

Image Processing for X-ray and Electron Detection Based on Neural Networks for Pixelated Semiconductor Detectors

DISSERTATION

Björn Eckert

Image Processing for X-ray and Electron Detection Based on Neural Networks for Pixelated Semiconductor Detectors

DISSERTATION
zur Erlangung des Grades eines Doktors
der Naturwissenschaften

vorgelegt von
M.Sc. Björn Eckert

eingereicht bei der Naturwissenschaftlich-Technischen Fakultät
der Universität Siegen
Siegen 2022

Betreuer und erster Gutachter
Prof. Dr. Lothar Strüder
Universität Siegen

Zweiter Gutachter
Prof. Dr. Ullrich Pietsch
Universität Siegen

Tag der mündlichen Prüfung
02.12.2022

To my wife Laura Elisabeth

Research: The distance between an idea and its realization.

David Sarnoff

Abstract

The analysis of large data sets in physics experiments profits from recent advances in machine learning techniques based on neural networks. The strength of this approach relies on the existence of validated calibration data called ground truth and reliably tested numerical simulations. Especially in recent years, the widespread use of neural networks gained momentum because of the continuous supply of software and a substantial increase in computer processing power. This continuous increase also impacts applications based on neural networks in analyzing data acquired by pixelated semiconductor detectors.

The user at a beamline experiment expects the event parameters such as the position of the radiation in terms of the point of entry on subpixel level and the amplitude, which describes the energy deposition, the number of photons, or the number of electrons. In electron microscopy, energetic electrons produce three-dimensional tracks in the detector volume and do not deposit their energy locally at their point of entry. The energy deposition happens along these tracks, typically extending over several pixels. However, the user is not interested in the tracks of the primary electrons but in the precise point of entry. In crystallography experiments, the online indexing of the Laue diffraction patterns enables new opportunities. That means all sensor and electronics artifacts need to be analyzed and corrected in real-time.

Four different methods based on neural networks are developed for different event rates on the detectors to precisely determine the point of entry and the intensity of the radiation on the detector. The developed methods enable the reconstruction of positional information and intensity images in real-time at high frame rates. For further physics analyses, no additional detector corrections need to be performed. For X-rays, subpixel resolution has been achieved of less than 10% of the pixel dimensions. For 300 keV electrons in transmission electron microscopy (TEM), the point of entry in the detector was determined precisely (40 μm), although they produce a track of more than 450 μm in the silicon.

Crucial for the development and testing of the neural networks are large data sets containing, on the one hand, the unanalyzed raw data of the detector system and, on the other hand, the associated exact entry points of the primary radiation, e.g., by simulations. The requirement for the unanalyzed dataset is a physically accurate description of the signal formation in the individual pixels.

For this purpose, the primary particles' behavior in the detector volume and

the signal response of the individual pixels are physically described and modeled. The results are implemented in a Monte Carlo simulation. Numerous measurements have verified the Monte Carlo data.

Finally, the newly developed neural networks are compared with the previously used methods in terms of parameters such as spatial precision and analysis speed based on simulated data and performed measurements. We have executed a series of measurements with a transmission electron microscope at different electron intensities and energies to test and verify the developed methods. The results based on the use of neural networks are in good agreement with the precisely known points of entry from the TEM and the simulated data. An X-ray data set with 1.3 keV X-rays yielded a position resolution of better than $3\ \mu\text{m}$ - again, in good agreement with experimental data from an X-ray microscope. The performance of the developed algorithms paves the way to real-time data analysis and data reduction.

In this sense, this work provides a basis and fundamental understanding for future advanced data analysis applications for pixelated semiconductor detector systems.

Zusammenfassung

Die Analyse großer Datensätze in physikalischen Experimenten profitiert von den neuesten Fortschritten bei maschinellen Lernverfahren, die auf neuronalen Netzen basieren. Die Stärke dieses Ansatzes beruht auf dem Vorhandensein von validierten Kalibrierungsdaten, der sogenannten Ground Truth, und zuverlässig getesteten numerischen Simulationen. In den letzten Jahren hat der breite Einsatz von neuronalen Netzen aufgrund der kontinuierlichen Bereitstellung von Software und einer erheblichen Steigerung der Computerleistung an Dynamik gewonnen. Diese Zunahme lässt sich auch im Bereich der pixelierten Halbleiterdetektoren wiederfinden.

Die Nutzer an Strahllinienexperimenten erwarten Eventparameter wie Position der auftreffenden Strahlung auf dem Detektor und die Amplitude, welche die Energiedeposition, die Photonenzahl oder die Elektronenzahl beschreibt. In der Elektronenmikroskopie erzeugen energetische Elektronen dreidimensionale Spuren im Detektorvolumen und geben ihre Gesamtenergie nicht punktförmig am Eintrittsort ab. Stattdessen erfolgt die Energiedeposition entlang der ausgebildeten Spuren, welche sich im Allgemeinen über einige Pixel erstrecken. Der Nutzer ist jedoch nicht an den Spuren interessiert sondern am genauen Eintrittspunkt der Elektronen. In der Kristallografie eröffnet die Onlineanalyse der Laue Diffraktionsmuster neue Möglichkeiten. Hierzu müssen alle sensorspezifischen und elektronischen Effekte in Echtzeit analysiert und korrigiert werden.

Es werden vier verschiedene Methoden für unterschiedliche Ereignisraten auf den Detektoren entwickelt, um den Eintrittspunkt und die Intensitätsverteilung der Strahlung auf dem Detektor genau zu bestimmen. Diese Methoden ermöglichen die Rekonstruktion von Positions- und Intensitätsinformationen in Echtzeit bei hohen Bildraten. Für weitere physikalische Analysen müssen keine zusätzlichen Detektorkorrekturen vorgenommen werden. Für Röntgenstrahlung wurde eine Subpixelauflösung von weniger als 10% der Pixelabmessungen erreicht. Für Elektronen mit einer Primärenergie von 300 keV in der Transmissionselektronenmikroskopie (TEM) wurde der Eintrittspunkt in den Detektor genau bestimmt (40 μm), obwohl sie eine Spur von mehr als 450 μm im Silizium erzeugen.

Entscheidend für die Entwicklung und den Test der neuronalen Netze sind große Datensätze, die zum einen die unanalysierten Rohdaten des Detektorsystems und zum anderen beispielsweise durch Simulationen die zugehörigen exakten Eintrittspunkte der Primärstrahlung enthalten. Die Voraussetzung

für den unanalysierten Datensatz ist eine physikalisch genaue Beschreibung der Signalentstehung in den einzelnen Pixeln.

Zu diesem Zweck wurden das Verhalten der Primärteilchen im Detektorvolumen und die Signalantwort der einzelnen Pixel physikalisch beschrieben und modelliert. Die Ergebnisse wurden in einer Monte-Carlo-Simulation implementiert. Die Monte-Carlo-Daten wurden durch zahlreiche Messungen verifiziert.

Abschließend werden die neu entwickelten neuronalen Netze mit den bisher verwendeten Methoden in Bezug auf Parameter wie räumliche Auflösung und Analysegeschwindigkeit anhand von simulierten Daten und durchgeführten Messungen verglichen. Zur Überprüfung und Verifizierung der entwickelten Methoden haben wir eine Reihe von Messungen mit einem Transmissionselektronenmikroskop bei verschiedenen Elektronenintensitäten und -energien durchgeführt. Die auf der Verwendung neuronaler Netze basierenden Ergebnisse stimmen gut mit den genau bekannten Eintrittspunkten aus den experimentell gemessenen und den simulierten Daten überein. Ein Röntgendatensatz mit 1.3 keV Röntgenstrahlung ergab eine Positionsauflösung von besser als $3\ \mu\text{m}$ - ebenfalls in guter Übereinstimmung mit experimentellen Daten aus einem Röntgenmikroskop. Die entwickelten Algorithmen bieten das Potential für Datenanalyse und Datenreduktion in Echtzeit.

In diesem Sinne bietet diese Arbeit eine Basis und ein grundlegendes Verständnis für zukünftige fortschrittliche Datenanalyseanwendungen für pixelierte Halbleiterdetektorsysteme.

Contents

Important Technical Term Definitions and Abbreviations	1
1 Introduction	7
2 Radiation Interaction with Matter	13
2.1 Photons	13
2.2 Electrons	20
2.3 Atomic Relaxation	28
3 Pixelated Semiconductor Detectors	31
3.1 Semiconductors as Detector Material	31
3.2 Electron-Hole Pair Generation	36
3.3 The pnCCD	37
3.4 Conversion from Signal Electrons to a Digital Signal	39
4 Simulation of the Energy Deposition by Photons and Electrons	43
4.1 Energy Deposition	44
4.2 Conversion and Transport of the Generated Signal Charges	46
4.3 Generation of Frames	56
4.4 Readout Noise	56
4.5 Results for Photons as Primary Particles	57
4.6 Results for Electrons as Primary Particles	60
4.7 Numerically Generated Data Set For Photons	71
4.8 Summary	79
5 Classical Data Analysis Methods	81
6 Artificial Neural Networks	85
6.1 Introduction	85
6.2 Neurons	89
6.3 Topology of a Neural Network	90

6.4	Activation Function	92
6.5	Training Process	94
6.6	Optimizer	102
6.7	Special Layers	109
6.8	8-Bit Quantization	120
6.9	Edge Tensor Processing Unit	121
6.10	Workflow	121
7	Reconstruction of the Point of Entry with a Compact Neural Network	125
7.1	Network Architecture	125
7.2	Training	128
7.3	Validation	129
8	Reconstruction of the Point of Entry with Convolutional Neural Networks	139
8.1	Network Architecture	140
8.2	Checkerboard Artifacts	151
8.3	Loss Function	153
8.4	Training	158
8.5	Validation	164
9	Super-Resolution of Intensity Images	169
9.1	Single Image Super-Resolution	170
9.2	Multi-Image Super-Resolution	172
9.3	Training	175
9.4	Validation	179
9.5	Summary	186
10	Analysis of Photon Detection and Electron Tracks	187
10.1	Photon Detection	187
10.2	Electron Detection	196
10.3	Summary	202
11	Conclusion and Outlook	205
A	Physical Considerations	209
A.1	Spatial Precision, Spatial Accuracy, and Spatial Resolution . . .	209
A.2	Physical Limitations	209
A.3	Units of the Bethe Bloch Formula	221
A.4	Landau Distribution Function	222

A.5 Electric Potential and Field Approximation in the Silicon Detector Volume for pnCCD	223
B Classical Data Analysis Methods	225
B.1 Event Pattern Analysis	227
B.2 Corrections to the Center of Gravity Method	235
B.3 Spatial Resolution in the Presence of Noise	256
B.4 Average Distance from the Mean in Comparison to the Standard Deviation	263
C Artificial Neural Networks	265
C.1 Pseudo-Code of the Adam Optimization Algorithm	265
C.2 Gradient of the Loss Function with Respect to the Weights and the Biases of the Neural Network	267
C.3 Pseudo-Code of the Backpropagation Algorithm	271
C.4 Hyper-Parameters of the Special Layers Used in Neural Networks	272
C.5 Topology and Number of Parameters of Used Neural Networks .	274
D Experiments	281
D.1 Modulation Transfer Function	281
D.2 Data Reference	287
List of Figures	287
List of Tables	293
Bibliography	297

Important Technical Term Definitions and Abbreviations

Accuracy

The spatial accuracy is defined as the average distance between the mean position of the individual reconstructed points of entry (PoEs) and its ground truth.

Activation Function

In a neural network, the activation function of a neuron defines the output of the neuron. For the last layer of a neural network, the activation function defines the output range of the neural network and has to be problem-related.

Charge Cloud

In semiconductors, primary particles create electron-hole pairs that are separated. The electrons form charge clouds. It increases during the drift process into the pixel structure due to repulsion and diffusion.

CNN

A convolutional neural network (CNN) is a neural network that consists of convolutional layers and is typically used for frames and images.

Counting Detector System

A counting detector system describes a detector system that outputs only the positional information of a PoE but no information about the amount of deposited energy.

Cross-entropy

The cross-entropy measures the difference between two probability distributions and can be used as a loss function to train a neural network. Depending on the number of classes of the classification task, the binary cross-entropy or the weighted cross-entropy can be used.

Diffusion

Diffusion describes the increase of the charge cloud size due to thermal effects.

Drift

The generated charge clouds move due to the local electric field in the detector volume into the pixel structure located at the backside of the detector volume. This process is called drift.

Event Pattern

The pixels of a frame that contain an energy deposition and are spatially connected are called event patterns.

Feature Map

A feature map describes the position of a particular feature (property) in the data. The feature maps of the input and the output layers represent physical properties. These properties have to be problem-related. The features described by the feature maps of the hidden layer of a convolutional neural network can be very complex and physically not meaning full.

Ground Truth

Ground truth is data that is known to be authentic or true. The neural network is trained to reconstruct the ground truth as best as possible.

Intensity Image

For very high primary particle rates, the energy depositions of individual particles can not be separated anymore. In nearly every illuminated pixel and its neighbors, energy depositions of at least one primary particle can be detected. The resulting image is called an intensity image.

Layer of a Neural Network

A neural network consists of several neurons. These neurons are organized in layers. The first layer of a neural network is called the input layer, the intermediate layers are called the hidden layers, and the last layer of a neural network is called the output layer.

Loss Function

The loss function maps the difference between the predicted result and the ground truth to a real number. The neural network is optimized during the training process to minimize the loss function.

MISR Approach

The multi-image super-resolution (MISR) approach increases the resolution and denoises using multiple images of the same scenario.

MTF

The modulation transfer function (MTF) describes how much contrast of the object function is transferred by the imaging process.

Neuron

The neurons are the basis for neural networks. A neuron receives one or more inputs, separately weights them, and sums them up. To this weighted sum, a bias is added and the result is passed to a non-linear activation function.

Neural Network

The neural network is an arrangement of many neurons and can be trained to solve complex tasks.

Particle Track

Charge particles deposit their energy by multiple scattering in the detector volume. Therefore, the energy deposition is randomly distributed along the particle trajectory, called the particle track.

Pattern Pile-up Event

A pattern pile-up event is an event pattern where the energy deposition of two or more different primary particles is combined to one cluster and recognized as one event pattern by the event pattern analysis algorithm.

PoE

The point of entry (PoE) denotes the point where the primary particle crosses the surface of the detector.

PoENN

The point of entry neural network (PoENN) reconstructs the PoE of primary particles frame-wise on pixel or subpixel level.

PnCCD

The pn charge-coupled device (pnCCD) is used as detector for the reference measurements presented in this thesis.

Precision

The spatial resolution is defined as the average Euclidean distance between

the individual reconstructed PoEs and their ground truth.

Primary Particle

The primary particle is the particle that the detector system should detect.

Repulsion

Repulsion describes the charge cloud size increase due to electrostatic repulsion of the signal electrons.

Resolution

The spatial resolution is defined as the average Euclidean distance between the individual reconstructed PoEs and their ground truth.

Signal Electron

Signal electrons are the electrons of the electron-hole pairs created by energy deposition in the detector volume.

Single Pattern Event

Event pattern to which only one primary particle contributes.

SISR Approach

The single image super-resolution (SISR) approach increases the resolution and denoises using a single image.

Subpixel

A physical pixel can be divided into several virtual pixels. These virtual pixels are called subpixels.

Super-Resolution

Super-resolution is a technique that increases the resolution of an image, typically providing images on a subpixel level.

Supervised Learning

Supervised learning is a type of machine learning that learns a function that maps a result to a corresponding input by examples.

Tracking Detector System

A tracking detector system describes a detector system that outputs the amount of energy deposition, its shape, and its spatial location.

Training

The training describes the optimization process of a neural network with a training data set. This training data set contains the input and the corresponding ground truth.

Chapter 1

Introduction

Advancements in the semiconductor fabrication make it possible to produce more powerful and performing pixelated detectors based on silicon technology. The detector systems can be readout faster, the signal-to-noise ratio is getting closer to the physical limit, and the active area of the detector becomes larger, whereas a smaller feature size enables finer pixel structures and more sophisticated electronics [1]. This enables applications that were unimaginable a few years ago.

However, better detector hardware is only one part. The analysis of raw data collected with the detector system is crucial as well for reaching the physical limits of spatial and energy resolution. Moreover, the amount of data generated with the new detector systems increases significantly because of the smaller pixels and the faster readout per unit area [2].

With the detector hardware being significantly improved over the last decades, new demands for high-performance data analysis software arise. Those demands can be divided into challenges originating from the detector operation principles and challenges given by the signal generator's physical behavior in the detector volume.

One of the biggest technical challenges is handling, storing, and processing large amounts of data. For example, particle accelerators such as the Large Hadron Collider at CERN produce around 250 TByte per day [3]. The data rates have increased in consumer applications and equally in many fields of science. For example, a two-dimensional detector at synchrotron light sources can generate tens of GByte per second [4].

In some applications, it is not possible to handle such large amounts of raw data. For example, satellite-based missions are limited due to the downlink bandwidth from the satellite to ground control [5]. One way of dealing with

this issue is to reduce the amount of data without losing any important information. One method is to reduce the number of pixels within the same total active area of the detector, which reduces the data rate at the cost of spatial resolution. To some extent, complex subpixel algorithms must be used afterward to restore the original spatial resolution.

Alternatively, one can process and analyze the data directly and send only analyzed results back to earth. This requires fast and energy-efficient algorithms.

Most of the current analysis methods process, validate, and analyze the data after an experiment measurement time due to computational limitations. Due to this, experimental constraints and errors are not immediately detected and cannot be solved ad-hoc. Processing the data in real-time can enable the detection of errors and unwanted imperfections during the experiment. This allows intervening and solving problems and optimizing data acquisition. Real-time data analysis can enable both detection of and recovery from problems and data acquisition optimization. [4]

The physical nature of the signal generation in the detector volume represents the other major challenge. The particles that should be detected are called primary particles in the following. Primary particles deposit their energy depending on their type and energy. *In this work, electrons and photons in particular are referred as primary particles.* Charged particles like electrons of hundreds of kilo electron volts deposit their energy in the detector volume along traces with a length of up to several hundred micrometers depending on their energy. In contrast, photons deposit their energy in a small detector volume. In both cases, the energy is deposited into the semiconductor-based detector volume via the creation of electron-hole pairs. The electrons of these electron-hole pairs are separated from the holes, form charge clouds, and drift due to the detector volume's local electric field into the pixel structure. The created holes drain at negatively biased electrical contacts and are not used for signal analysis. During the drift time, the electron charge clouds expand by diffusion and electrostatic repulsion. Therefore, the charge cloud can spread over several pixels. In addition, there is a large spread in the case of electron tracks due to the created traces. The detector system's output is the pixelated two-dimensional projection of the charges in the individual pixels. [6]

The quality of many applications is related to the spatial resolution of the detector system. These applications are widespread and are used, for example, in the fields of high energy physics [7], medicine [8], material science [9], or transmission electron microscopy [10]. However, to provide a good spatial resolution, not the energy deposition into the individual pixels is of interest

but *the point of entry (PoE)* into the detector volume. The PoE denotes the point where the primary particle crosses the surface of the detector. The spatial resolution is mainly limited by algorithms that convert the individual energy depositions into PoEs and not only by the pixel size. Ryll et al. [11] and Ihle et al. [12] show such state-of-the-art algorithms for pnCCD detector systems. In the following, these algorithms are called classical approaches.

Summarized, modern detector systems' developments demand high-performance software that can cope with the operational challenges and high data rates. Additionally, this software can enable new analysis methods and techniques. In general, classical algorithms developed to solve these challenges are very complex and often produce results far away from physical limits [13]. Hence, classical algorithms struggle to meet the increasing demands. Emerging machine learning algorithms show a way into the future.

In the last years, approaches based on neural networks have been established in many fields and show their potential, especially for pattern recognition and complex problems with a large amount of data. [14] There are mainly three fields in the area of machine learning in which the most rapid progress is currently taking place.

- First, the processing of speech, images, videos, and texts. Prominent examples are virtual assistants such as Siri by Apple Inc. [15] or Alexa by Amazon.com Inc. [16], the colorization of grayscale images [17], or translators such as Google translate by Google LLC [18] or DeepL translator [19].
- Second, the analysis of a large amount of data, also known as big data, and its representation. Machine learning applications are very powerful in finding patterns or clusters in big data structures. This work represents an example in this field.
- Third, the construction of semi-autonomous and autonomous machines. The best-known example here is self-driving cars [20].

These three segments are increasingly growing together.

The concept of machine learning is not new. The first research project on artificial intelligence, of which machine learning is a subcategory, was carried out in 1956 [21]. However, the rapid development of applications

based on machine learning has only recently started. The growth can mainly be attributed to three factors that positively influence each other.

- The biggest factor is the performance enhancement of the computer hardware. 25 years ago, the most powerful supercomputers managed about 100 billion operations per second. Today, any state-of-the-art smartphone can handle this performance. At the same time, costs have dropped by a factor of 10'000. [14]

Following Moore's law, the number of transistors doubles about every two years [22]. Besides, entirely new approaches such as quantum computing are being developed to create powerful hardware [23, 24]. Additionally, cameras, microphones, and sensors of all kinds such as gyroscopes, accelerometers, and light imaging detection and ranging (lidar) are getting smaller and cheaper [14]. These sensors are used for data acquisition, which can be used as input for the machine learning application.

- The next factor is the optimization of the algorithms used behind machine learning. These include, above all, much deeper neural networks (deep learning) with significantly more neurons. One can imagine the neural network as a non-linear function. The variable of this function is the input of the neural network. The function additionally has parameters, which are optimized for the problem. Deeper neural networks with more neurons have more parameters. This makes it possible to fulfill significantly more complex tasks, but also requires more computing power. Moreover, the architecture of neural networks is getting more and more problem-specific. [25]
- The third factor is a large number of data sets that are public and freely available [26]. These larger data sets are required to train the neural networks. For example, the MNIST [27] data set is one of the most used machine learning data sets. It contains hand-written digits and consists of 60'000 examples and a test set of 10'000 examples. Another big database is the open image data set [28]. It contains around 9 million real images notated with image-level labels, object bounding boxes, object segmentation masks, visual relationships, and localized narratives. These data sets are excellent for the training and validation process.

In scientific context, especially in the reconstruction, restoration, and analysis of images, neural networks have an extreme potential. In particular, biomedical applications such as denoising [29], super-resolution [30], Radon transform for computed tomography [31], tissue classification and segmentation [32], and diagnostic [33] are worth mentioning.

Machine learning is currently being tested for problems where complex algorithms would be required to solve the problem. The advantages of machine learning based approaches lead to more precise scientific results and accelerate the analysis processes. [34] Applications are, for example, the acceleration of data generation for fluid dynamic simulations [35], triggering decisions for large detector systems [36], or data analysis [37].

In this work, new methods for analyzing data produced by semiconductor tracking and spectroscopy imaging detectors are developed and investigated. Those methods are based on neural networks.

The goal in this thesis is to improve the spatial resolution, increase the analysis processes' speed, and, thus, enable data analysis in real-time. This work begins with a review of the fundamentals of the photons' and electrons' interaction with matter (Chapter 2). Chapter 3 introduces the concept of pixelated semiconductor detectors and their signal response. In particular, the pnCCD detector system is introduced since it is used for this work's reference measurements.

Crucial for the development and testing of neural networks are large data sets containing, on the one hand, the unanalyzed raw data of the detector system and, on the other hand, the associated precise PoE of the primary particle. The requirement for the unanalyzed data set is a physically accurate description of the signal distribution in the individual pixels. For this purpose, the interactions of the primary particles in the detector volume and the signal response of the individual pixels are physically described and modeled. The results are implemented in a Monte Carlo simulation and validated (Chapter 4). Using the data generated by the Monte Carlo simulation, the new approaches can be developed independently of other existing data analysis methods. The Monte Carlo simulation provides the basis for the training data set for the neural networks. The results of these simulations explain relationships between the recorded signals and properties of the detector system and the primary particles. Therefore, a detailed understanding of the physics behind the signal response and simulated data is necessary for a successful training process of the neural networks.

In order to provide the necessary fundamentals for the development of an approach based on neural networks, the main terms, and components of neural networks are introduced in Chapter 6.

Based on the application purpose, the developed neural networks are divided into three parts:

- A compact neural network (CoNN) that requires a classical event pattern

analysis is presented in Chapter 7. The compact neural network can reconstruct the PoE by using the output of a classical event pattern analysis on subpixel level.

- A convolutional neural network to reconstruct the PoE (PoENN) without a previous event pattern analysis is presented in Chapter 8. The convolutional neural network is applied frame-based and is applicable up to primary particle rates where energy depositions of individual primary particles are identifiable. The main advantage of this approach is that PoENN can separate overlapping electron traces. The classical methods cannot handle overlapping traces. Consequently, the primary particle rate can be increased, but all PoEs of the individual primary particles can still be reconstructed. Another advantage of this approach is that no additional data analysis is required. This fact leads to improving the reconstruction rate and, therefore, enables real-time applications. The approach can provide subpixel resolution for electrons with primary low energies below approximated one hundred kilo electron volt and X-ray photons.
- In Chapter 9, a framework for super-resolution also known as subpixel resolution of intensity images and denoising is presented. In this context, intensity images are recorded with very high primary particle rates. Consequently, the traces of individual particles can not be separated anymore. Super-resolution enhances the spatial resolution, whereas denoising reduces unwanted noise.

To investigate the accuracy with regard to physical parameters and performance of the developed neural networks, they are compared with state-of-the-art methods in Chapter 10. For comparison, data obtained by measurements are used.

The developed models lead to a significant improvement of the data analysis process approaching closer to the physical limits compared with the classical methods.

Chapter 11 closes with a conclusion and an outlook for the future.

Chapter 2

Radiation Interaction with Matter

In order to detect particles with a detector, the particles must interact with the detector material and deposit their energy in the detector volume. These interaction processes are described for photons and charged particles in this Chapter. The interaction processes are introduced using silicon as an example detector material without the loss of generality. They are the basis of the simulations implemented in GEANT4.

2.1 Photons

In the following, the interactions of photons with the detector material are described.

2.1.1 Physical Processes

Photons interact with matter to deposit energy mainly in three ways. These effects are the photoelectric effect, the Compton effect, and pair production. [38] Figure 2.1 shows the cross-sections for silicon. The total cross-section is the sum of the cross-sections of the individual processes. For energies below approximately 50 keV, the photoelectric effect is the dominating effect in silicon. The cross-section rapidly increases as soon as the photon's energy is sufficient to ionize an electron from an energetically higher shell. [39]

These edges are named after the shells of the ionized electron. The K-edge (1839 eV) and the L-edge (149.7 eV) of silicon can be seen in Figure 2.1 [40]. At energies between 60 keV and 10 MeV, the Compton scattering provides the main contribution to the photon's energy loss. For even higher energies, the pair production process is dominant. For the pair production process, at least a primary energy of the rest mass of an electron and a positron is necessary.

Rayleigh scattering describes the elastic scattering of photons with atoms and molecules. Therefore, only the direction of the primary photon is changed, and the atom is neither excited nor ionized. The photon conserves its total energy. It is only relevant for primary energies below a few tens kilo electron volts. The average deflection angle decreases with increasing energy. As a consequence, the influence is further restricted to low energies. [39]

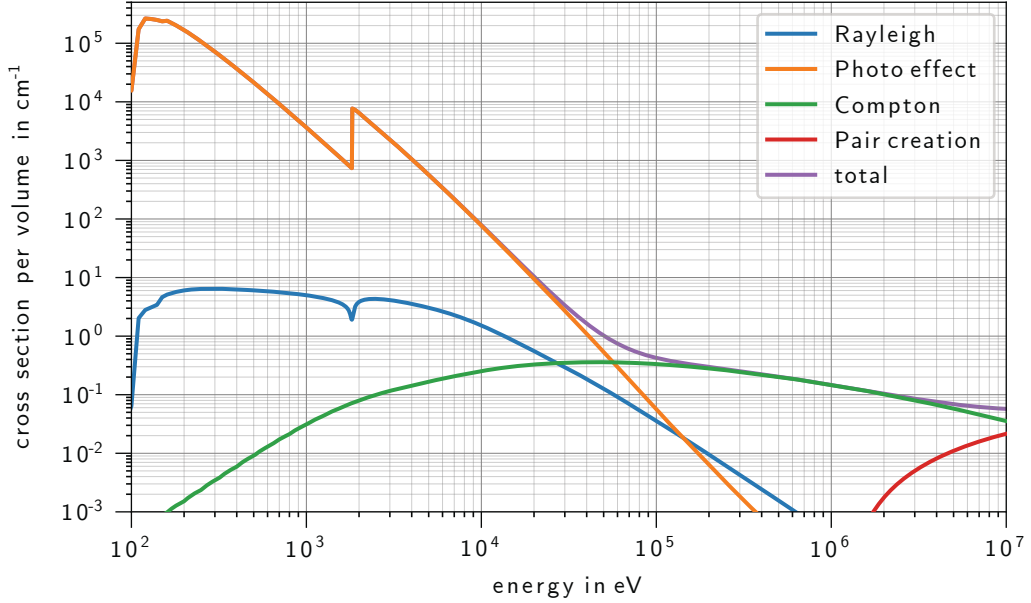


Figure 2.1: Cross-section of the processes which are dominant for the photon in silicon with a density of 2.33 g/cm^3 as a function of the primary energy. The photoelectric effect dominates for lower energies. The L-edge (149.7 eV) and the K-edge (1839 keV) can be seen as a kink in the cross-section. The Compton scattering is dominant for intermediate energies, and the pair production process becomes relevant for energies above two times the rest mass of the electron. The plotted data was obtained from the cross-sections, and the stopping power was used by the simulation with Geant4 and extracted directly from the physics tables and not from a Monte Carlo simulation [41, 42]. Rayleigh scattering describes the elastic scattering of photons by particles much smaller than the wavelength of the radiation. The plots in this work are created by using Matplotlib [43].

2.1.1.1 Photoelectric Effect

The photoelectric effect describes the process where the incoming photon is fully absorbed and does not exist after the interaction anymore [44]. A

schematic drawing of the photoelectric effect is shown in Figure 2.2. Here A is an atom, and A^* is the ionized atom. In silicon, the photon's energy E_{ph}

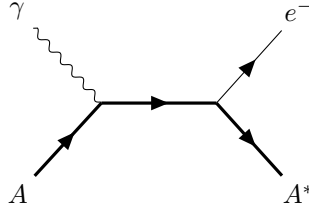


Figure 2.2: Schematic drawing of the photoelectric effect. A photon γ ionizes an atom A . The time axis is horizontal.

excites an electron over the band gap with a gap energy $E_{\text{gap}} \approx 1.12\text{eV}$ [45]. The remaining energy goes into the kinetic energy of the electron E_{kin} [39]:

$$E_{\text{kin}} = E_{\text{ph}} - E_{\text{gap}} \quad (2.1)$$

The kinetic energy of the electron generates either phonons or additional electron-hole pairs. Some of the photon's energy dissipates into the crystal lattice, which leads to an average energy of approximated 3.68 eV per created electron-hole pair at room temperature for photon energies above 50 eV [46]. Except for deviations for photons with an energy near the band gap energy, this value is independent of the incoming photon's energy [39]. The cross-section of the photoelectric effect depends on the energy of the incoming photon E_{ph} and the atomic number of the detector material Z [38]:

$$\sigma \propto \frac{Z^n}{E^m} \quad (2.2)$$

The exponents vary for different approximations. Here, the Born approximation is used, and the exponent n of the atomic number is between 4 and 5. The exponent of the energy m is smaller equal to 3.5, and becomes one for high energetic photons (energy much higher than the rest mass of the electron)[38]. The angular distribution of the emitted electron is, in the non-relativistic case, proportional to $\sin^2 \theta$ [38]. Here, θ is the polar angle of the emitted electron relative to the direction of the incoming photon. With increasing energy, the electron follows more and more the direction of the photon.

2.1.1.2 Compton Effect

The Compton effect is an incoherent scattering process between the incoming photon and a quasi-free electron. An electron of the atomic shell is called

quasi-free when the photon's energy E_{ph} is much larger than the electron's binding energy E_{bi} (Figure 2.3).

$$E_{\text{ph}} \gg E_{\text{bi}} \quad (2.3)$$

The photon loses a part of its energy during the scattering process, which is transferred to the electron. [47]

The photon's energy \tilde{E}_{ph} after the collision can be calculated as a function of

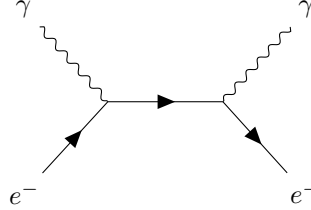


Figure 2.3: Feynman diagram of the Compton effect. The photon γ scatters with an electron e^- and transfers a fraction of its energy. The time axis is horizontal.

the scattering angle θ_{ph} and the photon's energy E_{ph} before the collision [38]:

$$\tilde{E}_{\text{ph}} = \frac{E_{\text{ph}}}{1 + \frac{E_{\text{ph}}}{m_e c^2} (1 - \cos \theta_{\text{ph}})} \quad (2.4)$$

Here, m_e is the rest-mass of the electron and c the speed of light in vacuum.

The Klein-Nishina formula describes the differential cross-section per solid angle Ω for free electrons [48]:

$$\frac{d\sigma_C}{d\Omega} = \frac{r_e^2}{2 [1 + E_{\text{ph}} (1 - \cos \theta_{\text{ph}})]^2} \left(1 + \cos^2 \theta_{\text{ph}} + \frac{\left(\frac{E_{\text{ph}}}{m_e c^2}\right)^2 (1 - \cos \theta_{\text{ph}})^2}{1 + \frac{E_{\text{ph}}}{m_e c^2} (1 - \cos \theta_{\text{ph}})} \right) \quad (2.5)$$

Here, r_e is the classical electron radius, with ϵ_0 being the vacuum permittivity [39]:

$$r_e = \frac{e^2}{4\pi\epsilon_0 m_e c^2} \quad (2.6)$$

Figure 2.4 depicts the probability distribution of the differential cross-section as a function of the photon's scattering angle. The scattering angles are forward orientated for higher primary energies.

The cross-section per atom can be approximated by the number of shell

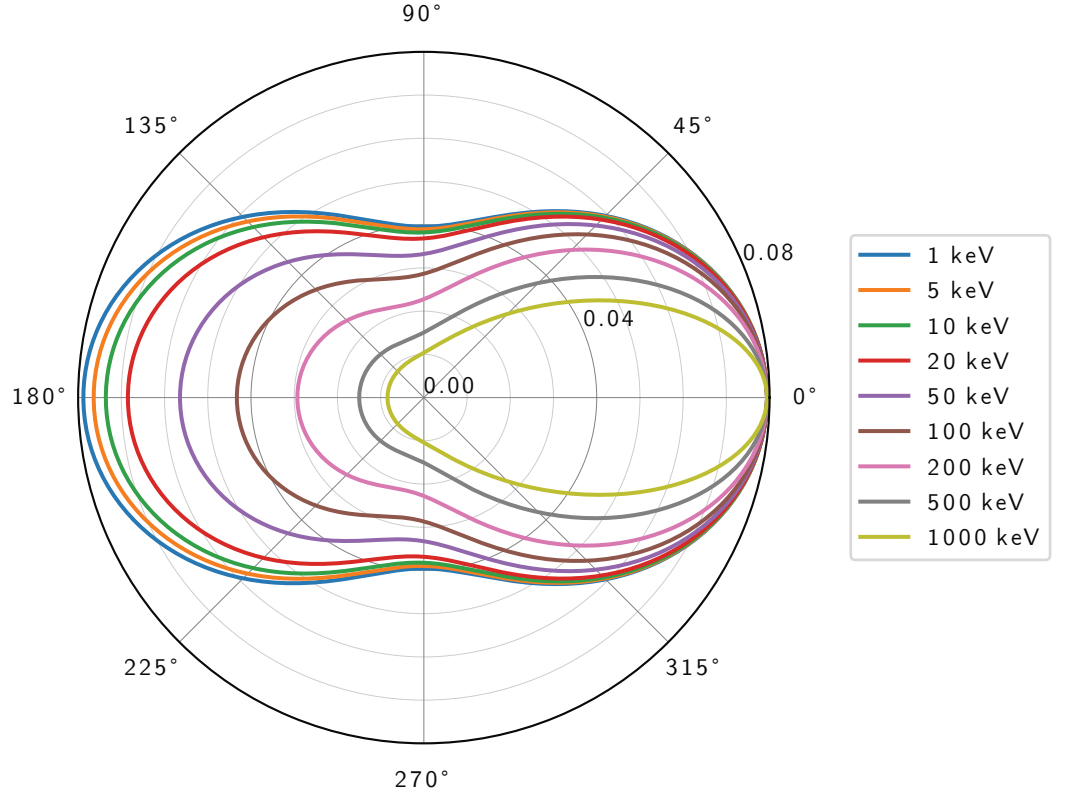


Figure 2.4: Klein–Nishina distribution as function of the scattering angle θ_{ph} . The unit of the r-axis is barn = 10^{-28}m^2 .

electrons times the differential cross-section for one shell electron [38]:

$$\sigma_{\text{C}}^{\text{atom}} \approx Z\sigma_{\text{C}} \quad (2.7)$$

The cross-section is directly proportional to the atomic number because every electron of the shell contributes to the incoherent scattering process. The cross-section's linear dependence on the atomic number holds as long as the photon energy is higher than the electrons' binding energy.

The shell electron's momentum on the scattered photon's energy leads to a deviation called Doppler broadening. The scattering process becomes coherent for lower energies, and the cross-section depends nearly quadratic on the atomic number. The angle is for low energies proportional to $(1 + (\cos \theta_{\text{ph}})^2)$. [38] The

energy dependence is obtained as follows [38]:

$$\sigma_C \propto \begin{cases} \left(1 - 2\frac{E_{\text{ph}}}{m_e c^2}\right) & E_{\text{ph}} \ll m_e c^2 \quad \text{Thomson limes} \\ \left(\ln\left(2\frac{E_{\text{ph}}}{m_e c^2}\right) + \frac{1}{2}\right) & E_{\text{ph}} \gg m_e c^2 \quad \text{highly relativistic} \end{cases} \quad (2.8)$$

The angle θ_e between the incoming photon and the moving direction of the electron after the scattering process can be calculated to [38]:

$$\cos \theta_e = \frac{E_e(E_{\text{ph}} + m_e c^2)}{E_{\text{ph}}\sqrt{E_e^2 + 2m_e c^2 E_e}} \quad (2.9)$$

Here, E_e is the kinetic energy of the electron. The maximum energy transfer to the electron is for backscattering ($\theta_{\text{ph}} = \pi$) of the photon corresponding to a forward scattering of the electron ($\theta_e = 0$). In this case, the kinetic energy of the electron is the maximum kinetic energy. This leads to the so-called Compton edge at the following energy in the spectrum [38]:

$$E_e^{\text{max}} = \frac{2E_{\text{ph}}^2}{2E_{\text{ph}} + m_e c^2} \quad (2.10)$$

Figure 2.5 shows the maximal transferable energy and the mean transferred energy as a function of the photon's primary energy. The mean transferred energy is the expected value of the product of the Klein-Nishina formula and the angle-dependent energy transfer to the electron:

$$\langle E_e \rangle = \frac{\int_S \int \frac{d\sigma_C}{d\Omega} (E_{\text{ph}} - \tilde{E}_{\text{ph}}) d\Omega}{\int_S \int \frac{d\sigma_C}{d\Omega} d\Omega} = \frac{\int_0^\pi \frac{d\sigma_C}{d\theta_{\text{ph}}} (E_{\text{ph}} - \tilde{E}_{\text{ph}}) \sin(\theta_{\text{ph}}) d\theta_{\text{ph}}}{\int_0^\pi \frac{d\sigma_C}{d\Omega} \sin(\theta_{\text{ph}}) d\theta_{\text{ph}}} \quad (2.11)$$

2.1.1.3 Pair Production

A photon can convert into an electron-positron pair in the presence of a charge and, therefore, a corresponding electric field. The photon's energy has to be larger than twice the electron's mass plus the recoil energy of the charge with mass m_n (eq. 2.12). This charge is typically a nucleus. [49]

In general, the mass m_n of the nucleus is much bigger than the rest mass of the electron, and therefore, the second term can be neglected [38]:

$$E_{\text{ph}} \geq 2m_e c^2 \left(1 + \frac{m_e}{m_n}\right) \approx 2m_e c^2 \quad (2.12)$$

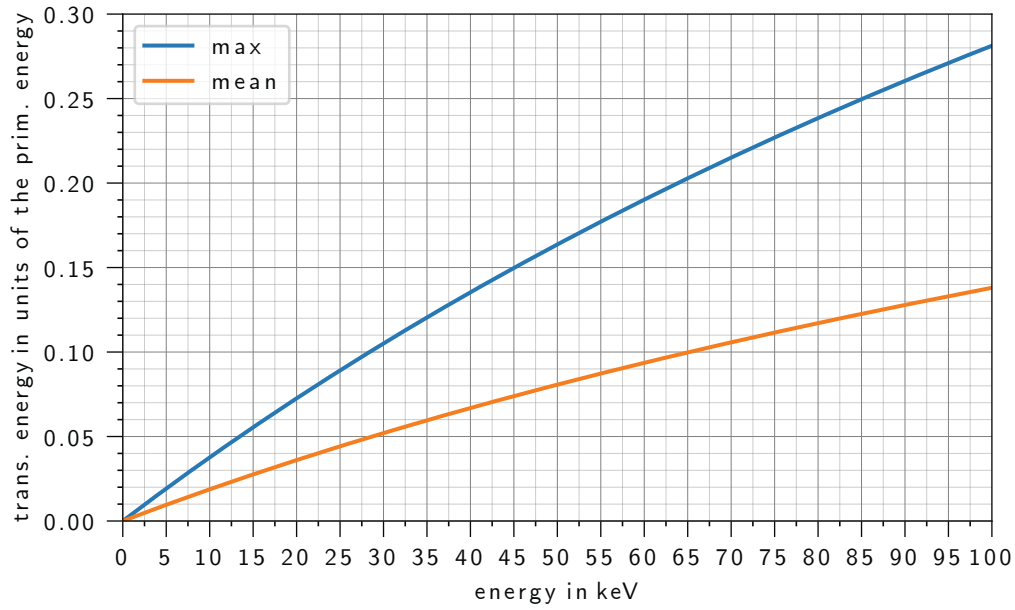


Figure 2.5: Transferred energy from incoming photon to electron during the Compton effect.

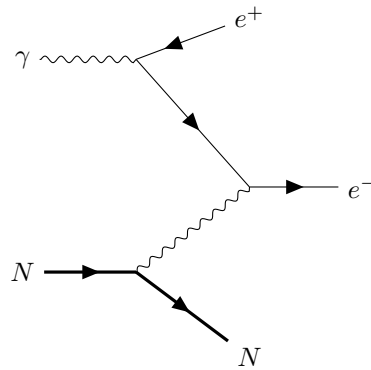


Figure 2.6: Schematic view of the pair production. A photon γ creates an electron e^- and a positron e^+ . N is the nucleus whose electric field is required to preserve the conservation of momentum. The time axis is horizontal.

The pair production quadratically depends on the atomic number and is for high energetic photons (energy is much higher than the rest mass of the electron) nearly constant for different energies of the photon. [38]

2.1.2 Lambert–Beer Law

Due to the described effects in Section 2.1.1, photons are absorbed or deviated from their original path with a probability proportional to the distance dz . The attenuation coefficient μ describes the probability of absorption per path length and is connected to the cross-section σ via the density of atoms n in the detector volume [38]:

$$\mu = -\frac{1}{N} \frac{dN}{dz} = n\sigma \quad (2.13)$$

Here, N is the number of photons. After a certain path length, the number of remaining photons is derived from solving this differential equation and is called Beer-Lambert-law.

$$N(z) = N_0 e^{-\mu z} \quad (2.14)$$

N_0 is the original number of photons. Another often used quantity is the mean free path λ , the inverse attenuation coefficient. The mean free path is the average distance traveled by a photon between two successive interactions with the detector material. Figure 2.7 shows the relative absorption $A(z)$ as a function of the layer thickness for various primary energies for silicon. Eq. 2.15 describes the relative absorption.

$$A(z) = 1 - e^{-\int_0^z \mu(z') dz'} = 1 - e^{-\mu z} \quad (2.15)$$

2.2 Electrons

The electron is an elementary particle of the standard model. Its rest mass is 511 keV, and it is charged with one negative elementary charge. Its energy loss in the detector material can be divided into three groups. These are ionization and excitation, which describe the interactions with the atomic shell's electrons, and Bremsstrahlung, for higher primary energies and the scattering processes with the nuclei. [38]

2.2.1 Physical Processes

The physical interactions between electrons and matter can be divided into interactions with the atomic shell and with the atomic nucleus. Figure 2.8 depicts the schematic interactions with the atomic shell (Figure 2.8a, 2.8b, and 2.8c) and the interactions with the atomic nuclei (Figure 2.8d, and 2.8e).

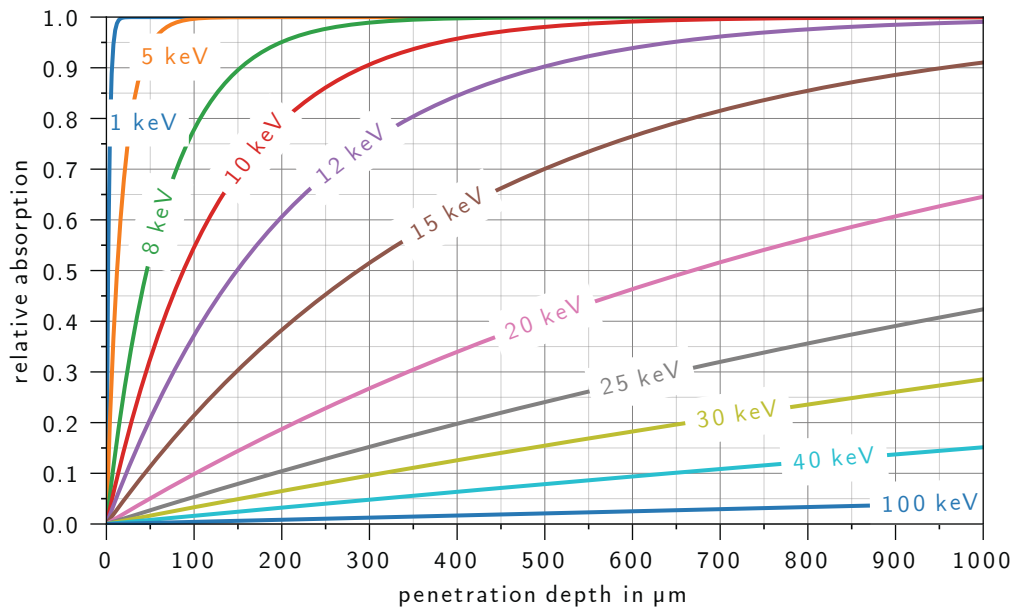


Figure 2.7: The relative absorption for photon beams with different primary energies as a function of the silicon’s penetration depth is shown. The intensity after a layer of silicon with a given thickness corresponding to the penetration depth in this plot is one minus the relative absorption. The data are obtained from the cross-sections and stopping power used by the simulation with GEANT4 [41, 42] and are extracted directly from the physics tables and not from a Monte Carlo simulation.

