a uniform distribution of photon positions in the network's training data and choosing a large-enough network architecture to reflect the complexity of the single-photon detection problem.

We tested if training the network on simulated data and evaluating it on real data will give reasonable results. This is done via evaluation of the invariant mass of two photons. The distribution shows two peaks: One for the pion mass and one for the eta mass. To compare Lednev to the network, the variances and signal strengths of the peaks were investigated. Both methods show the same performance within $\pm 5\%$. While the Lednev fit has the advantage of being calibrated on real data, the network can only reflect the precision of the simulation. To improve the network's performance, one might think of using electron beam data to train the network on. Electrons produce similar shower shapes as photons but can be tracked due to their charge with other detectors too. With the help of tracking detectors the electron position can be extrapolated to the ECAL, and we know the electron energy as they come from an electron beam that we can control. This approach has some difficulties. Electrons might scatter or interact via bremsstrahlung before reaching the ECAL. A careful event selection is mandatory for this approach to work. Still, the possibility remains that the data is nevertheless less accurate than the simulated data. One could also try making the simulation more precise. We still used the GEANT3 based COMGEANT[4] simulation framework as it is fine-tuned for the 2009 COMPASS setup. There is already a GEANT4 based simulation framework called TGEANT[18] which is currently being worked on to simulate shower shapes even more precisely.

Detecting two photons brought up challenges and the presented network could not outperform the Lednev fit in all quantities. The positions are determined better, as the spatial resolution, which is the average deviation of the reconstructed position to the nominal position, is smaller by a factor of 1.7, however the prediction of the photon-energy resolution is worse by a factor of 1.4. When analyzing the performance, we see that especially the sum of the energies is not learned correctly. This effect could be reduced by implementing the sum of the photons into the loss function. The problem also got more difficult compared to the single-photon detection as two-photon showers can overlap and only build one cluster. By training and evaluating a network on a dataset with disjoint clusters only, we were able to show that evaluating two disjoint clusters with the single-photon network provided better results than evaluating it with the two-photon network. Therefore, we can conclude that the two-photon network's architecture is not chosen sensibly enough to match the problem's complexity. We also saw this by separating the two tasks of position and energy determination. Learning the quantities separately showed an increase of performance especially on the energy determination. This is a hint that the chosen network is not complex enough to learn all quantities simultaneously. If the complexity of the two-photon detection problem is reduced by using mono-energetic photons only, the network shows comparable results to the single-photon detection. This shows that a network can generally detect two (overlapping) photons, but we probably need to change to a more complex approach to reflect the complexity of the general two-photon detection problem. Another question one might ask is how to combine the presented networks to apply them generally on photon-detection tasks with a varying number of photons. This question and the poor performance of the two-photon network lead to another approach: Neural networks with Slot Attention modules.

Even though convolutional neural networks are a strong tool in image-recognition problems, there are more efficient network architectures for object-centric learning. The two-photon problem

might be solved with more accuracy when considering Slot Attention modules as a possible network architecture. As the output vectors of Slot Attention modules hold permutation symmetry, one of the major prerequisites for the photon-detection problem as discussed in section 4.2 is met. Furthermore, Slot Attention holds the potential to detect a varying number of objects. This is an interesting feature for the photon-detection problem. The overall approach for photon detection presented in this thesis first needs a network which predicts the number of photons in an image to then apply the suitable single-photon network or two-photon network etc. to the image. Slot Attention does not need a fixed number of photons. It also separates the showers into the individual slots. The output of the slots are basically single-photon events whose properties can be easily extracted as discussed in section 4.1. Slot Attention might therefore not only provide more accurate results for two overlapping photons due to a problem-customized architecture but also seems to be an elegant overall solution for photon detection.

6 Bibliography

References

[1] About keras. https://keras.io/about/. Accessed: 2023-07-04.

[2] Tensorflow. https://www.tensorflow.org/. Accessed: 2023-07-04.

[3] Abbon, P., et al. The compass experiment at cern. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment 577*, 3 (2007), 455–518.

[4] Abbon, P., et al. The COMPASS setup for physics with hadron beams. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment 779* (apr 2015), 69–115.

[5] Adams, B., et al. COMPASS++/AMBER: Proposal for Measurements at the M2 beam line of the CERN SPS Phase-1: 2022-2024. Tech. rep., CERN, Geneva, 2019. The collaboration has not yet constituted itself, thus instead of a Spokesperson currently the nominated Contact Person is acting in place.

[6] Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., and Farhan, L. Review of deep learning: concepts, cnn architectures, challenges, applications, future directions. *Journal of Big Data 8* (2021), 53.

[7] Baum, G., et al. COMPASS: A Proposal for a Common Muon and Proton Apparatus for Structure and Spectroscopy.

[8] Bello, F. A. D., Dreyer, E., Ganguly, S., Gross, E., Heinrich, L., Ivina, A., Kado, M., Kakati, N., Santi, L., Shlomi, J., and Tusoni, M. Reconstructing particles in jets using set transformer and hypergraph prediction networks, 2022.

[9] Collaboration, T. C. COMPASS-II Proposal. Tech. rep., CERN, Geneva, 2010.

[10] Ecker, D. Personal Communication, 2023.

[11] Galeone, P. *Hands-On Neural Networks with TensorFlow 2.0*. Packt Publishing, Birmingham, 2019.

[12] Kuhn, H. W. The Hungarian Method for the Assignment Problem. *Naval Research Logistics Quarterly 2*, 1–2 (March 1955), 83–97.

[13] Lednev, A. Electron shower transverse profile measurement. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment 366*, 2 (1995), 292–297.

[14] Lednev, A. Shower separation program for ecal2, 2008.

[15] Locatello, F., Weissenborn, D., Unterthiner, T., Mahendran, A., Heigold, G., Uszkoreit, J., Dosovitskiy, A., and Kipf, T. Object-centric learning with slot attention, 2020.

[16] Povh, B., Rith, K., Scholz, C., and Zetsche, F. *Teilchen und Kerne. Eine Einführung in die physikalischen Konzepte*, ninth ed. Springer-Verlag GmbH, 2014.

[17] Sarker, I. H. Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions. *SN Computer Science 2* (2021), 420.

[18] Szameitat, T. C. *New Geant4-based Monte Carlo Software for the COMPASS-II Experiment at CERN*. PhD thesis, Albert-Ludwigs-Universität Freiburg im Breisgau, 2016.

[19] Uhl, S. *Photon Reconstruction and Partial-Wave Analysis of Three-Body Final States with Neutral Particles at COMPASS*. PhD thesis, Technische Universität München, 2016.

[20] Workman, R. L., et al. Review of Particle Physics. *PTEP 2022* (2022), 083C01.