2.2.1.1 Scattering at the Atomic Shell

The electron can lose some of its energy by scattering at the electrons in the shell of the atomic nuclei. The electron can scatter elastically to Coulomb interaction with the total atomic shell (Figure 2.8a). In this case, the charge distribution in the shell is deformed and polarized. The primary electron loses only a small amount of energy during this interaction. [38]

If, on the other hand, the interaction with the total atomic shell leads to excitation (Figure 2.8b) or ionization (Figure 2.8c) of an electron of the outer shells of an atom, the scattering process is inelastic. The primary particle deposits the amount of energy necessary to ionize, respectively, excite an electron of the outer shells of an atom. This amount of energy is in the order of a few electron volts. The released secondary electron has a small amount of energy and deposits it in the immediate environment. These two scattering processes lead only to a small energy deposition and are called soft collisions. Soft collisions make approximately 50 % of the energy deposition [50]. Because of the

small energy loss per interaction, the incoming particles are decelerated quasi continuously, which is called continuous slowing down. [38]

Inelastic scattering with a single electron of the atomic shell leads to ionization or excitation of the atom, leading to a larger energy transfer. These interactions are called hard collisions. These emitted secondary electrons are called δ -electrons if their energy is high enough to ionize other atoms themselves. The scattering angles of δ -electrons are bigger, and their energy is much higher than it would be with secondary particles generated by soft collisions. The tracks of the δ -electrons leave the primary particle track laterally. [38]

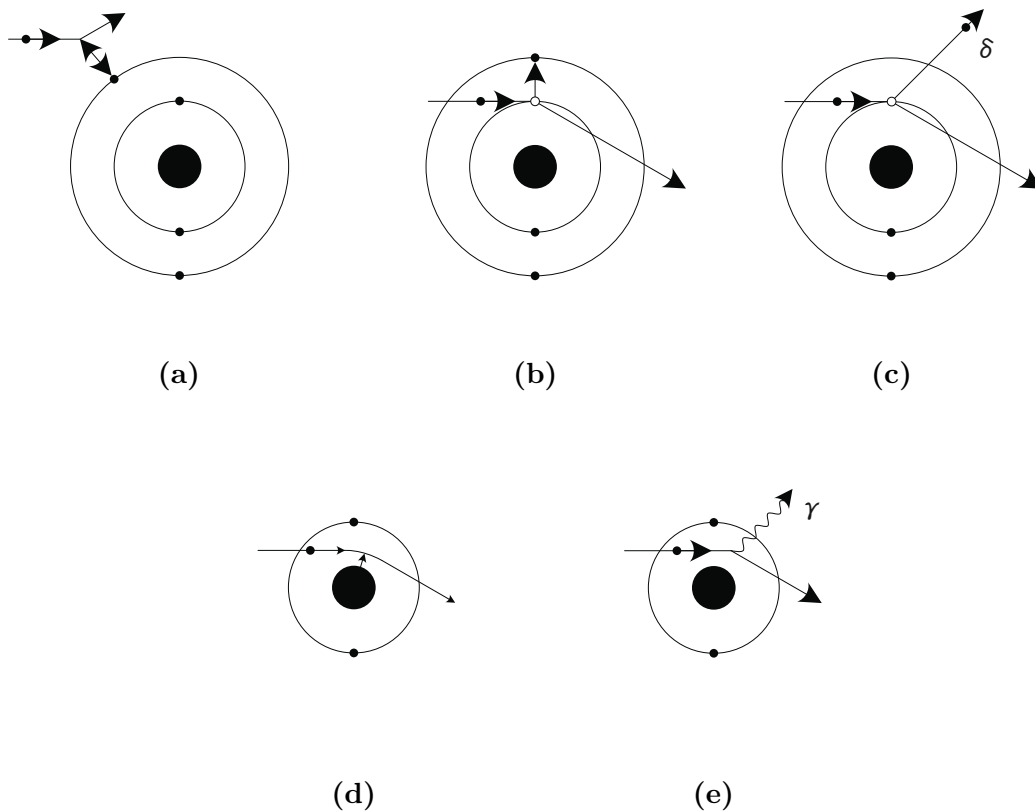


Figure 2.8: Schematic view of the interactions between the electron and matter. The atom shell's interactions are shown on the top, and the interactions with the atomic nucleus are shown below. The processes are **(a)** elastic scattering, **(b)** excitation of a shell-electron, **(c)** ionization, **(d)** elastic Coulomb-scattering, and **(e)** inelastic scattering (Bremsstrahlung). Figure adapted from [50].

2.2.1.2 Scattering at an Atomic Nucleus

In the first approximation, charged particles like electrons are scattered according to the Rutherford cross-section at the nuclei's Coulomb field (Figure 2.8d). The Rutherford cross-section is as follows [51]:

$$\frac{d\sigma_R}{d\Omega} = z^2 Z^2 \alpha \hbar \frac{1}{\beta^2 p^2} \frac{1}{4 \sin^4(\theta/2)} \quad (2.16)$$

Here, θ is the scattering angle relative to the incoming electron's direction, z is the charge of the scattered particle, Z the atomic number of the nucleus, α the fine-structure constant, \hbar the Planck constant, β the velocity of the scattered particle as the ratio of the speed of light, and p the momentum. The quadratic dependence on the atomic number is due to the coherent scattering at the entire nucleus. In comparison, ionization processes are incoherent and proportional to the atomic number since the contributions of the shell's individual electrons have to be summed up. Compared to scattering processes with the shell's electrons, the transfer of momentum and, therefore, energy is small for light projectiles. For energies much higher than the rest mass of the electron, the scattering processes with the nucleus dominate over scattering processes with electrons in the atomic shell. [38] The maximum electron energy used throughout this thesis is 300 keV and, therefore, the energy loss is dominated by scattering processes at the atomic shell.

2.2.1.3 Bremsstrahlung

Charged particles can also release energy by being deflected by a field. This is done by the emission of photons. This radiation is called Bremsstrahlung. In the detector volume, the field that mainly causes the deflection is the Coulomb field of an atomic nucleus. [39]

The process is schematically shown in Figure 2.8e. In silicon, Bremsstrahlung is the dominating process for electrons with a primary energy higher than approximately 40 MeV [52].

The radiation power is proportional to E/m^2 in the relativistic limit [53]. A quantum mechanical approximation of the emitted spectrum can be found in [54]. The directional distribution depends on the thickness of the detector volume and the primary particle energy. For low primary energies, the maximum of the emission intensity is around 60 to 90 degrees to the direction of the incoming particle and is radially symmetric. The higher the energy, the more the radiation is directed forward. For highly relativistic electrons, the Bremsstrahlung intensity distribution is sharply focused in forward direction [55].

For low primary energies, the thicker the material, the more important is the influence of the Bremsstrahlung's spatial diffusion. The diffusion leads to a more isotropic emission of Bremsstrahlung [56]. For the simulation in Chapter 4, the model by Berger and Seltzer is used [57].

2.2.1.4 Non-Ionising Energy Loss

Non-ionizing energy loss is the energy deposition due to a permanent displacement damage. Displacement damage could be generated by a so-called primary knocked-on atom, which means an atom is displaced from its lattice place due to irradiation. [58]

The displaced atom and its vacancy in the lattice are called Frenkel pair [38]. The displaced atom could have enough energy to migrate inside the lattice and displace other atoms of the lattice during further collisions. This displacement damage changes the bulk characteristics of the detector, which causes degradation called radiation damages. The non-ionizing energy loss due to Coulomb scattering on the nuclei can be described with the Wentzel–Molière differential cross-section. [59]

2.2.1.5 Multiple Scattering

A charged particle usually scatters many times in the detector material. The overall deflection angle of this multiple scattering process is the sum over the individual processes. Molière [60], [61] and Lewis [62] describe the angular distribution for a finite number of scattering processes. Lewis also describes the moments of the spatial distribution. The description by Molière can be approximated by the Gaussian distribution for small angles. The probability is larger than the probability of a pure Gaussian distribution for larger angles.

2.2.2 Mean Energy Loss

The Bethe Bloch formula describes the mean energy loss per track length of the charged primary particle as a function of its kinetic energy E_{kin} [63, 64]:

$$-\frac{dE}{dx} = \frac{z^2 e^4}{4\pi \epsilon_0^2 m_e c^2} \frac{N_A \rho Z}{A} \beta^{-2} \ln \left(1.166 \frac{E_{\text{kin}}}{I} \right) \quad (2.17)$$

Here, $\beta = v/c$ is the ratio between the velocity and the speed of light c , z the charge number of the primary particle, e the elementary charge, N_A the Avogadro constant, ϵ_0 the vacuum permittivity, m_e the rest mass of the electron, and c the speed of light. The material-dependent properties are the density ρ , the atomic number Z , the atomic weight A , and the mean excitation

potential I . For silicon, the mean excitation energy is $I = 174$ eV [65]. The relativistic relationship between particle velocity v and energy E with m_0 being the rest mass is as follows [39]:

$$v = c \sqrt{1 - \left(\frac{m_0 c^2}{E + m_0 c^2} \right)^2} \quad (2.18)$$

A detailed derivation of the units can be found in Appendix A.3. For energies below 1 keV, the Bethe Bloch formula fails, especially for light particles like electrons. A semi-empirical correction by Joy et al. [66] fixes the energy losses for small energies modifying the mean excitation potential (Figure 2.9):

$$I' = \frac{I}{1 + k \frac{I}{E}} \quad (2.19)$$

Substituting eq. 2.19 into the Bethe Bloch formula 2.17 results in an approximation for lower energies and electrons [66]:

$$-\frac{dE}{dx} = \frac{z^2 e^4}{4\pi \epsilon_0^2 m_e c^2} \frac{N_A \rho Z}{A} \beta^{-2} \ln \left(1.166 \frac{E_{\text{kin}} + kI}{I} \right) \quad (2.20)$$

The typical value for electrons in silicon is $k = 0.822$ [66]. This modified model shows a good agreement with experimental data [67].

2.2.2.1 Path Length

The path length is the distance a charged particle travels until it has deposited all its energy. In the continuous slowing down approximation, this length l can be calculated by integrating the inverse of the energy loss per unit way length with respect to the energy:

$$l = \int_0^{E_{\text{kin}}} \frac{dx}{dE} dE = l(E_{\text{min}}) + \int_{E_{\text{min}}}^{E_{\text{kin}}} \frac{dx}{dE} dE \quad (2.21)$$

For the continuous slowing down approximation, it is assumed that the charged particle deposits its energy continuously to the detector material exactly with the mean energy deposition. Since energy deposition is a statistical process, only the mean path length and the path length of individual particle tracks can vary. A practical approximation for the lower integration limit E_{min} is 200 eV. For silicon applies $l(E_{\text{min}} = 200 \text{ eV}) < 2 \text{ nm} \approx 0$. Figure 2.10 shows the integrated length as a function of the previous energy. For energies above

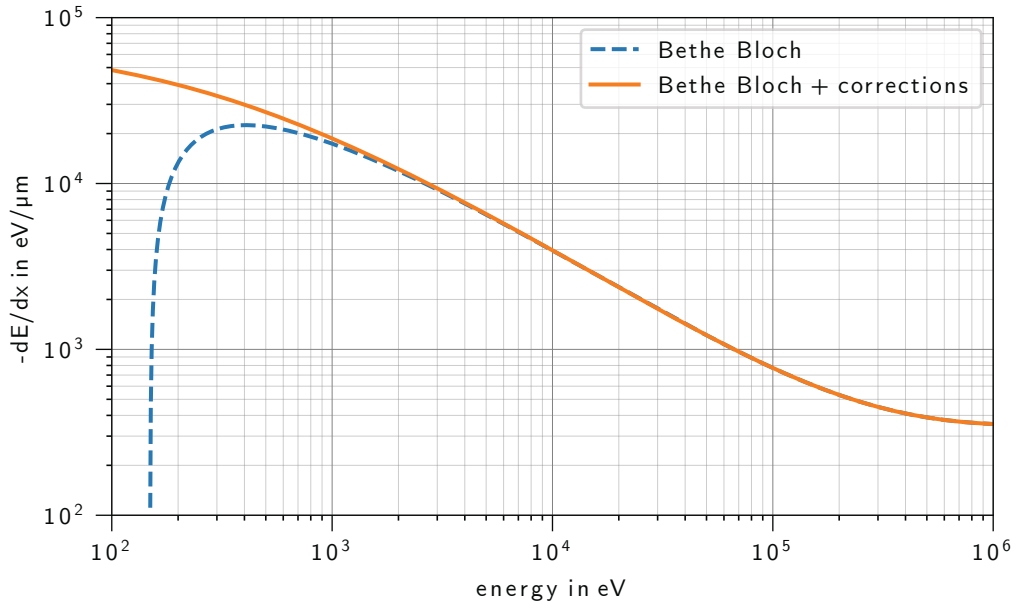


Figure 2.9: Energy deposition as a function of the energy. The dashed blue line shows the energy deposition according to the Bethe Bloch formula. The orange line shows the Bethe Bloch formula plus the correction for electrons with low energies.

300 keV, the mean length of the tracks obtained from the Monte Carlo simulation is slightly longer than the Bethe Bloch formula's predicted value with the corrections. The deviation is always smaller than ten percent of the length and neglectable compared to the statistical spread of the length.

Due to the multiple scattering process, the path length of an individual charged particle is always longer or at least of the same length as the particle's penetrating depth into the detector volume.

The energy deposition along the track of the particle is not constant and is known as Bragg curve. The highest energy deposition most likely is at the end of the particle track because the interaction cross-section increases as the energy of the charged particle decreases. Especially for higher primary energies, this behavior plays an important role in reconstructing the point of entry (PoE).

2.2.2.2 Landau Distribution

The above Section describes the mean energy loss per track length. The process of energy transfer is a randomly distributed process. This leads to a fluctuation

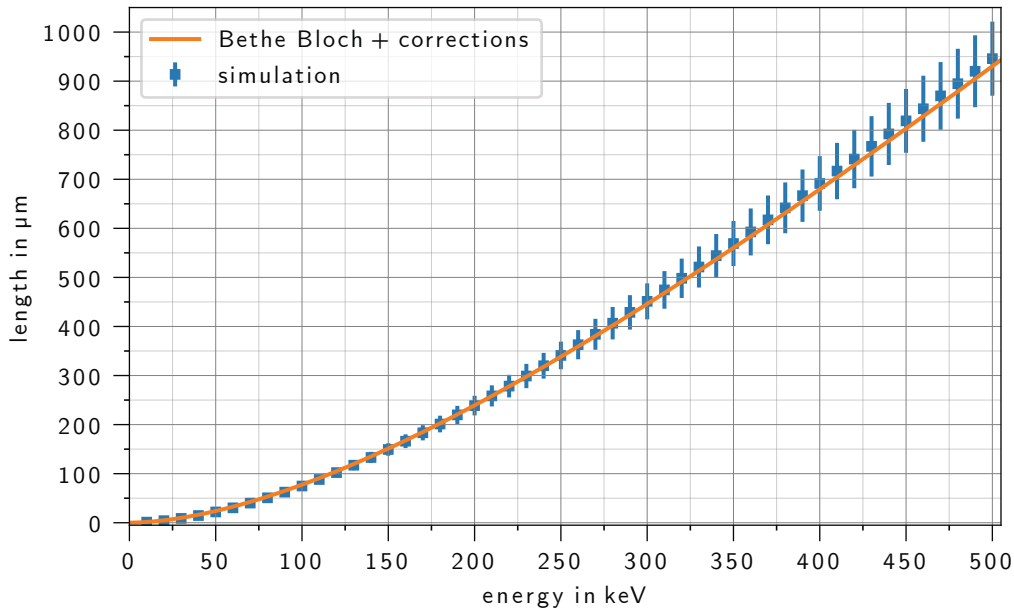


Figure 2.10: Trajectory length for electrons as a function of the primary energy in silicon. The orange line shows the theoretical result obtained from the Bethe Bloch formula with the correction for electrons. The blue dots indicate the results obtained from the Monte Carlo simulation with GEANT4 (Chapter 4). The error bars indicate the spread of the simulated events. The standard deviation is used to determine the spread.

in the amount of energy loss. The distribution of the energy loss f_L passing a thin absorber is described by the Landau distribution [68]. The Landau distribution function can be found in Appendix A.4. Its shape resembles a Gaussian distribution with a long tail at higher energies. This tail results from the small number of individual collisions, each with a small probability of transferring comparatively large amounts of energy. Theoretically, the upper limit is at infinite energies, while the energy deposited by an incoming particle cannot exceed its own primary energy.

2.2.2.3 Minimum Ionizing Particles

The mean energy loss decreases approximately proportional to $1/v^2$ with increasing velocity of the particle. This decrease can be explained by the decreasing time of interaction per path length for faster particles. For relativistic velocities, the stopping power increases again due to several effects: First, the secondary particle production must be considered at higher energies. Second,

the Lorentz factor increases with higher velocity, which increases the transferable energy. Furthermore, the electric field of charged particles close to the speed of light is deformed due to the Lorentz contraction. This deformation leads to a higher transverse field, increasing the interaction probability between the primary particle and an electron in the atomic shell. This effect cannot become arbitrarily large since the particle's field is shielded by the detector material's atomic nuclei. The size of the effect depends on the material properties, e.g., the density. [38]

The minimum of the mean energy loss can be found by derivation of the Bethe Bloch formula and is at $\beta \approx 0.95$. Particles with velocities around this minimum are called minimum ionizing particles. [38]

2.3 Atomic Relaxation

Atomic relaxation can be induced by any interaction that leaves the atom in an excited state. Excited state in this context means a vacancy in the inner shell of an atom.

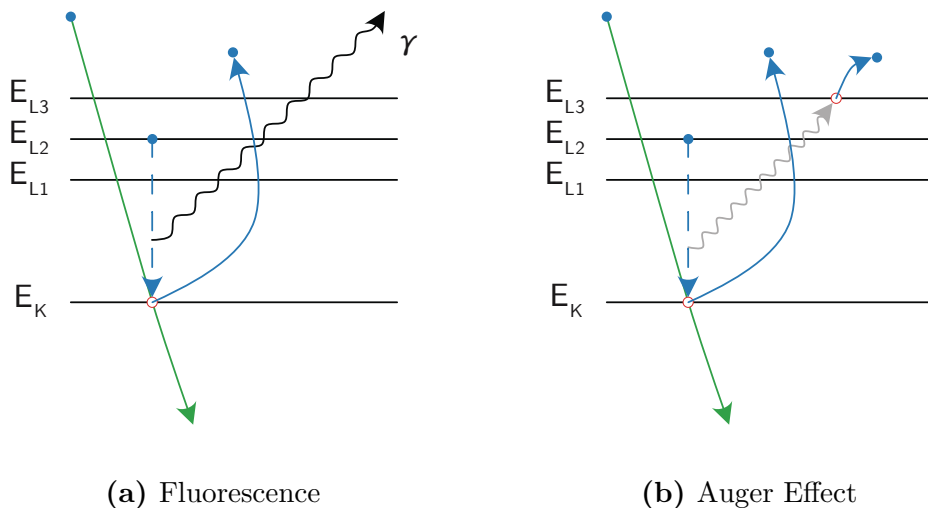


Figure 2.11: Atomic relaxation. The horizontal lines indicate the discrete energy levels of the atom. The primary particle is shown in green. Vacancies are shown as red circles and electrons as blue dots. The black sinuous line represents a photon. Figure adapted from [69].

2.3.1 Fluorescence

A vacancy in one of the inner shells of an atom is filled by an electron of a higher shell (Figure 2.11a). The binding energy of the higher shell is lower than the binding energy of a lower shell. During this transition, this energy difference is emitted as a photon. Depending on the element and shell number of the initial and final state, the energy difference is in the range of a few eV up to tens of keV [50]. The photon's energy is characteristic of the element, and the radiation is also called characteristic X-ray.¹ A special case is when the photon escapes from the sensitive detector volume and, thus, the energy of the emitted photon is not detected. The caused peak in the spectrum is called photo escape peak. The energy of this peak is the energy of the primary particle minus the energy of the characteristic X-ray. The probability of releasing energy by emission of an X-ray photon increases with the atomic number. [39]

2.3.2 Auger Effect

The complementary process to fluorescence is the Auger effect (Figure 2.11b). Again, a vacancy in one of the inner shells of an atom is filled by an electron of a higher shell. However, the excess energy is transferred to a second electron in a higher shell for the Auger effect. The second electron is ejected and called Auger electron. [38] In Figure 2.11b, the Auger electron's energy is given as [38]:

$$E_{\text{Auger}} = (E_K - E_{L2}) - E_{L3} \quad (2.22)$$

Auger electrons have an extremely short range because of their low energy of a few eV up to 3 keV [40]. Therefore, the Auger electron more likely deposits its energy in the active detector volume as characteristic X-ray. In contrast to the characteristic X-ray emission, the emission of Auger electrons decreases with higher atomic numbers. [38]

¹It is common use not to refer to X-ray emission, which is introduced by an electron as fluorescence [10]. However, the underlying effect is the same.

Chapter 3

Pixelated Semiconductor Detectors

In this Chapter, the principle of semiconductor detectors is introduced. This thesis's measurements are performed with pn charge-coupled devices (pnCCD). Therefore, the following introduction focuses on silicon as detector material. The further sections describe signal processing and data acquisition. In the last Section of this Chapter, signal processing is introduced separately for integrating and counting mode.

3.1 Semiconductors as Detector Material

Semiconductors are characterized by a small band gap from around 0.1 eV to 4 eV between the valence band and conduction band at room temperature[70]. Valence electrons are bound to individual atoms and can not contribute to an electrical current, whereas electrons in the conduction band can freely move within the atomic lattice. The energetic location of the valence band is below the conduction band. Due to the band gap, a valence band electron must be excited to enter the conduction band and contributes to the conductivity. This excitation can, e.g., be due to thermal or external irradiation. An atom in a pure silicon crystal has four neighbors. To each neighbor, a single covalent bond is built. If a bond is broken, an electron-hole pair is formed. The free electron moves to the conduction band and contributes to the conductivity. After a certain time, it can recombine again with the hole and relax into the valence band. [6]

An electron from a neighbor bond can fill the broken bond and create a hole in the new location. The hole can move through the crystal and, therefore, contribute to the valence band conductivity. The hole can move until it re-

combines with an electron from the conducting band. [6]

A more detailed discussion of the band structure can be found in [71].

3.1.1 Doping

A semiconductor without any significant impurity atoms is called an intrinsic semiconductor. Replacing some of the lattice atoms with other atoms can modify the properties of the semiconductor. The replacement process of silicon atoms by impurity atoms is called doping, and the result is an extrinsic semiconductor. Depending on the oxidation state of the foreign atoms, doping can either be n-doping (main group V elements of the periodic system of elements) or be p-doping (main group III elements). For silicon, the common elements for n-doping are phosphorus or arsenic and, for p-doping, boron. [6]

Figure 3.1 shows a substrate material of silicon. Figure 3.1a shows the lattice of an intrinsic semiconductor. In Figure 3.1b, one silicon atom is replaced by a phosphorus atom resulting in an n-type semiconductor, and in Figure 3.1c, one silicon atom is replaced by a boron atom resulting in a p-type semiconductor. The oxidation state of silicon is IV which means each silicon atom has four valence band electrons to form bonds. Replacing a silicon atom by an atom of the group V leads to an additional unbound electron, as shown in Figure 3.1b. This electron can be easily excited into the conduction band at room temperature. In contrast, atoms of group III have only three valence band electrons. This leads to an unsatisfied bond to a neighbor silicon atom (Figure 3.1c). The there missing electron can be filled with an electron from a neighbor bond, creating a hole. Elements of main group V act like electron donors and create an additional filled state close to the conduction band. Elements of main group III act like electron acceptors and create an additional empty state close to the valence band. [6]

3.1.1.1 The p-n Junction

A p-n junction is a spatial connection on crystal level between a p-doped and an n-doped region. The additional electrons from the n-doped area diffuse into the p-doped area and recombine with holes from the p-doped area. This creates a depletion zone that is free of mobile charge carriers, as shown schematically in the upper illustration of Figure 3.2. The depletion region is also called space charge region because this zone contains charges coming from the fixed unmovable doping atoms in the silicon lattice left during the diffusion process. These ions create an electric field that counteracts the diffusion current. In thermal equilibrium, the drift current generated by the electric field and the diffusion current cancel each other. [6]

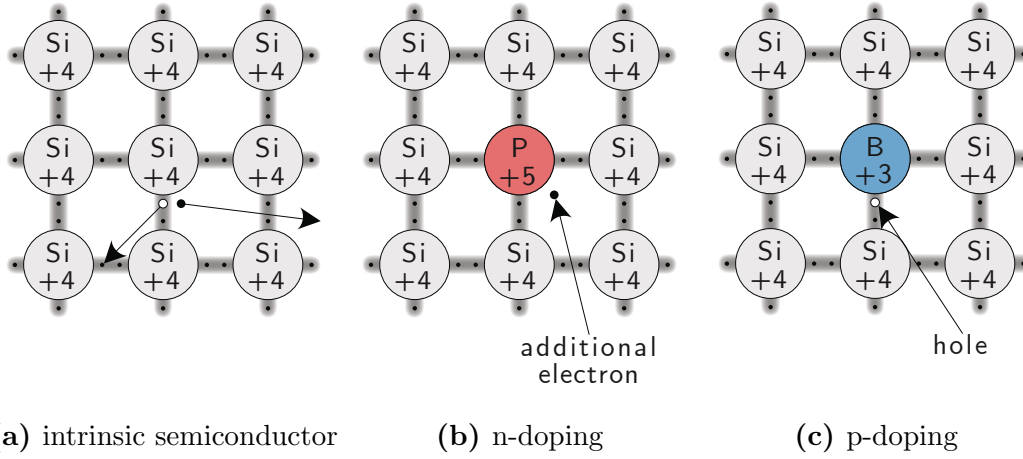


Figure 3.1: Doping of silicon. **(a)** In an intrinsic semiconductor for conduction, an electron has to be excited from the valence band to the conduction band. The left hole also contributes to the conduction. **(b)** N-doping adds an additional electron that can be excited easily to the conduction band. **(c)** P-doping creates an unsatisfied bond.

The charge density and the corresponding electrical field are shown in Figure 3.2.

The band structure of the p-n junction, including the size of the space charge region, can be explained with the concept of the Fermi potential. The Fermi potential refers to the energy at which the occupation probability of a state is one-half. Because of the additional filled states, the Fermi potential of n-doped semiconductors is closer to the conduction band. For the p-doped semiconductor, it is closer to the valence band. When bringing the n-doped and the p-doped regions together, both regions' Fermi potentials match in thermal equilibrium. This leads to a shift in the band structure. [6]

A diode formed by a p-n junction has two electrical contacts on each of the doped parts. The width of the space charge region of a diode depends on the doping level and the applied voltage. [72]

The space charge region becomes larger with higher external reverse bias voltage and can be calculated to [6]:

$$d = \sqrt{\frac{2\epsilon_0\epsilon_{\text{Si}}(N_A + N_D)}{eN_A N_D}}(V_{\text{bi}} - V) \quad (3.1)$$

Here, $\epsilon_0\epsilon_{\text{Si}}$ is the relative permittivity times the vacuum permittivity, N_A the acceptor doping concentration, N_D the donor doping concentration, e the ele-

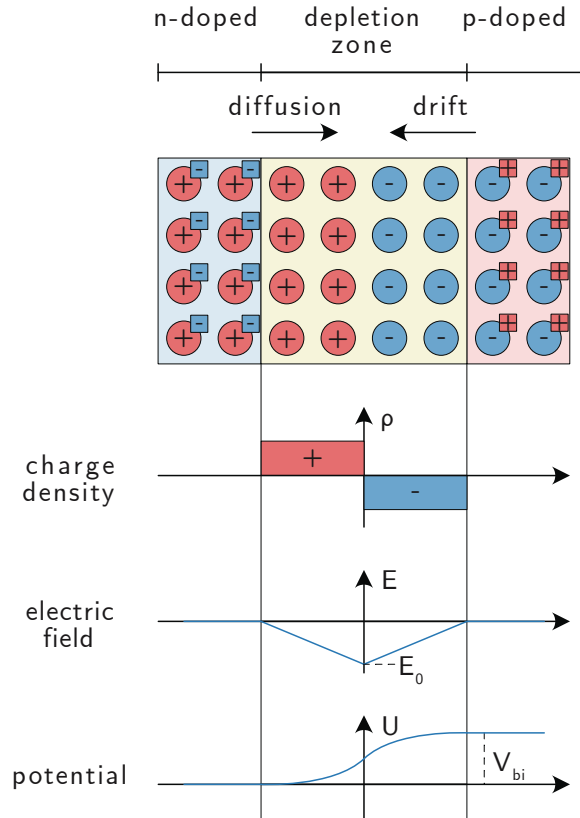


Figure 3.2: The unbiased p-n junction in the one-dimensional model. From top to bottom, the atomic structure's schematic drawing, the charge density, the electric field, and the electrostatic potential are shown. In the depletion zone, the charge density zone is unequal zero, which results in an electric field. The electric field is the negative gradient of the electrostatic potential and can be calculated using the Poisson equation. Figure adapted from [6].

mentary charge, V the external bias voltage which is applied to the diode, and V_{Bi} is the built-in voltage [6]:

$$V_{Bi} = \frac{k_B T}{e} \ln \left(\frac{N_A N_D}{n_i^2} \right) \quad (3.2)$$

Here, $n_i \approx 1.45 \cdot 10^{10} \text{cm}^{-3}$ is the intrinsic charge carrier density at room temperature, k_B is the Boltzmann constant, and T is the temperature. For asymmetric doping, the space charge region expands more into the region of the lower doping dose. [6]

3.1.2 Sideward Depletion

Sideward depletion allows it to deplete the same volume of the bulk with a quarter of the bias voltage, and it allows to decouple the detector area from the readout anode and, therefore, reduce the readout capacity. [73]

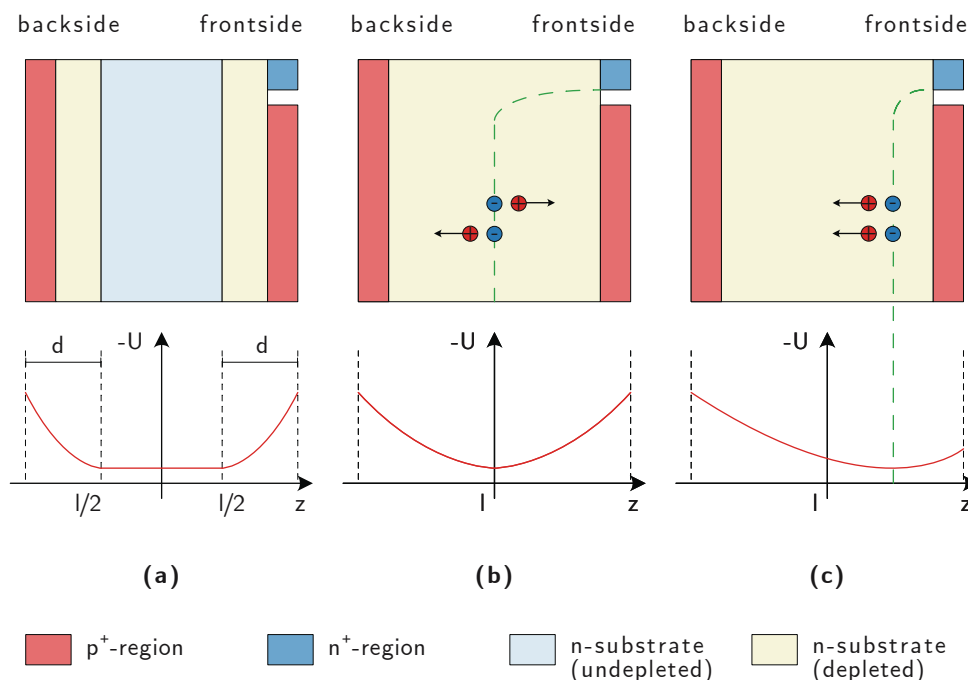


Figure 3.3: Sideward depletion. A schematic cut through the wafer is shown. The wafer surfaces are right and left at the p⁺-junctions. **(a)** Unbiased setup. **(b)** Fully depleted setup. The applied bias voltages at the device surfaces are symmetric. This leads to a parabolic potential with a potential minimum for electrons (dashed green line) centered at the center of the device. **(c)** The potential minimum for electrons (dashed green line) can be shifted towards a device surface by applying asymmetric bias voltages. Figure adapted from [73].

Figure 3.3 shows the principle of sideward depletion. The frontside is structured into a small n-doped and a large p-doped part. The p-doped backside and the p-doped frontside are biased with negative voltages. The depletion zones expand from both sides into the low n-doped bulk, as shown in Figure 3.3. When fully depleted, the potential is parabolic from each side with a potential minimum for electrons in the middle (Figure 3.3b). The position of

the potential minimum for electrons can be moved by applying the voltages to the front and back asymmetrically (Figure 3.3c). [73]

The holes move to the p^+ contact. The electrons move towards the anode by an additional applied drift field that superpose the depletion field. [73]

3.2 Electron-Hole Pair Generation

Chapter 2 describes the energy deposition in the detector volume. In this Section, the statistics behind the conversion from this energy deposition to electron-hole pairs is described. The generation of electron-hole pairs is subject to Fano statistics. For a given energy deposition E , the mean number of created electron-hole pairs is given as the energy deposition divided by the mean energy ω , which is needed for the creation of an electron-hole pair [6]:

$$N = \frac{E}{\omega} \quad (3.3)$$

For room temperature in silicon, the pair creation energy is $\omega = 3.68$ eV [46]. The mean energy ω slightly increases for lower temperatures. Since silicon is an indirect semiconductor, the average energy required to generate an electron-hole pair is much higher than the band gap energy, which is approximately 1.12 eV, with the exact value depending on the doping state at room temperature [45]. The variance of the number of generated charges is

$$\langle \Delta N^2 \rangle = FN = F \frac{E}{\omega} \quad (3.4)$$

with the Fano factor $F = 0.115$ at room temperature [45, 74]. The phase space for generating electron-hole pairs becomes with decreasing energy smaller. Therefore, the generation of electron-hole pairs is correlated and not uncorrelated. The Fano factor absorbs the deviation from an uncorrelated process. [74]

For a perfect detector, the Fano factor and the ratio of scattered particles exiting the active detector volume, hence only depositing a certain ratio of energy in the detector, are the energy resolution's physical limits and the peak to valley ratio in the energy spectrum. A detailed explanation of the physical limits can be found in Appendix A.2.

The electron-hole pairs created by incident photons or charged particles are free charge carriers in the depletion region and are efficiently separated due to the local electric field. This efficient separation is necessary to minimize the recombination of the electron-hole pairs. The electrons are called signal

electrons in the following. Many signal electrons created by high-energy irradiation form a charge cloud that drifts to the anode due to the electric field. During this drift, the size of the charge cloud becomes larger by diffusion and repulsion. [6]

Both effects are described in detail in Section 4.2.4.

3.3 The pnCCD

For benchmark measurements, a detector based on the concept of pnCCD is used. The methods presented in this thesis are by default independent of the detector's operating principle and are transferable to all pixelated semiconductor detectors for imaging without further treatment.

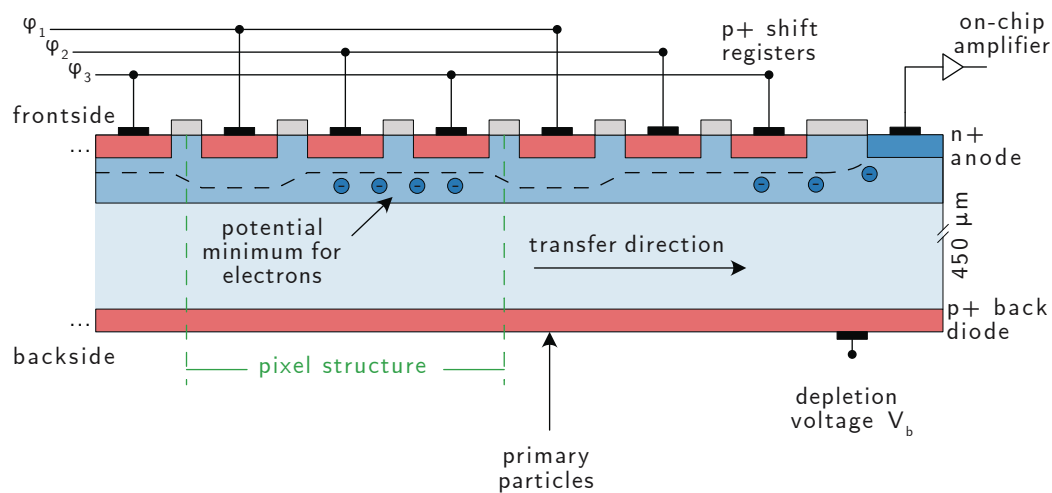


Figure 3.4: Schematic view of the pnCCD (not to scale). The sectional view is perpendicular to the shift register direction. The red areas denote p^+ -junctions, the dark-blue areas n^+ -junctions, and the light blue area the n-substrate. Three adjacent shift registers form one pixel. Figure adapted from [69].

In this Chapter, a short overview of the pnCCD is given. The structure and working principle of the pnCCD are described in many publications in detail [6, 75, 76].

Figure 3.4 shows the schematic view of a pnCCD. The pnCCD works on the principle of sideward depletion (Section 3.1.2). The detector has to be divided into sections, so-called pixels, to obtain spatial information. This structuring is achieved by dividing the p-doped frontside into strips,

so-called shift registers. Every third shift register is connected to the same voltage. This leads to three different supply voltages φ_1 , φ_2 , and φ_3 of the frontside. Applying a more negative voltage to one of the three shift register groups yields a local potential minimum for electrons under both other shift register groups. This leads to pixelation in one dimension of the detector. Typically, three neighboring shift registers form one pixel indicated by the green dashed line in Figure 3.4 (three-phase CCD). Additional n-doped stripes are implanted perpendicularly to the shift register leading to an attractive potential for electrons. This implantation avoids charge spreading below the shift registers and forms the second dimension of the pixelation and is called channel guide. [6]

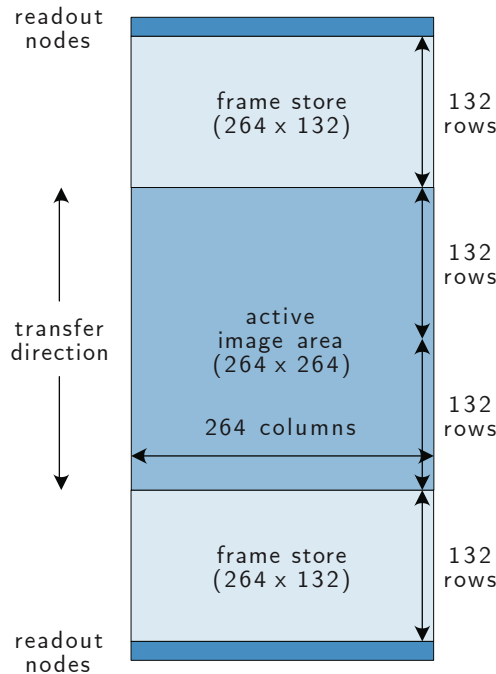


Figure 3.5: Schematic view of the pnCCD readout. The active image area is 264 times 264 pixels. The upper half of the active image area is transferred into the top frame store, and the lower half is transferred into the bottom frame store. At the readout nodes, one row of the top frame store and one row of the bottom frame store is read out parallel. Figure adapted from [69].

By switching the supply voltages with a reasonably defined sequence, the potential minimum for electrons and, therefore, the collected electrons can be transferred to the detector edge without mixing the electrons from the

individual minima. The row-by-row readout is achieved by an n-doped anode at the edge of the frontside and an on-chip amplifier based on field-effect transistor (FET) for each column. [6]

It can either be processed with an alternating sequence of shifting and readout or the total image can be fast transferred to a radiation-shielded area called frame store. During the acquisition of the next frame, the frame store can then be readout in the same way as the sensitive detector area. The frame store reduces so-called out-of-time events. Such events occur when an incident photon or charged particle hits the detector when the electrons are shifted to the edges. These events are then assigned to a row that is further away from the readout edge than the actual PoE. Using a frame store requires additional supply voltages to shift the potential minima in the frame store area independent of the remaining area of the pnCCD. [6]

Figure 3.5 depicts the readout scheme used in this work with two frame stores at the top and the bottom of the pnCCD. The two frame stores double the readout speed compared to using only one frame store. The device is irradiated from the backside. The thickness of the sensor is 450 μm .

3.4 Conversion from Signal Electrons to a Digital Signal

The readout chain and signal processing of pixelated detectors can be divided into two concepts. Figure 3.6 schematically illustrates a random energy deposition in one pixel as a function of the time and the corresponding response of both detector systems. In integrating mode (Figure 3.6b), the amount of energy deposited in each pixel is converted into an analog voltage that is digitized. Therefore, the analog voltage represents the amount of deposited energy. In counting mode (Figure 3.6c), individual particle hits are identified and processed to a binary output. The counting mode is typically used for monochromatic irradiation.

3.4.1 Integrating Detector Mode

In this detector mode, the total amount of energy deposition during the integration time is integrated for each pixel. The integration time is the inverse of the frame rate. If two or more particles deposit their energy during this time in the same pixel, the deposited energy can not be assigned to the individual particles, as the system only perceives the overall energy deposition. The pnCCD used in this work is connected to the input of

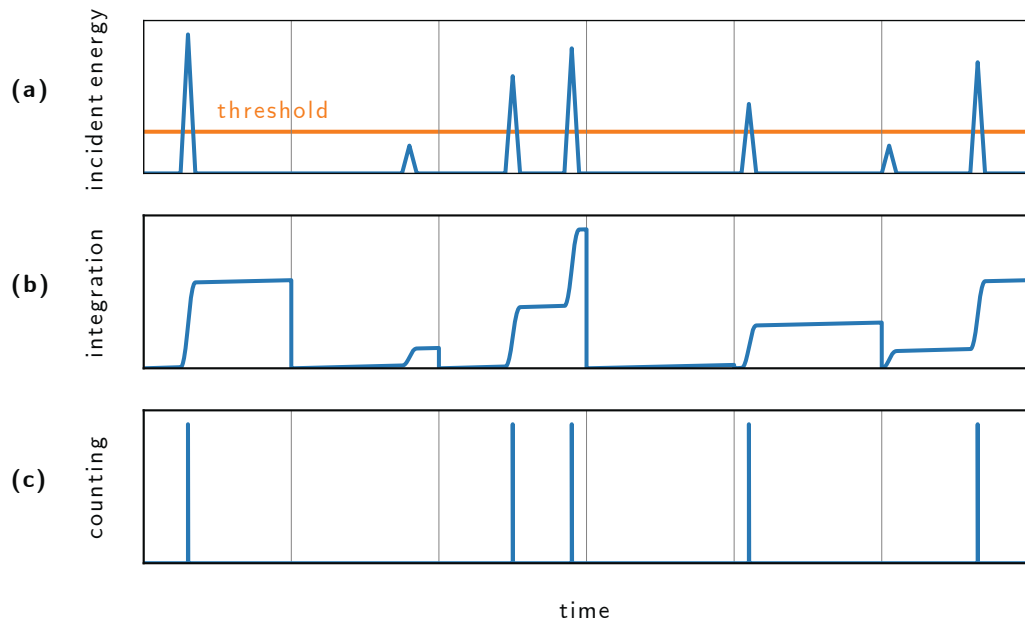


Figure 3.6: Comparison of counting and integrating mode. The graphic schematically shows the concepts of one pixel in counting mode and the same pixel in integration mode. The x-axis is for all plots the time. **(a)** shows the energy deposition as a function of time. **(b)** shows the signal in the pixel. Note that due to the frame-based readout, the signal's shape is not recorded but only the signal at the time when the detector is readout (vertical gray line). **(c)** shows the output of a counting system with one threshold. The system's output is a binary signal whenever the energy deposition exceeds a certain value. The information about the magnitude of the energy deposition is lost.

a readout application-specific integrated circuit (ASIC) via wire bonding. Here, the used ASIC is called CAMEX (**C**MOS **a**mplifier and **m**ultiplexer) [77]. The CAMEX has one amplification stage for each column of the detector so that the pixels of every row are readout simultaneously. Each amplification stage has one output for each column. An on-chip multiplexer creates a time-coded single output from the multiple lines. A synchronized analog-to-digital converter (ADC) digitizes the multiplexer's output and sends it to a computer. The computer sorts the values obtained from the ADC into frames and stores them. Each quadrant of the active detector area has its own CAMEX [69].

At this point, it should be noted that each of the described steps can be parallelized to achieve higher frame rates. For other detector systems, for

example, based on DEPFETs [78], in the fastest cases, each pixel has its own readout chain, and the signal processing and data acquisition are fully parallelized.

3.4.2 Counting Detector Mode

In counting detector mode, the systems count the number of times an energy deposition in a pixel exceeds a certain threshold. In general, this happens pixel by pixel. It can, for example, be achieved with a discriminator, which compares the signal with an adjustable voltage or digitally. Using two or more thresholds enables the possibility of energy windowing. The more thresholds levels are used, the finer the differentiation of a counted event into the different energy windows can be achieved. [79]

Modern counting systems such as the Timepix3 [80] also store the timestamps of the counts. Hybrid counting systems discriminate the signal not pixel by pixel but count more complex event structures. Therefore, individual event clusters are identified at the time they reach the detector. [81].

State-of-the-art systems, moreover, measure the actual energy deposition of the pixels above the threshold level. These energy depositions are directly processed and analyzed by a high-performance computing system or more problem-specific hardware such as a field-programmable gate array (FPGA) or ASICs. The additional information of the energy depositions can be used to determine the event's position in the subpixel regime. [82]

Figure 3.7 shows an energy deposition in the different representations. Figure 3.7a shows the actual PoE of the primary particle, Figure 3.7b the charge cloud after the drift and diffusion, and Figure 3.7c the energy deposition in the pixel structure. This energy deposition is the output of a detector system in integration mode. Figure 3.7d shows the output obtained in counting detector mode, and Figure 3.7e shows the output in a 2×2 subpixel regime.

For monochromatic radiation, counting requires that the individual particles' tracks do not overlap and, therefore, requires a fast readout electronic or a low particle rate [83].

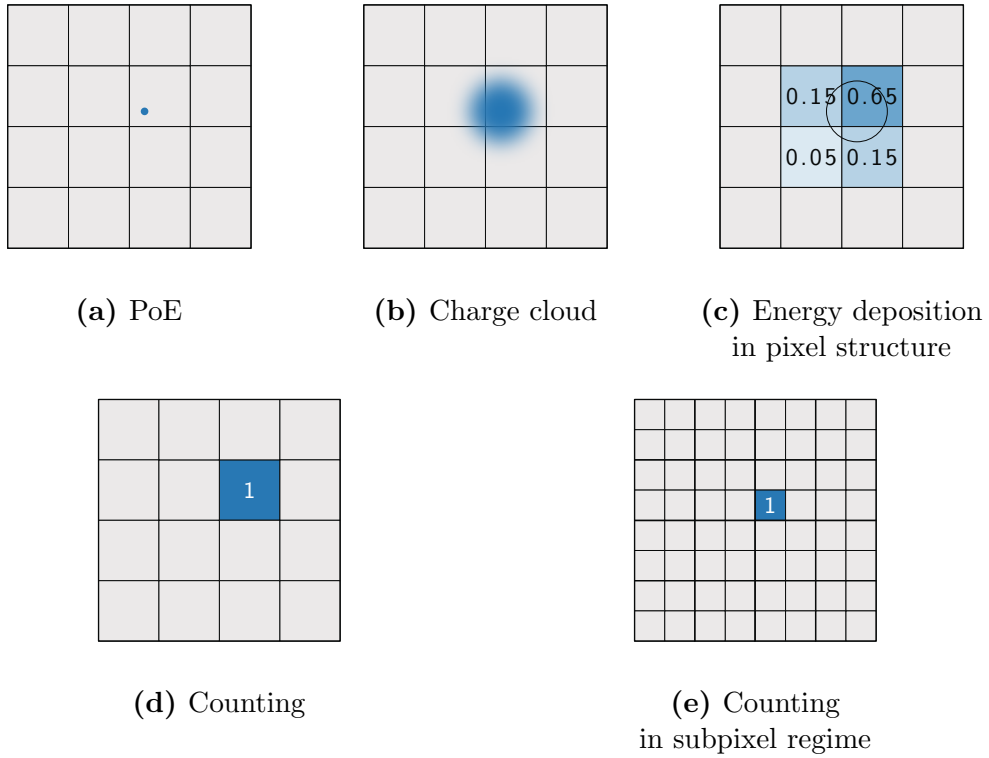


Figure 3.7: Energy deposition in counting detector mode. A detector section of four by four pixels is shown. The blue dot indicates the PoE (a). An expanded charge cloud arrives in the pixel structure (b). See Section 4.2.4 for details. The amount of charges is proportional to the energy deposition. The energy deposition is detected for each pixel. The values show the fraction of the primary energy in the individual pixels. In integrating mode, these energy depositions are integrated over the frame time and stored (c). In counting mode, only a binary response is stored. This response can either be in the regime of the physical pixel (d) or in the subpixel regime (e). Figure adapted from [83].

Chapter 4

Simulation of the Energy Deposition by Photons and Electrons

The trajectory of light charged particles with nonrelativistic energies in the detector material can be approximated as a random walk. Every scattering process of the particle in the detector material is a statistical process that follows well-known physical laws, and the trajectory of the particle is created by a chain of such scattering processes (Section 2.2.1.5). This behavior makes it impossible to predict the trajectory of a single particle precisely. Directly measuring the trajectory with a pixelated detector is also not possible because the detector only measures the accumulated deposited energy in the individual pixels. This means the three-dimensional trajectory is projected onto two dimensions and, subsequently, binned in the two-dimensional pixel grid.

Hence, Monte Carlo simulations of many individual particles are performed to investigate the photons' and electrons' physical behavior in silicon and the corresponding detector response. The obtained knowledge can be used to optimize parameters such as the pixel size or the detector's backside voltage and to understand the particles' behavior in the detector volume. The results of the simulations have been carefully verified. No significant deviations between simulation and measurement have been observed in the energy range relevant for our study. The results of these Monte Carlo simulations are required to train the neural networks presented in this work.

Basically, these simulations are divided into two parts: First, the energy deposition of the particles, and second, the charge carrier creation, separation, and the formation of the pixel by pixel signal. For simulating the energy de-

position, the toolkit GEANT4 (version 10.5) [57] implemented in C++ [84] is used. The second part is implemented in Python 3 [85] and is constructed modularly. The modules correspond to the different physical effects. It is possible to save, respectively, load the output (energy deposition, signal electron distribution, frames) and the used parameters after each module.¹ This makes it possible to simulate, for example, different particle rates with exactly the same individual particle tracks.

The total simulation is implemented scalable. This means the degree of parallelization of the individual processes can be adapted to the used machine.

4.1 Energy Depositon

GEANT4 with the low energy electromagnetic library Livermore [88] is used to individually simulate each event's energy deposition as a function of the three space dimensions. Information such as, for example, the time, the physical process of the energy deposition, and the physical step length are optionally logged. The physical step length describes the path length the particle has traveled between two simulated energy depositions. The generated simulation data are sorted hierarchically. The top-level is a so-called run and has a unique run identification. A run contains several events. Each of these events has the same physical settings, such as energy or type of primary particle. Due to the possible creation of secondary particles, each event can have more than one particle track. The particle track with the id zero is the primary particle, and the secondary particles get their identification sorted by the time they are created. The PoE of the primary particle on the simulated detector surface always stays the same. The distribution over the detector is done in a later step. This procedure enables different hit patterns and particle rates with the same events.

The Livermore model is valid for electromagnetic processes down to 250 eV and, in principle, usable down to 10 eV with a lower accuracy. [88]

The relevant processes for the energy range used by the Monte Carlo simulation in this thesis are **ionization**, **bremsstrahlung**, and **scattering processes** for electrons (Section 2.2) and the **photoelectric effect**, **Compton scattering**, and **pair production** for photons (Section 2.1) [89]. In contrast to the standard physics list of GEANT4, which uses parameterized data of the cross-section, the Livermore model uses the energy dependent cross-sections'

¹The data handling and analysis are mainly done with the package pandas [86] and the package numpy [87].

tabulated data.¹ Relaxation processes such as atomic deexcitation such as fluorescence, Auger electron emission, including Auger cascades, and particle induced X-ray emission (PIXE) are also part of the Monte Carlo Simulation [92]. A detailed mathematical description of the processes can be found in [93].

The particle trajectories and energy depositions are simulated step by step. For all processes, the probability of an interaction is calculated and subsequently used in combination with a random number generator to randomly decide which interaction process is occurring. GEANT4 uses a condensed algorithm for the scattering processes resulting in a multiple scattering theory developed by Lewis [59]. This theory describes the individual scattering events by using Legendre polynomials [62]. After several scattering events, the analytic solution of the deflection angle is calculated using the additive properties of the Legendre polynomials. The Lewis theory provides the moments of the spatial displacement distribution. A simulation of the single scattering processes would also be possible and is a bit more accurate but takes a lot more computational power, which is not suitable for higher numbers of simulated particles and does not increase the precision for our purpose.

The simulation's important parameters are the range cut, the low energy edge, and the lowest electron energy. The range cut decides which secondary particles are simulated. It is set to 5 nm. This means only secondary particles which travel at least 5 nm are simulated, otherwise their energy is directly deposited. The influence of a range cut of 5 nm is neglectable since the size of the charge cloud arriving in the pixel structure and the size of the pixel structure of the detector is much larger. The low energy edge is set to 250 eV and overwrites the range cut if the range cut is equivalent to an energy below the low energy edge. The lowest electron energy is set to 250 eV and forces full energy deposition at one step independently for electron energies below this value. The mean path length of an electron with 250 eV is approximated with 5 nm. The mean free path for photons with an energy of 250 eV in silicon is approximately 100 nm. Both lengths are much smaller than the pixel structures. For photons, the mean free path for this range of energies is dominated by the photoelectric effect, which means the complete energy is deposited and not only a fraction as, for example, by the Compton scattering process.

A detailed analysis of the performance of the GEANT4 simulation using

¹The used data sets are the evaluated atomic data library (EDAL) [90], the evaluated electrons data library (EEDL) [90], the evaluated photons data library (EPDL97, EPICS2014) [41, 42], and the binding energies are taken from Scofield [91].

the Livermore model of electrons and photons in semiconductors can be found, for example, in [94]. In summary, the GEANT4 simulations and experimental data have an overall good agreement.

4.2 Conversion and Transport of the Generated Signal Charges

Conversion and transport of the generated signal charges are implemented in separate modules. The advantage of this modular design is to save computational power and time for sweeps over different parameter sets. The run identification assigned in the GEANT4 simulation to events created within the same physical settings will be maintained and supplemented by a conversion identification. The simulation parameters for these identifications are automatically saved in a database structure. The results of the conversion and the transport modules are in good agreement with the measurements performed by Kimmel [95].

4.2.1 Thermalization of Excited Silicon Atoms

The first step in the conversion and transport module is called thermalization. In this step, the conversion from the incident particle energy deposition to electron-hole pairs is performed. This conversion is described by the Fano statistics (Section 3.2). The distribution of the number of electron-hole pairs N generated from the energy deposit ΔE can be calculated as [96]:

$$p(N(\Delta E)) = \begin{cases} \frac{1}{\sqrt{2\pi F \frac{\Delta E}{w}}} \exp \left[-\frac{1}{2} \left(\frac{N - \frac{\Delta E}{w}}{\sqrt{F \frac{\Delta E}{w}}} \right)^2 \right], & \text{if } N \geq 0 \\ 0, & \text{otherwise} \end{cases} \quad (4.1)$$

Here, F is the Fano factor, and w is the pair creation energy. The actual number of the generated electron-hole pairs for each energy deposition is drawn from these distributions.

4.2.2 Charge Collection Efficiency

With the charge collection efficiency, effects such as charge loss or recombination are modeled. This recombination mainly occurs at the detector's entrance window because only there, the electric field is too weak to separate the electrons and holes fast enough. Moreover, dangling bonds at the Si-SiO₂ interface lead to trapping and recombination. The charge collection efficiency (CCE) is

a value between zero and one and describes the ratio of collected to generated charge carriers. Goto [97] describes the CCE with an exponential approach phenomenologically:

$$\text{CCE}(z) = 1 - (1 - R) \cdot e^{-\frac{v_s \cdot z}{D}} \quad (4.2)$$

Here, v_s is the electron saturation velocity, D the diffusion constant, and R a parameter to describe effects by a finite recombination velocity. The approximation for the CCE can be rewritten as a parameterized exponential function [97]:

$$\text{CCE}(z) = 1 - \gamma \cdot e^{-\frac{z}{\tau}} \quad (4.3)$$

To take the charge carrier recombination's statistical characteristic into account, this can be implemented as a binomial distribution. The binomial distribution counts the number of successes (charge carriers do not recombine) in a series of similar and independent experiments, each with exactly two possible results (recombine or not recombine). This binomial distribution's sample size is the number of charge carriers generated by the point-like energy deposition, and the probability is the CCE. Typical parameter values for pnCDDs are $\gamma = 0.09$ and $\tau = 0.1 \mu\text{m}$ [98].

Since τ is very small, the CCE effects are only relevant for energy depositions near the surface. Therefore, the correction is only relevant for photons. The effect of recombination near the entrance window can be neglected for energetic electrons penetrating the detector much deeper.

4.2.3 Clustering of Signal Charges

During the drift process, electrostatic repulsion occurs. The strength of this repulsion depends on the amount of charge per volume. The simulation of the forces and the movements of the distributed charges is an n-body problem. In general, no analytical solution for an n-body problem exists for $n > 2$ [99]. The problem could be solved numerically, but this is computationally intensive and not necessary in the given context.

Up to this point, each energy deposition obtained from GEANT4 is treated separately. In this step, for each primary particle, the generated electrons, which are spatially less separated than $1 \mu\text{m}$, are clustered.

Clustering is a technique to group similar samples that are more similar to samples in the same group than samples in another group. Similarity in this particular case means spatially less distanced.

An efficient approach is to cluster charges close to each other to a point charge. This point charge's spatial position is the center of mass of the

electrons that contribute to the cluster. The amount of charge of the point charge is the sum of the charges of the contributing electrons. The charge density of the electrons drifting away from the center of mass over time due to their electric field is radially symmetric and can be analytically calculated.

In the used clustering algorithm¹, each sample is initially treated as an individual cluster. At each iteration, close clusters are merged. After each merge, the proximity matrix, which contains all distances between all clusters in pairs, is calculated. The termination criterion of the iterations can be either a distance threshold or the number of final clusters. In the Monte Carlo simulation, the distance threshold is used. The algorithm's output is an identifier for each sample that assigns it to a cluster.

The parameters of the used clustering algorithm are the affinity, the distance threshold, and the linkage. The affinity is the Euclidean distance between two samples. The distance threshold determines the distance above which clusters will not be merged. Maximum linkage means the distance between two clusters is obtained from the maximum distance between all electrons of those two clusters. In the following, those clusters are named charge clouds. The number of contributing electrons and the weighted centroid are calculated for each charge cloud.

4.2.4 Drift of Signal Charges

Because of the detector's negative backside contact, the charge clouds drift in the direction of the pixel structure located on the opposite side. During this drift, the charge clouds expand by diffusion and electrostatic repulsion. Therefore, even a point-like energy deposition can be spread over several pixels.

The integrated signal in the pixels is the two-dimensional projection of these expanded charge clouds. The charge cloud's size mainly depends on the drift time, temperature, and the number of charge carriers in this charge cloud. In addition, the drift time depends on the point of the conversion, the charge separation depth, which denotes the depth at which the charges are separated into several neighboring pixels, and the applied electric field.

The electric field in the detector volume can be approximated by solving the one-dimensional Poisson equation. The pnCCD is assumed as a p-n-p structure [95]. In this approximation, the pixel structure at the detector's

¹Technically, this is done by a hierarchical agglomerative clustering algorithm [100] which belongs to the unsupervised machine learning methods. In particular, the implementation of scikit-learn [101] is used.

frontside is neglected to simplify the internal electric field structure. This approximation can be made because the pixel structure plays a subordinate role for the field and the potential in the bulk volume. The additional implantation for the pixel structure changes the electric field significantly only near the frontside. However, the electric fields in this region are not relevant because the charge separation depth parameter introduces their influence in the simulation process. The charge separation depth in this simulation is set to $17\ \mu\text{m}$ away from the frontside [98].

In a second approximation, the thin p^+ -doped contacts at the front and back sides are neglected [96]. This leads to a structure with two regions: The bulk, which is the original substrate of the silicon wafer and weakly n^- -doped, and an n -doped region near the frontside. The simplified schematic of the pnCCD bulk is shown in Figure 4.1.

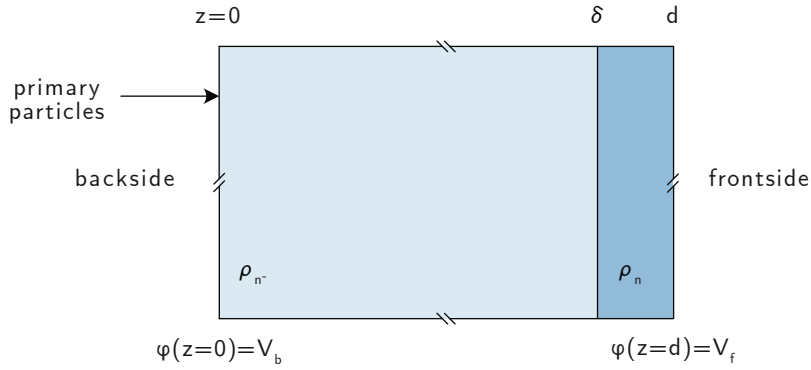


Figure 4.1: Schematic cut through the approximated model used to calculate the electric field and electrostatic potential of the pnCCD. The drawing is not to scale. The p -regions of the p - n - p junction are neglected in this model. The coordinate system is chosen in a way that the backside is at $z = 0$, and the frontside is at $z = d$. The transition between the bulk with a doping concentration of ρ_{n^-} and the doped region with doping concentration ρ_n is at $z = \delta$. The boundary conditions for the potential φ are shown at the bottom.

The electric field in the z -direction and the potential in the detector volume are given by the following equations [95]:

$$E_z(z) = -2A_i \cdot z - B_i \quad (4.4)$$

$$\varphi(z) = A_i \cdot z^2 + B_i \cdot z + C_i \quad (4.5)$$

The x - and y -component of the electric field is zero. The coefficients A_i , B_i , and

C_i depend on the thickness d of the detector, the thickness δ of the n^- -doped region, and the doping concentrations ρ_{n^-} and ρ_n :

$$A_i = \begin{cases} A_{n^-} = -\frac{\rho_{n^-}}{2\epsilon_0\epsilon_{Si}} & 0 < z \leq \delta \\ A_n = -\frac{\rho_n}{2\epsilon_0\epsilon_{Si}} & \delta \leq z < d \end{cases} \quad (4.6)$$

$$B_i = \begin{cases} B_{n^-} = \frac{V_f - V_b}{d} + \left(\frac{\delta^2}{2d} - \delta\right) \cdot \frac{\rho_{n^-} - \rho_{n^-}}{2\epsilon_0\epsilon_{Si}} + \frac{d \cdot \rho_n}{2\epsilon_0\epsilon_{Si}} & 0 < z \leq \delta \\ B_n = \frac{V_f - V_b}{d} + \frac{\delta^2 \cdot (\rho_{n^-} - \rho_{n^-})}{2d \cdot \epsilon_0\epsilon_{Si}} + \frac{d \cdot \rho_n}{2\epsilon_0\epsilon_{Si}} & \delta \leq z < d \end{cases} \quad (4.7)$$

$$C_i = \begin{cases} C_{n^-} = V_b & 0 < z \leq \delta \\ C_n = V_b + \frac{\delta^2 \cdot (\rho_{n^-} - \rho_{n^-})}{2\epsilon_0\epsilon_{Si}} & \delta \leq z < d \end{cases} \quad (4.8)$$

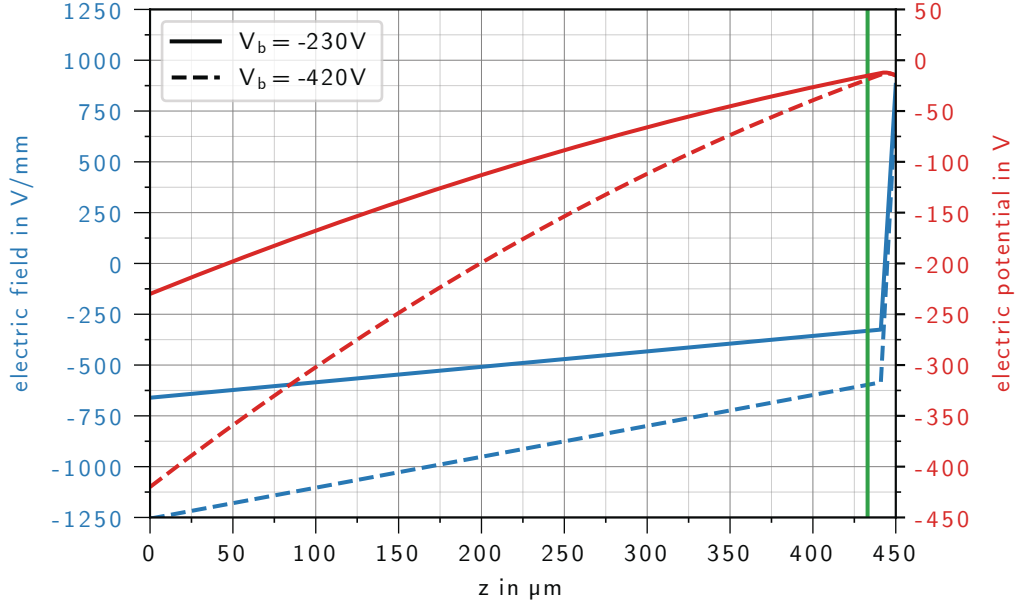


Figure 4.2: Electric field in blue and electric potential in red in a pnCCD with a thickness of $450 \mu\text{m}$ as a function of z . The underlying model and the parameters are described in the text. The charge separation depth is shown in green.

The origin of the coordinate system is the backside, and the z -direction points into the silicon bulk. A detailed derivation of these equations by solving the Poisson equation can be found in Appendix A.5. In this thesis, the following parameters are used: The thickness d is $450 \mu\text{m}$, the thickness of the n -doped bulk δ is $441 \mu\text{m}$ in this approximation, the frontside voltage V_f and

the backside voltage V_b are application-dependent chosen, the space density of the donors in the bulk ρ_{n-} is $5 \cdot 10^{11} \text{cm}^{-3}$, and the space density ρ_n is $9 \cdot 10^{13} \text{cm}^{-3}$ [96]. Figure 4.2 shows the resulting electric field and electrostatic potential for a frontside voltage of -15 V and two different backside voltages.

For realistic parameters, the electric field vanishes in the n -doped region at:

$$z_0 = -\frac{B_n}{2A_n} \quad (4.9)$$

In the case of standard operations parameters ($V_b = -230 \text{ V}$) z_0 results in $442 \mu\text{m}$. For a different backside voltage setting ($V_b = -420 \text{ V}$) z_0 reaches $445 \mu\text{m}$.

The drift time t_d of an electron at an initial position z_i to a final position z_f is the integral over the inverse drift velocity v_d , which depends on the electric field $E(z)$ and the mobility $\mu(E)$ [6]:

$$t_d = \int_{z_i}^{z_f} \frac{1}{v_d(z)} dz = - \int_{z_i}^{z_f} \frac{1}{\mu(E) \cdot E_z(z)} dz \quad (4.10)$$

The final position z_f is the charge separation depth. At this position, the charge is confined by the pixelated structure and cannot drift to other pixels anymore. Figure 4.3 shows the drift time as a function of the initial position z_i . The drift time increases with increasing temperature and decreases with increasing the absolute value of the backside voltage.

If the mobility was constant, high electric fields and arbitrarily high drift speeds would be possible. Since this is not physical, there is a maximum drift velocity called saturation velocity. In the simulation, this is realized by temperature and electric field-dependent mobility. The mobility is implemented by an empirical model [102], [103], [104]:

$$\mu(E, T) = \frac{1.42 \cdot 10^9 T^{-2.42}}{\left[1 + (E/1.01 \cdot T^{1.55})^{2.57 \cdot 10^{-2} T^{0.66}}\right]^{1/(2.57 \cdot 10^{-2} T^{0.66})}} \quad (4.11)$$

In eq. 4.11, the units of the temperature T is K, of the electric field E is V cm^{-1} , and the mobility of the electrons μ is $\text{cm}^2 \text{V}^{-1} \text{s}^{-1}$.

During the drift time (eq. 4.10), the charge cloud expands because of diffusion and repulsion. Drift, diffusion, and electrostatic repulsion are described

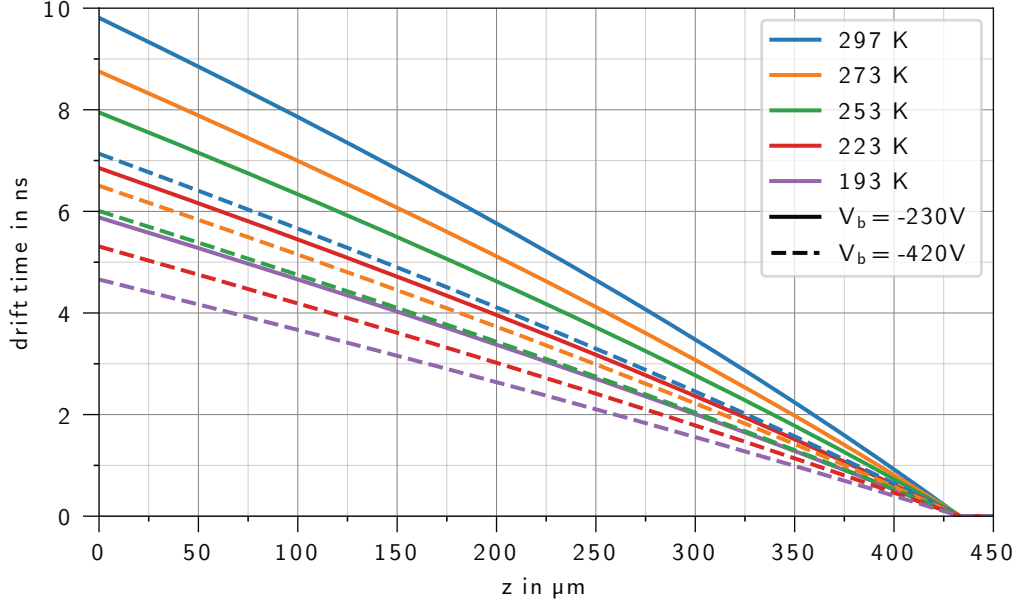


Figure 4.3: Drift time of the charge cloud into the pixel structure as a function of the depth of the energy deposition for different temperatures. The charge separation depth is assumed at $z_f = 433 \mu\text{m}$ [98].

with the continuity equation[6]:

$$\frac{\partial n_e}{\partial t} = \mu \nabla n_e \mathbf{E} + \mu n_e \nabla \mathbf{E} + D_e \Delta n_e \quad (4.12)$$

Here, n_e is the electron density, μ the mobility, \mathbf{E} the electric field, and $D_e = (k_B \cdot T/e) \cdot \mu$ the diffusion coefficient. An analytic equation can be found if either diffusion or electrostatic repulsion is neglected [105]:

- **Diffusion:** If electrostatic repulsion is neglected, the continuity equation modifies to

$$\frac{\partial n_e}{\partial t} = \mu \nabla n_e \mathbf{E} + D_e \Delta n_e \quad (4.13)$$

which can be solved by a spherical charge cloud with a Gaussian shape, which moves with the drift velocity $\mu \cdot E_z$ along the z -axis [105]. The diffusion does not depend on the number of charge carriers in the charge cloud. The diffusion into the x - y -plane is perpendicular to the electric field and, therefore, is free of external forces. The standard deviation of

the Gaussian-shaped charge cloud is given by [105]:

$$\sigma_{\text{diff}}(t) = \sqrt{2D_e(t + t_0)} = \sqrt{2\frac{k_B T}{e}\mu(t + t_0)} \quad (4.14)$$

Here, t is the drift time, and t_0 takes the charge cloud's size at the initial time into account. The projection into the x-y-plane, which is perpendicular to the electric field and parallel to the pixel structure, is a two-dimensional Gaussian distribution with the same standard deviation. [96, 95]

- **Electrostatic repulsion:** In the other case, the diffusion is neglected. If one integrates the continuity equation in spherical coordinates θ and ϕ and assuming a zero external electric field, one obtains the following equation for the charge Q [105]:

$$-\frac{\partial Q(r, t)}{\partial t} = Q(t, r) \cdot \frac{\partial Q(r, t)}{\partial r} \cdot \frac{\mu}{r^2 \cdot 4\pi\epsilon_0\epsilon_{\text{Si}}} \quad (4.15)$$

This partial differential equation can be solved by a sphere with a uniform charge density of the radius

$$R(t) = \sqrt[3]{3\frac{\mu e}{4\pi\epsilon_0\epsilon_{\text{Si}}}Nt} \quad (4.16)$$

with N being the number of electrons in the charge cloud [105]. The projection $\tilde{n}(r, t)$ into the x-y-plane, which is perpendicular to the electric field and parallel to the pixel structure, can be calculated by integrating over the z dimension of the spherical density [96]:

$$\tilde{n}(r, t) = \begin{cases} \frac{3N}{2\pi} \frac{\sqrt{R(t)^2 - r^2}}{R^3} & r \leq R(t) \\ 0, & \text{otherwise} \end{cases} \quad (4.17)$$

It is a disk with sharp edges and an increasing charge density from the edge to the center. The radius of the charge cloud depends on the number of electrons in this cloud. The maximum of the charge density is at $r = 0$ with a value of $3N/(2\pi R^2)$. The projected density is half of its maximum at a radius of $r = \sqrt{3}R/2$. This relation is used to approximate the repulsion by a Gaussian distribution, based on the (half) full width at half maximum (FWHM) and the maximal amplitude. This leads to the

following sigma of the Gaussian distribution[96]:

$$\sigma_{\text{rep}}(t) = \frac{\sqrt{3}}{2\sqrt{2\ln 2}} \sqrt[3]{\frac{2\mu e}{4\pi\epsilon_0\epsilon_{\text{Si}}} Nt} \quad (4.18)$$

The Gaussian approximation of the overall expansion of the charge cloud σ_{radius} is the quadratic addition of the individual effects:

$$\sigma_{\text{radius}} = \sqrt{\sigma_{\text{diff}}^2 + \sigma_{\text{rep}}^2} \quad (4.19)$$

The charge cloud σ_{radius} depends on the electric field, the number of charges in the cloud, and the drift time.¹

The Gaussian approximation of the electrostatic repulsion is valid for electrons with higher primary energies ($\gtrsim 100$ keV) since the dominant effect on the charge cloud is due to the primary electron's energy deposition along the ionization trajectories itself. The energy deposition is spatially very limited for lower energetic electrons and photons. Therefore, the uncertainty of the Gaussian approximation is more significant. The deviation from a pure Gaussian shape is the largest for short drift times and charge clouds with many charges.

Figure 4.4 shows the cutaway drawing through the center of the charge cloud for a pure diffusion, a pure repulsion, and the convolution of both effects. The charge cloud is radially symmetric around the PoE. The Gaussian approximation overestimates the size of the charge cloud. The kurtosis of the convolution of the repulsion and diffusion is negative. The normal distribution has a kurtosis of zero. [106]

The charge cloud's normalized shape can be interpreted as a probability distribution for the position of individual electrons that contribute to the charge cloud. For the simulation of the charge cloud propagation, this fact is used, and the probability distribution is used to distribute the individual electrons around the center of mass of the charge cloud.

The charge cloud's charge distribution cannot be approximated as continuous distribution for a minimal energy deposition since the charges are quantized. As a consequence, the simulated result differs from the probability

¹The diffusion and the electrostatic repulsion are independent of the assumed mobility model because the mobility contributes inversely to the drift time.

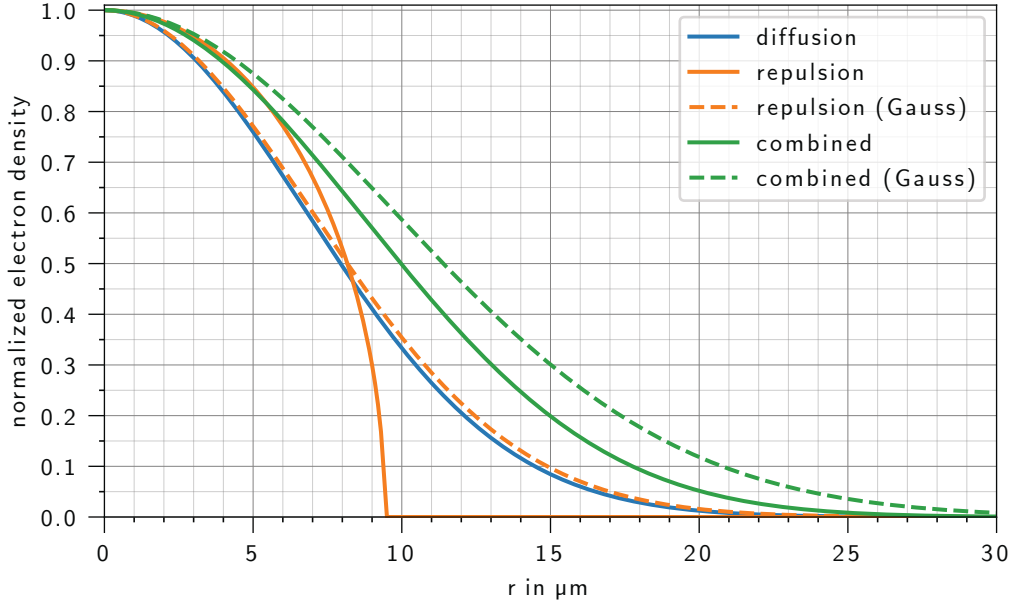


Figure 4.4: Shape of the charge cloud as a function of the radius r . The PoE is at $r = 0$. The different contributions to the expanded charge cloud are color-coded. The dashed lines show the Gaussian approximation. For the calculation, a characteristic X-ray photon emitted by copper ($E = 8048$ eV [40]) and the parameters $V_b = -230$ V, $V_f = -15$ V, and $T = 253$ K are assumed.

distribution, and the charge distribution is no longer point-symmetric.

4.2.5 Pixelation of Signal Charge Clouds

In this step, the charge clouds are collected and binned into the pixel structure for each event. The collection and binning can mathematically be described as integral over two spatial dimensions. The signal $S_{x,y}$ in the pixel whose center is located at (x, y) can be calculated with eq. 4.20 [107].

$$S_{x,y} = \int_{y-\Delta y/2}^{y+\Delta y/2} \int_{x-\Delta x/2}^{x+\Delta x/2} \rho(x', y') dx' dy' \quad (4.20)$$

Here, Δx and Δy are the pixel size in the x- and the y-dimension of the rectangular pixel, and $\rho(x, y)$ is the charge density. The integral turns into a sum over the discrete charges for the discrete charge density obtained from the drift submodule.

The PoE of the particles and, therefore, the relative origin of the charge density with respect to the limits of the pixel structure can be moved over the pixel. The PoE can either be randomly selected or systematically swept over a two-dimensional grid. After the binning into the pixels, the deposited charge per pixel is retransformed in an energy deposit. The conversion factor is typically the mean energy required to generate an electron-hole pair in silicon. In order to achieve maximum flexibility with the lowest possible computer effort, no frames or pattern pile-up events are generated at this point but in a later step.

4.3 Generation of Frames

This step of the conversion module provides the simulated frames. The affected pixels are obtained from the pixelation step. The input is the particle rate, the distribution of the particle rate, and spatial distribution of the PoEs. The spatial distribution can, for example, be related to shapes such as slanted edges, homogeneous illumination, a single point, or related to a specific experiment. The distribution of the rate can be constant or Poisson distributed.

In combination with the spatial distribution and the temporal distribution, a hit list containing all necessary information of the time and the PoE is generated and stored. This hit list is used to distribute the events to the frames. If two or more events contribute to the same pixel in a frame, these energy depositions are summed up, and a so-called pattern pile-up event is created. For housekeeping, the shapes and the pixels which contribute to an event are stored.

For the counting results, the time, the row, and the column of the pixel which has an energy deposition are stored. Additionally, a different number of thresholds with different values can be applied. In such a case, not the energy deposition but the index of the threshold which has been crossed is stored. In this context, the counting result counts the pixels over the threshold.

4.4 Readout Noise

Additional to the uncertainties obtained from back-scattering or out-scattering effects and the Fano statistics in this step, an additional noise term can be added pixel by pixel. With this noise term, readout noise from the signal processing can be included in the simulation. This additional noise is Gaussian-shaped around zero. Its standard deviation can be indicated in units of equivalent noise charges. For the first time, in this step of the simulation process,

negative values can occur. This can be explained because no offset is simulated in the simulation. Therefore, the result behaves such as offset corrected data.

4.5 Results for Photons as Primary Particles

In the Monte Carlo simulation for the energy deposition used here, a beam of monochromatic photons hits a fully sensitive silicon detector with a thickness of $450\ \mu\text{m}$. Primary energies over a wide range between $0.5\ \text{keV}$ and $100\ \text{keV}$ were simulated. For each primary energy, ten million primary photons were simulated.

Figure 4.5 shows the energy deposition in the silicon for primary energies of $6\ \text{keV}$ and $8\ \text{keV}$. The x-axis is normalized to the energy of the primary photon to make the individual spectra comparable.

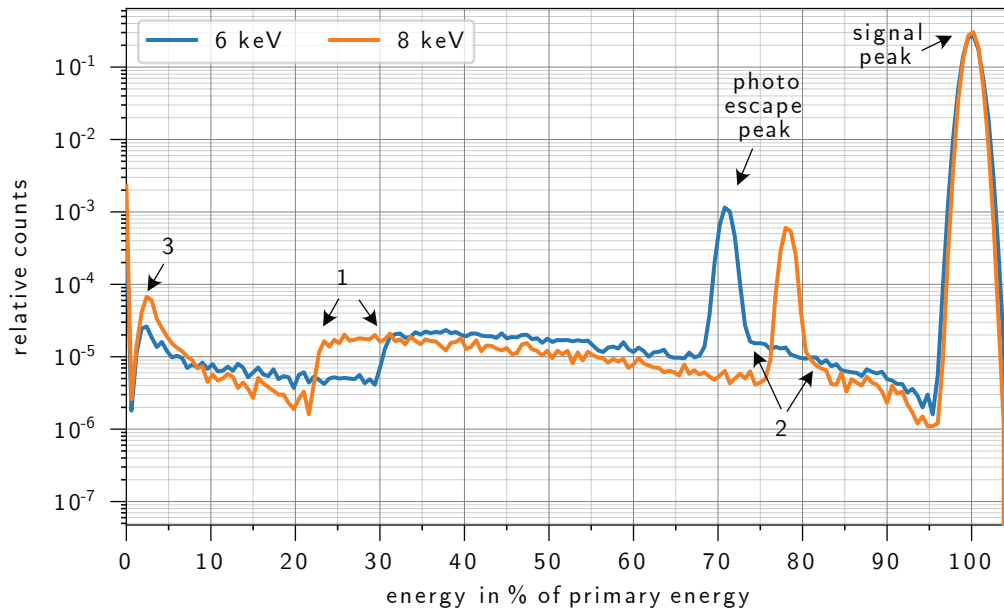


Figure 4.5: Simulated spectrum of deposited energy in silicon for photons. A beam of monochromatic photons hits a fully sensitive silicon detector with a thickness of $450\ \mu\text{m}$. The different colors indicate the primary energies.

The most prominent peak is at the primary energy itself. This peak is called signal peak in Figure 4.5. Here, the full energy of the primary photon is deposited in the silicon. The peak is not sharp but is widened by the ionization statistics described by Fano. The best achievable energy resolution for a given primary energy can be found in Appendix A.2.2.

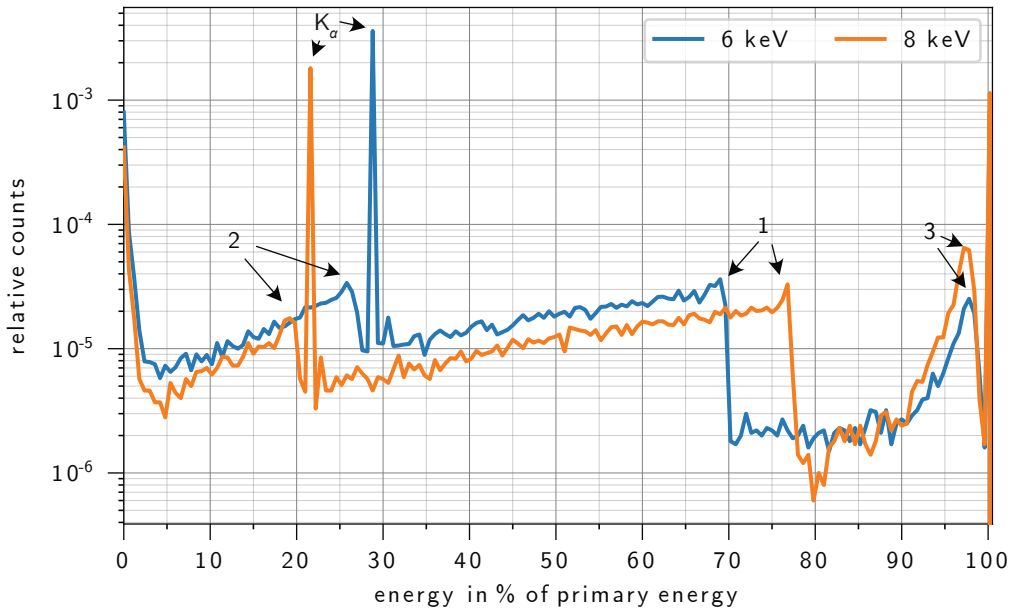


Figure 4.6: Energy distribution of the backscattered particles leaving the silicon. The different colors indicate the primary energies.

All spectra show the characteristic escape peak at the primary photon energy minus the energy of the escaping X-ray photon. The escape peak is produced when an electron of a higher shell relaxes into a created vacancy in the K-shell. The released energy is most probable in the form of a K_α X-ray photon which is generated and escapes the detector material. The energy of this photon is 1740 eV in silicon [40].

The other features of the spectrum highlighted with the numbers 1, 2, and 3 can be better understood by looking at the spectrum of the out-scattered particle leaving the detector volume. Since energy conservation applies, for each incoming photon, the deposited energy plus the escaping particle's energy has to be equal to the primary energy.

For low primary energies¹, the ratio of forward-scattered particles leaving the sensitive detector volume at the frontside can be neglected (Figure A.4 on page 215), and the energy distribution of the backscattered particles (Figure 4.6) mainly influences the spectrum. The backscattered particles are mainly electrons. An exception are the silicon characteristic X-ray photons at 1740 eV which cause the escape peak. The characteristic edges (1,2, and 3 in Figure 4.5) are caused by the energy-dependent cross-section (Figure 2.1 on page 14)

¹This energy range increases with the thickness of the sensitive silicon bulk and is for a thickness of 450 μm in the order of 15 keV.

and are explained by combining several effects:

The first effect is the photoelectric effect. The produced free electron from the K-shell has the energy of the primary energy minus the K-shell's binding energy (1839 eV [40]). This electron either escapes the detector without any additional energy deposition, which produces the edge 1 in Figure 4.6 or deposits a fraction of its energy. The more energy is deposited, the less likely it is for the electron to escape the detector. As a consequence, the energy distribution decreases from the edge towards lower energies.

The vacancy in the K-shell can be filled by emitting a characteristic K_α X-ray photon. This photon has not enough energy to perform a second photoelectric effect with a K-shell electron, but it has enough energy to perform a less probable photoelectric effect with an L-shell electron. The energy of the emitted electron is $1590 \text{ eV} = 1740 \text{ eV} - 150 \text{ eV}$ [40]. This electron can again either escape the detector without any additional energy deposition, which produces the edge 2 in Figure 4.6, deposit a fraction, or all of its energy. In the spectrum of the energy deposition, the edge is overlaid by the photo escape peak, but one can see it indirectly by comparing the levels of the valley at a slightly lower and a slightly higher energy than the escape peak.

The edge at energies close to the primary energy 3 in Figure 4.6 is generated by the photo escape peak of a characteristic L X-ray photon and an escaping electron from the L-shell. Its maximal energy is the primary energy minus the binding energy of the L-shell.

In this Section, a fully sensitive silicon bulk was assumed. Insensitive layers on the top of the sensitive silicon bulk lead to a deviation between a measured and the simulated spectrum. For details, see the work of Granato [98]. The insensitive layers are often called entrance windows. A finite entrance window smears out the silicon characteristic structures to a flat shelf. The level of the smearing depends on the properties of the entrance window. However, this deviation is neglectable since the reconstruction methods presented in this work's framework are mainly sensitive to the spatial charge distribution.

With higher primary energies, forward-scattered particles which escape the detector on the opposite side also play a more significant role. Therefore, the characteristics of the detector material are superimposed by characteristics from forward scattering.

Figure 4.7 shows the average number of pixels with an energy deposition by one primary X-ray photon. Diffusion and repulsion were taken into account. As expected, from the size of the charge cloud (Figure 4.19), the number of pixels containing an energy deposition first increases up to approximately

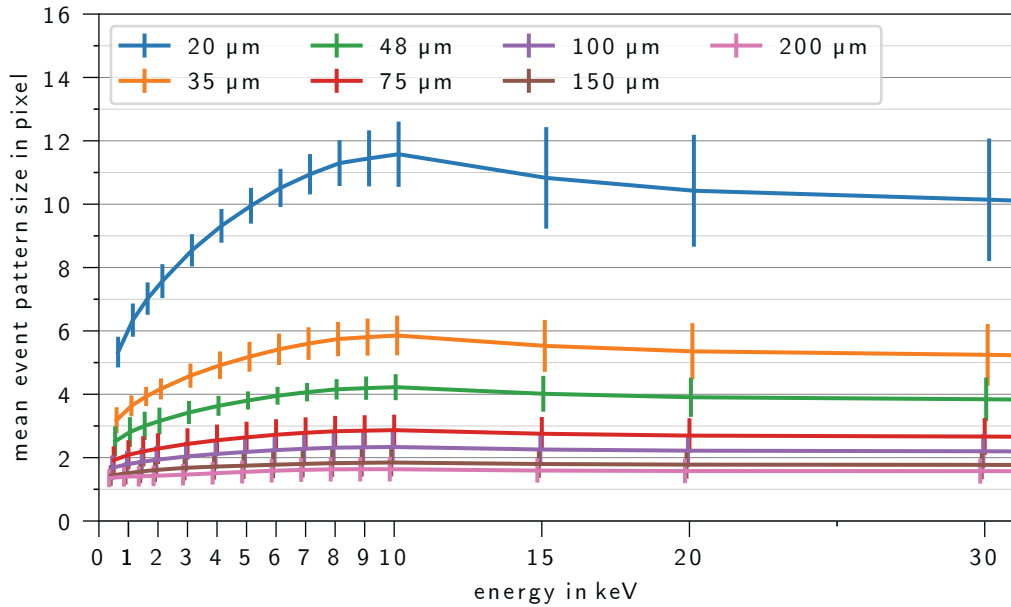


Figure 4.7: Averaged single event pattern size as a function of the primary energy. The different colors denote different sizes of the pixel structure. The error bars indicate the standard deviation over the used event patterns.

10 keV and subsequently drops again. Because the dimensions of the charge clouds are independent of the pixel size, the number of pixels containing an energy deposition decreases with increasing pixel size.

4.6 Results for Electrons as Primary Particles

In contrast to photons, electrons deposit their energy not point-like but energy-dependent by producing three-dimensional tracks in the detector volume. The energy deposition happens along the track. Figure 4.8 and Figure 4.9 show arbitrarily selected tracks for different primary energies. Figure 4.8 shows the projection into the x-z plane, and Figure 4.9 the projection into the x-y plane parallel to the detector surface. The colors in both Figures refer to the same individual tracks. The shapes of the tracks are stochastic and created by multiple scattering processes. However, due to the electron's initial momentum, the first scattering processes are preferably in a forward direction. Tracks of electrons with lower primary energies are more concentrated around the PoE.

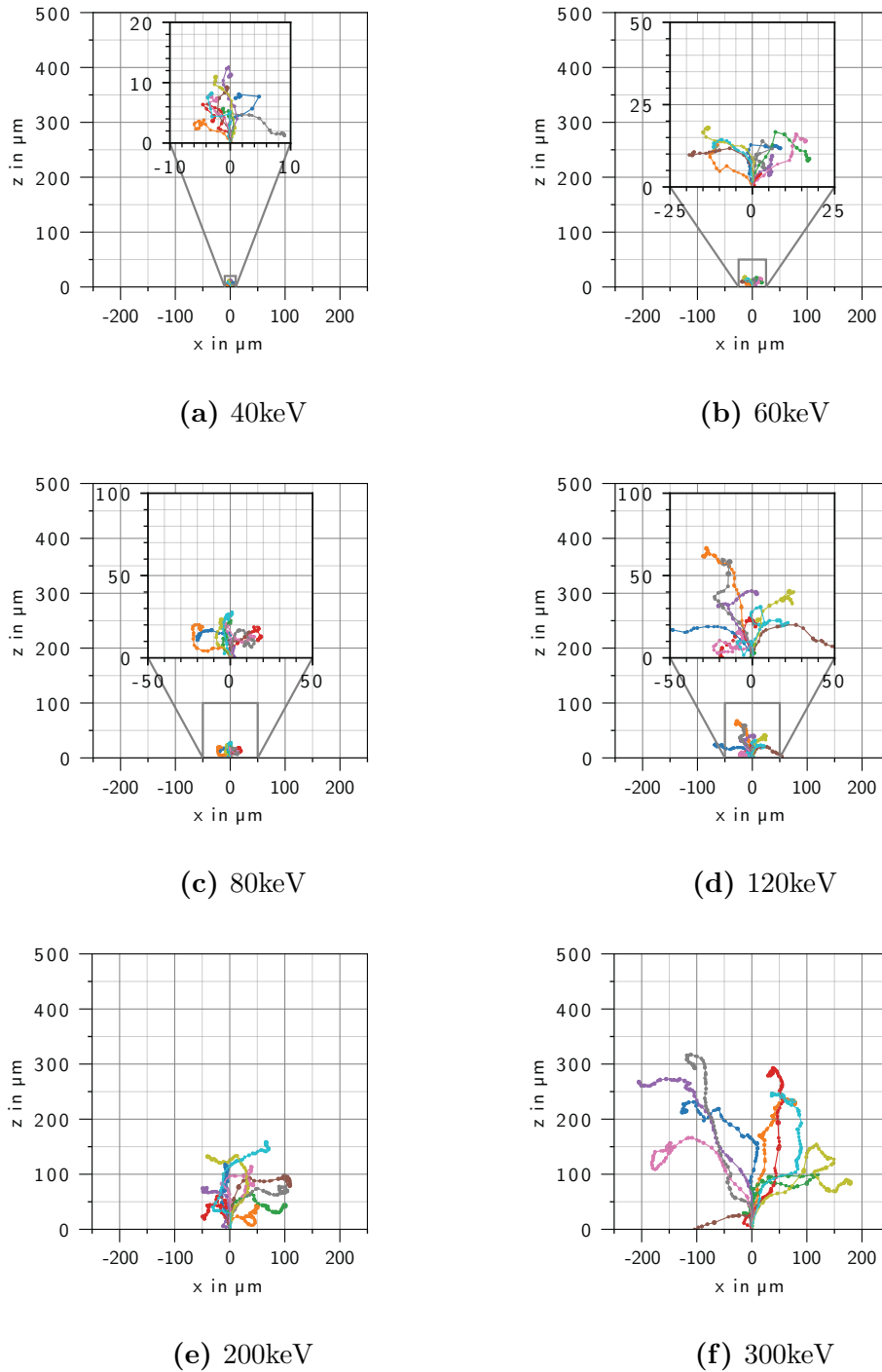
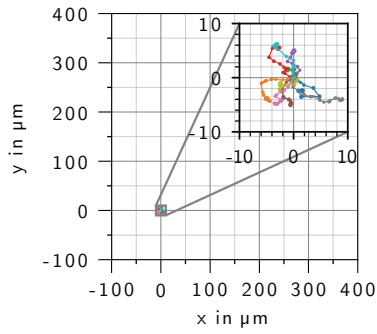
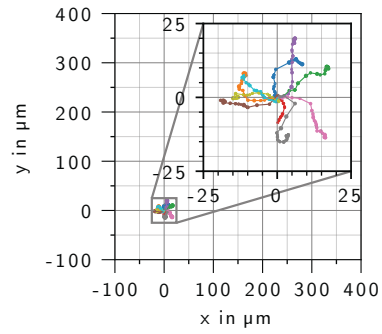


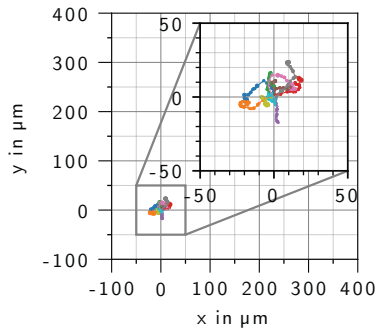
Figure 4.8: Projection of the energy deposition of ten randomly selected primary electrons into the x-z-plane. The electrons enter the sensitive detector volume at the origin (0,0,0). The primary momentum is parallel to the z-axis. The different tracks are color-coded, and the dots' size indicates the amount of deposited energy in arbitrary units.



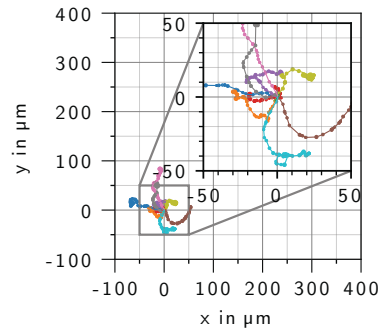
(a) 40keV



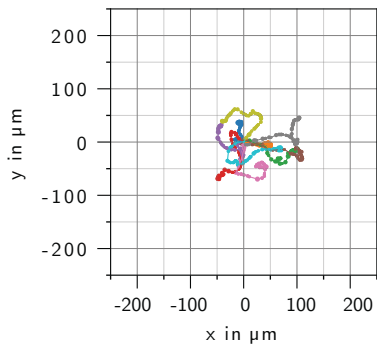
(b) 60keV



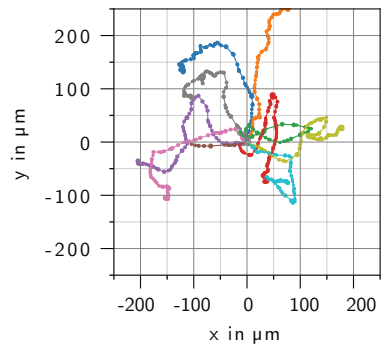
(c) 80keV



(d) 120keV



(e) 200keV



(f) 300keV

Figure 4.9: Projection of the energy deposition of ten randomly selected primary electrons into the x-y-plane. The electrons enter the sensitive detector volume at the origin (0,0,0). The primary momentum is parallel to the z-axis. The different tracks are color-coded, and the size of the dots indicates the amount of deposited energy in arbitrary units.

To get an idea that is independent of the statistics of individual electrons, one has to average over the behavior of many electrons. Therefore, for every individual primary particle, the average over the z -coordinate weighted by the corresponding energy deposition was calculated. The result is one weighted average for each primary particle. These individual, weighted averages were binned in a histogram and are shown in Figure 4.10. The histogram can be interpreted as the probability distribution $\rho_{\text{dep}}^{(z)}$, which can be written as a function of the projection of the energy deposition $E_{\text{dep}}^{(z)}(z)$ on the z -coordinate:

$$\rho_{\text{dep}}^{(z)}(z) = \frac{1}{\int E_{\text{dep}}^{(z)}(z)dz} \frac{dE_{\text{dep}}^{(z)}(z)}{dz} \quad (4.21)$$

$$E_{\text{dep}}^{(z)}(z) = \int \int E_{\text{dep}}(x, y, z)dydx \quad (4.22)$$

The maximum of $\rho_{\text{dep}}^{(z)}(z)$ depends on the primary energy. The higher the primary energy, the further inside the silicon material is the maximum of the penetration depth. As a consequence, the influence of surface effects and the detector entrance window becomes less pronounced with increasing energy. For higher energies, the distribution becomes broader. The energy deposition is less concentrated to one slice parallel to the detector surface but more spread out over the depth.

Figure 4.11 shows the probability distribution $\rho_{\text{dep}}^{(x)}(x)$ of the lateral energy deposition. The calculation is similar to the calculation of $\rho_{\text{dep}}^{(z)}(z)$:

$$\rho_{\text{dep}}^{(x)}(x) = \frac{1}{\int E_{\text{dep}}^{(x)}(x)dx} \frac{dE_{\text{dep}}^{(x)}(x)}{dx} \quad (4.23)$$

$$E_{\text{dep}}^{(x)}(x) = \int \int E_{\text{dep}}(x, y, z)dzdy \quad (4.24)$$

Averaging over many particles, an energy deposition at the same x -position as the PoE is most probable. However, this is not valid for individual particles. The probability distribution indirectly includes the particle density. The particle density decreases radially around the PoE. Even if the largest energy deposition is far from the PoE for individual particles, a high average energy deposition occurs at the PoE, caused by the summation of all particles.

The higher the primary energy, the less concentrated is the energy deposition around the PoE.

Because of the rotation invariance along the z -axis, $\rho_{\text{dep}}^{(y)}(y)$ is the same as

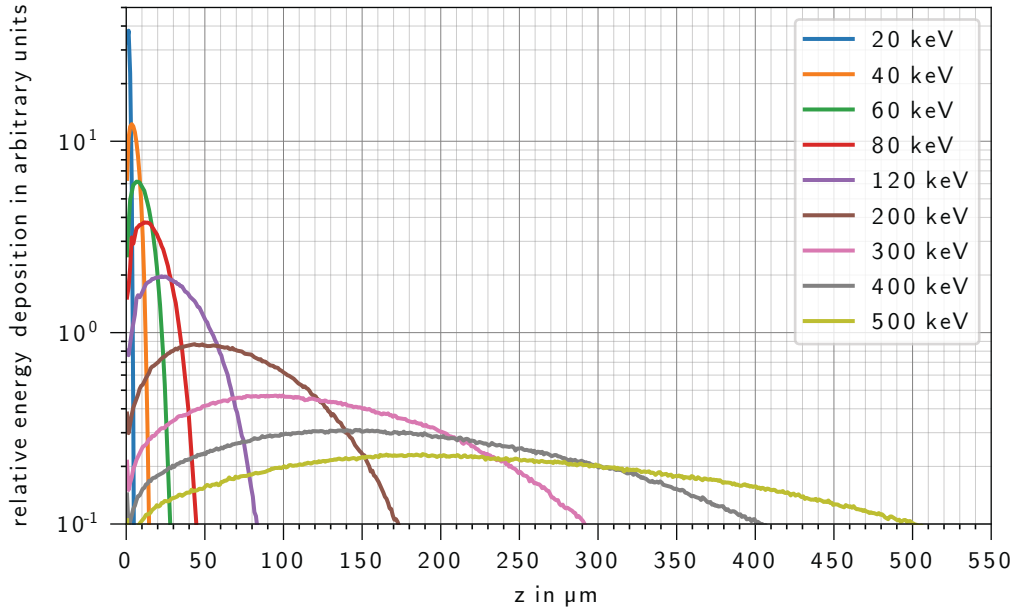


Figure 4.10: Relative energy deposition as a function of the depth z for various energies of the primary electrons.

$\rho_{\text{dep}}^{(x)}(x)$. To use this symmetry, instead of the coordinates x and y , the radius r is used to describe the lateral component. The PoE is at $r = 0$. The radius r is calculated as the two-dimensional Euclidean distance:

$$r = \sqrt{x^2 + y^2} \quad (4.25)$$

Figure 4.12 shows the range of the primary electrons and secondary particles in the detector volume as a function of the primary electron's energy. The orange curves correspond to the depth z parallel to the electron's primary momentum, and the blue curves to the radius on a plane parallel to the detector surface.

First, for every individual particle, the maximal depth, respectively, the maximal radius of the energy depositions is determined. It does not matter how large this energy deposition is. The continuous line shows the averaged ranges over all individual particles.

Second, for every individual particle, the average of the positions of the deposited energy weighted with the energy deposition is determined. The dashed lines show the depth, respectively, the radius of the mean of these weighted averages over all particles.

For all primary energies, the position of the mean energy deposition is closer

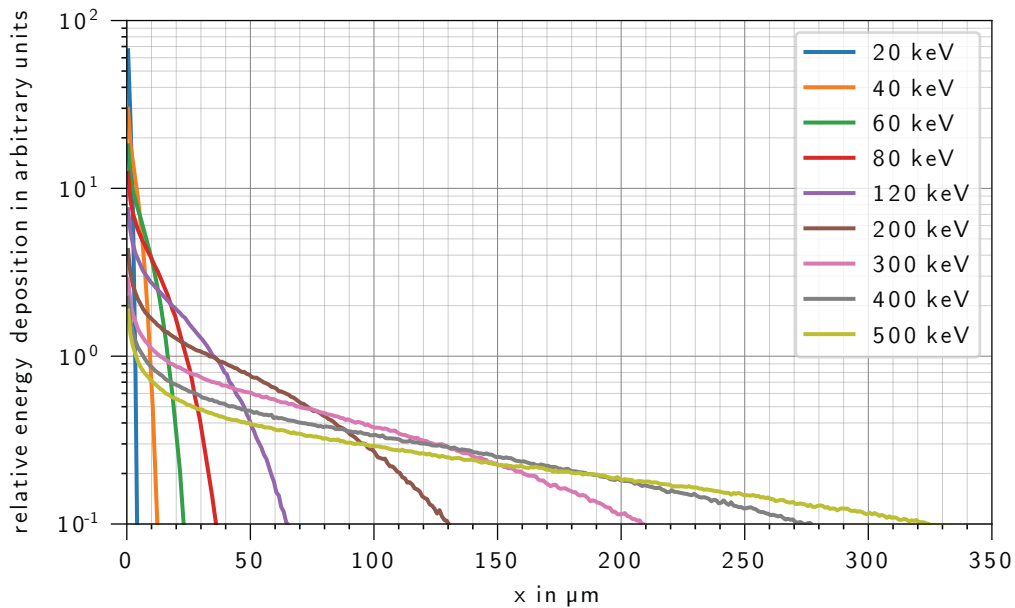


Figure 4.11: Relative energy deposition as a function of the lateral coordinate x for various energies of the primary electrons.

to the PoE as the maximal range. The lateral range and the penetration depth are similar and increase with the primary energy. The mean range in three dimensions is on a hemisphere with the PoE in the center. The radius of the hemisphere is the same as the lateral range and the penetration depth. Since the electrons do not form straight tracks but trajectories dominated by multiple scattering, this hemisphere's radius is always smaller or at most equal to the trajectory lengths. The trajectory length as a function of the primary energy can be found in Figure 2.10 on page 27.

The mean depth over the weighted energy depositions is larger than the lateral mean for all primary energies. This is a result of the fact that a particle that deposits a part of its energy laterally far away from the PoE has to scatter several times.¹ During these scattering processes, it loses parts of its energy. As a consequence, the remaining energy of the particle is smaller than the remaining energy of a particle that travels the same length but parallel to the z -axis.

Figure 4.13 shows the spectrum of the deposited energy in a silicon detector with a thickness of 450 μm . A fully sensitive volume is assumed. A

¹Small scattering angles are more likely than larger angles [108].

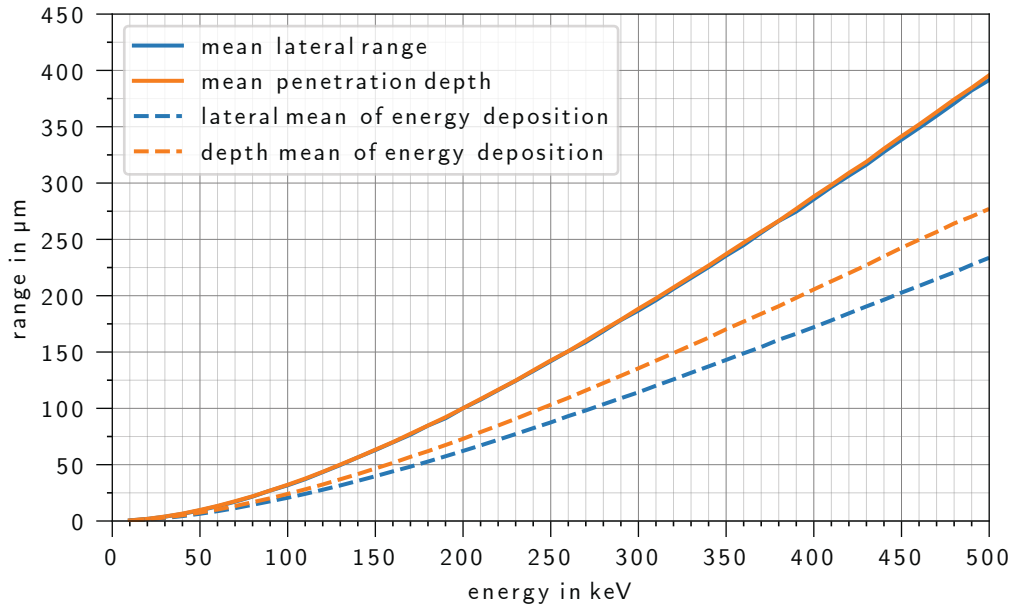


Figure 4.12: Lateral range (blue) and depth (orange) as a function of the energy of the primary electron. The solid line describes the average range of the primary and secondary particles. The dashed lines describe the averaged position of the energy deposition.

finite entrance window leads to a small deviation between a measured and the simulated spectrum. The entrance window can be considered as a thin layer. The energy loss of an electron passing through a thin layer can be described by the Landau distribution (Appendix A.4). The energy loss leads to an asymmetric broadening of the signal peak. The tail to lower energies is larger than the tail to higher energies. However, this deviation can be neglected since the reconstruction methods presented in this work's framework are mainly sensitive to the spatial energy distribution and not the amount of total energy deposited. A detailed study of the spectrum's features can be found in [69].

The dominant peak in Figure 4.13 of all spectra is at 100 %, corresponding to full energy deposition. The width of the peak is mainly caused by Fano statistics.

Additionally, all spectra show the characteristic escape peak at the primary electron's energy minus the energy of the escaping X-ray photon. The escape peak is produced when an electron of a higher electron shell occupies a created vacancy in the K-shell. The released energy is most probable in the form of a

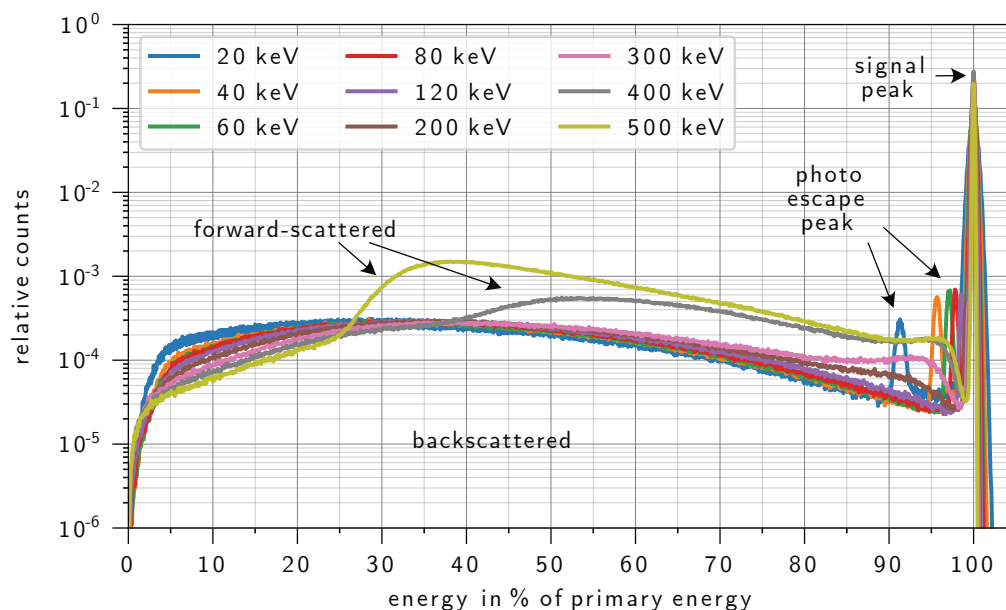


Figure 4.13: Simulated spectrum of deposited energy in silicon for primary electrons. A beam of mono-energetic electrons hits a fully sensitive silicon detector with a thickness of $450\ \mu\text{m}$. The different colors indicate the primary energies.

K_α X-ray photon which is generated and escapes the detector material. The energy of this photon is $1.74\ \text{keV}$ [40].

The broad distribution with a lower energy than the primary energy is caused by electrons scattered out of the detector material. The electrons deposit only a fraction of their primary energy and leave the sensitive detector volume. Figure 4.14 shows the energy distribution of the backscattered particles. The sharp lines are caused by the photon escape peaks. The shape of the broad distribution is given by two effects [69]:

First, small scattering angles are more probable than large ones due to the differential cross-section [108]. Therefore, the probability of being backscattered increases with the number of scattering processes. The primary particle loses a fraction of its energy during each scattering process. As a consequence, the probability of being backscattered increases with decreasing energy of the escaping particle. Because the distribution of scattering angles is more narrow for higher primary energies [108], the effect is more dominant for higher primary energies.

Second, the probability of escaping the detector volume at the backside decreases with the increasing detector depth the primary electron reaches. On average, the lower the particle's remaining energy, the longer the previously

traveled distance and, therefore, the penetration depth. As a consequence, for low remaining energies, it becomes less probable to escape the detector, and the probability distribution decreases to lower energies of the backscattered particle.

The amount of forward-scattered particles depends on the thickness of the used detector. For a detector thickness of 450 μm , forward-scattering becomes relevant for primary energies above around 350 keV. A detailed study of the fraction of back- and forward-scattered particles can be found in Appendix A.2.3.

In addition to the broadening of the backscattered electron spectrum, also the spectrum of forward-scattered particles is broadened. This is due to the fact that the forward-scattered particles deposit a fraction of their energy in the detector volume and then leave the detector. Because the probability for forward-scattered particles increases with higher primary energies, the broad distribution is more pronounced for higher primary energies. For higher energies, the tail's left position moves to lower relative energies as the relative energy of the forward-scattered particles increases (Figure 4.13). The higher the primary energy or the thinner the detector volume, the higher is the probability of only a small relative energy loss.

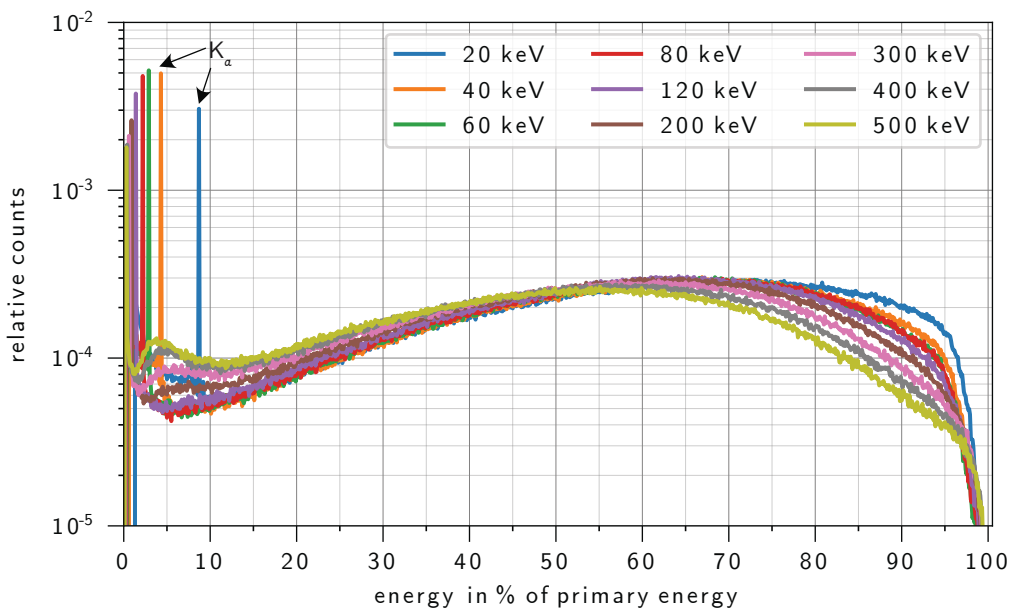


Figure 4.14: Energy distribution of the backscattered particles leaving the silicon. The different colors indicate the primary energies.

Figure 4.15 shows the averaged number of pixels with an energy deposition by one primary electron. Diffusion and repulsion were taken into account. As expected from the lateral range (Figure 4.12), the number of pixels containing an energy deposition increases with higher primary energies. Because the size of the tracks is independent of the pixel size, the number of pixels containing an energy deposition decreases with increasing pixel size.

The comparison between the measured and simulated data is performed not frame-based but event-based. Instead of looking at individual frames, the comparison is focused on investigating extracted event patterns. The event-based comparison focuses on the event patterns' size, shape, and structure since these properties are relevant for the training process of the neural networks in this thesis's framework and their evaluation. Table 4.1 shows the average single event pattern size obtained from the conventional event analysis with a primary threshold of 5σ and a secondary threshold of 2σ of the pixel-wise noise for various primary energies (Appendix B.1). Using only one threshold of 2σ would lead to too many noise events not associated with a signal generated by a primary particle. Using 5σ determines correct events, and a subsequent lower secondary threshold avoids detecting false events and characterizes correct events better.

The uncertainties describe the variation of the pattern size within the data sets caused by the statistical behavior of the electron tracks. The measured data are referenced in Appendix D.2 and offset and gain corrected.

Table 4.1: Average single event pattern size obtained from the conventional event analysis for various primary energies. The ENC is determined by the measured data and used for the simulation. Due to different detector settings, the ENC varies for different primary energies. The data reference is shown in Table D.3.

primary energy	simulation	measurement	ENC per pixel
keV	pixel	pixel	e^-
20	2.3 ± 1.0	2.1 ± 0.7	14
40	2.6 ± 1.2	2.8 ± 1.2	14
60	3.4 ± 1.0	3.7 ± 0.9	19
80	3.8 ± 0.7	4.2 ± 0.8	19
120	4.6 ± 1.2	4.8 ± 1.2	19
200	7.3 ± 1.9	7.5 ± 2.1	30
300	11.4 ± 3.2	11.4 ± 3.1	44

Figure 4.16 shows the multiplicity for electrons with a primary energy of 300 keV and a quadratic pixel size of $48 \times 48 \mu\text{m}^2$. The multiplicity $\langle m \rangle$ mea-

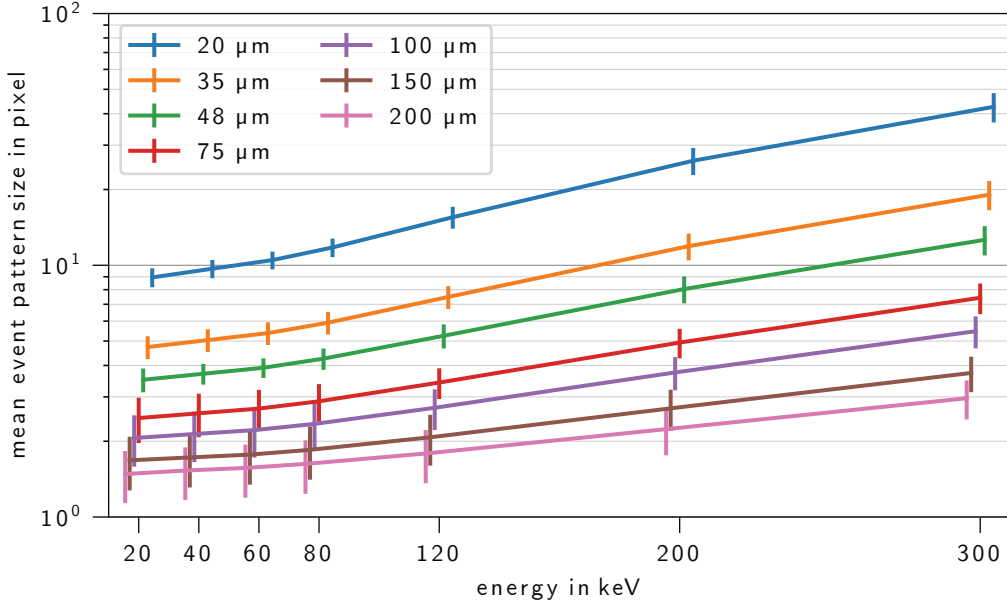


Figure 4.15: Averaged single event pattern size as a function of the primary energy. The different colors denote different sizes of the pixel structure. The error bars indicate the standard deviation over the used event patterns.

asures for how many pixels the energy deposition is above a certain threshold on average [109] and is related to the event pattern size for single events where only one primary particle contributes to the energy deposition. In the case of the event pattern analysis described in Appendix B.1, the secondary threshold would correspond to the above-described threshold.

The multiplicity can be calculated as the ratio of pixels above the threshold (triggered pixel) N_{trig} and the incident particles N_{true} .

$$\langle m \rangle = \frac{N_{\text{trig}}}{N_{\text{true}}} = \frac{\sum_i i N_i}{\sum_i N_i} = \sum_i i p_i \quad (4.26)$$

The number of triggered pixels N_{trig} is the sum of the particles weighted with the event pattern size. p_i denotes the probability that a primary particle triggers i pixels. In Figure 4.16, the blue distribution shows the relative amount of single event patterns for one incident particle. The orange curve shows the distribution of the relative amount of double event patterns for one incident particle times two. The other colored distributions show the probability for one incident particle with an accordingly higher number of

hits times the number of hits. The averaged multiplicity is the envelope of the individual distributions.

The higher energy limit of the individual distributions has to be smaller than the total primary energy divided by the number of hit pixels because the sum of the energy depositions of all hit pixels has to be the total primary energy. The multiplicity diverges for low thresholds in the order of the detector system's intrinsic noise since the number of pixels triggered but only containing noise and no primary particle's energy deposition becomes larger.

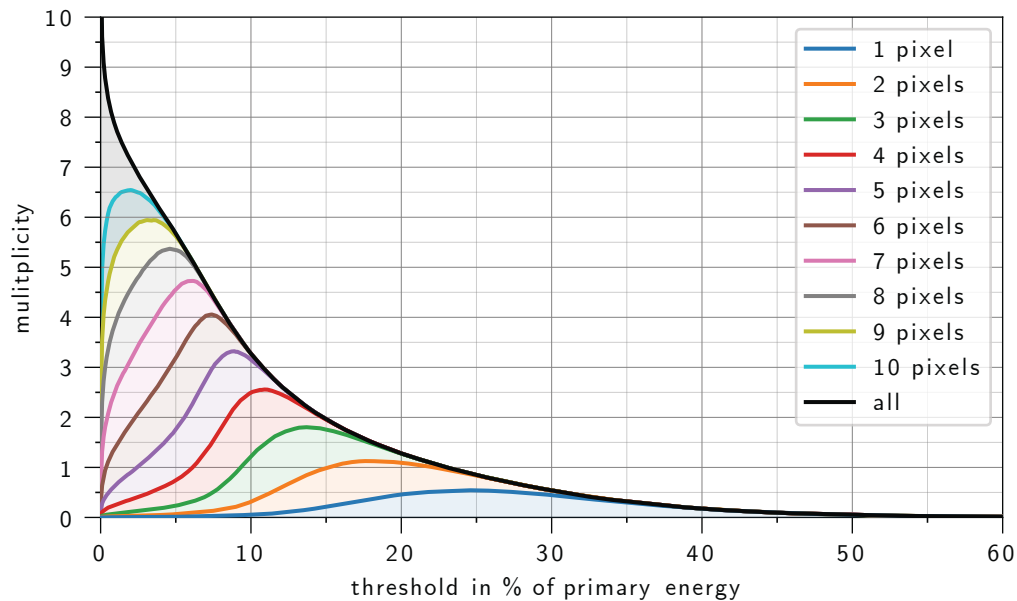


Figure 4.16: Multiplicity for electrons with a primary energy of 300 keV and a pixel size of $48 \times 48 \mu\text{m}^2$. The colored distributions show the contribution of the different amounts of triggered pixels per incident electron to the multiplicity.

4.7 Numerically Generated Data Set For Photons

According to the scattering cross-section (Figure 2.1 on page 14), the energy deposition for lower energies mainly is caused by the photoelectric effect. This process transfers the complete energy of the photon to an electron. For low energies, the range of this generated free electron is, according to the integrated Bethe-Bloch equation (Figure. 2.10 on page 27), short in comparison to the

size of the generated charge cloud due to repulsion and diffusion (Section 4.2.4).

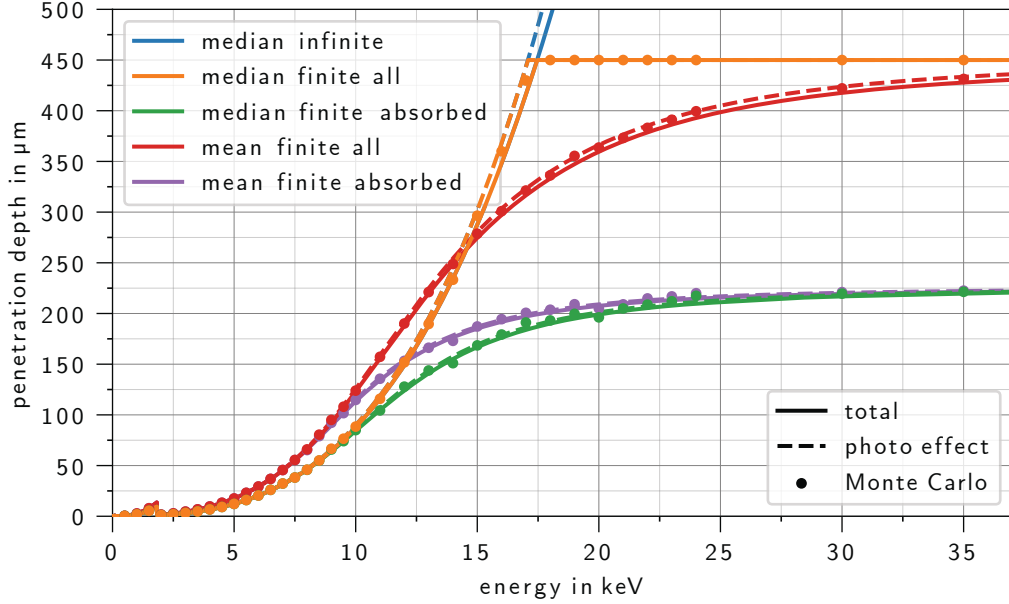


Figure 4.17: Penetration depth of photons as a function of the primary energy. The primary photons penetrate the detector perpendicular to the surface. The thickness of the sensitive silicon bulk is $450\ \mu\text{m}$. Points indicate the results obtained from the Monte Carlo simulations. The lines and the dashed lines show the theoretical result obtained from the total cross-section, respectively, the cross-section of the photoelectric effect, which dominates the total cross-section for the low energy range. The colors indicate different analytical methods.

Without scattering and with a full energy deposition localized at one point, the depth of the primary photon's absorption is the penetration depth. The depth and the mean amount of the energy deposition can be described by the Lambert-Beer law (Section 2.1.2). Figure 4.17 shows the depth of conversion as a function of the energy of the primary photon.

The blue curve shows the behavior for an infinite thick detector volume. The median of the penetration depth can be expressed by the half-value thickness $\delta_{1/2}$:

$$\delta_{1/2} = \frac{\ln(2)}{\mu(E)} \quad (4.27)$$

The half-value thickness describes the thickness where only half of the intensity remains. The attenuation coefficient $\mu(E)$ decreases with the photon's

energy E and is linearly related to the cross-section (eq. 2.13 on page 20). The increase of the depth of absorption around 1.8 keV is related to the K-edge of silicon.

The other curves in Figure 4.17 show the behavior for a detector's finite thickness of 450 μm . The finite analyses of the depth of absorption handle either all photons (also those which are not absorbed and traverse the silicon bulk) or only the photons absorbed in the silicon bulk.

- The median of the penetration depth for all photons $\tilde{\delta}_{\text{all}}$ (orange line in Figure 4.17) in a silicon bulk of the thickness d is composed of two parts:

$$\tilde{\delta}_{\text{all}} = \begin{cases} \delta_{1/2} & \delta_{1/2} \leq d \\ d & \text{else} \end{cases} \quad (4.28)$$

The median of the penetration depth for all photons $\tilde{\delta}_{\text{all}}$ describes the depth after which only half of all primary photons remain. At a depth of 450 μm , which is in our case the thickness of the sensitive silicon bulk, the slope of the penetration depth's median shows a discontinuity with a pronounced kink following a saturation.

- The median of the penetration depth $\tilde{\delta}_{\text{absorbed}}$ (green line in Figure 4.17), which takes only the photons into account which are absorbed in the silicon bulk, can be calculated as follows:

$$\tilde{\delta}_{\text{absorbed}} = \frac{\ln\left(\frac{2}{1+\exp(-\mu(E)d)}\right)}{\mu(E)} \quad (4.29)$$

$\tilde{\delta}_{\text{absorbed}}$ can be calculated as the half-value thickness of the modified distribution $\hat{N}(z)$, which can be derived from eq. 2.14 on page 20:

$$\hat{N}(z) = \begin{cases} \frac{e^{-\mu(E)z} - e^{-\mu(E)d}}{1 - e^{-\mu(E)d}} & z \leq D \\ 0 & \text{else} \end{cases} \quad (4.30)$$

Like $N(z)$ in eq. 2.14 on page 20, $\hat{N}(z)$ expresses the ratio of remaining photons after a certain depth but takes only the photons absorbed in the silicon as reference. The photons' probability of escaping the silicon bulk through the backside is very low for low energies. Therefore, $\tilde{\delta}_{\text{absorbed}}$ can be approximated by $\tilde{\delta}_{\text{all}}$ for primary energies below 10 keV.

As the primary energy increases, this approximation is no longer valid, and the ratio of photons that pass through the silicon bulk increases. For higher energies, $\mu(E)$ becomes smaller, and $\hat{N}(z)$ can be approximated over the silicon bulk's thickness as a constant. Therefore, $\bar{\delta}_{\text{absorbed}}$ converges to $D/2$.

- The mean penetration depth of all photons $\bar{\delta}_{\text{all}}$ (red line in Figure 4.17) in a silicon bulk of the thickness d is the expectation value of the distribution, which describes the probability for a photon being absorbed at a certain depth:

$$\begin{aligned}
 \bar{\delta}_{\text{all}} = \langle z \rangle &= \frac{1}{\int_0^\infty \frac{dN}{dz} dz} \left[\int_0^d \frac{dN}{dz} z dz + d \int_d^\infty \frac{dN}{dz} dz \right] \\
 &= - \left[Nz \Big|_0^d - \int_0^d N dz + d \int_d^\infty \frac{dN}{dz} dz \right] \\
 &= \frac{1 - e^{(-\mu d)}}{\mu}
 \end{aligned} \tag{4.31}$$

The Lambert-Beer law's derivative describes the probability distribution of a photon being absorbed in a certain depth. In eq. 4.31, the prefactor of the integral normalizes the integral of the probability distribution to one and is minus one.

The first integral in the squared bracket treats the expected value of the penetration for primary particles absorbed in the silicon bulk. The integral can be solved by an integration by parts.

The maximum penetration depth of primary particles is the thickness of the silicon bulk. The second integral in the squared bracket describes the contribution to the mean penetration depth of the particles which traverse the silicon bulk. For primary energies below approximately 14 keV, the mean of the penetration depth (red line in Figure 4.17) is larger than the median (orange line in Figure 4.17). The distribution of the penetration depth is skewed to larger values for low primary energies. For higher primary energies, the second integral in the squared bracket becomes more and more dominant, and therefore, the mean approaches asymptotically to d .

- The mean penetration depth $\bar{\delta}_{\text{absorbed}}$ (purple line in Figure 4.17), which

takes only the photons that are absorbed in the silicon bulk into account, can be calculated with a similar approach to eq. 4.31:

$$\begin{aligned}
 \bar{\delta}_{\text{absorbed}} = \langle z \rangle &= \frac{1}{\int_0^\infty \frac{d\hat{N}}{dz} dz} \int_0^\infty \frac{d\hat{N}}{dz} z dz \\
 &= \frac{1}{\int_0^d \frac{d\hat{N}}{dz} dz} \int_0^d \frac{d\hat{N}}{dz} z dz = - \left[\hat{N}z \Big|_0^d - \int_0^d \hat{N} dz \right] \\
 &= \int_0^d \hat{N} dz = \frac{1}{\mu} - \frac{d}{e^{(\mu d)} - 1}
 \end{aligned} \tag{4.32}$$

Eq. 4.32 calculates the expected value of the modified distribution \hat{N} (eq. 4.30). For low primary energies, the mean is larger than the median. Again, $\bar{\delta}_{\text{absorbed}}$ can be approximated by $\bar{\delta}_{\text{all}}$ below 10 keV. For higher energies, $\mu(E)$ becomes smaller, and $\hat{N}(z)$ can be approximated over the thickness of the silicon bulk as a constant. The mean of the absorption converges for higher primary energies to $d/2$. For the numerically generated data set, a conversion depth calculated by eq. 4.32 is used.

An approximation of the mean deposited energy for many photons (Figure 4.18) can be calculated by multiplying the equation for the relative amount of absorption (eq. 2.15 on page 20) with the mean transferred energy per interaction. The ratio between the transferred energy and the photon's primary energy is one for the photoelectric effect. Eq. 2.11 on page 18 approximates the mean energy transfer of the Compton scattering process. The photoelectric and Compton effects are both dominant effects for the energy transfer in the selected energy range. Other energy transfer mechanisms are neglected in this approximation.

In Figure 4.18, the sum of the mean transferred energy by the photoelectric effect and the Compton effect is indicated as total energy transfer. For higher primary energies ($\gtrsim 35$ keV), the Compton process's contribution to the total energy transfer results in a slightly higher total energy transfer than the pure contribution of the photoelectric effect. This contribution increases with the primary energy.

For low primary energies, the energy transfer is linear with energy because the probability of a photon being not absorbed in the silicon bulk is very small.

The maximum of the deposited energy is around a primary energy of 14 keV. For higher energies, the ratio of photons that are not absorbed becomes larger, and, therefore, the mean deposited energy becomes smaller. However, individual photons either deposit their total energy due to the photoelectric effect or pass the silicon bulk without an energy deposition. A detailed study of the ratio of the photons which are forward-scattered can be found in Appendix A.2.3.

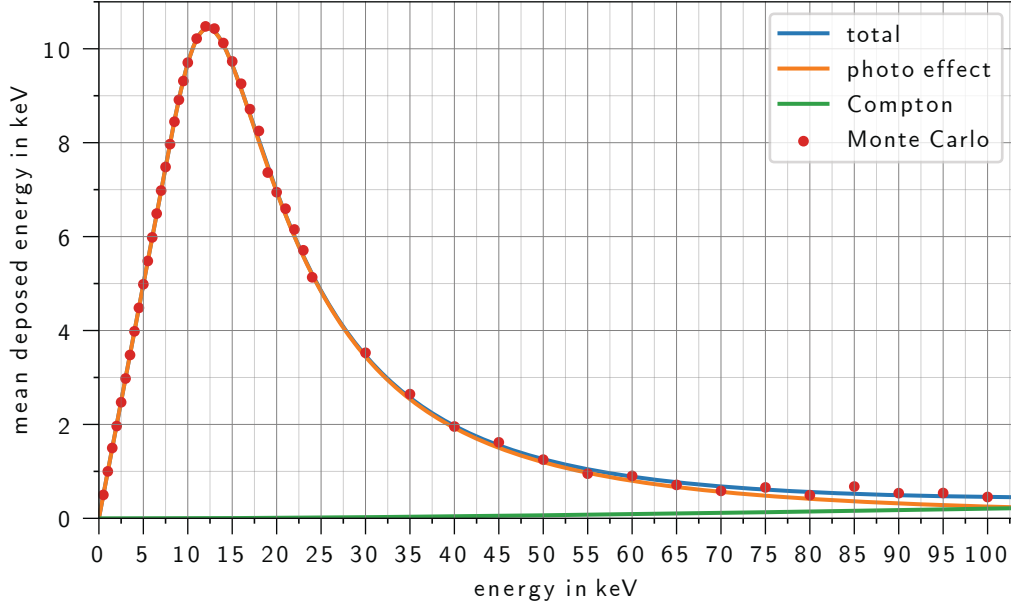


Figure 4.18: Mean deposited energy for photons with a low primary energy in a silicon bulk with a thickness of 450 μm .

Since for primary energies below 35 keV, the Compton effect's energy deposition can be neglected, and since the photoelectric process transfers the total energy of the primary photon, the deposited energy can be approximated as a constant.¹ In this model, the charge cloud arriving at the pixel structure looks similar for all events and is independent of the energy deposition mechanism but only depends on the drift process, the primary energy, the depth of the energy deposition, the backside voltage of the detector, and the temperature. The required drift time is calculated from the mean penetration depth $\bar{\delta}_{\text{absorbed}}$ (eq. 4.32).

¹Other effects such as those introduced by the entrance window or the photo escape peaks are neglected in this model.

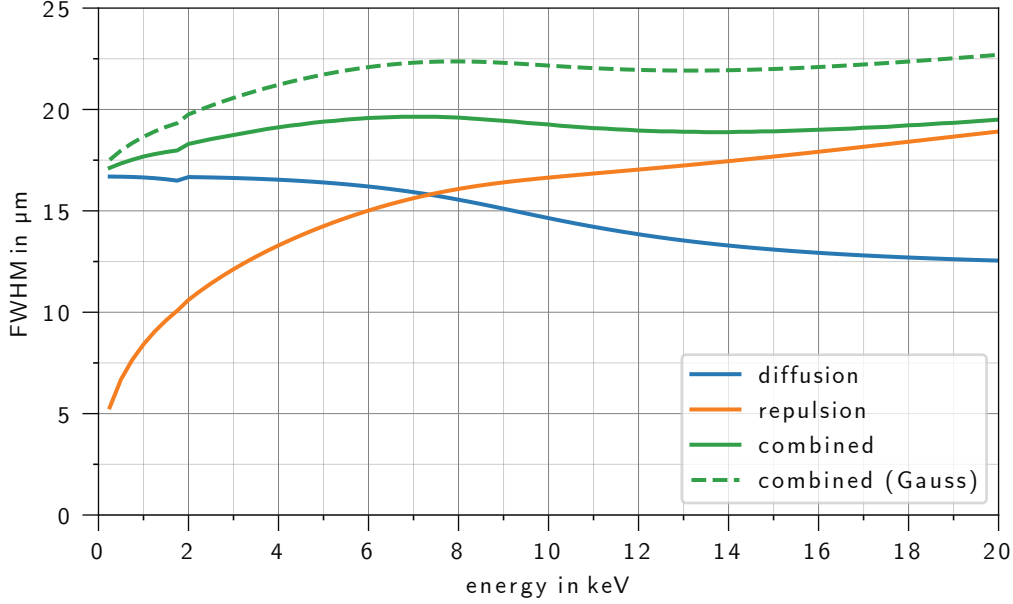


Figure 4.19: Size of the charge cloud as a function of the primary photon’s energy. The colors indicate the different effects. The green curves labeled with ”combined” result from the convolution of the individual effects. The FWHM for repulsion is due to its definition the same for the actual shape and the Gaussian approximation. The parameters $V_b = -230$ V, $V_f = -15$ V, and $T = 253$ K are assumed.

Figure 4.19 shows the average size of the charge cloud as a function of the primary photon’s energy. The combined size results from the convolution of diffusion and repulsion. The combined charge cloud size becomes larger for higher primary energies.

The charge cloud size decreases due to diffusion with higher primary energies. The average depth of the energy deposition is deeper for higher energies. As a consequence, the drift time into the pixel structure decreases and, therefore, the spread of the charge cloud due to diffusion. For higher primary energies, the penetration depth becomes independent of the primary energy in this model’s scope. The spread of the charge cloud becomes constant due to diffusion. The small kink at an energy around 1.8 keV results from the K-edge of silicon. At this energy, the cross-section increases, and the mean penetration depth decreases.

The effect of repulsion increases with the primary energy because the increasing number of electrons in the charge cloud dominates over the decreasing drift time.

The numeric generation of the data set for photons uses the projection of the initial position of the energy deposition parallel to the detector surface, the drift time, and the number of electrons in the charge cloud as parameters. The systematic generation of the data set contains the following steps if the Gaussian approximation for repulsion is not applied:

- **Calculation of the charge density as a function of the radius:** The first step is to calculate the charge density as a function in the polar coordinates domain. It is radially symmetric and, therefore, independent of the polar angle. The polar coordinate system's origin is the projected initial position of the energy deposition onto a plane parallel to the detector surface.
The charge density is calculated by a numerical convolution of the Gaussian-shaped diffusion and the distribution of the repulsion, which are both described in Section 4.2.4. The spacing of the r-coordinate is $0.1 \mu\text{m}$.
- **Coordinate transformation from polar coordinates to Cartesian coordinates:** In this step, the charge density, which is calculated on a polar coordinate grid, is converted to a Cartesian grid via interpolation. The result is the charge density on a Cartesian grid as a function of the x- and the y-coordinate. The spacing of the grid is $0.1 \mu\text{m}$ for the x- and the y-coordinate.
- **Varying the charge cloud over the pixel structure:** The charge density is positioned relative to the grid to simulate results for different PoEs relative to the pixel structure. The result is a list of charge clouds in the Cartesian domain and their corresponding position of the PoE.
- **Pixelation:** In this step, the actual amount of charge carriers in the pixels is calculated. Therefore, the fine grid is down-sampled to a coarser grid by local averaging [110]. The coarser grid is chosen in a way that it represents the pixel structure. The coarser grid's spacing is the pixel size in each dimension, and the number of grid points is seven times seven. Mathematically this procedure is described with eq. 4.20. A seven times seven pattern is large enough that all non-neglectable amounts of charges are represented.¹ The PoE is scanned across the central pixel of the seven times seven pixel array with a spacing of the fine grid ($0.1 \mu\text{m}$). The result is a list of seven times seven patterns and their corresponding PoEs. Optionally, random noise can be added to each pixel.

¹The Gaussian distribution is never zero but for the range of the used parameters outside the seven times seven pattern smaller than one percent of the total charge and, therefore, neglectable.

The created data are used to generate the correction map to the center of gravity method in Appendix B.2 and to train the neural network in Chapter 7. To calculate the correction map for the center of gravity method, it is, due to symmetry, sufficient to push the charge cloud over the pixel structure in only one dimension.

4.8 Summary

The distribution of the signal in the individual pixels strongly depends on the primary radiation's type and energy.

For photons, the energy deposition in the detector volume is very concentrated. The charge cloud's projection to a plane parallel to the detector surface can be approximated as radially symmetric. The charge cloud's size slightly depends on the primary photon's energy but is mainly influenced by detector parameters such as the backside voltage or the temperature. The signal distribution is dominated by the repulsion and diffusion of the electrons in the charge cloud and is, therefore, deterministic.

Electrons deposit their energy along tracks. The shape of the tracks is caused by multiple scattering processes and, therefore, stochastic. The length of the tracks increases with the primary energy and, therefore, the charge distribution's size strongly increases with the primary energy. Especially for higher primary energies, the electrons' repulsion and diffusion in the charge distribution play a subordinate role. The signal distribution is dominated by the energy loss of the primary particle and is therefore stochastic.

Chapter 5

Classical Data Analysis Methods

This Chapter introduces the classical data analysis methods to reconstruct the PoE of primary particles. A detailed description of the classical analysis methods and the state-of-the-art algorithms to determine the PoE of primary particles can be found in Appendix B. The used methods are defined in Appendix B.1.4. A good understanding of these classical methods is necessary to understand, on the one hand, the physics and the mechanisms behind the data reconstruction and, on the other hand, provide the benchmark for this thesis's framework introducing neural networks.

All introduced classical analysis methods have in common that they need, as in the previous step, an event pattern analysis (Appendix B.1) and cannot handle pile-up events. This has the consequence that, in contrast to the methods based on neural networks, the conventional methods are only applicable for low particle rates. As a criterion for a single event or pile-up event, the simple summation of the energy depositions can be used.

The center of gravity method (with corrections) (Appendix B.1.4) performs best for photons and electrons with low primary energies, whereas the furthest away method (Appendix B.1.4) leads to the best results for electrons with higher primary energies and strong directional behavior of the track. The achievable resolution depends on the noise of the detector system as well as the type and energy of the primary particle. A mathematical derivation and calculation of the systematical error made by the center of gravity method can be found in Appendix B.2.

Figure 5.1 shows the normalized hit-map and the resolution map over one pixel for simulated photons. Due to the presence of noise, the normalized hit-map (Figure 5.1a) shows a slightly lower probability of a central hit for

the x- and the y-dimension and a slightly higher probability near the axes. This inhomogeneity can be explained by looking at the distributions for the individual PoEs in Figure B.13 on page 252. The contribution of PoEs near the axes shows an asymmetric spatial distribution. This leads to an inhomogeneous distribution. The resolution map (Figure 5.1b) is defined as the averaged distance between the true PoE and the reconstructed PoE and shows a worse resolution in the center of the pixel, as expected from the theoretical description (Appendix B.3 and Appendix B.2.4).

The inhomogeneity can be corrected with the method described in Appendix B.2.3. The result is a homogeneous image (Figure 5.1c). However, the spatial precision worsens by applying this second correction since the reconstructed center of gravity method (with corrections) already describes the best guess of the true PoE (Figure 5.1d).

Figure 5.2 shows the Euclidean distance between the reconstructed PoE and the true PoE for electrons. All methods have in common that they do not consider the statistical energy deposition behavior of charged particles in the detector volume. The center of gravity method (with corrections) (Appendix B.1.4) performs best for electrons with low primary energies. The furthest away method (Appendix B.1.4) leads to the best results for higher primary energies. For the furthest away method, the large difference between the mean and the median can be explained by the shape of the Euclidean distance's distribution: An average that is larger than the median suggests a right-skewed distribution. The median is more robust against outliers than the mean. [111] Due to the furthest away method algorithm it is more likely that an error in the reconstruction is larger than an average error made by the other reconstruction methods. For the furthest away method, the reconstructed PoE is always at a border of the event pattern, whereas for the other methods, the reconstructed PoE is, in most cases, more central. Under the assumption that the true PoE is in the event pattern, for a made error, the Euclidean distance between the reconstructed PoE and the true PoE can be larger for the furthest away method.

However, the probability of making such an error is relatively low, and, therefore, the median of the distance between the reconstructed PoE and the true PoE is small.

The classical reconstruction of a PoE requires, in general, a preceding event pattern analysis. This event pattern analysis usually represents the bottleneck of performance and, thus, limits the speed of reconstruction.

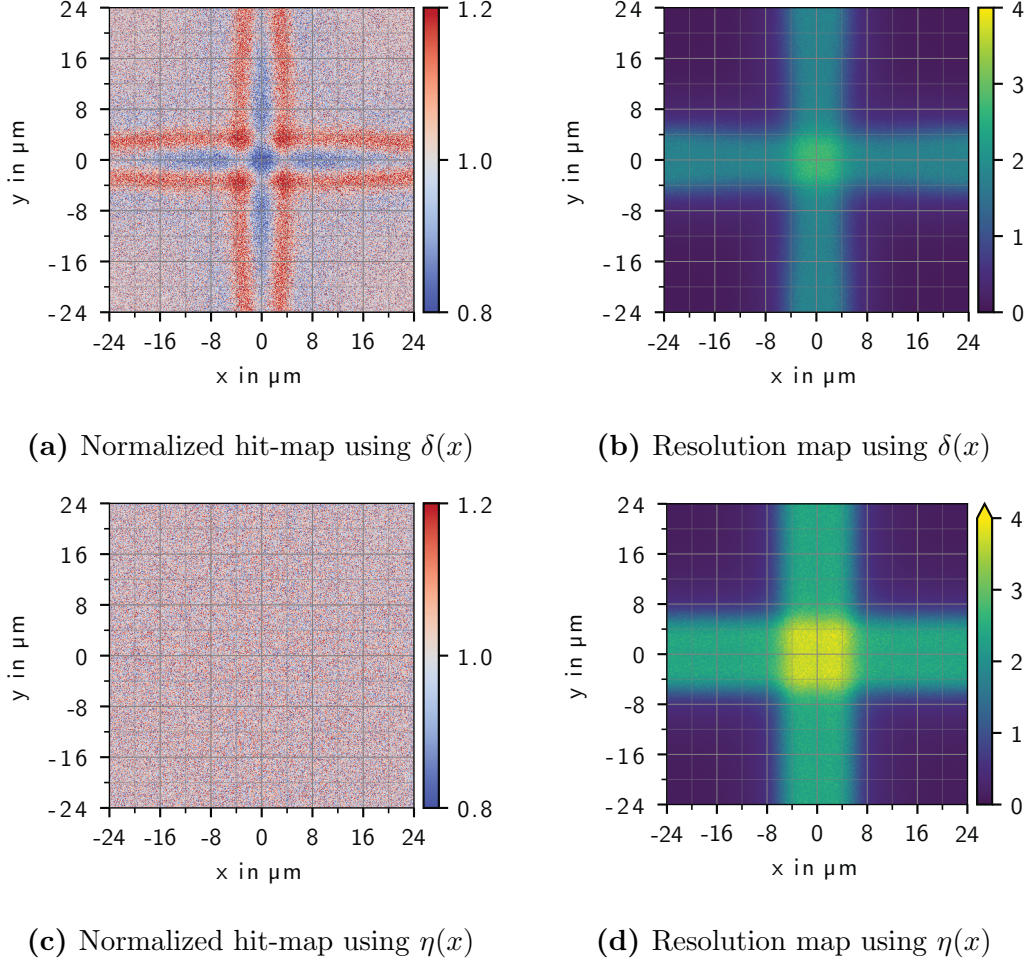


Figure 5.1: Normalized hit-map and resolution map for the center of gravity with corrections obtained from a simulated homogeneous illumination with around two million simulated primary particles. The physical pixel size is $48 \times 48 \mu\text{m}^2$ with $(0,0)$ in the center of the pixel, and the energy of the primary photons is 8048 eV. A non-correlated Gaussian distributed pixel-wise noise with a mean of 11.3 eV is assumed. These are the same parameters as for the presented measurement in Section 10.1.2. The hit-map's color scale is unitless due to the normalization, and the color scale's unit of the resolution map is μm .

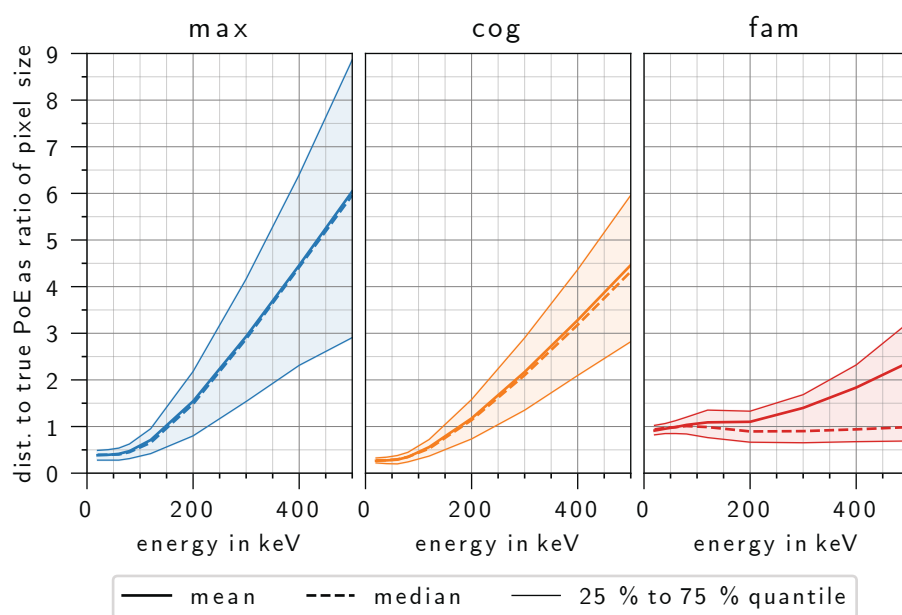


Figure 5.2: Euclidean distance between the reconstructed PoE and the true PoE as a function of the primary energy of the electron. The mean, the median, and the 25% and 75% quantile are shown. The simulated pixel size is $48 \times 48 \mu\text{m}^2$.

Chapter 6

Artificial Neural Networks

The following Chapter introduces the main terms and components of artificial neural networks used in this thesis and briefly explains the operating principle behind them. In literature, the principle and structure of machine learning and especially artificial neural networks are described in detail [112, 113, 26]. Artificial neural networks are named neural networks in the following. Neural networks are common tools to approximate an unknown complex function by investigating the structure of large amounts of data. The optimization process of the parameters which approximate this complex function is called training. [114]

In contrast to the parameters optimized during the training process, there are also parameters called hyper-parameters. The hyper-parameters define the structure and properties of the neural network. These hyper-parameters are defined before the neural network's actual training and optimization process. Their choice is crucial for a functional neural network. [112]

All neural networks in this thesis are implemented using Keras [115] with the TensorFlow backend [116].

6.1 Introduction

Approaches based on artificial intelligence (AI) have been established in many fields in recent years and especially show their potential in complex problems with large amounts of data [114]. Famous examples are voice assistants [15], autonomous driving [20], or image recognition [117]. AI refers to the ability of a machine to mimic the skills of the human brain. For example, the imitation can be done by learning from examples, understanding and responding, and

making decisions to solve problems [118].

Machine learning is a part of AI and describes the ability to learn and adapt a model based on data instead of explicitly defining and programming the solution to a problem. Neural networks are a special kind of machine learning. [119]

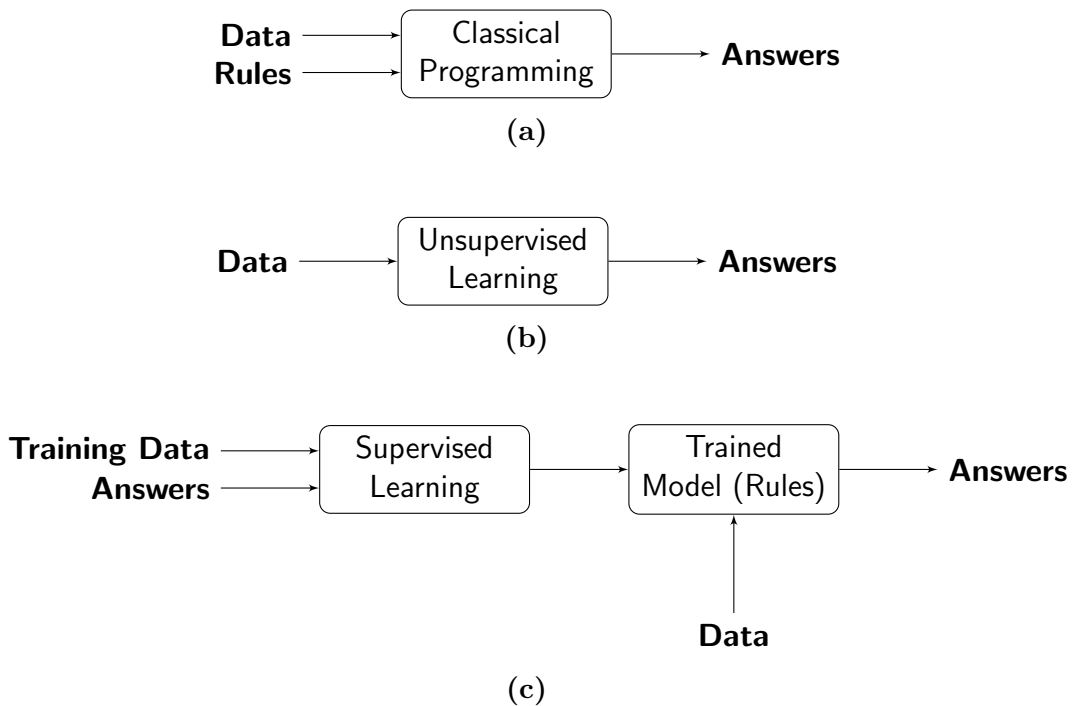


Figure 6.1: Conceptual approach of machine learning. As a comparison, the concept of classical programming is shown in (a). (b) shows the concept behind unsupervised learning and (c) the concept behind supervised learning. The goal of all approaches is to find the desired answer for the data. The learning data are data with the same properties and features such as the data to be analyzed and help the algorithm learn the desired task.

6.1.1 Types of Machine Learning Algorithms

Machine learning algorithms can be divided into three types.

First, **unsupervised learning**, where the algorithm runs on an unlabeled data set. Unlabeled data sets are data sets in which samples are not tagged with labels. Those labels can be characteristics, properties, or classifications.

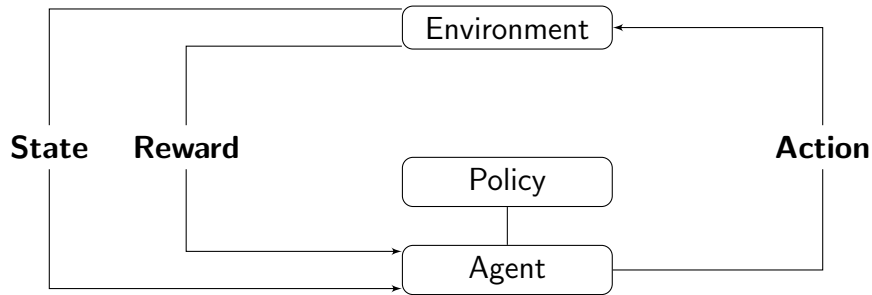


Figure 6.2: Conceptual approach of reinforcement learning. The required parts are an environment and an agent which interact with each other. The environment gives a state to the agent. The agent acts with an action and gets a reward for this action and the new state. The goal for the agent is to maximize the reward. The rules for the actions are given by the policy.

Understanding an affiliation or a sorting behind the data is the goal of unsupervised learning. The algorithm progressively predicts patterns in the data and organizes and clusters them accordingly. The number of clusters can either be predefined or predicted by the model itself. Figure 6.1b shows the conceptual approach. For better comparison, the concept of classical programming is shown in Figure 6.1a. The algorithm based on unsupervised learning can be directly applied to the data and discover the structure behind the data during the clusterization. [114]

The second type is **supervised learning**. Here, an algorithm tries to find a complex hypothesis that makes accurate predictions. A hypothesis is a mapping that assigns an assumed output value to each input value. The difference between unsupervised learning and supervised learning is the algorithm's optimization process called training. The concept is shown in Figure 6.1c. The algorithm learns on samples and their corresponding pre-defined answers to the hypothesis during the training. The samples and their corresponding pre-defined answers are called labeled data sets. The algorithm gradually learns to predict the correct answer to a sample. The learning process results can be compared with the known, correct answers, i.e., "supervised". After finishing the training, the algorithm is able to predict answers of unlabeled data sets whose content is similar to the labeled data sets. [114]

The third type is **reinforcement learning**. Reinforcement learning conceptually differs from unsupervised and supervised learning because the algorithm is not trained with sample data. The concept of reinforcement

learning is shown in Figure 6.2. An algorithm (agent) takes an action in an environment to maximize a reward. The environment can be abstract or a physical world. The agent learns by trying different actions (trial and error). The goal of the agent is to perform a sequence of actions to solve the task. The reward can be designed to be short-term or long-term. Short-term means the reward appears quickly after the action. A long-term reward appears some steps delayed. The agent tries to find a balance between exploration and exploitation of the current knowledge. The knowledge is stored in the policy, which recommends a particular action for a given state. [114]

An example for reinforcement learning is balancing an inverted pendulum by a cart [120].

The frameworks presented in this thesis use supervised learning since the goal is to predict the point of entry of the primary particles for new data while knowing upfront the type of results to expect. The data set for the training process was generated with Monte Carlo methods.

6.1.2 Types of Neural Networks for Image Analysis

Another way to classify neural networks refers to the task they perform. In the context of this thesis, models in the area of image analysis are used. These are typically divided into three classes. An example of these three classes is shown in Figure 6.3.

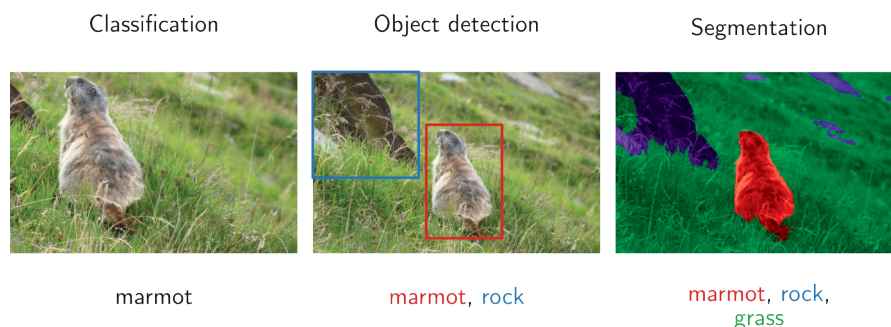


Figure 6.3: Types of image analysis. The first type is classification, the second type is object detection, and the third type is segmentation. The labels are shown under the images.

The first class is **classification**. Here, each image contains only one object which shall be classified. The output of such a neural network is the class of

the object. The neural network neither makes a prediction concerning the object's location in the image nor any other objects' prediction within the image. [121]

The second class is **object detection**, which is sometimes called tracking in the literature. Here, the model predicts the class of the object and the position in the image. Normally, this position is predicted by a rectangular box. Tracking models can predict the classes and positions of multiple objects per image. [122]

The last category is **segmentation**. In image segmentation, every pixel of the image is labeled with a class. Segmentation models predict not only the class and the position of an object, but also the shape in pixel precise resolution. [123]

In the context of this work, the concept of segmentation is adapted to predict the point of entry of the incoming particle on pixel and subpixel, respectively.

6.2 Neurons

In this Section, the basic elements of a neural network are introduced. These basic elements are called neurons [119]. A neuron has one or more inputs and one output. Exemplary, Figure 6.4 shows a neuron with n inputs. Between

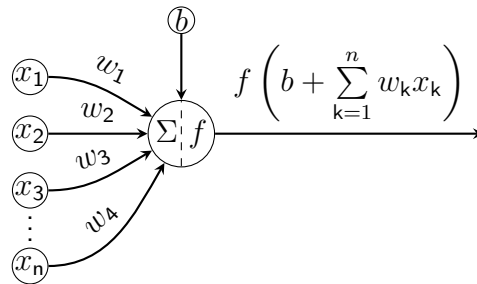


Figure 6.4: Schematic of a neuron with n input nodes x_1 to x_n with weights w_1 to w_n , bias b , and an activation function f .

the input and the output, three mathematical steps are performed [124, 125]:

- Each input is multiplied by a weight w_k . This weight can be a positive or a negative real number. The letter k describes the index of the input to the neuron.

$$x_k \rightarrow x_k \cdot w_k \quad (6.1)$$

- The next step is to sum up all weighted inputs and, optionally, add a bias b .

$$b + \sum_k x_k w_k \tag{6.2}$$

- In the last step, this sum is used as an argument for the function f , which is called activation function. In general, the activation function can be arbitrary. A detailed introduction to commonly used activation functions can be found in Section 6.4.

$$y = f \left(b + \sum_k x_k w_k \right) \tag{6.3}$$

The output y of the neuron is called activation.

6.3 Topology of a Neural Network

A neural network is a combination of connected neurons. These neurons are organized in layers. Every neural network has at least two layers. These are called input layer and output layer. Between the input layer and the output layer, there can be a varying number of so-called hidden layers. Figure 6.5 shows an example neural network with one hidden layer and no additional bias. [113]

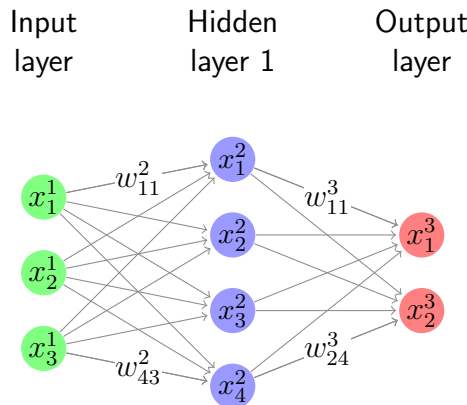


Figure 6.5: Architecture of a simple neural network with three input nodes, one hidden layer with four nodes, and two output nodes. For simplicity, the bias is zero.

Mathematically, the activation x_j^i of the j^{th} neuron in the i^{th} layer can be

described in the more general form of eq. 6.3 as [113]:

$$x_j^i = f \left(b_j^i + \sum_k w_{jk}^i x_k^{i-1} \right) \quad (6.4)$$

In this notation, b_j^i is the bias of the j^{th} neuron in the i^{th} layer and w_{jk}^i is the weight from the k^{th} neuron in the $(i-1)^{\text{th}}$ layer to the j^{th} neuron in the i^{th} layer.

The weights of one layer could also be represented by a matrix. In this notation, the neurons' bias and activation in one layer are vectors. Then the activation function f acts element-wise on each entry element of the vector. This leads to a description in a vectorized form of eq. 6.4 [113]:

$$\mathbf{x}^i = f(\mathbf{b}^i + \mathbf{w}^i \mathbf{x}^{i-1}) = f(\mathbf{z}^i) \quad (6.5)$$

Here, \mathbf{w}^i is a matrix. The number of columns is the number of neurons in the $(i-1)^{\text{th}}$ layer and the number of rows is the number of neurons in the i^{th} .

The argument \mathbf{z}^i is called weighted input to the neurons of the layer i .

Neuronal networks are generally named according to the type and amount of hidden layers. Solving a problem with a neural network with multiple hidden layers is called **deep learning** [126]. There is no definite answer at which number of hidden layers a neural network is considered as deep [127, 128]. The output of a hidden layer is increasingly abstract with increasing deepness of the position in the network. In the standard case, neurons in one layer are only connected with neurons in the previous and the next layer. An exception are the so-called **residual neural networks**, which have skip connections or shortcuts to jump over a layer or several layers [129]. In this thesis, skip connections are used in Section 9.1.1.

If all neurons in one layer are connected with all neurons in the previous layer, the layer is called fully-connected. Fully-connected layers are also referred to as dense layers in literature [130].

One physical representation of a neural network relevant for this thesis is a neural network in which each neuron of the input layer represents one pixel of a frame or the detector. However, the output of neurons of the hidden layers can be very abstract and not necessarily have an obvious physical meaning. The output of the last layer again has a physical meaning. This meaning depends on the network's task and can describe, for example, a hit-map, a frame with higher resolution as the input, the x- and y-coordinate of a point of entry, or a binned energy spectrum.

6.4 Activation Function

The activation function calculates each neuron's activation and usually takes only the neuron's weighted input as an argument. It is common for the output layer to also use the other neurons' output of the layer as input to enable the normalization of the predicted result. The activation function can be defined individually for each neural network layer. In most cases, the activation functions are the same for the input layer and the hidden layers. Typically, the activation function of the output layer is different. In general, activation functions can be divided into binary, linear, and nonlinear activation functions.

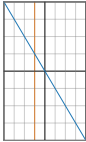
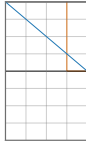
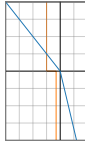
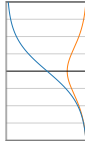
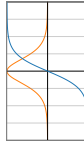
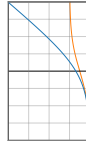
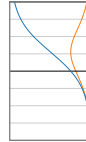
The **binary** activation function activates the neuron if the weighted input is above a threshold; otherwise, the neuron is inactive. Instead of continuous values, binary activation functions allow only two discrete values at the output. The neuron is either active (1) or inactive (0). This behavior has a low flexibility.

The simplest possible **linear** activation function is the identity function. However, using only identities as activation functions leads to a collapse of the hidden layer. In this case, the input of the output layer is always a linear function of the input. This is because a linear combination of linear functions is still a linear function. A network simply transforms into a linear regression model that is not able to predict more complex problems. In addition, linear functions cannot be optimized with backpropagation (Section 6.6.3) because the derivative is a constant and independent of the input of the neuron. It is not possible to retrieve the inputs and understand their influence. [131]

State-of-the-art neural networks use **nonlinear** activation functions. They allow to create complex relations between the input and the output of the neural network and allow to stack together multiple layers to a deep neural network. This is necessary to learn and predict advanced problems. In principle, every function can be used as an activation function. However, the choice of the activation function of the last layer should be problem-related. Especially, the co-domain of the activation function of the output layer should be problem-related. For example, if the output represents a probability, the co-domain of the activation function should be between zero and one. [131]

Commonly used functions are shown in Table 6.1. For Leaky ReLU, the typical value for α , which describes the slope for negative weighted inputs, is 0.3. For ReLU, α is zero. For the SoftMax activation, the sum in the denominator sums over all output layer neurons. The sum ensures normalization

Table 6.1: Commonly used activation functions, with their plot, mathematical equation, derivative, range, monotonic behavior (Mono), and monotonic behavior of the derivative (Mono'). The blue curve in the plot shows the activation function, and the yellow curve shows its derivative. The distance between the grid is one for all plots. [130]

Name	Plot	Equation	Derivative	Range	Mono	Mono'
Linear		$f(x) = x$	$f'(x) = 1$	$(-\infty, \infty)$	yes	yes
Rectified linear unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x \leq 0 \\ x & \text{for } x > 0 \end{cases}$ $= \max(0, x)$	$f'(x) = \begin{cases} 0 & \text{for } x \leq 0 \\ 1 & \text{for } x > 0 \end{cases}$	$[0, \infty)$	yes	yes
Leaky ReLU		$f(x) = \begin{cases} \alpha x & \text{for } x \leq 0 \\ x & \text{for } x > 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for } x \leq 0 \\ 1 & \text{for } x > 0 \end{cases}$	$(-\infty, \infty)$	yes	yes
Sigmoid		$f(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$	$(0, 1)$	yes	no
Tanh		$f(x) = \tanh(x)$	$f'(x) = 1 - f(x)^2$	$(-1, 1)$	yes	no
SoftPlus		$f(x) = \ln(1 + e^x)$	$f'(x) = \frac{1}{1 + e^{-x}}$	$(0, \infty)$	yes	yes
SoftMax		$f_i(x_i) = \frac{e^x}{\sum_j e_j^x}$	$\frac{\partial f_i}{\partial x_j} = f_i(\delta_{ij} - f_j)$	$(0, 1)$	(yes)	(no)

of the output, which is, for example, necessary for multi-class classification, which is presented in Figure 6.8.

6.5 Training Process

In this Section, the training process is described. Before the training process, the training data have to be prepared. The essential thing about the data set used in the training process is that it contains typical data to be predicted and the corresponding result called ground truth. The training data set, therefore, has to be a labeled data set. It is crucial to have good labeled data because the neural network is trained on them.

6.5.1 Training Data Set

The neural networks described in this thesis were trained with simulated data. The input data are normalized or standardized to make different data more comparable, and, therefore, the training faster. Standardization reduces the influence of outliers and is calculated as [132]

$$x^0 \leftarrow \frac{x^0 - \mu}{\sigma} \quad (6.6)$$

where μ is the mean of the samples, and σ is the standard deviation. After the standardization process, the standardized sample's mean is zero, and the standard deviation is one. Normalization does not reduce the influence of outliers and is calculated as [133]

$$x^0 \leftarrow \text{factor} \cdot \left(\frac{x^0 - \min(x^0)}{|\max(x^0) - \min(x^0)|} - \text{shift} \right). \quad (6.7)$$

The minimum $\min(x^0)$ and the maximum $\max(x^0)$ are the extrema of the sample. The factor and the shift of the data are an optional hyper-parameter and depend mainly on the range of the input layer's activation function. The factor changes the scale of the variable, whereas the shift changes its position. The method of this scaling is another important hyper-parameter. The same scaling used during the training is applied to the data fed into the neural network.

6.5.2 Training Algorithm

The training algorithm is outlined in the following and contains eight steps:

1. **Random initialization:** The weights of the neural network are initialized randomly. The distribution of random data depends on the layer. The biases are initialized with zeros.
2. **Split of labeled data:** Before training, the labeled data are split into a so-called validation data set and a training data set. It is essential to separate these data. The ratio of validation data to training data is typically around 10% to 20%. The validation data set is used to check the accuracy of the prediction. The training data set is used for the actual optimization.
3. **Forward propagation:** In this step, one sample of the training data set is passed through the neural network, and thus, a prediction of the neural network is calculated. The result is the predicted output of the neural network to the corresponding sample.
4. **Calculation of loss:** The predicted output of the neural network is compared with the expected output, called ground truth for this sample. This comparison is done by the loss function (Section 6.5.3). The closer the loss function is to zero, the better is the prediction. The goal of the training is to minimize the loss function for all samples. The accuracy is also calculated. Even if it is not necessary for the training algorithm, it provides good monitoring for the user.
5. **Backpropagation:** The loss is propagated backward to the neural network starting from the output layer. In this step, every neuron of every layer receives a fraction of total loss, based on the relative contribution of each neuron to the output. This relative contribution depends on the weights and the connection of the neurons of the following layers. A detailed description of backpropagation can be found in Section 6.6.3.
6. **Updating parameters:** Weights and biases are updated in a way that their contribution to the loss and, therefore, the total loss is reduced. Updating the parameters is not done for each training sample individually but in batches of a certain size. This batch size depends on the machine's available amount of memory on which the training is performed.
7. **Validation Loss:** After each sample of the training data set has been passed through the neural network once, the validation data are used to test how successful the training step was. To get a valid statement, it

is important that the neural network does not know these data, i.e., has not been optimized for it. Typically, the accuracy and the loss function for the validation are the same as for the training.

8. **Iteration:** Each round of the whole set of training data is called an epoch. The amount of epochs is another hyper-parameter. Again, it is crucial that only the training data, not the validation data, is passed through the neural network again. It is possible to set up a so-called early stopping criterium to avoid unnecessary epochs, as the result is already good enough. This criterion is usually defined by a threshold of the loss function or the validation data set accuracy. The pure sequence with which the samples are passed again through the neural network is shuffled to avoid correlation by successive samples.

The amount of epochs is an important hyper-parameter. Using fewer epochs could result in a not well-trained neural network. Whereas using too many epochs could lead to overfitting. Overfitting means the neural network is able to predict the training data very well but does not understand the underlying structure (Figure 6.6). A larger amount of training data can help to reduce overfitting.

6.5.3 Loss function

The loss function \mathcal{L} compares the predicted output of each individual sample with the ground truth. The goal of the training and the optimizer is to update the neural network's parameters in a way that the loss function is minimized for all samples of the training data set. The loss function has to be problem-related. It is crucial to understand its behavior for a successful training and, therefore, a reliably working neural network. The only arguments to the loss function are the ground truth Y_{gt} and the prediction of the neural network $Y_{\text{predict}} = x^I$. Here, x^I is the output of the last layer of the neural network.

A detailed view of the loss functions included in the TensorFlow framework can be found at the TensorFlow's online documentation [130]. Often used loss functions are mean squared error, binary cross-entropy, and categorical cross-entropy.

One of the intuitive loss functions \mathcal{L} is the mean squared error which is mathematically defined as follows [134]:

$$\mathcal{L} = \frac{1}{n} \|Y_{\text{gt}} - Y_{\text{predict}}\|_2^2 = \frac{1}{n} \sum_{i=0}^{n-1} (Y_{\text{gt}, i} - Y_{\text{predict}, i})^2 \quad (6.8)$$

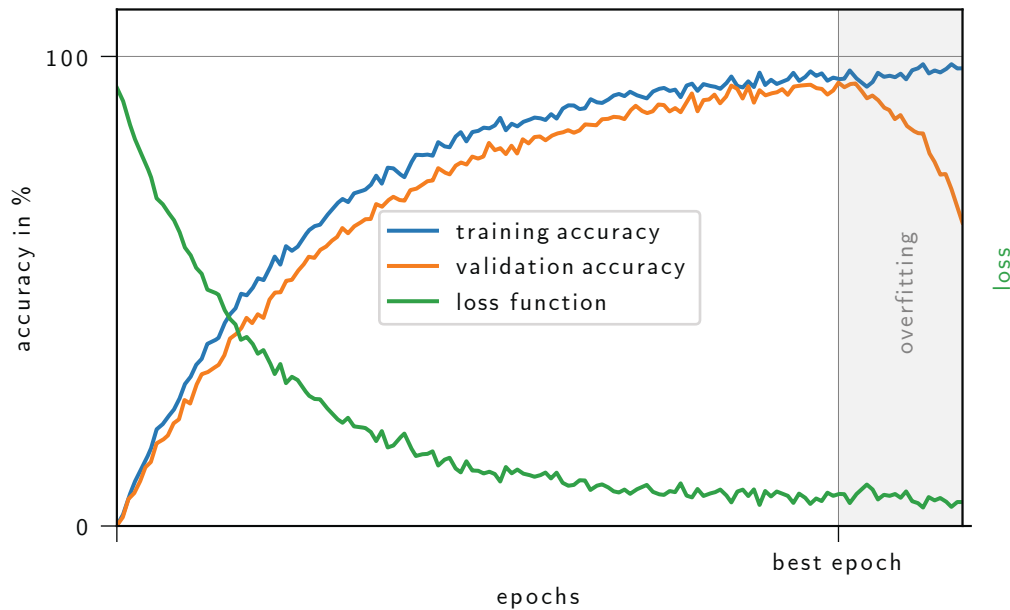


Figure 6.6: Ideal amount of epochs. The graph shows the amount of epochs on the horizontal axis and the accuracy and the loss after each epoch on the vertical axis. At a certain point, the accuracy of the validation data decreases again. This is the best point to stop the training process.

Here, $\|\cdot\|_2$ is the Euclidean norm, also known as L2 norm. n is the output size, and i denotes the index within the output, which can be interpreted as vector. The mean squared error calculates the square of the difference between each output of the output layer neurons and the ground truth (gt) individually and then takes the arithmetic mean.

To use the loss functions binary cross-entropy and categorical cross-entropy, the labels of the data set have to be one-hot encoded. Figure 6.7 shows the idea behind one-hot encoding. If there is one output node that can have different values (e.g. 0,1,2 for three classes), the classification is not one-hot encoded. The one-hot encoding transfers this classification problem to four output nodes where every node represents one class and can take either one for being in the class or zero for not being in the class.

To understand the principle of the loss functions binary cross-entropy (eq. 6.9) and categorical cross-entropy (eq. 6.10), one must think about binary classification, multi-class classification, and multi-label classification. Figure 6.8 shows an example for the different types of classifications:

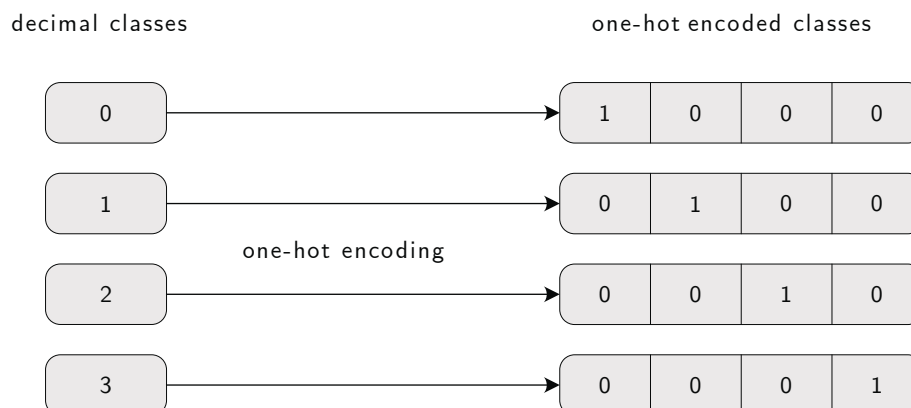


Figure 6.7: Principle of one-hot encoding with four classes. On the left are the not encoded classes in decimal notation. The required amount of output neurons to represent the different classes is one, and the output represents the class in decimal notation. On the right, the same classes are shown after one-hot encoding. The required amount of output neurons is one neuron per class. If the class is active, the corresponding neuron's value is one; otherwise, the value is zero.

- **Binary classification** divides the elements of a data set into two classes. These classes are complimentary. [101]
- **Multi-class classification** means that the classification task has at least three classes, and a sample can only be assigned to one class. Due to the conservation of probability, the sum over all output nodes for multi-class classification has to be one. [101]
- **Multi-label classification** means one example can be assigned to more than one class.

Binary cross-entropy is used for multi-label classification, whereas categorical cross-entropy and spares categorical cross-entropy are used for multi-class problems where only one result can be correct. Multi-class problems can be one-hot encoded. [101]

Multi-label classification cannot be represented by the decimal classes but only by one-hot encoded classes. Therefore, it is possible that more than one output node has a high output value. The sum over all output nodes for multi-label classification can be higher than one. The mathematical computation of the

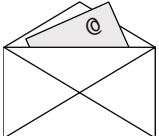









Binary classification	C = 3	Multi-class classification		Multi-label classification	
 Spam No spam	 (1, 0, 0)	Samples	Labels	Samples	Labels
	 (0, 1, 0)		(1, 0, 0)		(1, 0, 1)
	 (0, 0, 1)		(0, 0, 1)		(0, 0, 1)
			(0, 1, 0)		(1, 1, 1)

Figure 6.8: Types of classification. The three types are binary classification, multi-class classification, and multi-label classification. Binary classification can, for example, decide whether an e-mail is spam or no spam. In this example are three classes (C=3). The classes are one-hot encoded into (1, 0, 0) for the red sun, into (0, 1, 0) the blue moon, and into (0, 0, 1) for the yellow star. Three sample images and their corresponding labels are shown for both multi-class and multi-label classification. For multi-class classification, each sample contains exactly one class. For multi-label classification, the number of classes per sample can vary. Figure adapted from [135] and [136].

binary cross-entropy is done by [134]:

$$\mathcal{L} = -\frac{1}{n} \sum_{i=0}^{n-1} [Y_{gt, i} \cdot \log(Y_{predict, i}) + (1 - Y_{gt, i}) \cdot \log(1 - (Y_{predict, i}))] \quad (6.9)$$

Here, n is the output size, and i denotes the i^{th} index within the output. Whereas the ground truth Y_{gt} is zero or one, the prediction $Y_{predict}$ of the neural network contains values from zero to one for each class to represent the corresponding possibility. For binary cross-entropy, the activation function of the last layer has to be the sigmoid function (Table 6.1). The equation takes the arithmetic mean over all output nodes. The first part of the arithmetic mean represents the error made by the model of the labels, which should actually be true, whereas the second part handles the error of the nodes that should be actually false. The ground truth labels can either be one or zero.

The categorical cross-entropy is described by the following equation [134]:

$$\mathcal{L} = -\frac{1}{n} \sum_{i=0}^{n-1} \left\{ \sum_{c=0}^C [Y_{\text{gt}, i,c} \cdot \log(Y_{\text{predict}, i,c})] \right\} \quad (6.10)$$

Here, n is the output size, and i denotes the i^{th} index within the output, and c denotes the one-hot encoded classes. The equation sums the error for the different classes, and in a second step, it sums these errors over the output size. Due to the logarithm, the categorical cross-entropy is minimized for $Y_{\text{predict}, i} = 1$, which does not describe physical systems correctly. Therefore, the activation function of the last layer has to be SoftMax. The SoftMax layer guarantees to preserve the probability of one. The definition and properties of the SoftMax activation can be found in Table 6.1.

For multi-label classifications, binary cross-entropy like eq. 6.9 can be used. Each class can be treated separately as an independent binary classification. Therefore, multi-label classifications can be interpreted like many binary classifications.

The previous considerations about the loss function were only for single samples of the training data set. Using batches with a batch size greater than one, the used loss $\tilde{\mathcal{L}}$ for the optimization is the average over the values of the loss function of all samples in the batch:

$$\tilde{\mathcal{L}} = \langle \mathcal{L} \rangle \quad (6.11)$$

These loss functions are a good point to start, but a user-defined loss function has to be implemented in many cases.

6.5.4 Metric

A metric is used to quantify the model's performance and calculate the accuracy. While the loss function is used for optimization, the metric is a measure to help the user estimate how good the model is working. Typically, metrics are defined in a way that better predictions result in higher values. The TensorFlow framework provides some predefined metrics, but also user-defined problem-specific metrics are possible. The metric does not influence the neural network and is only a tool for the user.

6.5.5 Transfer Learning

Transfer learning is a method that improves the learning of a new problem by transferring already learned knowledge from a related problem [137]. It is a common method to save computation power and time during the training process. The intention behind transfer learning is that a network is trained for general purposes with a large amount of data. The resulting network can then be refined with a small modification of the network parameters for similar or more special tasks. For transfer learning, a pre-trained neural network is required. Typically, this model is trained with a large data set. Transfer learning can be implemented in two ways:

- **Full model retraining**, where each layer of the neural network is re-trained with the new data set, can take a significant amount of time and requires a big new data set to avoid overfitting. Full model retraining can be performed similarly to the normal training process. The only difference is that the initial parameters are not chosen randomly but are loaded from the pre-trained model. [138]
- **Last layer only retraining**, where only the last layer with the final classification is altered. The last layer only retraining is less computation power-intensive and requires a smaller new data set. Moreover, the topology of the last layers could be changed, and additional layers can be added at the end of the neural network. This is, for example, used in this thesis to vary the number of subpixels without changing the neural network's main task (Section 8.4.3). Only the last layers and the amount of output nodes is modified. [138]

For the last layer only retraining, two steps have to be processed [138]:

- **Feature extraction:** The pre-trained model is loaded, and the parameters of the layers which should not be retrained are kept constant during the retraining. Optionally, a new classifier trained completely new can be added on top of the existing pre-trained model. Typically, the total pre-trained network is kept constant if new layers are added. The complete neural network is part of the retraining process because the layers which should be modified depend on the frozen layers' specific output.
- **Fine-tuning:** This step is only necessary if additional layers are added to the end of the pre-trained neural network. A few of the last layers of the pre-trained models are subject to alterations. These

layers and the new layers are retrained jointly. This allows fine-tuning of the higher-order feature representations in the pre-trained model.

6.6 Optimizer

Training a neural network is an optimization problem. The algorithm which does the optimization process is called optimizer. In this subsection, the used optimization algorithm is introduced and shortly explained. One important step of all optimization algorithms is to calculate the function's gradient, which should be optimized. The gradient describes the direction in which the parameter should be updated to reach a minimum of a function. For neural networks, a commonly used method is called backpropagation. These update-rules are implemented in different optimizers. The optimizers should be problem-related and vary in stability and convergence speed. In this thesis, two of them are presented:

- The **gradient descent optimizer GD** [139] is a simple optimizer that is well suited for easy understanding and can be used as a showcase for the optimizer principle.
- The state-of-the-art **Adam optimizer** [140]. The Adam optimizer is used within the framework of this thesis. The pseudo-code of the algorithm can be found in Appendix C.1.

More information on other optimizers can be found in [134].

6.6.1 Gradient descent optimizer

The gradient descent (GD) uses only the function's first derivative (gradient) $\mathbf{g}_t \approx \nabla_{\boldsymbol{\theta}} f_t(\boldsymbol{\theta}_{t-1})$ to optimize the function. The gradient always points in the direction in which the value of the function increases. Therefore, the negative of the gradient is used to obtain the direction in which the parameters should be updated. The size of this step is determined by the value of the gradient and by the learning rate. Eq 6.12 shows the update rule for the parameter vector $\boldsymbol{\theta}$. [130]

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} - \alpha \cdot \mathbf{g}_t \quad (6.12)$$

Here, α is the global, defined learning rate.

A slight modification is the momentum optimizer. The momentum optimizer's concept accelerates the gradient descent optimizer by adding a fraction γ of

the update vector of the previous time step. Eq. 6.13 and eq. 6.14 show the modified update rules. Vector \mathbf{v} is called velocity, and $\gamma \geq 0$ is the global, defined momentum. [130]

$$\mathbf{v}_t = \gamma \mathbf{v}_{t-1} - \alpha \cdot \mathbf{g}_t \quad (6.13)$$

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} - \alpha \cdot \mathbf{v}_t \quad (6.14)$$

The frequently used stochastic gradient descent (SDG) works similarly to the gradient descent but is unlike the ordinary gradient descent stochastic. Instead of using all training data set samples, the stochastic gradient descent uses a subset of the training data set. This subset is called batch¹. As a consequence, the direction of the velocity \mathbf{v} , which describes the update of weights, has not to be perpendicular to the equipotential lines in the parameter space. However, many little steps of the stochastic gradient descent approximate the gradient over all samples [141].

The convergence behavior under mild conditions [142] can be found in [143] by using the Robbins-Siegmund theorem [144].

6.6.2 Adam Optimization Algorithm

The Adam [140] optimization algorithm has been designed specifically for training deep neural networks. It needs the gradient $\mathbf{g}_t \approx \nabla_{\boldsymbol{\theta}} f_t(\boldsymbol{\theta}_{t-1})$, which can be obtained from backpropagation (Section 6.6.3). Here, t is the index of the iteration. It takes advantage of the moving average of the gradient instead of the gradient itself. It also uses the variance, often referred to as second moment, of the gradient to scale the learning rate. Therefore, the corrected learning rate is computed for each parameter individually. The pseudo-code of the algorithm can be found in Appendix C.1. Adam uses the exponential moving average to approximate the first moment \mathbf{m}_t and the second moment \mathbf{v}_t of the batch with the gradient \mathbf{g}_t :

$$\mathbf{m}_t = \beta_1 \cdot \mathbf{m}_{t-1} + (1 - \beta_1) \cdot \mathbf{g}_t \quad (6.15)$$

$$\mathbf{v}_t = \beta_2 \cdot \mathbf{v}_{t-1} + (1 - \beta_2) \cdot \mathbf{g}_t^2 \quad (6.16)$$

The moving average vectors are initialized with zero, and the hyper-parameters β_1 and β_2 have default values of 0.9 and 0.999. The iterative equation can be

¹Traditionally, the stochastic gradient descent was used with only one sample. The batch size was one. However, today the stochastic gradient descent is used to get an approximation to the gradient on a subset of the training data.

rewritten as [145]:

$$\mathbf{m}_t = (1 - \beta_1) \sum_{i=0}^t (\beta_1^{t-i} \cdot \mathbf{g}_i) \quad (6.17)$$

$$\mathbf{v}_t = (1 - \beta_2) \sum_{i=0}^t (\beta_2^{t-i} \cdot \mathbf{g}_i^2) \quad (6.18)$$

The expectation value of \mathbf{g}_t is then a function of the first moment [145]:

$$E[\mathbf{m}_t] = E \left[(1 - \beta_1) \sum_{i=0}^t (\beta_1^{t-i} \cdot \mathbf{g}_i) \right] \quad (6.19)$$

$$E[\mathbf{m}_t] = E[\mathbf{g}_t] \cdot (1 - \beta_1) \sum_{i=0}^t \beta_1^{t-i} + \boldsymbol{\xi} \quad (6.20)$$

$$E[\mathbf{m}_t] = E[\mathbf{g}_t] (1 - \beta_1^t) + \boldsymbol{\xi} \quad (6.21)$$

Note that the superscripts of the parameters β_1 and β_2 are exponents and not upper indices. In eq. 6.20, the gradient \mathbf{g}_i is approximated with the gradient of the last step \mathbf{g}_t . The additionally made error is absorbed in $\boldsymbol{\xi}$.

To obtain eq. 6.21, we apply the geometric series [146] to eq. 6.20. The equation for the second momentum can be derived in the same way.

Comparing the expectation value of the gradient's mean $E[\mathbf{g}_t]$ and the gradient's variance $E[\mathbf{g}_t^2]$, which are the first and the second moment to $E[\mathbf{m}_t]$ and $E[\mathbf{v}_t]$, \mathbf{m}_t and \mathbf{v}_t have to be corrected by a factor. After the correction, estimators can be written as [145]:

$$\hat{\mathbf{m}}_t = \mathbf{m}_t / (1 - \beta_1^t) \approx E[\mathbf{g}_t] \quad (6.22)$$

$$\hat{\mathbf{v}}_t = \mathbf{v}_t / (1 - \beta_2^t) \approx E[\mathbf{g}_t^2] \quad (6.23)$$

Finally, we update the parameter vector $\boldsymbol{\theta}$ with those bias-corrected moving averages [145]:

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} - \alpha \cdot \hat{\mathbf{m}}_t / \left(\sqrt{\hat{\mathbf{v}}_t} + \epsilon \right) \quad (6.24)$$

The first moment representing the mean of the gradient is scaled by the inverse square of the second moment. Here, $\alpha = 10^{-3}$ is the user-defined global training rate and $\epsilon = 10^{-8}$ a parameter that avoids the division by zero. The specified values of α and ϵ are good default settings [140]. These steps are repeated until the parameter $\boldsymbol{\theta}_t$ converges. A detailed proof that the Adam

algorithm converges can be found in the Appendix of [140]. The pseudo-code of an implementation of the Adam optimization algorithm can be found in the Appendix C.1.

6.6.3 Backpropagation

The task of backpropagation is to obtain expressions for the partial derivatives of the loss function with respect to the weights and the biases of the neural network. The algorithm is called backpropagation because the error δ_j^i is calculated backward, starting with the output layer. A detailed derivation and mathematical proof can be found in [112]. A derivation of the gradient can be found in Appendix C.2. To use backpropagation, one has to make two assumptions about the loss function [113]. First, the loss function is averaged for individual training sets m :

$$\mathcal{L} = \frac{1}{n} \sum_{m=0}^{n-1} \mathcal{L}_m \quad (6.25)$$

Second, the loss function can be written as a function of the output \mathbf{x}^I of the neural network with I layers:

$$\mathcal{L} = \mathcal{L}(\mathbf{x}^I) \quad (6.26)$$

The loss function also depends on the desired output. But similar to the input to the neural network, the desired output is a fixed parameter and not a variable that the training process can change. To calculate the gradient, an intermediate step is required. In this intermediate step, the error δ_j^i of the j^{th} neuron in the i^{th} layer is calculated. A little perturbation Δz_j^i of the weighted input to the j^{th} neuron in the i^{th} layer changes the weighted input of the j^{th} neuron in the i^{th} layer:

$$x_j^i = f(z_j^i) \rightarrow x_j^i + \Delta x_j^i = f(z_j^i + \Delta z_j^i) \quad (6.27)$$

Using eq. 6.27 in eq. 6.26 we obtain eq. 6.28.

$$\mathcal{L} \rightarrow \mathcal{L} + \frac{\partial \mathcal{L}}{\partial z_j^i} \Delta z_j^i \quad (6.28)$$

Eq. 6.28 is similar to a Taylor expansion up to the first order[146]. However, z_j^i is not a variable but a complex function that depends on the neurons of the previous layers. A perturbation of a neuron in layer i influences the inputs of the neurons of layer $(i + 1)^{\text{th}}$ and the following layers and, therefore, the perturbation propagates through the neural network and causes an overall

change of the loss function. The goal is to choose δ_j^i in a way that the loss function is minimized. Here, two cases are possible:

- $|\frac{\partial \mathcal{L}}{\partial z_j^i}| \gg 0$
Here, Δz_j^i with an opposite sign in comparison to the derivative makes the loss smaller.
- $|\frac{\partial \mathcal{L}}{\partial z_j^i}| \approx 0$
A change Δz_j^i has no big influence on the loss function. In addition, the weighted input z_j^i is chosen in a way that the loss function is already close to the extremum for a variation of this weighted input.

This leads to the assumption that the partial derivative of the loss function with respect to the weighted input is a good quantity to measure the error indicated by a neuron. This motivates the following definition of the error δ_j^i influenced by the activation of the j^{th} neuron in the i^{th} layer:

$$\delta_j^i := \frac{\partial \mathcal{L}}{\partial z_j^i} \quad (6.29)$$

In the following, the vector notation is used:

$$\boldsymbol{\delta}^i = \frac{\partial \mathcal{L}}{\partial \mathbf{z}^i} \quad (6.30)$$

For the backpropagation algorithm, four equations are needed where the index I describes the last layer of the neural network [147]:

$$\boldsymbol{\delta}^I = \nabla_{\mathbf{x}^I} \mathcal{L} \odot \frac{df(\mathbf{z}^I)}{d\mathbf{z}^I} \quad (6.31)$$

$$\boldsymbol{\delta}^i = \left((w^{i+1})^T \right) \boldsymbol{\delta}^{i+1} \odot \frac{df(\mathbf{z}^i)}{d\mathbf{z}^i} \quad (6.32)$$

$$\frac{\partial \mathcal{L}}{\partial b_j^i} = \delta_j^i \quad (6.33)$$

$$\frac{\partial \mathcal{L}}{\partial w_{jk}^i} = x_k^{i-1} \delta_j^i \quad (6.34)$$

\odot is the so-called Hadamard product [148]. It is defined as the element-wise product of two vectors or matrices with equal dimensions. The result of the Hadamard product has the same dimensions as its inputs. $f(\mathbf{z}^i)$ is the activation function of the i^{th} layer, \mathbf{z}^i is the weighted input of the i^{th} layer, b_j^i the bias of the j^{th} neuron in i^{th} layer, x_j^i the bias of the j^{th} neuron in i^{th}

layer, and w_{jk}^i is the weight from the k^{th} neuron in the $(i - 1)^{\text{th}}$ layer to the j^{th} neuron in the i^{th} layer.

Eq. 6.31 describes the influence of an error of the output layer on the loss function. The gradient ∇_{x^I} is defined as a vector whose components are the partial derivatives $\partial/\partial x_j^I$ with respect to the output of the output layer. Equation 6.31 can be interpreted as a component-wise use of the chain rule because the activation x_j^I is the same as the output of the activation function applied on the weighted input $f(z^I)$.

Eq. 6.32 describes the error δ^i in terms of the error of the next layer δ^{i+1} . $(w^{i+1})^T$ is the transposed matrix of w^{i+1} . Supposing the error δ^{i+1} , eq. 6.32 moves the error backward by multiplying the transposed weights. The influence of the activation function of layer i is absorbed in the derivative, which is component-wise multiplied with the Hadamard product. The Hadamard multiplication propagates the error backward through the activation function. This leads to an expression for the error at the weighted input into layer i .

The influence on the error δ^i of the weighted input of every layer can be calculated by combining eq. 6.31 and 6.32. First, the error δ^I is calculated, then by applying the second equation recursively, the errors $\delta^{I-1, I-2 \dots 1, 0}$ are calculated.

Eq. 6.33 describes the rate of change of the loss function with respect to any bias in the network. The right side of eq. 6.33 is already known from eq. 6.31 and 6.32.

Eq. 6.34 describes the rate of change of the loss function with respect to any weight in the network. The right side can be interpreted as a multiplication of the activation of the input times the errors of the output of layer i . As a consequence, small activation leads to a small gradient. The optimization process of the weight with a small activation is slow.

The loss function depends directly on the output of the neural network. To calculate the influence of the weights and the biases of the hidden layer or the input layer, one has to apply the chain rule. A pseudo-code implementation of the backpropagation algorithm can be found in Appendix C.3.

6.6.4 Learning Rate

The learning rate is one of the most important hyper-parameters. The learning rate describes the change of the weights and biases after each batch. Low learning rates yield highly reliable results but slow down the training process because more steps are needed to reach the minimum. Too high learning rates can lead to large changes in the parameter in one step. This can lead to the optimizer overshooting the minimum and leads to a not converging or even

diverging training. An optimum training process uses relatively high learning rates at the beginning with a continuous decay during the optimization. This accelerates the training process while still allowing fine-tuning of the parameters at the end. Typically, an exponential decay or a repeated sequence from high to low rates of the learning rate is used. A repeated sequence of sawtooth patterns could lead to an additional improvement [149].

6.6.5 Visualisation of the optimization process

A simple classification problem with two features and two classes is assumed to get a visual impression of the different optimizers' behavior. The dots in Figure 6.9 show the training data set. Because the problem is linearly separable, no hidden layers are necessary. The input layer has two neurons representing the two features, and the output has one neuron. Since the bias is constant at zero, the number of trainable parameters is two. The colormap in Figure 6.9 indicates the prediction of the neural network. This very simple neural network is able to classify the training data correctly.

Instead of finishing the training process after several epochs or with an early stopping criterion as used in the normal training processes, the training is finished if the loss function is below 0.03. This fixed value enables a better comparison of the required iterations.

Figure 6.10 shows the visualization of the different optimizers in the parameter space.

The gradient descent (GD) steps with a momentum equal to zero update the weights perpendicular to the equipotential lines in the parameter space. Because one step of the gradient descent algorithm requires a prediction for each sample of the training data set, it can take a long time.

For a batch size of one, the gradient descent and the stochastic gradient descent are the same.

The gradient descent with a momentum larger than zero (GDwM) accelerates the optimization process compared to the GD but tends to overshoot the optimum due to its update rules. In the parameter space, this leads to oscillations near the minimum.

The Adam optimizer needs fewer iterations than the GD optimizer. Due to the adaptable adjustment of the learning rates, the Adam optimizer does not overshoot the minimum such as the momentum optimizer but requires a few more iterations.

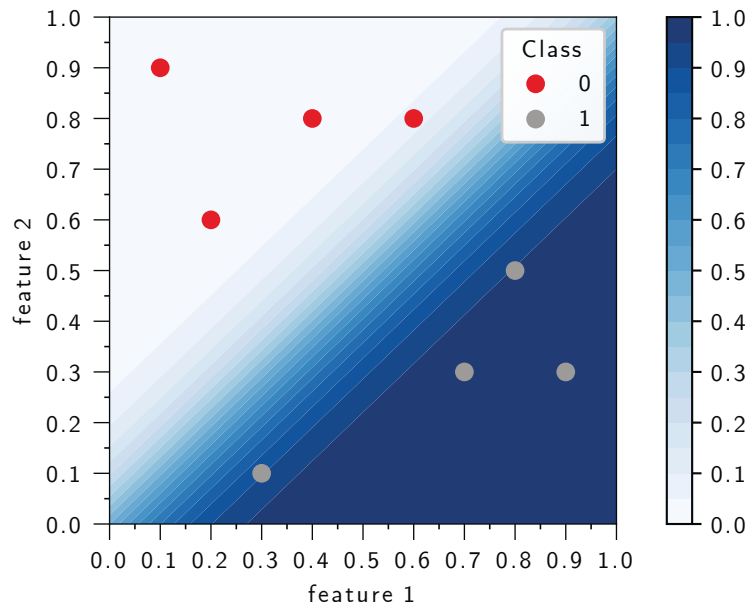


Figure 6.9: Representation of the training data set in the feature space. The dots describe the data of the training data set. A grey dot indicates data belonging to class 1. A red dot indicates data belonging to class 0. The colormap visualizes the prediction of the trained neural network.

6.7 Special Layers

Neural networks with dense layers are very powerful, but many parameters are required. The use of problem-related special layers increases the neural network's efficiency and accuracy, saving computation power and time. A list comprising all layers with their corresponding hyper-parameters used in this thesis' framework can be found in Appendix C.4.

In the case considered so far, the layers of the neural network are one-dimensional. In the following, the neurons will be arranged in a two- or three-dimensional structure to illustrate the utilized additional layer structures better. For example, the input layer's two dimensions are the pixel's two-dimensional position, represented by the corresponding neuron. In the case of color pictures, the third dimension of the data structure could be the magnitudes for red, green, and blue (RGB) values. One slice in the third dimension is called a feature map or channel. [150]

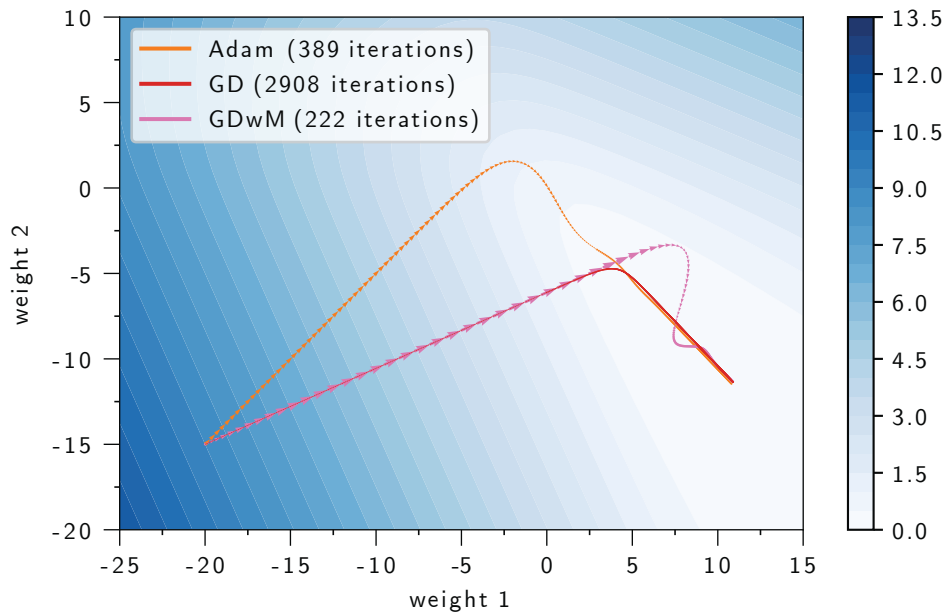


Figure 6.10: Optimization process in the parameter space. The colormap visualizes the loss function as a function of the two parameters, weight 1 and weight 2. The quiver plot shows the optimization results converging towards the minimum of the loss function with a learning rate of 0.5

6.7.1 Convolution

For image analysis, one of the commonly used operations are convolutions since they can be used for translation invariant structures.¹ An example of a two-dimensional convolution is illustrated in Figure 6.11. For simplicity, the example's bias is zero, and the activation function is the identity function. Here, we apply the same weights to multiple nodes of the input I . Their usage is very common in image processing. The light blue matrix is called the kernel K and is moved across the input feature map. Each element of the kernel is multiplied with the corresponding input node, summed up with a bias, and then used as an argument in the activation function. The bias is the same for all elements of one kernel. This procedure is done for each kernel position on the input feature map. For multiple input feature maps, a three-dimensional kernel is used. This means every input feature map has its own two-dimensional kernel, slid across the width and the height. [150]

¹An object or a feature in an image should be treated similarly no matter at which position it is located in the image.

The step size of the kernel moving across the input feature map is called stride and a hyper-parameter of the convolution. It can be a single value for the width and the height or can be individual for each dimension. Since particle tracks have no preferred direction, a single value for all dimensions is chosen in the following. [151]

Figure 6.12a shows the two-dimensional convolution for an input with four feature maps. For an input with multiple feature maps, the third dimension of the kernel has the same size as the number of feature maps of the input. The kernel is moved in the two spatial dimensions across the input.

Multiple convolutions with individual kernels are applied to generate multiple feature maps at the output. Figure 6.12b shows a generation of an output with six feature maps. Therefore, six kernels of the same size but with individual weights are used. The number of feature maps at the output is an additional hyper-parameter. In general, the individual feature maps of the hidden layers represent abstract features.

The last important hyper-parameter for convolutions is called padding. Padding describes the behavior at the edges of the input feature map of the kernel. For so-called same padding, the width and the height of the input and the output feature map are the same. This is achieved by adding (padding) rows and columns to the input. Preferentially, the rows and columns are added evenly at the edges of the input feature map. If this is not possible, the right or bottom edge gets one extra column or row. The entries of these additional rows and columns are usually zeros. In such a case, the padding is called zero padding. It is important to mention that by applying two or more convolutions behind each other, zero padding can lead to artifacts (Section 8.2). To avoid such artifacts, mirror padding or mean padding is used. Mirror padding is realized in two ways, either reflected or symmetric. In reflect mode, the padded regions do not include the borders, while in symmetric mode, the padded regions do include the borders. In mean mode, the average of each input feature map is used. [130, 151, 152]¹

The output feature layer size depends on the input feature layer's size, the size of the kernel, the stride, and the padding. A detailed description can be found in [153].

¹Of course, all other numbers and distributions of adding columns and rows are possible but unusual.

The convolution can be one-, two-, or three-dimensional. The dimension describes the number of slide directions. Three-dimensional convolutions additionally slide the kernel along the axis of the feature maps.

$$\begin{array}{c}
 \begin{pmatrix}
 \boxed{\begin{matrix} 2 & 1 & 2 \\ \times_1 & \times_0 & \times_1 \\ 2 & 0 & 1 \\ \times_1 & \times_2 & \times_0 \\ 0 & 1 & 1 \\ \times_1 & \times_0 & \times_1 \\ 2 & 0 & 3 \end{matrix}} & \begin{matrix} 1 \\ 2 \\ 1 \end{matrix} \\
 I & K
 \end{pmatrix} * \begin{pmatrix} 1 & 0 & 1 \\ 1 & 2 & 0 \\ 1 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \boxed{7} & 6 \\ 10 & 6 \end{pmatrix} \\
 \\
 \begin{pmatrix}
 \begin{matrix} 2 & \boxed{\begin{matrix} 1 & 2 & 1 \\ \times_1 & \times_0 & \times_1 \\ 2 & 0 & 1 \\ \times_1 & \times_2 & \times_0 \\ 0 & 1 & 1 \\ \times_1 & \times_0 & \times_1 \\ 2 & 0 & 3 \end{matrix}} & 1 \\
 I & K
 \end{matrix} * \begin{pmatrix} 1 & 0 & 1 \\ 1 & 2 & 0 \\ 1 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 7 & \boxed{6} \\ 10 & 6 \end{pmatrix} \\
 \\
 \end{array}$$

Figure 6.11: Computing the output values of a convolution. The input matrix I is shown on the top left corner, whereas K denotes the so-called kernel, which is multiplied. The sum of the multiplication produces the output indicated by the green square. The kernel is moved over the input matrix I producing the output O . In the example, the kernel size is (3,3), the strides that describe the step size are (1,1), and the bias is zero. The activation function is the identity function. There is no padding.

Another more practical description of a convolution layer is in the form of one matrix multiplication. For this, the input feature map and the output feature map, which are matrices, are written as vectors. The reshaping happens from left to right and then from top to bottom. The convolution can be written as a matrix. The width of matrix w is

$$w = \text{input height} \cdot \text{input width} \tag{6.35}$$

and the height of the matrix h is

$$h = \text{output height} \cdot \text{output width}. \tag{6.36}$$

After the matrix multiplication, the output vector is reshaped to the output feature map. The corresponding matrix C to the in Figure 6.11 shown convo-

lution is as follows:

$$\begin{pmatrix} w_{0,0} & w_{0,1} & w_{0,2} & 0 & w_{1,0} & w_{1,1} & w_{1,2} & 0 & w_{2,0} & w_{2,1} & w_{2,2} & 0 & 0 & 0 & 0 & 0 \\ 0 & w_{0,0} & w_{0,1} & w_{0,2} & 0 & w_{1,0} & w_{1,1} & w_{1,2} & 0 & w_{2,0} & w_{2,1} & w_{2,2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & w_{0,0} & w_{0,1} & w_{0,2} & 0 & w_{1,0} & w_{1,1} & w_{1,2} & 0 & w_{2,0} & w_{2,1} & w_{2,2} & 0 \\ 0 & 0 & 0 & 0 & 0 & w_{0,0} & w_{0,1} & w_{0,2} & 0 & w_{1,0} & w_{1,1} & w_{1,2} & 0 & w_{2,0} & w_{2,1} & w_{2,2} \end{pmatrix} \quad (6.37)$$

For reasons of clarity, not the values of the weightings but $w_{i,j}$ were taken. i and j are the index of the row and the column of the kernel.

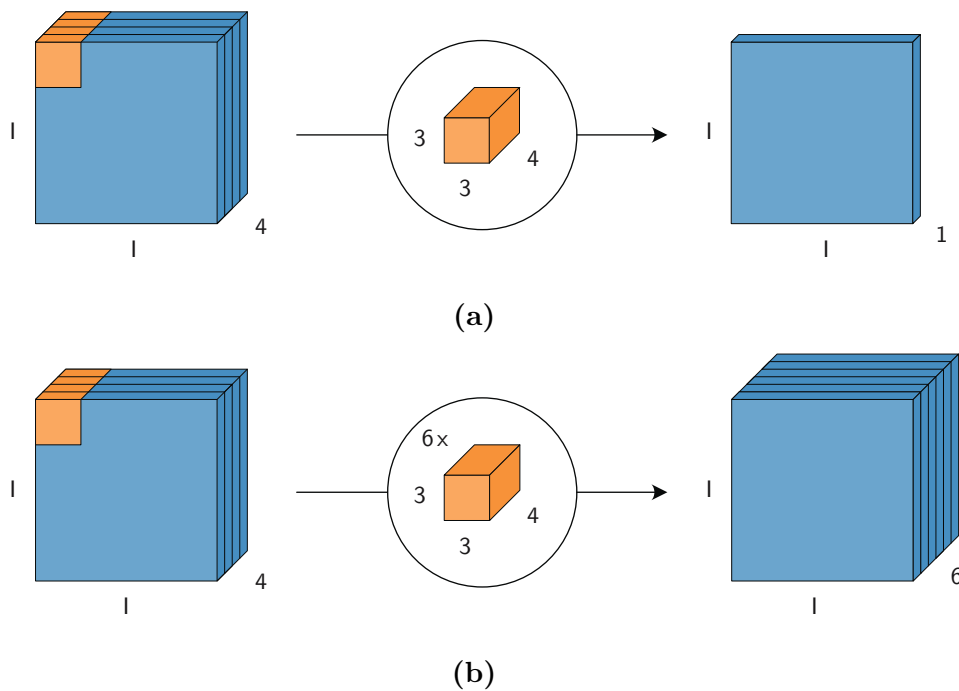


Figure 6.12: Two-dimensional convolution with same padding and multiple feature maps at the input. **(a)** One kernel creates one feature map. **(b)** To get multiple feature maps at the output, multiple kernels with individual weights and biases are used. In the example, six feature maps at the output are created. Therefore, six kernels with individual weights are needed.

6.7.2 Separable Convolutions

Separable convolutions do the same operation as the discrete convolution with significantly fewer multiplications and parameters. This leads to a reduced computing time of the neural network. The number of multiplications $\#_{\text{multi}}$ for all convolutions, no matter whether separable or not, is the following prod-

uct[154]:

$$\#_{\text{multi}} = w_{\text{kernel}} \cdot h_{\text{kernel}} \cdot \#_{\text{kernel}} \cdot \#_{\text{feature maps}} \cdot \#_{\text{vertical}} \cdot \#_{\text{horizontal}} \quad (6.38)$$

Here, w_{kernel} and h_{kernel} are the width and the height of the $\#_{\text{feature maps}}$ kernels, and $\#_{\text{vertical}}$ and $\#_{\text{horizontal}}$ are the number of vertical and horizontal slides of the kernel. The smaller number of parameters of the separable convolution leads to much faster predictions but could lead to lower accuracy. There are two types of separable convolutions which will be discussed in the following [155].

6.7.2.1 Spatial Separable Convolutions

This kind of separable convolution is called spatial because it separates the spatial dimensions (width and height) of the feature maps and the kernel. It divides the kernel into smaller kernels. An $N \times M$ kernel is divided into an $N \times 1$ and an $1 \times M$ kernel. Here N is the width, and M is the height of the kernel:

$$\begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix} \cdot (-1 \ 0 \ 1) \quad (6.39)$$

As one can see from the example, the matrix has to have a special characteristic, and not all matrices can be separated via spatial separation. This is a significant restriction during the training process, costing a lot of flexibility. Thus, spatial separable convolutions are typically not used for neural networks, but they are a simple example to understand the separations' concepts.

6.7.2.2 Depth-Wise Separable Convolutions

Figure 6.12 shows a simple convolution, and Figure 6.13 shows the same convolution performed as separable convolution. Depth-wise separable convolutions are not factorized into smaller kernels but split the kernel into two kernels in another way. The convolution is split into the so-called depth-wise convolution and point-wise convolution [155]:

- **Depth-wise convolution:** The three-dimensional kernel with dimensions $N \times M \times L$ is separated channel-wise to L times $N \times M$ kernels (Figure 6.13a). Here, N is the width, M is the height of the kernels, and L is the number of feature maps of the previous layer. Each of the new kernels iterates over only one of the feature maps of the previous layer. In contrast to the discrete convolution, the result of the multiplication

with the kernel has the same amount of feature maps as the input and not only one.

- **Point-wise convolution:** The result of the normal convolution always has the same number of feature maps as the input. A second convolution has to be applied to change the number of feature maps. This point-wise convolution has a kernel with the size $1 \times 1 \times L$ and iterates over every single node (Figure 6.13b). The number of output feature maps can be selected by the number of used kernels for the point-wise convolution (Figure 6.13c).

There are six kernels with a size of $3 \times 3 \times 4$ that moves $I \times I$ times for the normal convolution. This leads, for example, for $I = 64$ to

$$6 \cdot 3 \cdot 3 \cdot 4 \cdot 64 \cdot 64 = 884736 \quad (6.40)$$

multiplications. The number of multiplications for the separable convolutions is the sum of the multiplications for the depth-wise convolution and the multiplications of the point-wise convolution. There are four $3 \times 3 \times 1$ kernels and six $1 \times 1 \times 4$ kernels which are both slid $I \times I$ times. This leads only to

$$(4 \cdot 3 \cdot 3 \cdot 1 + 6 \cdot 1 \cdot 1 \cdot 4) \cdot 64 \cdot 64 = 147456 + 98304 = 245760 \quad (6.41)$$

multiplications which is more than a factor of three less multiplications compared to the normal convolution in eq. 6.40.

This strongly reduces the computing time resulting in a faster training process.

6.7.3 Pooling

The advantage of pooling layers is to reduce the amount of storage of the model and, therefore, increase the speed. The pooling does not reduce the accuracy of the network despite the data reduction. However, deeper networks can be created due to the low memory requirements. The deeper networks sometimes lead to better accuracy and allow to solve even more complex tasks. The main idea of pooling layers is to combine the output of neighboring neurons. The hyper-parameter for the number of combined neurons is called pool size. The hyper-parameter strides and padding can be chosen in the same way as for a discrete conventional layer. The combination of the outputs of neurons can be, for example, done by taking the maximum or the average, which is called maximal pooling or average pooling. An example of maximal pooling is shown in Figure 6.14. [156, 112]

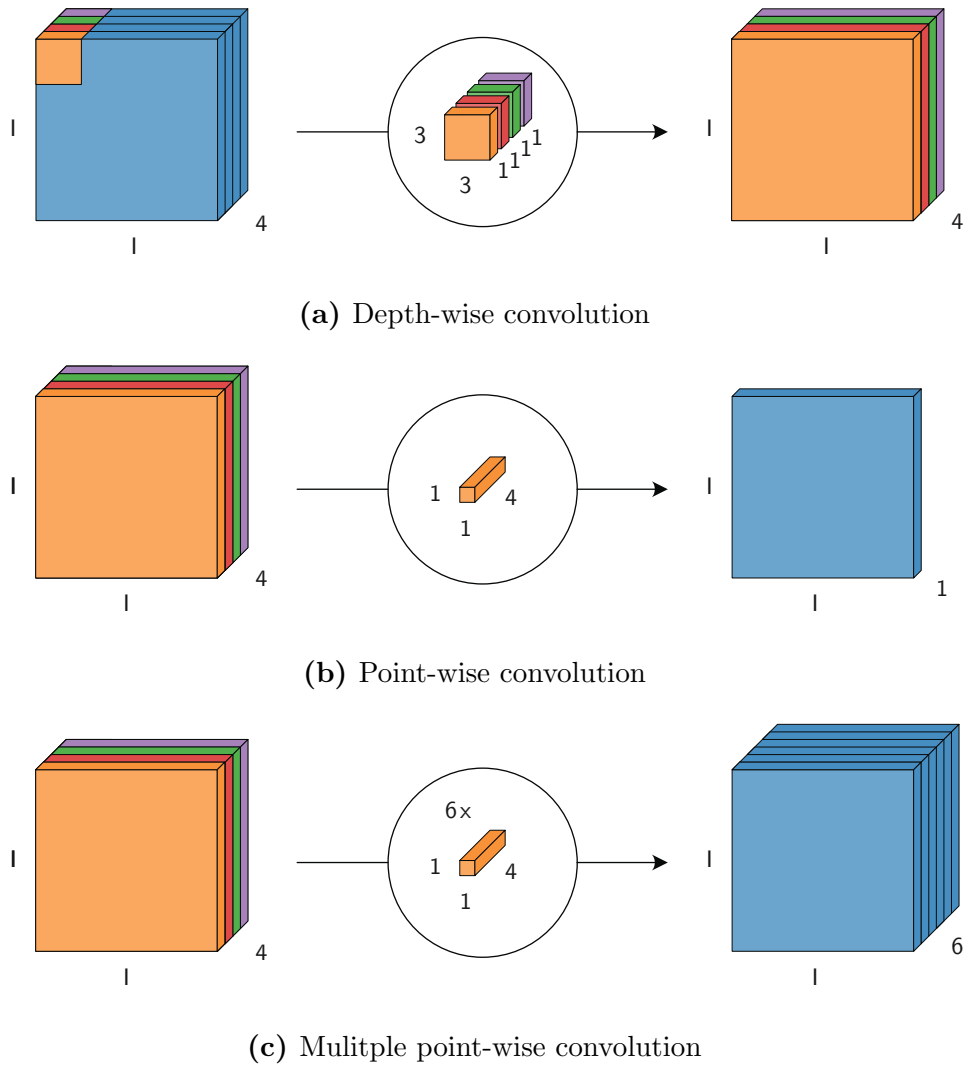


Figure 6.13: Two-dimensional separable convolution with same padding. (a) Depth-wise convolution with four kernels. Each kernel corresponds to one feature map at the input. (b) A point-wise convolution with one kernel results in one feature map at the output. (c) To get multiple feature maps at the output, multiple kernels with individual weights and biases are used.

6.7.4 Dropout

Dropout is a regularization technique to reduce overfitting (Section 6.5) during the training process. When training the network, a specified amount of neurons in each layer of the network are switched off and not considered for the batch [157]. These "dropped out" neurons are varied for each batch. The

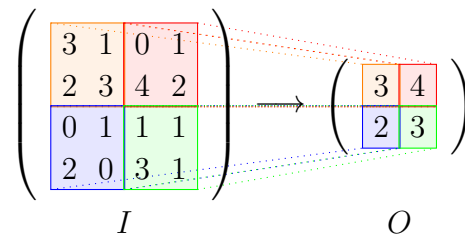


Figure 6.14: Maximal pooling with a pool size of two times two and a stride of the same size as the pooling.

dropout layer is only active during the training process and inactive during the prediction phase. This can lead to the effect that the validation set's accuracy during training is higher than the training set's accuracy. Conversely, there could be a smaller value of loss function (Section 6.5.3) for the validation set compared to the training set.

6.7.5 Transposed Convolution

Transposed convolutions are used to upscale an input and increase the number of neurons in each feature map. This means the spatial dimensions of the feature maps are increased. They are the transposed operation of the convolution [158]. As convolutions, transposed convolutions have a kernel size, strides, and padding as hyper-parameters. There are two conceptual descriptions¹ and a mathematical description to understand transposed convolution discussed in the following Sections [159].

6.7.5.1 Distributing Values Model

Interpreting Figure 6.11 from right to left results in the interpretation of a transposed convolution of Figure 6.15a, which shows this model schematically. In this interpretation of transposed convolution, the input in this Figure 6.15a is the output of Figure 6.11 and vice versa. In this conceptual model, the point of view is a single input value. In this case, the kernel describes how the value should be distributed in its neighborhood in the output. This is done for every input node. The individual amounts of different input nodes for the same output node are summed up.

¹Note that both conceptual models describe the same operation but from another point of view.

6.7.5.2 Collecting Values Model

The interpretation of a transposed convolution of Figure 6.15b shows the idea of the distributing values model. In contrast, in this interpretation, the point of view is a single output value. This interpretation is very similar to a discrete convolution. A single output cell sums up the weighted inputs that are distributed into it. The kernel again gives the rules and the weights for the distribution. Compared to the distributing values model, the kernel is flipped point symmetrically about the center. The flipped kernel is not a transpose of the kernel of the distributing values model, despite the name transposed convolution.

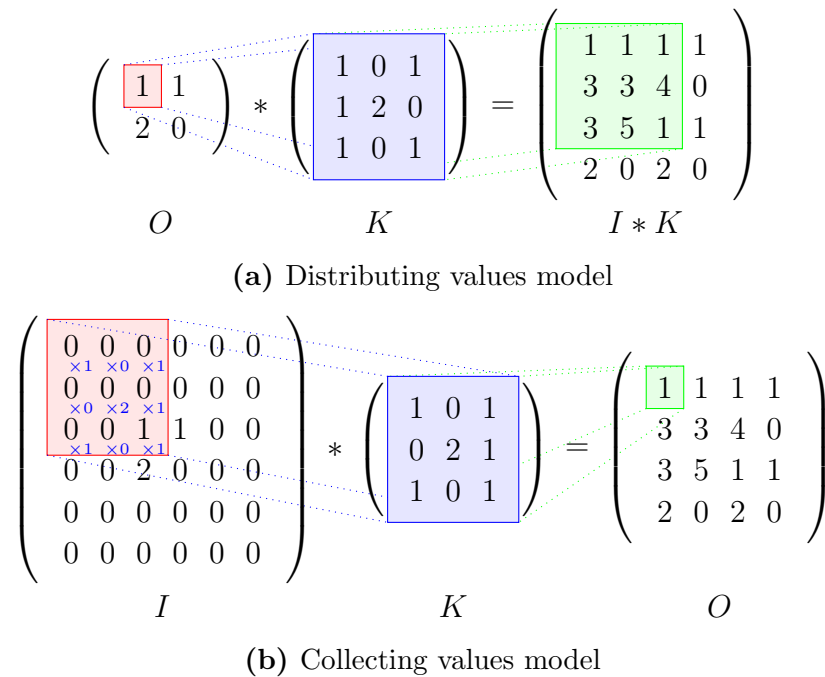


Figure 6.15: Computing the output values of the same transposed convolution with the two mental models.

6.7.5.3 Mathematical Description

For a discrete convolution matrix C (eq. 6.37), the corresponding transposed convolution matrix is labeled C^T . C^T is obtained from the mathematical operation of transposing C . To apply C^T , a similar reshaping as described in Section 6.7.1 is used.

6.7.6 Batch Normalization

The batch normalization layer normalizes its input. The mean of the layer's output is close to zero and the standard deviation close to one. [134]

Batch normalization works differently during the training process and the prediction interface.

During the training process, the batch normalization layer uses the mean $\mu(x^i)$ and the standard deviation $\sigma(x^i)$ of the current batch of inputs to normalize its output [130]:

$$x^i \leftarrow \gamma \frac{x^i - \mu(x^i)}{\sigma(x^i) + \epsilon} + \beta \quad (6.42)$$

Here, x^i is the input to the batch normalization layer of the current batch, γ is a learned scaling factor, β is a learned offset factor, and ϵ is a small constant used to avoid a division by zero. The parameters β and γ are trainable parameters.

For the prediction process after the training, the batch normalization layer uses the moving average $\tilde{\mu}$ of the mean and the moving average of the standard deviation $\tilde{\sigma}$ of the batches used during the training to normalize its output [130]:

$$x^i \leftarrow \gamma \frac{x^i - \tilde{\mu}}{\tilde{\sigma} + \epsilon} + \beta \quad (6.43)$$

$\tilde{\mu}$ and $\tilde{\sigma}$ are non-trainable parameters, which are updated each time the layer is called during the training process and are frozen for the prediction process. Their updating rules are shown in equation 6.44 and 6.45, with m being the momentum of the moving average, which is typically 0.99. As a consequence, only data with similar statistics to the data used for the training process are normalized correctly. [130]

$$\tilde{\mu} \leftarrow m \cdot \tilde{\mu} + (1 - m) \cdot \mu(x^i) \quad (6.44)$$

$$\tilde{\sigma} \leftarrow m \cdot \tilde{\sigma} + (1 - m) \cdot \sigma(x^i) \quad (6.45)$$

If the variance across the data samples is too high, the gradient of a small input is completely suppressed by a high input. Applying batch normalization ensures a constant mean and standard deviation. Therefore, the amount of variation over the different inputs is reduced, which potentially leads to a better and faster training process. [160] [161]

However, batch normalization adds extra noise to the training sample because any sample depends on the other samples of the batch. This additional

noise can have either positive or negative effects, depending on the neural network. The positive effect is a regularization effect. The optimizer always observes slightly different noise because the batch normalization is computed and applied over batches and not the entire training data set during the training process. This different noise acts as regularization and helps to avoid overfitting (Section 6.5). Since the noise is small, additional dropout layers as described in Section 6.7.4 are typically used. [161]

Batch normalization in combination with convolution layers works in a very similar way. The normalization is applied along the axis of feature maps to take the convolution character into account. As a consequence, each feature map has a single mean and a single standard deviation. [161]

6.8 8-Bit Quantization

To improve the storage and memory requirements and the speed of the neural network, a so-called 8-bit quantization is common after the training process. 8-bit quantization approximates the floating point values using the formula [162]:

$$\text{real value} = (\text{int8 value} - \text{zero point}) \cdot \text{scale} \quad (6.46)$$

A distinction is made between per-axis (also known as per-channel) and per-tensor quantization. Per-tensor quantization has one scale and one zero point for the complete tensor. Per-axis quantization has one scale and one zero point for each slice in the so-called quantized dimension. The arrays' size where the scales and the zero points are stored depends on the matrix, which should be quantized itself, but also on the quantized dimension. A post-training quantization could lead to a smaller accuracy of the model. Up to a certain point, this phenomenon can be counteracted by separating the last layer of the model and train only the last layer after the post-training quantization, similar to transfer learning. The last layer's retrained weights and biases soften the inaccuracy of the quantization of the previous layers. After this second training, the last layer is also quantized, and the layers are merged into one model.

However, for the in this work presented neural networks, the deterioration is not significant.

6.9 Edge Tensor Processing Unit

The edge tensor processing unit (edge TPU) is a specially designed ASIC optimized for the requirements of neural networks and low power consumption. The current generation can perform 4 trillion (fixed-point) operations per second (4 TOPS) per ASIC with a power consumption of 2 watts [163]. The edge TPU has roughly 8 MB of static random-access memory (SRAM) that can cache the model's parameter data. Ideally, the model's parameters, such as the topology, the weights, and the biases, should all be saved on the TPU SRAM, which ensures faster speed than fetching the parameter data from external memory. The ASIC's architecture requires an 8-bit quantized model, an 8-bit quantized input, and provides an 8-bit quantized output. The quantization of the input and dequantization of the output are not done by the TPU but by the host system. Since the edge TPU and the corresponding framework are in an early phase of development, not all kind of layers are supported to be run on it. If the neural network contains unsupported layers, it can still compile, but only a portion of the model will execute on the edge TPU. At the first point in the neural network graph where an unsupported operation occurs, the compiler partitions the neural network into two parts. The first part runs on the edge TPU, and the second part is executed on the CPU. This means not the ratio between the supported and unsupported operation of the neural network determines the speed, but the first unsupported layer's position. [163]

6.10 Workflow

In this Section, the workflow used for modeling a neural network is described. The necessary steps are shown in Figure 6.16. The first step is the design of the problem-related topology of the neural network. In this step, it is important to include the dimensions of the input data and to consider the desired output of the neural network. The topology should be kept complex enough to achieve good accuracy. However, more complex typologies lead to more parameters. This reduces the speed of the training process and, more importantly, the speed of the prediction. The choice between complexity and speed should be made problem-related. Next, one has to define the hyper-parameters. Especially, the loss function should be adjusted to the problem to ensure successful training. The trainable parameters such as weights and biases are now randomly initialized as 32-bit float numbers and must be optimized during the training process. The training data must be prepared for the training process. A detailed procedure of the training can be found in Section 6.5. It is important

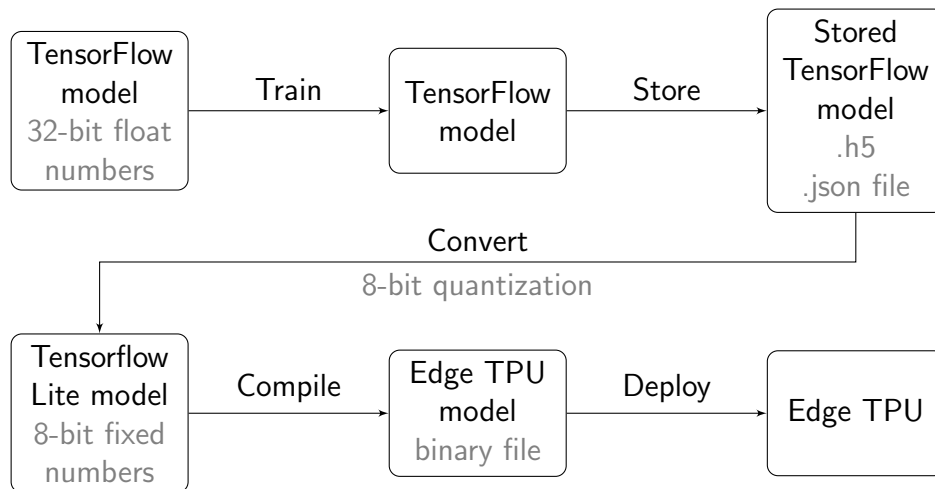


Figure 6.16: Typical workflow to provide a problem-related neural network. Figure adapted from [163].

to note that the training process is only necessary once per problem, apart from optimization processes to optimize the neural network's topology and hyper-parameters. Small changes in the settings that do not affect the whole network, such as the number of subpixels, can be realized by transfer learning. Since transfer learning does not require all parameters to be re-optimized, it is much faster. In principle, after this step, the model is ready to be used or tested, and we can start to optimize the accuracy. This optimization process is an iterative process that combines the knowledge of the topology, the hyper-parameters, and the accuracy from the last training processes.

The model with all parameters and important information is stored in two files to extend the model to other systems and other program languages (interfaces with existing analysis tools). This is done in a binary hierarchical data format (hdf5) file for the weights and the biases and a human-readable JavaScript object notation (JSON) file with the topology and other important information. The next steps are necessary to use the model on smaller devices, e.g., mobile devices or special hardware such as edge TPUs. The stored model is converted from two files to one binary file, which can be used by the TensorFlow lite framework [162].¹ An 8-bit quantization is performed during the conversion, which may require an additional training step (Section 6.8). This is necessary because the edge TPU only supports 8-bit fixed numbers [163]. The TensorFlow lite model can be compiled to an edge TPU model, which is a TensorFlow Lite model with edge TPU support. The model is deployed to the edge TPU

¹TensorFlow Lite is a framework for running the deep learning models on mobile devices, microcontrollers, and embedded devices with low latency [162].

during the first prediction. Therefore, the first inference on the edge TPU is slower because it includes loading the model into the edge TPU memory [163]. It should be noted that the reconstruction process is nothing else than a multitude of matrix multiplications, which can be parallelized easily to increase the performance.

Chapter 7

Reconstruction of the Point of Entry with a Compact Neural Network

In this Chapter, a lightweight neural network to reconstruct the PoE for event patterns is introduced. An event pattern analysis is required before the neural network can be applied to the data. This event analysis can be similar to the event analysis explained in Section B.1 or, in principle, also based on a neural network. The compact neural network (CoNN) can only handle single events. Compared to the classical analyses presented in Appendix B, this is not a limitation, as those can also only handle single events and not so-called pattern pile-up events.

The goal of the CoNN is to produce a hit-map in the subpixel regime, as shown in Figure 3.7e on page 42.

7.1 Network Architecture

The input to the CoNN is a stack of event patterns. The size of the pattern is uneven and typically three times three pixels for photons and low energetic electrons. Since the clusters for higher energetic primary electrons are larger, the pattern size for higher energetic electrons has to be larger. The pixel with the maximal energy deposition is the central pixel of the pattern, and all neighboring pixels are taken into account.

Before feeding the neural network with the patterns, the mean of each pattern is subtracted and afterward divided by its standard deviation. The standardization is described with eq. 6.6 on page 94. This standardization is reasonable as the reconstructed PoE should depend on the ratio between the

number of single events in the individual pixels and should not depend on the absolute energy deposition in a pixel.

The output layer of the CoNN for single events contains two output neurons. One neuron represents the normalized x-coordinate of the PoE, and the other one the y-coordinate within the central pixel of the pattern¹. The activation function of the last layer is the sigmoid function with a range between zero and one. The output of the first neuron represents the projection between the left edge and right edge of the central pixel to the range between zero and one. The normalization can be described with eq. 6.7 on page 94. For the normalization, the minimum is zero, and the maximum is the pixel size. Because the range of the sigmoid function is between zero and one, the normalization factor of eq. 6.7 on page 94 has to be one and the shift zero. This applies analogously to the upper and lower boundary and the second neuron.

A neural network that reconstructs event patterns analogously with two primary particles needs four neurons at the output layer. In general, two neurons at the output are required for each primary particle that contributes to an event pattern. This leads to several drawbacks of this neural network architecture.

First, every category of events needs its own trained neural network in this representation. This is computational power-intensive but achievable as it has to be done only once.

The second issue occurs during the training process and can not be fixed for events containing multiple particles.

Due to the neural network's structure, it is not invariant under the exchange of two incoming particles of one pattern. This can be understood by looking at the training process in detail. Assume we have an event pattern with two incoming particles. One of the particles hits the detector at the upper left of the event pattern and the other at the lower right. Due to the ground truth, the upper left particle is labeled with the first two neurons and the lower right with the last two neurons. With this sample, the neural network learns that the first PoE is in the upper left corner and modifies the weights. The same goes for the second PoE.

The next event pattern looks similar to the first one, but now the assignment of the ground truth is exchanged. For this sample, the first PoE is at the lower right border of the pattern. During the forward propagation of the

¹The PoE for photons and low energetic electrons is located in the pixel with the maximum energy deposition. Therefore, a representation limited only to the central pixel and not the complete pattern is sufficient.

second training step, the neural network predicts the PoE of the first particle in the upper left region as it learned from the training step with the first sample. It learns that the first particle has arrived at the bottom right when adjusting the weights. It changes the weights in a way that the predicted PoE moves from the upper left in the lower right direction. As a consequence, the predicted PoEs tend to be more centered in the middle of the pattern in comparison to the ground truth.

This behavior can be fixed partially by a different training data preparation. For example, the first particle always has the smaller x-coordinate of the ground truth. This solves the issue for the x-coordinate but not necessarily the issue for the y-coordinate. An example of a pattern that still leads to issues is a PoE at the lower left and an upper right PoE. There is no general preparation or representation of the training data, which can resolve this issue; hence it always leads to a loss of accuracy for multiple events.

Nevertheless, the CoNN has its applicability: Because of its low number of parameters, it is fast and leads to very good accuracy for single events.

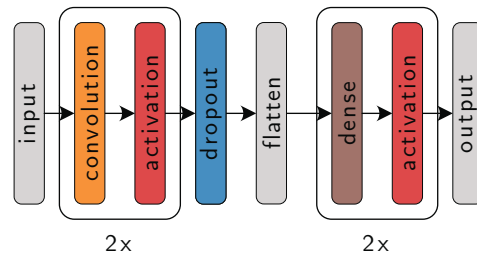


Figure 7.1: Schematic of the CoNN. The CoNN can be divided into a block that contains convolutional layers and a flat block. The convolutional layers can be optionally separated by a max pooling layer. The number of features of each convolution layer is 64, the number of neurons of the first dense layer is 100, and the number of neurons in the second dense layer is two.

The schematic of the CoNN is depicted in Figure 7.1. The network structure is a combination of a convolution part and a flat, dense part.

The first block contains two convolution layers (Section 6.7.1). These two convolution layers extract the ratio of relative energy deposition of the individual adjacent pixels of the event patterns.

The second part contains several fully-connected layers. The number of neurons of the fully-connected layers decreases from layer to layer down to two neurons for the output layer. The intermediate part consists of a dropout layer (Section 6.7.4) to avoid overfitting, and a flatten layer that

reduces the dimensions of the feature maps to one. A detailed list of the used hyper-parameters can be found in Table C.3 in Appendix C.5.

Except for the output layer, which uses the sigmoid function, the hyperbolic tangent is used as an activation function because it leads to the best results compared to the other activation functions. The hyperbolic tangent can be found in Table 6.1 on page 93, referenced as TanH.

The normalized output of the CoNN can be transferred back to physical coordinates in the frame with two steps. The first step is to apply the inverse function of the normalization. The output of this transformation is the physical coordinates of the PoE in the reference systems of the individual pixels. In the second step, these PoEs are transferred to the global reference system of the frame. Therefore, the positions of the patterns are required, which are stored by the housekeeping of the classical event analysis. The result is a list of the absolute positions and the frame identifier of each PoE in the reference system of the frame.

The individual PoEs are binned into virtual pixels and summed up over all frames to create an intensity image. The size and position of these virtual pixels are independent of the physical pixels. However, it is common to align the positions of the virtual pixels to the physical pixels. The size of these virtual pixels is independent of the compact network itself and limited by the amount of data and the accuracy of the reconstruction.

7.2 Training

The training and optimization process is implemented in a highly automated pipeline that is scalable to different machines. The data used for the training are analytically generated (Section 4.7). The analytically generated training data set consists of approximately 15 million event patterns and their corresponding PoEs. The samples are different in their PoE position and pixel-wise noise. Data obtained by measurements were not used for the training because their actual PoE is not easy to be obtained. Therefore, it is not possible to create an accurately labeled experimental data set.

The loss function is the mean square error since its square root is also the quantity that should be as small as possible at the benchmark. Compared to the mean absolute error, the mean square error penalizes large

errors more. The optimization process focuses on reducing large errors. The mean square error could lead to the behavior that many small errors are acceptable if this reduces the number of large errors. [164]

Especially, the treatment of events whose true PoE is close to the central axis is influenced by the selection of the loss function. The distributions of those events' reconstruction are asymmetric, with a large tail towards the pixel center. The selection of the loss function determines the influence of the events in the tails on the optimization process. The mean square error reduces the asymmetry and the tail and, therefore, increases the accuracy for larger uncertainties. A smaller asymmetry leads to a more homogeneous reconstruction (Section B.2.4).

7.3 Validation

In this Section, the validation for the CoNN is presented. For the validation, a simulated data set that was not used for the training process of the neural network is used. The same data set as in Chapter 5 is used to make the results comparable. The validation is divided into the accuracy and the performance of the CoNN. A detailed comparison with the classical methods and the accuracy with measured data can be found in Chapter 10.

7.3.1 Accuracy of the Neural Network

The accuracy of the CoNN is divided into two applications. These applications are the reconstruction of a sharp beam's position and a (summed) intensity image of many individual PoEs caused by a structure such as a grid. The evaluation for both applications is split into a normalized hit-map for homogeneous illumination and a spatial resolution map.

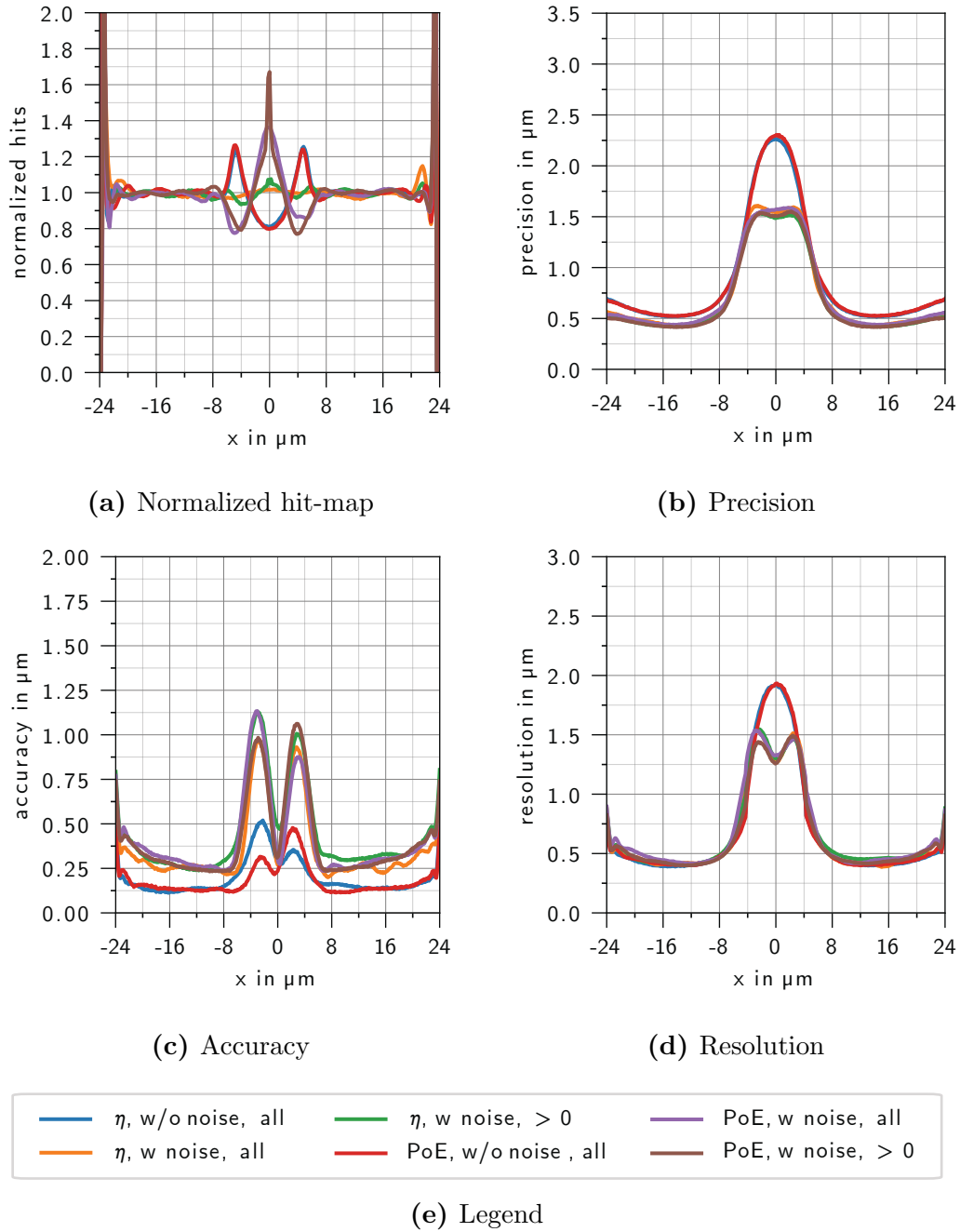


Figure 7.2: Accuracy of CoNN in terms of normalized hit-map, spatial accuracy, spatial precision, and resolution. The physical pixel size is $48 \times 48 \mu\text{m}^2$ with (0,0) in the center of the pixel, and the energy of the primary photons is 8048 eV. A non-correlated Gaussian distributed pixel-wise noise with a mean of 11.3 eV is assumed. The peaks near the pixel borders compensate for the normalized hit-map of zero on the pixel borders. A detailed description of the legend can be found in the text.

Like for the conventional methods in Section B.2.4, Figure 7.2 depicts the accuracy for CoNN. The legend refers to the data fed to CoNN during the training process. “ η ” means that η corrected values are used as ground truth, and “PoE” denotes the usage of the simulated PoEs for the training process. “w/o noise” means no noise is added to the training data, and “w noise” means a non-correlated Gaussian distributed pixel-wise noise with a mean of 11.3 eV is added to the three times three pattern during the training process¹. “all” describes a training process where all pixels of the pattern are used, and “>0” means that negative values are set to zero during the training and the prediction process.

Due to the design of the CoNN, the plotted quantities are not necessarily symmetric around the pixel center. However, these asymmetries are on a sub-micrometer level and are introduced by the training samples fed to the CoNN during the training process. Consequently, the asymmetries are different for different training processes, but the overall structure is independent of the training process and the same for all training processes.

Especially, the hit-map (Figure 7.2a) is sensitive to checkerboard artifacts introduced by the CoNN, but due to the binning of the subpixel structure, these fine checkerboard artifacts are averaged over the virtual pixel’s area and, therefore, reduced (Figure 10.7 on page 195). Since the CoNN is not able to correctly handle PoEs located exactly on the pixel border due to the sigmoid activation function, the number of hits is zero at the pixel borders. These PoEs are reconstructed near the pixel borders leading to a high number of hits of approximated three near the pixel borders. The peak at the pixel centers results from events that deposit their energy only in one pixel. Therefore, reconstruction at the pixel center is the best guess for the PoE. Except for the training, which neglects noise during the training process, the ground truth corrected by η leads to a homogeneous result in the pixel center.

The spatial precision is defined as the standard deviation of the individual reconstructed PoEs. The spatial accuracy is defined as the average distance between the mean position of the individual reconstructed PoEs and their ground truth. The resolution is defined as the average Euclidean distance between the individual reconstructed PoEs and their ground truth. A detailed comparison between the spatial precision, spatial accuracy, and resolution can be found in Appendix A.1.

The spatial precision (Figure 7.2b) and resolution (Figure 7.2d) are worse near the central axes of the pixel. As expected, CoNN leads to the best results in terms of resolution by using a training data set which is as close as possible to

¹Since a η correction is not necessary for noise-free data, the result of CoNN is the same as using no η correction neglecting fluctuations during the training process.

the measured data. The CoNNs trained with noise-free data show peaks with a spatial precision higher than $2\ \mu\text{m}$ and a resolution higher than $1.75\ \mu\text{m}$. In comparison, CoNNs trained with noise are approximately 25% better in terms of resolution.

Since the accuracy (Figure 7.2c) averages over many samples, the noise contribution is reduced by the number of samples. Therefore, the reconstruction of individual PoEs plays a minor role and the statistic average approximated by the noise-free PoE is of interest.

As expected, using η corrected values during the training leads to a worse result compared to the uncorrected values in terms of spatial precision, spatial accuracy, and resolution. However, using η corrected values with noise during the training leads to a homogeneous response and is, therefore, used for intensity images. The best result for intensity images is obtained using all values of the three times three pattern. For individual beam positions, using uncorrected PoEs with noise contribution leads to the best results. The best precision is obtained by applying no noise during the training.

The structures of the results created by CoNN look similar to the conventional methods (Section B.2.4). However, the results for the spatial accuracy, spatial precision, and resolution are better than the conventionally obtained results.

7.3.1.1 Individual Beam Positions

Figure 7.3a shows the normalized hit-map. Similar to the classical method (Figure 5.1a on page 83), the normalized hit-map shows a higher probability of a central hit for the x- and the y-dimension and a slightly lower probability near the axes. The inhomogeneity is more pronounced than for the classic method but spatially more localized.

The inhomogeneity occurs due to the presence of noise and can be explained analogously to the inhomogeneity in the classical method (Chapter 5). It is more pronounced since, due to the sharper distributions, the skewness of the reconstructed distributions is more significant. Figure 7.4 shows the distribution of the reconstructed positions for five different PoEs. The skewness is introduced by the same effects as for the classical methods (Section B.2.4). Due to the sigmoid activation of the last layer, the CoNN is not able to predict a PoE directly on the pixel border. As a consequence, the normalized hit-map is zero direct on the borders, and reconstructed PoEs are slightly shifted towards the pixel center resulting in a frame with higher hit density around each frame. This is expressed by a frame around the pixel containing zero hits and a slightly smaller inner frame around the pixel with a higher number of hits (Figure 7.3a). Since this effect is on sub-micrometer level, it normally introduces no artifacts for a subpixel reconstruction due to binning.

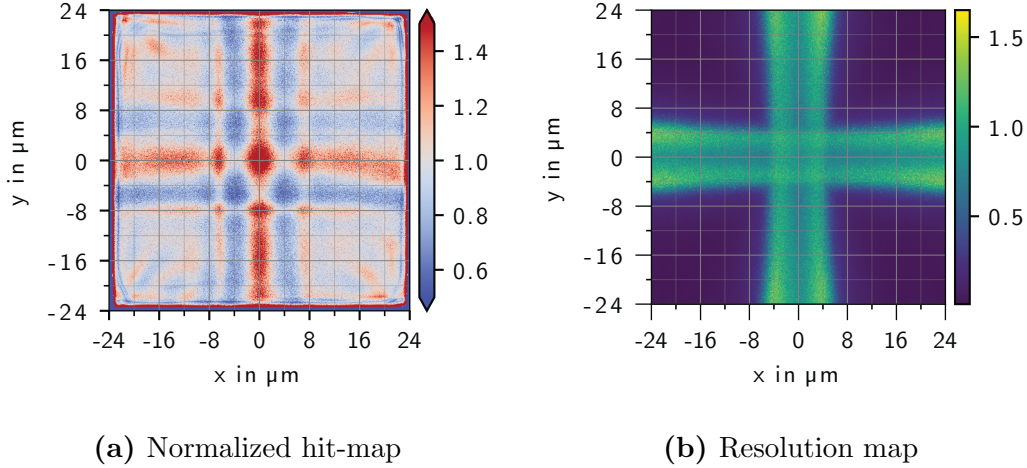


Figure 7.3: Normalized hit-map and resolution map for the CoNN obtained from a homogeneous illumination with approximately two million simulated primary particles. The physical pixel size is $48 \times 48 \mu\text{m}^2$ with $(0,0)$ in the center of the pixel, and the energy of the primary photons is 8048 eV. A non-correlated Gaussian distributed pixel-wise noise with a mean of 11.3 eV is assumed. These are the same parameters as for the presented measurement in Section 10.1.2. The hit-map's color scale is due to the normalization unit-less, and the color scale's unit of the resolution map is μm .

The resolution of the neural networks is measured spatially resolved within the pixel structure. Therefore, the Euclidean distance in the x- and y-dimension between ground truth and predicted PoE is calculated individually for each event. Since each event pattern contains exactly one primary particle and since the CoNN predicts exactly one PoE per event pattern, the assignment is unique. The obtained distances are averaged over events with the same true PoE and binned into a two-dimensional histogram (Figure 7.3b). In comparison to the classical method (Figure 5.1b on page 83), the spatial resolution obtained with the CoNN is better. Due to the skewness of the reconstructed distributions near the central axis (Figure 7.4), outliers exist, which are reconstructed. These outliers increase the averaged Euclidean distance between the ground truth and the predicted PoE, leading to a worse spatial resolution than central hits.

Figure 7.5 shows the reconstruction of 50 randomly chosen PoEs within the pixel structure by the CoNN. For every PoE, approximately 4000 event patterns are created. The only difference between those event patterns is the

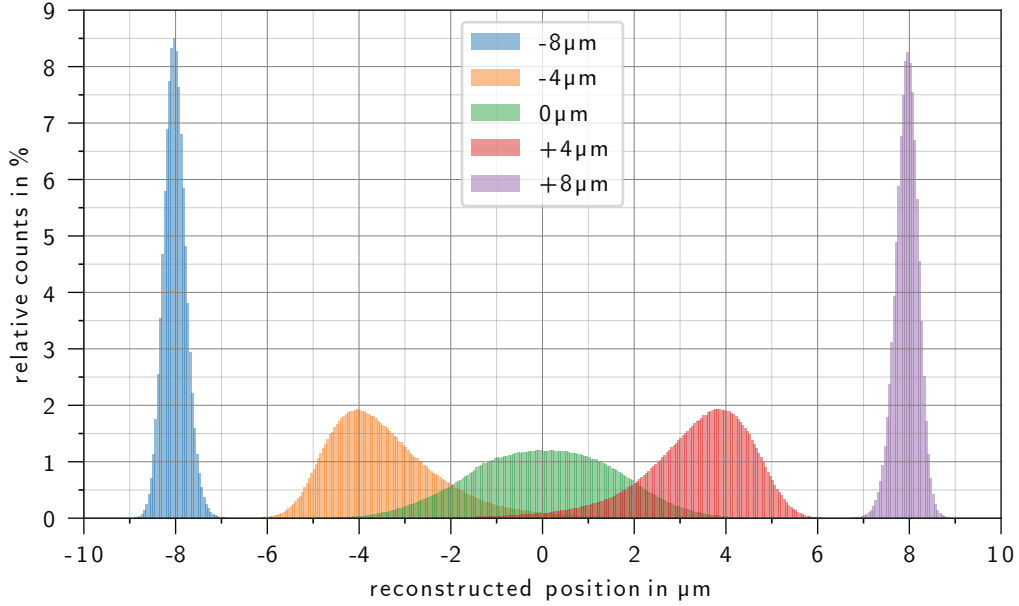


Figure 7.4: Reconstruction with CoNN of five chosen PoEs within the pixel structure. The physical pixel size is $48 \times 48 \mu\text{m}^2$ with $(0,0)$ in the center of the pixel, and the energy of the primary photons is 8048 eV. A non-correlated Gaussian distributed pixel-wise noise with a mean of 11.3 eV is assumed. The color code refers to the different PoEs. For the sake of clarity, only the central area of the pixel is shown.

contribution of the noise to the different pixels. Consequently, the histogram of the reconstructed PoE of the event pattern leads to broad distribution and not a sharp spot for each PoE position. The structure can be explained similarly to the classical reconstruction (Figure B.12 on page 251). Positions near the edges of the pixel structure are less sensitive to noise. Therefore, the spot created by many individual particles at the same PoE with different noise contributions is sharp (1 in Figure 7.5). Distributions of PoEs near the central axes are broad in the dimension in which they are near the central axis and sharp in the other dimension (2 in Figure 7.5). Distributions of PoEs in intermediate positions between the pixel border and the central axis cause a medium spread (3 in Figure 7.5). Distributions of PoEs near both central axes are broad in both dimensions (4 in Figure 7.5). In general, the distributions are broad in the dimension in which their position is near the central axis. The shape of those distributions depends on the relative position of the PoE within the pixel structure.

Especially the distributions of central hits are sharper than with the classical

reconstruction (Figure B.12 on page 251). However, the position-dependent shapes of the distribution look similar to the classical method since two events with different PoEs can have a similar event pattern due to the presence of noise. Therefore, event patterns can not be uniquely assigned to a PoE. The validation of the accuracy of the CoNN with measured data can be found in Section 10.1.1.

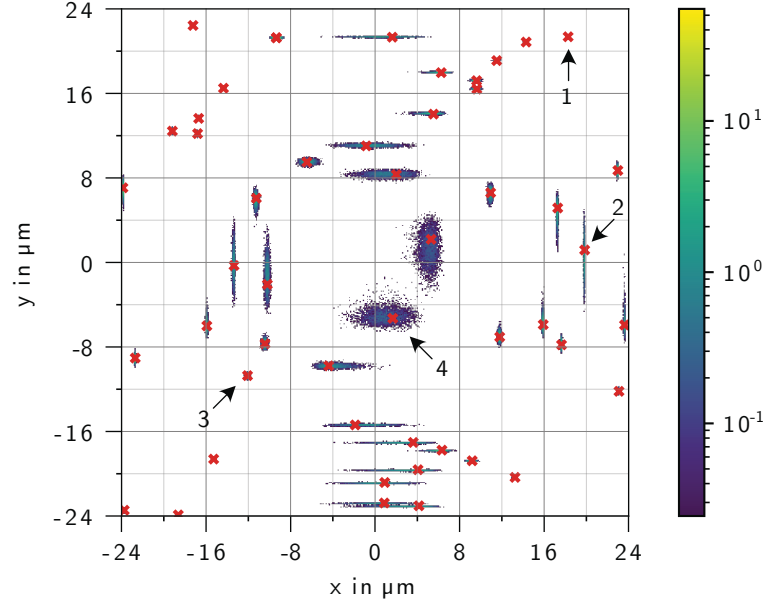


Figure 7.5: Reconstruction by CoNN with correction of 50 randomly chosen PoEs within the pixel structure. The physical pixel size is $48 \times 48 \mu\text{m}^2$ with $(0,0)$ in the center of the pixel, and the energy of the primary photons is 8048 eV. A non-correlated Gaussian distributed pixel-wise noise with a mean of 11.3 eV is assumed. For each PoE position, approximately 4000 primary particles are simulated. The red crosses indicate the positions of those PoEs, and the color map shows the amount of individually reconstructed PoEs on a virtual grid with a spacing of $0.1 \mu\text{m}$ in percent. The structure of the reconstructed distribution is exemplarily explained with the help of the four numbered PoEs in the text.

7.3.1.2 Intensity image of a structure

If the PoE of individual photons is not of interest but an intensity image of many photons without artifacts, the CoNN can be trained alternatively not with the actual PoEs but with PoEs shifted by $\eta(x)$ (Appendix B.2.3). $\eta(x)$ can be determined either by simulated or measured data with a homogeneous

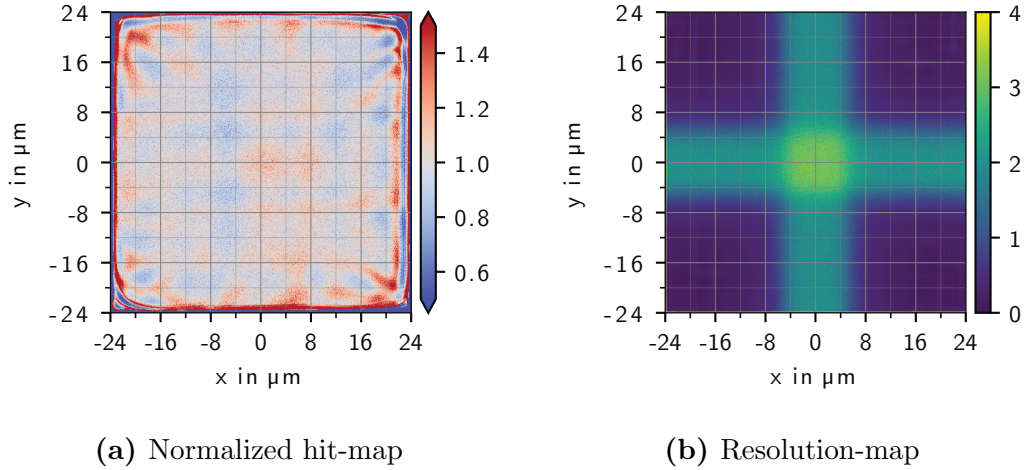


Figure 7.6: Normalized hit-map and resolution map for the CoNN obtained from a homogeneous illumination with approximately two million simulated primary particles. The training data are not the PoEs but the by $\eta(x)$ corrected PoEs. The physical pixel size is $48 \times 48 \mu\text{m}^2$ with $(0,0)$ in the center of the pixel, and the energy of the primary photons is 8048 eV. A non-correlated Gaussian distributed pixel-wise noise with a mean of 11.3 eV is assumed. These are the same parameters as for the presented measurement in Section 10.1.2. The hit-map's color scale is due to the normalization unit-less, and the color scale's unit of the resolution map is μm .

illumination. Like for the conventional methods, the result is a homogeneous image. However, the spatial accuracy worsens by applying this second correction since the reconstructed center of gravity method (with corrections) already describes the best guess of the true PoE. Figure 7.6 shows the normalized hit-map and the spatially resolved resolution map. In comparison with the CoNN, which is trained by the ground truth PoEs (Figure 7.3), the normalized hit-map is more homogeneous, but as expected, the resolution is worse. In comparison with the classical approach (Figure 5.1c on page 83 and Figure 5.1d on page 83), CoNN leads to a better spatial resolution but a slightly worse homogeneity. These artifacts in the homogeneity mainly occur at the pixel borders and become not so pronounced towards the center. However, this inhomogeneity is on sub-micrometer level and, therefore, introduces no artifacts for a subpixel reconstruction with a reasonable binning. Reasonable in this sense means a subpixel reconstruction in the order of 8×8 subpixels per physical pixel. The validation of the accuracy of the CoNN with measured data can be found in Section 10.1.2.

7.3.2 Performance

The evaluation of the performance of the CoNN can be divided into two parts.

The first part is the performance of the training process. Because the training process has to be done only once, the performance of the training process plays a minor role.

A suitable value is 60 epochs of training. On the GPU NVIDIA[®] GeForce[®] GTX 960 [165], one epoch with approximately 15 million samples and a batch size of 128 takes approximately 30 seconds, which leads to an overall training time of approximately half an hour. This short training time enables an easy and fast adjustment to other primary energies or other detector parameters.

Second, the performance and the associated computing time of the reconstruction process. Since the event analysis is part of both the classical analysis methods as well as the reconstruction with the CoNN, the necessary computational time is not considered. Only the PoE from the patterns is taken into consideration. On a NVIDIA[®] GeForce[®] GTX 960 [165] the reconstruction rate is approximately 11 MHz and on the edge TPU [163] the rate is approximately 10 kHz¹.

The model was run with TensorFlow lite [162] on a Raspberry Pi 3 Model B [166] to demonstrate the small resource requirements of the CoNN. On the Raspberry Pi 3 Model B, the reconstruction rate is approximately 1 kHz, which is faster than the conventional event pattern analysis (Section B.1). Therefore, even if the neural network is running on a Raspberry Pi 3 Model B, the bottleneck for the performance is the event pattern analysis. In Chapter 8, an approach based on a convolutional neural network that does not require an event pattern analysis in a previous step is presented.

¹The edge TPU and the TPU first need the model to be deployed. As a consequence, the reconstruction of the first batch is slower, since it also contains the step of deployment.

Chapter 8

Reconstruction of the Point of Entry with Convolutional Neural Networks

The reconstruction of the PoE for particles on pixel-level or subpixel-level without a previous event analysis can be achieved with a convolutional neural network (CNN). Due to the connectivity of the individual layers of the CNN, it is well suited for image processing [167]. The CNN enables the similar treatment of features, e.g. particle tracks, independent of their position in the frame¹. This is physically valid as the behavior of a particle should be independent of its position in the detector volume. The second advantage is due to the shared weights in one convolutional layer. As a consequence, all events, no matter what their positions are in the frame, contribute to the optimization of the same weights. This significantly reduces the training effort.

The output of the introduced point of entry neural network (PoENN), which is based on a CNN, is a hit-map representing the probability of a hit for each pixel in each frame. This automatically leads to a very unbalanced data set for the training process. Unbalanced data set in this context means that the probability of a pixel containing a PoE is much lower than the probability of a pixel containing no PoE. Especially for particles with higher primary energies, the energy deposition is distributed over many pixels (Figure B.1 on page 227). This means that if only events with one particle or pile-up events with a few particles are allowed, significantly more pixels have no PoE than a PoE. It becomes even more significant for smaller pixel sizes and subpixel resolution.

¹An exception is the border pixels of the active detector volume. Here, particle tracks can occur whose energy deposition is only partially detected.

Applying a threshold to the hit-map representing the probability leads to a binary hit-map. In this binary hit-map, each pixel can take two values. Zero if there is no PoE in the pixel predicted and one if there is a PoE predicted. The binary hit-map is similar to the output of a detector in counting mode as described in Chapter 3.4.2 and schematically looks like Figure 3.7d respectively Figure 3.7e on page 42 if super-resolution is enabled.

In terms of machine learning, the creation of hit-maps is a typical segmentation problem with two classes. Each pixel of each frame can contain or can not contain a PoE of a primary particle. For segmentation problems, the most commonly used loss functions are pixel-wise binary cross-entropy or pixel-wise categorical cross-entropy [168]. However, they can not handle this very unbalanced data set. Without high effort, the neural network learns that the loss function is small if it predicts that no pixel contains a PoE. But this does not achieve the desired result. Therefore, a newly developed loss function based on the confusion matrix is presented in Section 8.3.

8.1 Network Architecture

In this Section, the architecture of the developed PoENN is presented. The architecture is designed modularly to keep the structure flexible and simple. The core of the PoENN (Figure 8.1) is the u-net module that is built itself from nested submodules presented in Section 8.1.1. The u-net module is embedded by several layers described in Section 8.1.2 to realize a useful interface.

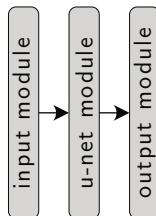


Figure 8.1: Architecture of the PoENN that contains the u-net module and the embedding. The individual modules are sequentially connected.

8.1.1 U-net Module

The structure of the developed PoENN is based on the u-net [152], which was initially developed to detect two-dimensional borders of structures for

biomedical imaging¹. The u-net consists only of convolutional parts and not fully connected layers to ensure the handling of images of nearly arbitrary size. Figure 8.2 shows the simplified structure of the u-net. In this example, its depth is four. The basic structure can basically be divided into three parts[152]:

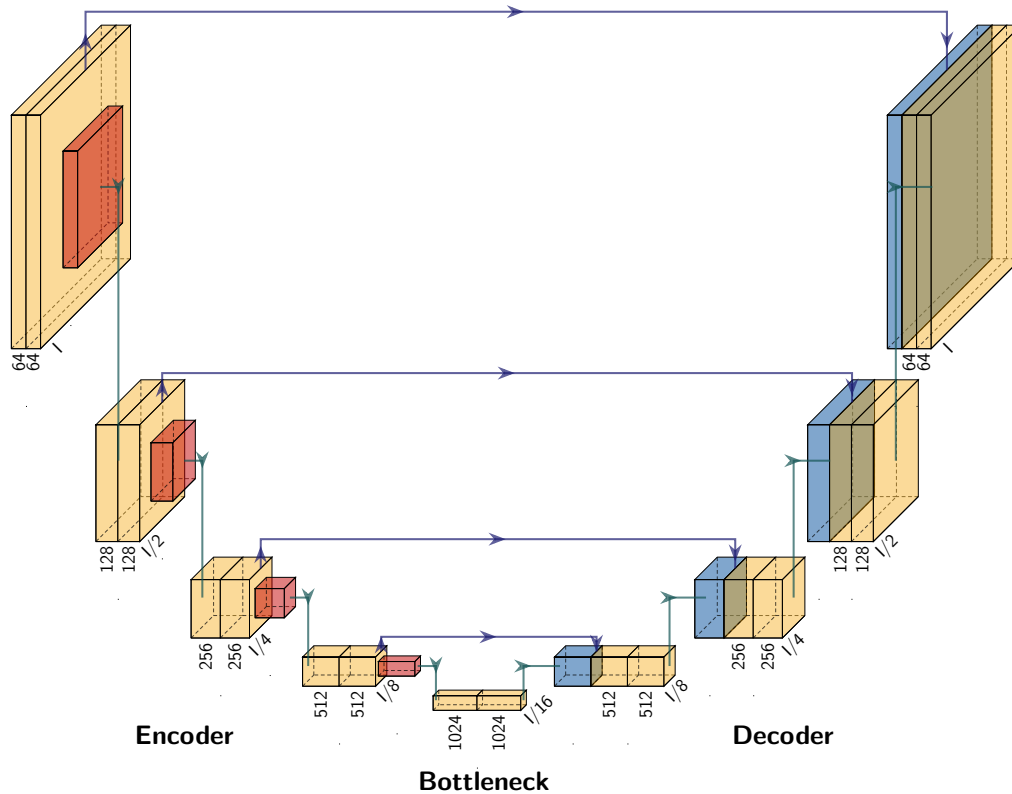


Figure 8.2: Simplified schematic of the u-net. The width and the height of the input image is I . The number under the layer denotes the number of features. In the shown example, the initial number of features is 64. The tilted number is the spatial width and height of the layer. The yellow layers denote two-dimensional convolutions, the red layers denote down sample processes, and the blue layers denote the upsampling processes. The blue lines show the shortcuts for the spatial allocation of the extracted features. The repetitive block structure is clearly visible. Figure adapted from [13].

¹In the following, the approaches aim to predict points. However, the architecture of the u-net is suitable since the prediction should be translation invariant and local. This means every pixel should be treated in the same way independent of its position within the detector, and only pixels spatially close to each other influence the prediction. The translation invariance is a valid assumption if the pixel-to-pixel variation caused by the detector system is small in comparison to the signal caused by the energy deposition.

The **encoder** is the first submodule of the neural network. It encodes the input and extracts features stepwise by repeated blocks. The deeper the block, the more abstract the features are. The number of feature maps is doubled with each repetition, whereas the spatial size is decreased. For the original u-net, the initial number of features is 64, such as shown in Figure 8.2 [152]. However, it turned out that using only 16 feature maps is sufficient (Section 8.5).

The **bottleneck** is the deepest point of the neural network. This layer has the most feature maps and the smallest spatial size. At this point, all information of the features is extracted, but only very little information where these features in the original frame occur is left. It connects the encoder with the decoder.

The **decoder** combines the extracted features with its spatial position. This is also done stepwise to enable a symmetric architecture. The number of feature maps is halved with each repetition, whereas the spatial size is doubled. The shortcuts between the decoder and encoder (blue arrows in Figure 8.2) guarantee the correct local allocation of the extracted features.

The structure of the u-net makes two requirements to the shape of the input. First, the number of rows and columns of the input must be a power of two, and second the spatial dimensions of the input must be large enough to perform all encoder steps.

One of the big advantages of a u-shape in comparison to other architectures is the relatively small amount of labeled data needed for the training process. This is due to of the efficient use of the training data set [152].

The following paragraph explains the technical realization of the u-net module. A list of the hyper-parameters used to define the network structure can be found in Table C.4 in Appendix C.5. Figure 8.3 shows the sequences used to build the u-net module. The encoder and the decoder are constructed from repeated submodules. The number of repetitions of the encoder submodule and the decoder submodule defines the depth of the u-net module.

The submodules again are built from convolution blocks (Section 8.1.3). Each block consists of the same arrangement of layers. These convolution blocks are supplemented by specific layers completing the submodules.

Figure 8.4a shows the sequence used for the encoder submodule. The max pooling has a pooling size of two times two and ensures a reduction of the spatial dimensions by a factor of two.

The bottleneck consists of a number of repeated convolution blocks, each followed by a batch normalization (Figure 8.4b). The number of feature maps and the spatial size of the feature maps stays constant over the whole

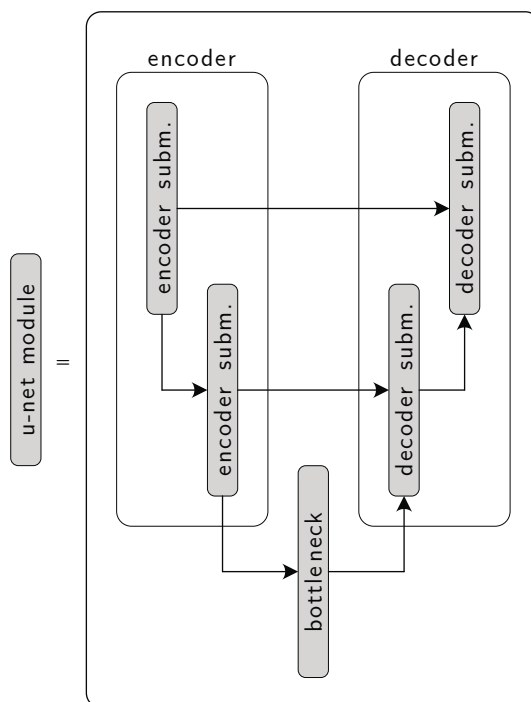


Figure 8.3: Architecture of the u-net module. Such as the u-net by Ronneberger et al. [152], the u-net module consists of an encoder, a bottleneck, and a decoder. The encoder and decoder consist of repetitive submodules. To ensure the symmetry of the u-net module, the number of submodules is the same for the encoder and decoder. This number is a hyper-parameter of the PoENN and by default two.

bottleneck.

The decoder submodule (Figure 8.4c) has two tasks. First, increasing the spatial size of the feature maps. This is achieved by upsampling (Section 8.1.5.2). Second, the merging of the extracted features with their corresponding spatial position, which is done by concatenating the feature maps from the corresponding encoder submodule with the feature maps of the previous decoder submodule. After each decoder submodule, the number of feature maps halves to maintain the symmetry with the encoder.

The depth of the original u-net is four which leads to a minimal input size of 16 times 16 [152]. However, in the framework of this thesis, a depth of two is sufficient (Section 8.5). A depth of two leads to four convolution layers (two times two) in the encoder plus at least two additional convolution layers in the bottleneck. Due to the limited spatial expansion of the individual events'

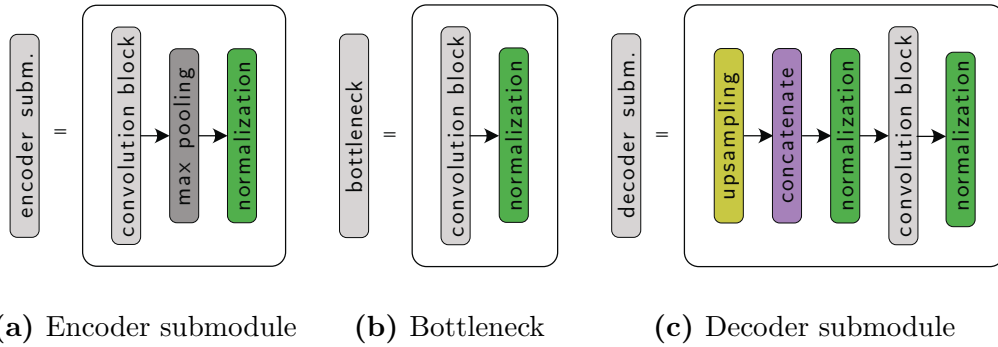


Figure 8.4: Architecture of the u-net’s submodules. The u-net’s submodules are the encoder submodule, the bottleneck, and the decoder submodule. The layers and blocks in each submodule are sequentially connected.

energy deposition, a deeper u-net structure and, therefore, the capability to investigate larger structures is unnecessary and requires only additional performance. The minimum input size of the u-net with a depth of two is four times four. However, smaller input sizes and, respectively, smaller frame sizes lead to an increased ratio between border pixels and not-border pixels of the frame. Events near the frame’s border potentially deposit a fraction of their energy outside the frame. This undetected energy is not taken into account by the neural network and leads to a lower accuracy. [169]

8.1.2 Embedding of the u-net Module

The u-net module is embedded by several layers, which handle the interface between the input of the neural network and the input of the u-net module and the output of the u-net module and the output of the neural network.

The input to the neural network has a dimension of three (batch size, frame width, frame height). However, the convolution layers require a four-dimensional tensor (batch size, frame width, frame height, feature maps). Therefore, the input module of the neural network expands the dimension by one (Figure 8.5a).

The output module contains two layers plus an activation function (Figure 8.5b). The first layer reduces the number of feature maps to one. The activation of the last layer is sigmoid (Table 6.1 on page 93). Therefore, the range of the output neurons is between zero and one, representing the probability of containing a PoE for the individual pixels. The third layer of



Figure 8.5: Architecture of the necessary embedding modules. These modules are necessary to ensure a smooth IO.

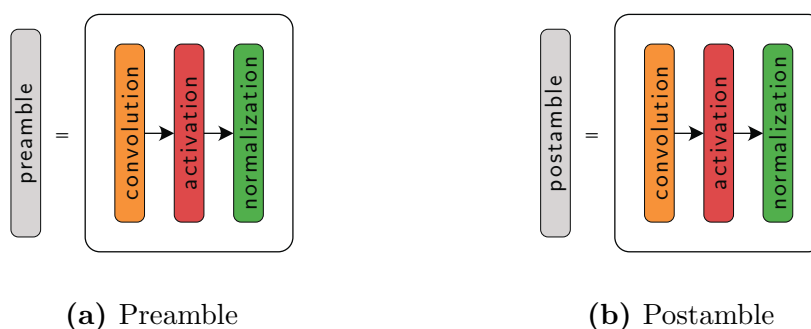


Figure 8.6: Architecture of the optional embedding modules. These optional embedding modules surround the u-net module.

the output module reduces the dimension by removing the feature dimension to get a three-dimensional output (batch size, frame width, frame height).

Additionally, a preamble can be used.¹ This preamble adds an additional convolution block in front of the u-net module to increase the number of channels at the neural network’s input. To keep the neural network symmetric with the preamble, also a postamble is added after the u-net module. The sequence of the preamble and postamble is shown in Figure 8.6. The preamble and the postamble are deactivated by default and activated if the super-resolution module is used (Section 8.1.4).

¹See Table C.4 in Appendix C.5 for the hyper-parameter for the PoENN implemented in the framework of this thesis.

8.1.3 Convolution Block

The convolution blocks always have the same structure (Figure 8.7) and differ only by the number of feature maps. The convolution block consists of two convolution layers as discussed in Section 6.7.1 with a default kernel size of three times three, followed by a rectified linear unit (ReLU) activation (Table 6.1 on page 93). By default, the two convolution layers are separated by a batch normalization (Section 6.7.6) and a dropout layer (Section 6.7.4), which is only active during training.

The hyper-parameters of the convolution block are the applied activation

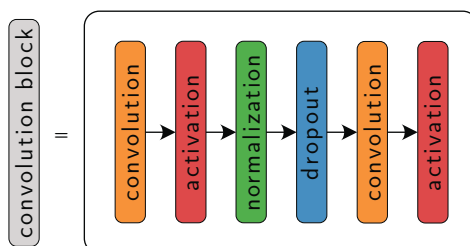


Figure 8.7: Schematic of the convolution block.

function and the kernel size of the convolution layers (Table C.4 in Appendix C.5). Additionally, the batch normalization can be switched off and on, and separable convolutions can replace the convolution layers (Section 6.7.2).

8.1.4 Super-Resolution Module

One way to achieve a better spatial resolution is to introduce subpixels. This means dividing the physical pixel into smaller virtual pixels. The dividing process is called upsampling. In the context of CNNs, this upsampling can be realized by combining several layers (Section 8.1.5.2). In the context of neural networks, increasing the resolution by using subpixels is known as super-resolution (SR) [170].

The SR module handles the upsampling process and extends the u-net module on the right side with several layers, and makes it asymmetric around the bottleneck. It acts similar to the decoder of the u-net module, but the corresponding counterpart of the encoder is missing. In the overall neural network structure (Figure 8.1), the SR module is supplemented between the u-net module and the output module, as shown in Figure 8.8. The additional

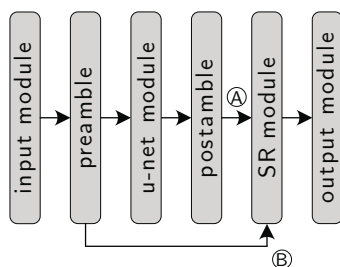
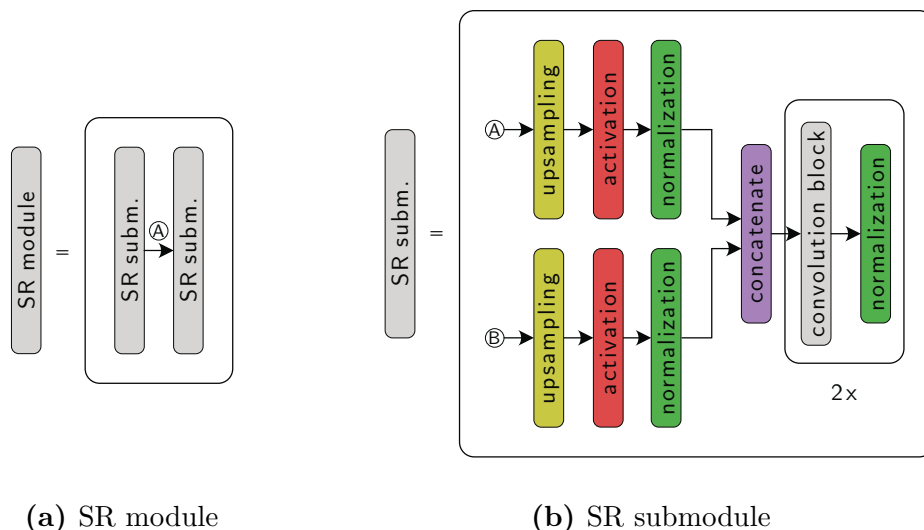


Figure 8.8: Architecture of the neural network for PoENN with super-resolution extension. The SR module gets the output of the postamble (A) and the output of the preamble (B) as inputs.

hyper-parameters can be found in Table C.5 in Appendix C.5.



(a) SR module

(b) SR submodule

Figure 8.9: Architecture of the SR module and the SR submodule. In this example, two SR submodules are repetitively applied to build the SR module. The input at (A) is from the postamble or a previous SR submodule, respectively. The input (B) is the output of the preamble.

The splitting of the physical pixel into the virtual subpixels can be done recursively step by step by using multiple repetitive SR submodules or in one big step by using one SR submodule. The idea behind the step-by-step approach is that performing many simple steps is easier learnable than performing only one complicated step.

Technically, this is achieved by the SR module [169]. In the case of a recursive expansion into the subpixel regime, the SR module consists of repetitive sequences of SR submodules. In the case of an upsampling in one step, only one SR submodule is used.

The SR module takes as input the postamble's output (A in Figure 8.8). The SR module additionally gets output of the preamble (B in Figure 8.8) to enable the spatial allocation. This can also be seen as a very large shortcut over the total u-net module.

The first SR submodule takes the same inputs as the SR module (Figure 8.9). Each subsequent SR submodule takes the output of the previous one as input as schematically shown as input A in Figure 8.9a. The inputs to the SR submodule are upsampled (Section 8.1.5.2). The factor of the individual upsamplings has to be accordingly adjusted. Therefore, a preceding concatenation before the upsampling is, in general, not possible. The upsampled feature maps are merged via concatenation. The concatenation is followed by two convolution blocks (Section 8.1.3) by default.

8.1.5 Upsampling

The goal of upsampling is to increase the spatial size of a feature map. Upsampling is required in the decoder of the u-net module and the super-resolution module.

8.1.5.1 Conventional Upsampling Methods

The conventional counterpart of super-resolution in machine learning approaches is interpolation.

Many of the upsampling methods in the context of neural networks make use of these conventional approaches.

All upsampling operations implemented as interpolation methods are algorithms that expand discrete data from an old grid to a new finer grid. Figure 8.10 shows the pixel structures of the different subpixel levels and their corresponding grids. The original grid is fixed due to the pixel structure of the detector system. Typically, each pixel is divided into an even number of subpixels in each dimension. Therefore, the new grid does not contain the grid points of the old grid.

The **nearest neighbor interpolation** is the simplest approach. This method simply determines the nearest neighbor in the old grid of a grid point in the new grid and assumes the value is also valid for the new grid point.

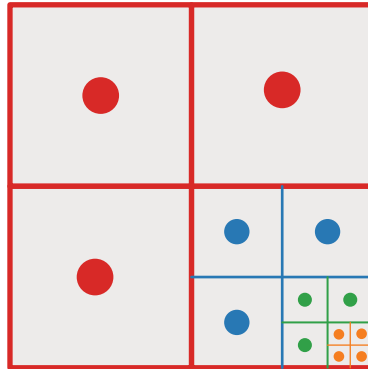


Figure 8.10: Pixel structure and grids for different subpixel levels. The squares indicate the pixel structure and the corresponding grid points as circles. The red pixel structure is fix due to the pixel structure of the detector system. The blue pixels have a 2×2 subpixel representation, the green structure has a 4×4 , and the orange has an 8×8 .

The **bilinear interpolation** is the extension of the linear interpolation for two-dimensional grids. It is a sequence of two linear interpolations in each dimension with one linear interpolation for each dimension. Although the individual operations are linear, the result of the bilinear interpolation is not linear. The bilinear interpolation takes the nearest neighbor and the second nearest of each of the two dimensions into account. The weighting of the four used values results from the distance between the new grid point and the old grid points.

The **bicubic interpolation** takes 16 pixels (four times four pixels) into account. The bicubic interpolation consists of 16 determined coefficients for each reconstructed point. These coefficients can be determined by solving a linear system of equations. The system of equations can be built by using the values, the first and the second (mixed) partial derivatives, which are evaluated at the physical adjacent pixel positions. Compared to the nearest neighbor interpolation and the bilinear interpolation, the bicubic interpolation is more computationally intensive, but the result is smoother and has fewer interpolation artifacts. [171]

There are also kernel-based upsampling methods. The kernel determines the weight of the surrounding points of the old grid. The window describes the used points in the old grid for each point in the new grid. Three kernel-based upsampling methods are briefly discussed in the following. For a detailed discussion, the reader is referred to [172]. The used **Gaussian interpolation**

uses a Gaussian kernel with a window of three and a sigma of 0.5 [130]. The **Mitchell-Netravali Cubic filter** is designed especially for lacking proper prefiltering synthetic images [173].

For the **Lanczos kernel**, the normalized sinc function¹ multiplied by the window is used [175]. The used size of these windows is a radius of three (Lanczos3) or a radius of five (Lanczos5). The Lanczos filter leads to good results but may lead to the so-called ringing artifact. This artifact appears near sharp transitions. It causes overshoots and oscillations. Near the edges of the physical frame, parts of the kernel may be outside the image boundaries. In this case, only the pixels inside the image are included in the filter sum and afterward appropriately normalized. [176]

TensorFlow also enables the use of even more complex scaling methods such as **Area interpolation** that uses polygons [130].

A convolutional layer follows these upsampling methods in the convolutional neural network. The convolutional layer compensates for the conventional upsampling methods' strengths and weaknesses, leading to no significant differences between the results. An exception is Area interpolation which is conceptual not suited for a detector with quadratic pixels. A more detailed discussion can be found in Section 8.4.2.

8.1.5.2 Upsampling Methods in the Context of Neural Networks

Upsampling methods in the context of neural networks increase the number of neurons per layer. The three common layers or combinations of layers to achieve upsampling are as follows:

- **Transposed convolution** is the most common method to achieve super-resolution (Figure 8.11a). The basic idea is described in Section 6.7.5. It can be implemented in only one additional layer. The major disadvantage of the transposed convolution is the tendency to periodical artifacts in the reconstructed frame. This tendency becomes more pronounced for a larger number of subpixels per physical pixel. [177]

Reducing these artifacts is one of the main challenges for the neural network's design and training process. A detailed discussion can be found in Section 8.2.

- **Resize convolution** is a combination of a conventional upsampling process followed by one or more convolution layers (Figure 8.11b). The padding of these convolutions is chosen to guarantee that the output of

¹ $\text{sinc}(x) := \sin(x)/x$ [174]

the convolution has the same size as the output of the upsampling process. The accuracy of the neural network strongly depends on the choice of the conventional upsampling method. There is a trade-off between performance and accuracy. Simple upsampling methods are fast but lead to lower accuracy, whereas interpolation methods that take more values of the feature map into account could lead to better accuracy but are more computationally intensive and, therefore, slower. Since the conventional upsampling methods are not necessarily used on physical images but on very abstract feature maps, nothing can be concluded about their applicability from the comparison with classical pictures. [177, 178]

- **Subpixel convolution** is performed in two steps (Figure 8.11c). Figure 8.12 schematically shows the idea of the subpixel convolution. The first step is a convolution in two dimensions. The padding of this convolution is chosen in a way that conserves the spatial dimension of the input. The number of feature maps is chosen as the number of new subpixels per pixel times the amount of desired output feature maps. In the second step, the output of the convolution is reshaped in a way that reduces the number of feature maps to one. The sorting of the individual entries of the feature maps ensures that each feature map describes one fix subpixel position in relation to the old pixel structure. For example, the first feature map describes the upper right subpixel. This method shows less pronounced artifacts in comparison to the transposed convolution (Section 8.2). [179]

Especially for super-resolution, the usage of transposed convolution tends to checkerboard artifacts. Resize convolution and subpixel convolution leads to a more homogeneous response. However, due to the depth to space layer in subpixel resolution, the frame size and, therefore, the size of the used detector must be constant during the prediction process. This is only a minor limitation since the size can be set before the prediction process individually instead of before the training process. A detailed comparison of the methods can be found in [180] and [181].

8.2 Checkerboard Artifacts

One of the major challenges of super-resolution tasks is to avoid checkerboard artifacts. Checkerboard artifacts are spatially periodical patterns caused by the neural network's architecture. [180]

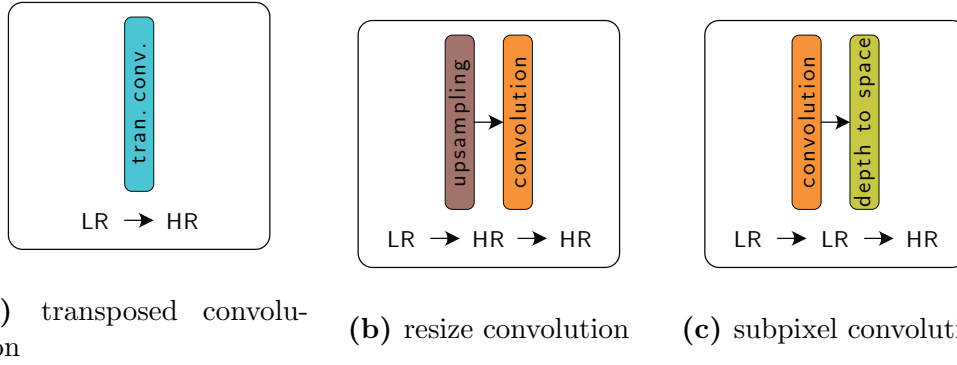


Figure 8.11: Schematic viewing of the different upsampling methods in the context of neural networks. LR denotes low resolution and HR high resolution and refer to the spatial size of the feature maps.

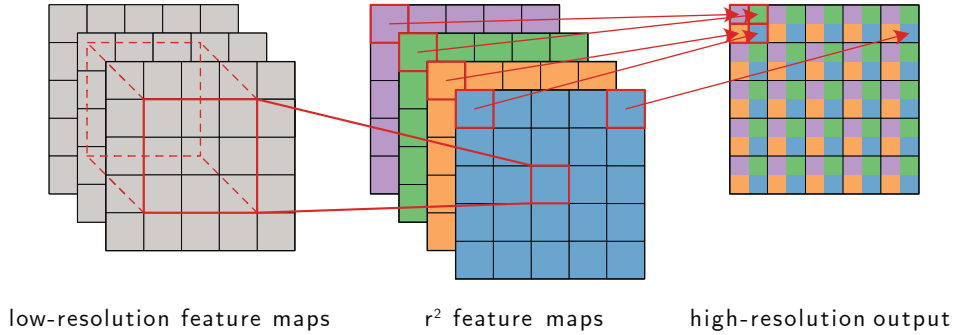


Figure 8.12: Schematic view of a subpixel convolution. The viewed subpixel convolution increases the resolution by a factor of $r = 2$. The input are three low-resolution feature maps. The convolution conserves the spatial dimensions and has $r^2 = 4$ output feature maps. The sorting ensures that each feature map describes a fix subpixel position in the context of the old pixel structure.

Due to the checkerboard artifacts' nature, they are constructively reinforced for a generated image which is typically obtained by summing over many individual frames [169].

The detector is homogeneously illuminated in order to investigate checkerboard artifacts. The normalized counting rate $c_{x,y}$ is defined as follows [169]:

$$c_{x,y} = \frac{\sum_{\text{frames}} \tilde{p}_{i,x,y} - g_{i,x,y}}{\sum_{\text{frames}} g_{i,x,y}} \quad (8.1)$$

Here, $\tilde{p}_{i,x,y}$ is the prediction of the pixel of the i^{th} frame at position (x, y) after

applying the threshold. The nomenclature for the ground truth g is similar. An example can be found in [13].

Due to the periodical character, the checkerboard artifacts can also be investigated by comparing the ground truth’s spatial frequency spectrum with the predicted result’s spectrum.

Since checkerboard artifacts spatially repeat themselves, a Fourier analysis in one or two dimensions can also help during the optimization process of the neural network structure to investigate the nature of the artifacts.

The presence of these artifacts strongly depends on the used layers and is most pronounced for certain transposed convolution layers. Transposed convolution layers can transform a homogeneous input to an output with checkerboard artifacts due to uneven overlap [182]. For example, a two-dimensional transposed convolution with a stride of two and a kernel size of three has some outputs with four times the input, some outputs with twice the value of the input, and some outputs which are the identity function of the input. These differences become more pronounced and the pattern even more complex for two or more transposed convolutions after each other. [182]

In principle, the neural network could learn to compensate these artifacts, but in practice, this is very complex, and most neural networks struggle to compensate them completely [182, 183]. Resize convolution and subpixel convolutions do not show this behavior intrinsically. But checkerboard artifacts can also occur during the training process, for example, due to the training data structure. If some pixels in the training data are hit more often than others, this can lead to an inhomogeneous result. Hence training data needs to be selected carefully.

8.3 Loss Function

The introduced loss function is optimized for unbalanced data sets, and the focus on which the neural network should be optimized can be varied with three parameters. It is necessary to handle unbalanced data sets, since on average, especially for super-resolution, the ratio of pixels with a PoE ($\#\text{PoE}$) to pixels with no PoE ($\overline{\#\text{PoE}}$) is even for higher rates much smaller than one:

$$\frac{\#\text{PoE}}{\overline{\#\text{PoE}}} \ll 1 \quad (8.2)$$

For example, the pixel-wise binary cross-entropy, which is typically used for binary segmentation, the optimization process would result in no PoE being predicted. In most cases, the neural network is correct in predicting no PoE.

The loss function shows a local minimum for predicting no PoE at all. But this is not the desired result. Nevertheless, the idea behind binary cross-entropy is also applicable to this problem. But additional weightings have to be introduced, showing the optimization process on what the focus should be. The basis of the definition of the loss function is the confusion matrix shown in Table 8.1.

Table 8.1: Confusion matrix. PoE denotes a PoE and $\overline{\text{PoE}}$ no PoE.

		Ground truth	
		PoE	$\overline{\text{PoE}}$
Prediction	PoE	True Positive (TP)	False Positive (FP)
	$\overline{\text{PoE}}$	False Negative (FN)	True Negative (TN)

The loss function (eq. 8.3) is calculated frame by frame and contains three components [13]. Each of these three components controls a particular property to ensure an optimal optimization process with the desired focus.

$$\mathcal{L}(g, p) = a \cdot TPR(g, p) + b \cdot TNR(\bar{g}, \bar{p}) + c \cdot \Delta(g, \bar{g}, p) \quad (8.3)$$

Here, TPR is derived from the true positives of the confusion matrix, TNR from the true negatives, and Δ explicitly handles the number of points of entry per frame. The positive prefactors a , b , and c can be individually adjusted to the used training set. A detailed discussion of the individual components is presented in the following.

The only two arguments to the loss function are the ground truth frame g , and the from the model predicted frame p . The entries of the ground truth frame can be zero for no PoE or one for a PoE in the corresponding pixel. The values of the pixels of the predicted frame are between zero and one, representing the probability of a PoE in this pixel. In addition, the inverse is always specified for each frame, which is called dark-class here. The to the ground truth frame corresponding dark-class is $\bar{g} = 1 - g$. For the predicted frame, the dark-class is $\bar{p} = 1 - p$. $\mathbf{1}$ is the matrix with the same shape as a frame, and each entry of it contains a one. The sum of the value of each pixel and its corresponding dark-class is due to the conservation of probability always one. The contribution of a pixel to the loss function depends only on its value and not its position within the frame.

The dark class can be calculated at each training step or can be explic-

itly integrated into the output layer of the neural network. For an explicate formulation, the number of neurons of the output layer has to be doubled. The explicit formulation is based on one-hot encoding, which is introduced in Section 6.5.3. In this case, the number of feature maps at the output has to be two. One feature map represents the probability for each pixel containing a PoE, and the other feature map represents the dark-class for each pixel. For the explicit formulation of the dark-class, the activation function of the last layer has to change from sigmoid to softmax activation (Table 6.1 on page 93). The softmax activation ensures the conservation of probability for each pixel over the different feature maps. However, during the optimization process, it turns out that a calculation of the dark-class and for every training step, a sigmoid activation leads to a slightly better accuracy than the explicit formulation of the dark class.

In eq. 8.3, the small letters a, b, c are positive factors that control the priorities of the optimization process. Because of the unbalanced data set for the training process, these factors have to be chosen asymmetrically. The ratio between a and b should be similar to the ratio between pixels with a PoE to pixels without PoE [184]. Deviations from this ratio shift the sensitivity of the model. If the ratio of a and b favors the contribution to the loss of the true positive rate, the model is trained to predict a PoE rather than no one at all, thus, increasing the multiplicity. In contrast, if the ratio of a and b favors the contribution to the loss of the true negative rate, the model predicts fewer PoEs. The choice of c is in principle not restricted but should be between a and b or smaller. A too-large factor c potentially leads to a disregard of the terms with the factors a and b . [13]

The explicit formulas for calculating the three parts of the loss function are the true positive rate (TPR), true negative rate (TNR), and Δ , which handles the ratio between the number of predicted PoEs and the number of PoEs in the ground truth [13]:

$$TPR = -\frac{\sum (g \odot \log p)}{\sum g} \quad (8.4)$$

$$TNR = -\frac{\sum (\bar{g} \odot \log \bar{p})}{\sum \bar{g}} \quad (8.5)$$

$$\Delta = -\log \left(1 - \frac{|\sum g - \sum p|}{\sum g + \sum \bar{g}} \right) \quad (8.6)$$

Here, \odot is the so-called Hadamard product [148], g the ground truth, p the prediction of the neural network, and \bar{g} the inverse ground truth. The structure of the three parts is discussed in the following.

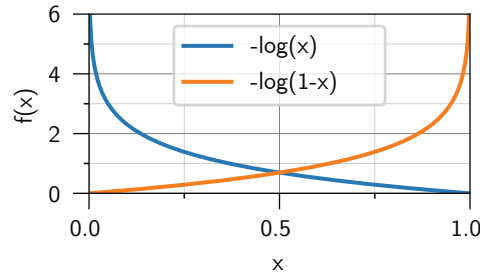


Figure 8.13: Behavior of logarithmic function

The normalization factor is the ground truth because it is independent of the prediction or the used model. The contribution to the loss of the true positive rate TPR (eq. 8.4) handles the true positive pixels. True positive in this context means that the positive result is used. Therefore, the probability for a PoE in the predicted pixel is taken into account, and the statement is true, respectively, the ground truth contains a PoE. The first step is to calculate the logarithm of the probability of the prediction pixel by pixel. In comparison with, i.e., a linear or polynomial function, the logarithm has some advantages (Figure 8.13) and is also used to define cross-entropy [130]. Compared to a linear response, larger errors have a bigger impact on the loss than small errors. This is necessary for a fast optimization process. The logarithm has no vanishing gradient in the domain of definition between zero and one, which is necessary to ensure a successful use of the backpropagation algorithm. For the best prediction of the PoE, denoted with a one, the logarithm is zero and makes no contribution to the loss. For predictions close to zero, which is the biggest possible error for a ground truth pixel with a PoE, the logarithm diverges to minus infinity. In combination with the minus sign in front of the fraction, the loss becomes very large. The next step is to multiply the logarithm component by component with the ground truth frame via the Hadamard product [148]. In a less computing power-intensive step, this ensures to consider only the positive pixels. The result of the Hadamard product is a matrix with the same size as the input matrices. However, the loss function should be a scalar. To achieve this, the individual results for each pixel are summed up. The sum over the ground truth frame in the denominator is for normalization. It makes the contributions of the true positive rate to the loss function independent of the amount of PoEs in the investigated frame.

The contribution to the loss of the true negative rate TNR (eq. 8.5) han-

dles the true negative pixels. The definition is analog to the definition of the contribution to the loss of the true positive rate. The dark class is used because this component of the loss handles the pixels with no PoE in the ground truth. True negative pixels are pixels where the ground truth pixel contains no PoE. This means the loss should be small for high values of the predicted dark-class.

The last component named Δ (eq. 8.6) explicitly handles the absolute difference between the predicted number of particles in a frame and the ground truth number of particles in this frame. The difference is also handled implicitly with the contributions to the loss of the true positive rate and the true negative rate. However, an explicit term enables a more detailed control by the user. The numerator describes the difference between the ground truth and the prediction. The denominator is the total number of pixels per frame. The averaged difference is in the best case zero and in the worst case one. The optimal model should have a small loss. As a result of this fact in combination with the behavior of the logarithmic function, the argument of the logarithm should be one minus the averaged difference (Figure 8.13).

The false negatives and the false positives are also part of the confusion matrix but are not considered explicitly for the loss function. Since their contributions are already considered implicitly, an explicit formulation would be redundant and only require unnecessary computing power. This would result in a slower optimization process. Analog to the contribution to the loss of the true rates, the contribution to the loss of the false negative rate is:

$$FNR = -\frac{\sum [g \odot \log(1 - \bar{p})]}{\sum g} \quad (8.7)$$

With the same argumentation as for Δ , the argument of the logarithm is one minus the dark-class in eq. 8.7.

Substituting the relation between the predicted frame and its corresponding dark-class leads to the result that in this definition, the contribution to the loss of the false negative rate FNR equals the contribution to the loss of the true positive rate TPR :

$$FNR = -\frac{\sum [g \odot \log(1 - (1 - p))]}{\sum g} = -\frac{\sum (g \odot \log p)}{\sum g} = TPR \quad (8.8)$$

An analog argumentation leads to the equality of the false positive rate

FPR and the true negative rate TNR :

$$FPR = TNR \quad (8.9)$$

Figure 8.14a shows the total loss function with parameters $a = b = c = 1$ for a frame with two pixels. The ground truth of pixel 1 contains a PoE, and the ground truth of pixel 2 contains no PoE. The symmetry of the parameters a and b leads to a symmetric contribution of pixel 1 and 2 to the loss. The loss function has a global minimum for a value of pixel 1 of one and for a value of zero for pixel 2. Figure 8.14b shows the gradient of the loss function. The derivative becomes larger for values further away from the minimum. In combination with the Adam optimizer described in Section 6.6.2, the behavior of the gradient leads to a faster and, therefore, less computing power-intensive optimization.

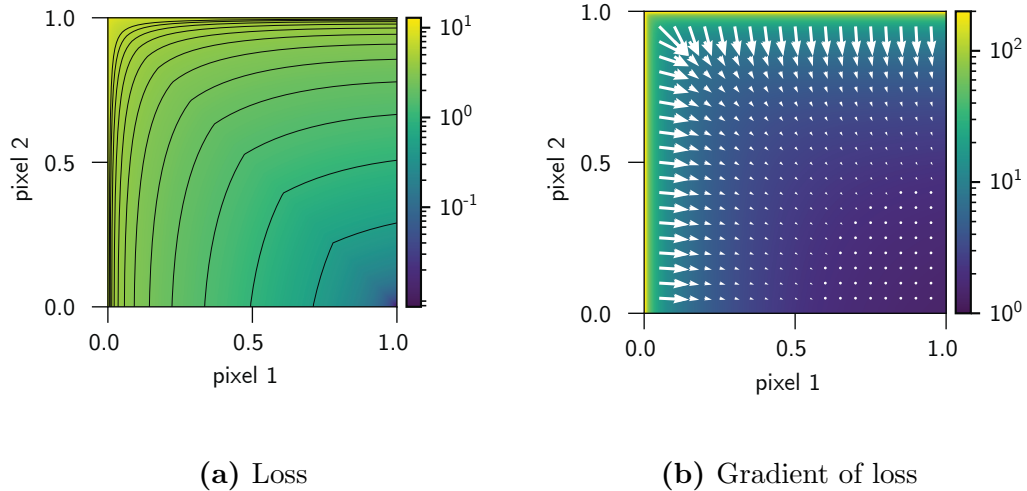


Figure 8.14: Loss function based on the confusion matrix for two pixels. The ground truth of pixel 1 is one, and the ground truth of pixel 2 is zero. (a) shows the loss function with equipotential lines with a distance of 0.5 starting at 0.5. (b) shows the gradient of the loss function. The arrows denote the direction of the negative gradient. The length of the arrows indicates the magnitude of the gradient.

8.4 Training

As for the compact neural network, the training and optimization process is highly automated in a pipeline. It works exactly such as the process for the

compact network, except that no event analysis is implemented. This means there is no housekeeping of the events and the events do not need to be assembled into frames after passing through the neural network for validation. The highly automated pipeline allows testing many hyper-parameters and design variations. The preparation of the training data set should be problem-related.

The data set used for the training is either obtained from a Monte Carlo simulation or analytically generated. The amount of training data is increased by additional mirroring and rotating the analytically generated data set. Data obtained by a measurement were not used for the training as their actual PoE is unknown, and, therefore, it is not possible to create an accurately labeled data set.

A Monte Carlo simulation of the energy deposition is required whenever the energy deposition is dominated by many small energy depositions over a large spatial area. Random processes generate these small energy depositions, and it is impossible to predict the track of an individual particle. This is the case for electrons. Therefore, the training data set to train the model for electrons is obtained from the simulation described in Chapter 4.

For photons that penetrate the detector perpendicular to the entrance window, the energy deposition of low energetic photons up to several tens kiloelectronvolts is very selective in the dimensions parallel to the detector surface and not spread over a wide range. Therefore, the shape of the event pattern mainly depends on the drift processes of the generated charges in the detector volume and not on the energy deposition of the primary photon into the silicon bulk. The generation of the numerically generated data set for photons is described in Section 4.7.

For the optimization process and evaluation of the different neural network configurations, simulated data and their ground truth, which the neural network did not see during the training, are used. The optimization of the hyper-parameters is done with electrons at a primary energy of 300 keV and 60 keV for super-resolution. The upper limit of the choice of the primary energies is made in such a way that with a high probability, the primary particle or secondary particles do not escape on the front side of a detector with a thickness of 450 μm under backside illumination (Figure 4.12 on page 66). The tracks of the electrons become longer and more complex for higher energies (Section 4.6). Therefore, a high primary energy of 300 keV is used for optimization.

8.4.1 Statistical Validation Parameters

Before checking the model's functionality with physical quantities, a statistical validation is performed. Therefore, statistical quantities such as the accuracy, precision, recall (sensitivity), and the F1 score is used. All these quantities can be calculated using the confusion matrix shown in Table 8.1. The accuracy describes the ratio of correctly classified pixels [134]:

$$\text{accuracy} = \frac{TP + TN}{TP + FP + FN + TN} = \frac{TP + TN}{\text{all}} \quad (8.10)$$

The dominant class has much more influence on the accuracy than the rare class. In the context of the PoE prediction, this means for low rates, predicting no PoE at all leads to a good accuracy but not to a good physical result. [185] This effect is referred to as high accuracy paradox [186].

The precision specifies the accuracy of a positive outcome predicted correctly [187]:

$$\text{precision} = \frac{TP}{TP + FP} \quad (8.11)$$

The recall (also known as sensitivity) measures the ability of the model to predict a positively predicted pixel correctly [187]:

$$\text{recall} = \frac{TP}{TP + FN} \quad (8.12)$$

The F1 score is a combined metric from precision and recall [188]:

$$F1 = 2 \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \quad (8.13)$$

The F1 score is only high if the precision and the recall are high. All quantities can have values between zero and one, where one denotes a good result.

The neural network's output is a hit-map representing the probability of a hit for each pixel in each frame. A threshold t is applied to analyze the statistical validation parameters, which converts the probability maps into binary maps:

$$S_{i,x,y}^{(\text{binary})} = \begin{cases} 0 & S_{i,x,y} < t \\ 1 & S_{i,x,y} \geq t \end{cases} \quad (8.14)$$

Here, t is the applied threshold, $S_{i,x,y}$ the probability for a PoE in the pixel of the i^{th} frame at position x and y , and $S_{i,x,y}^{(\text{binary})}$ the binary response.

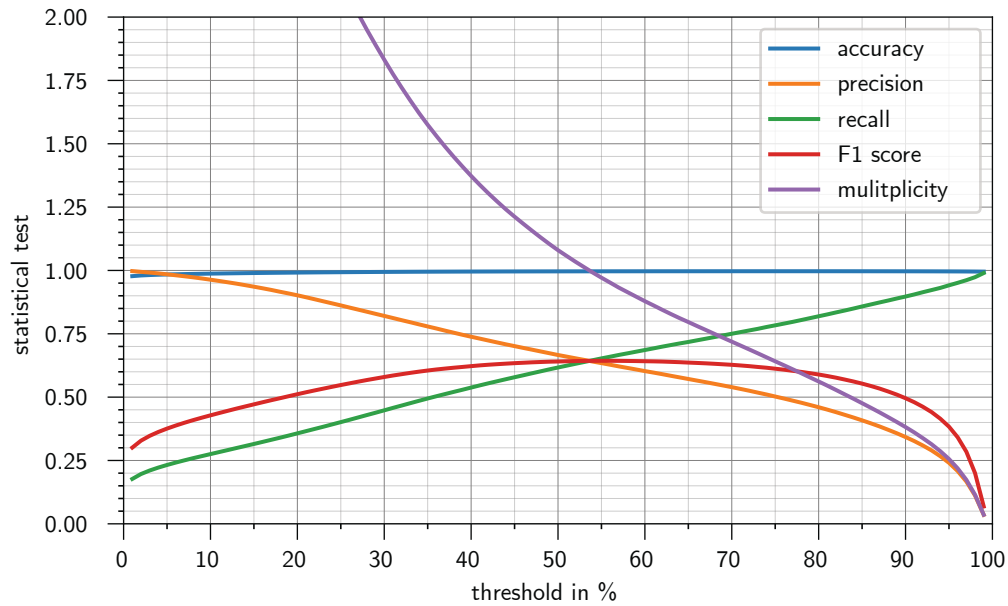


Figure 8.15: Various statistical validation parameters as a function of the applied threshold. Accuracy is plotted in blue, precision in orange, recall in green, F1 score in red, and multiplicity in purple.

Figure 8.15 shows the statistical validation parameters as a function of the applied threshold for the validation data set.

An applied threshold of approximated 50% aims to set the multiplicity (eq. 4.26 on page 70) to one. The multiplicity describes the ratio between predicted PoEs in the binary map and actual PoEs. This threshold is determined by the validation data set and kept constant for later analyses. Since the value of the threshold is indirectly correlated with the number of pixels above the threshold, the multiplicity decreases with the value of the applied threshold. For a threshold of zero, multiplicity is the number of pixels per frame divided by the average number of primary particles per frame. For a threshold of one, the multiplicity is zero.

The accuracy is close to one and, therefore, not suitable as a statistical validation parameter for all applied thresholds due to the unbalanced data set.

As expected from the definition (eq. 8.11), the precision decreases with increasing threshold, since for low thresholds, the probability to label all positive pixels (pixels that contain an actual PoE) is high. Predicting that all pixels contain a PoE leads to a high precision but is not meaningful in physical terms. The recall (eq. 8.12) increases with increasing threshold since it is sensitive to

false negatives. Predicting that very few pixels contain a PoE leads to a high recall because the model is very confident that a pixel which is predicted to contain a PoE actually contains a PoE.

As a consequence, a high precision or a high recall are not meaningful on their own. The F1 score combines both parameters, and maximizing the F1 score aims for an optimal balance of recall and precision.

8.4.2 Optimization Process

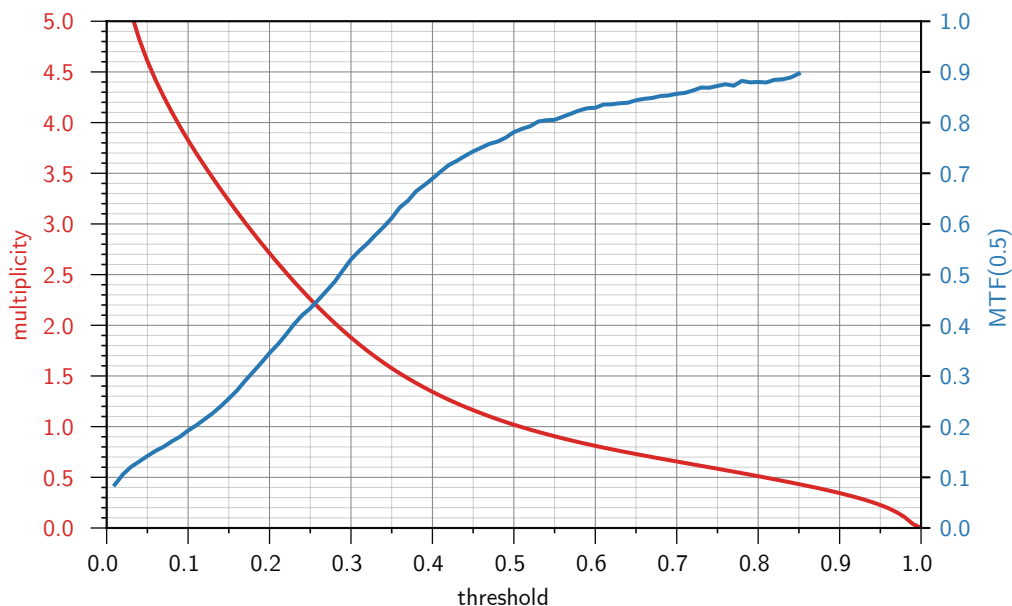


Figure 8.16: Multiplicity and MTF as a function of the applied threshold for a primary energy of 300 keV and a primary particle rate of $0.01 e^-/\text{pixel}/\text{frame}$. The MTF is obtained via the slanted edge method and graphed at 0.5 times the Nyquist frequency.

Besides, the architecture of the neural network hyper-parameters can be varied to obtain the best result. One key hyper-parameter is the method used for upsampling (Section 8.1.5.2). The most important characteristic for the different methods is the reduction of checkerboard artifacts (Section 8.2). For the different upsampling methods, the modulation transfer function and the multiplicity as a function of the applied threshold to the neural network's output are determined, such as shown exemplarily in Figure 8.16. A multiplicity of one is obtained for all upsampling methods with a threshold between 47% and 53%. The primary particle rate was for all

trainings $0.02 e^-/\text{pixel}/\text{frame}$, and except for the upsampling method, all hyper-parameters were kept constant. Except for the Area Interpolation, all methods lead to a modulation transfer function (MTF) at a ratio of 0.5 of the Nyquist frequency between 0.76 and 0.77.

Especially for super-resolution approaches, using subpixel convolution leads to results with fewer artifacts [182]. In combination with the obtained results for the multiplicity and the MTF, subpixel convolution is used in the following.

The other two important hyper-parameters are the number of features of the convolution layers and the depth of the neural network (Section 8.1.1). More features of the convolution layers and deeper neural networks lead to more trainable parameters. The result is a neural network that is more powerful but requires more computational resources and, therefore, the prediction process requires more time. However, due to physical limits, the improvements by more parameters are limited at a certain point, and the accuracy can not be significantly increased anymore.

To test the neural network approach for a different number of features and depths, the number of features was varied between 2, 4, 8, 16, 32, 64, and 128 for a depth of two, and the depth was varied between 1, 2, 3, 4 for 16 features. A PoENN with only 2 or 4 features does not have enough free parameters to train on the complicated tracks produced by primary electrons and, therefore, leads to non-useful results. Eight features lead to an MTF at a ratio of 0.5 of the Nyquist frequency of 0.72 and 16 features to 0.77. For more than 16 features, the MTF increases only slowly and not significantly. As a consequence, and as a trade-off between performance and resolution, a PoENN with 16 features is selected.

A depth of one leads to an MTF at a ratio of 0.5 of the Nyquist frequency of 0.59. For deeper depths, the MTF saturates at 0.77. Compared to the number of free parameters, therefore, it is better to use more features instead of making the neural network deeper.

8.4.3 Transfer Learning

In this Section, the applicability of transfer learning is investigated. The idea behind transfer learning and the working concept are explained in Section 6.5.5.

The training process of optimizing the PoENN is computational power intensive compared to the training process of the neural network described in Chapter 7. Training the compact neural network only takes a few hours, whereas training the PoENN needs time in the order of a few days. Therefore, transfer learning should be used to optimize the PoENN for the different

applications.

Transfer learning focuses on transferring stored knowledge to different but related problems [189]. This concept is used for three tasks in this thesis.

The first scenario is the extension to different subpixel levels. A detailed explanation of the architecture can be found in Section 8.1.4. Due to the neural network's architecture, only the last layers handle the extension to super-resolution applications. Therefore, the weights and biases of the first layers, which are u-shaped, are independent and universally applicable to the various super-resolution applications. This means the basis of the neural network can be reused for the various super-resolution applications without modifying or retraining the weights and biases of the basis. Therefore, the training process of the various super-resolution applications needs less computational power and is faster, leading to equivalent accuracy results.

The other two scenarios of using transfer learning instead of training the model from scratch are transferring the model to other pixel sizes or to other primary energies of the incoming particle. Training from scratch requires more iterations and, therefore, more epochs, more data for the training process, more computational power, and time. However, other pixel sizes or other primary energies lead to a different task. For small variations in the pixel size or the primary energy, this task is related and comparable to the original task of the previously trained model. The new task needs a slightly modified model. Due to the different boundary conditions, the modification has to be in all layers of the neural network and not only in the last layers. As a consequence, full model retraining is required.

8.5 Validation

The following Section is divided into two parts. The first part describes the accuracy in physical terms such as the MTF obtained by a slanted edge (Appendix D.1) and the multiplicity as a function of the applied threshold to the neural network's output and the primary particle rate. The second part describes the neural network's performance by analyzing the reconstruction frame rate for different hardware approaches. The topology of the neural networks can be found in Appendix C.5.

8.5.1 Accuracy of the Neural Network

In this Section, the accuracy of the results is described with two physically accessible quantities. On the one hand, the MTF, which provides information about the obtained resolution, and on the other hand, the multiplicity, which describes the number of reconstructed PoEs in comparison to the ground truth number of PoEs. Figure 8.16 shows the MTF and the multiplicity as a function of the applied threshold to the neural network's output. As expected, the multiplicity decreases with the applied threshold and is zero for a threshold of one. A multiplicity of one means that, on average, each primary electron creates one reconstructed PoE. The MTF behaves in the opposite way and increases with the threshold. If the neural network's output is interpreted as a probability for each pixel containing a PoE, this behavior is to be expected. With a higher threshold, only pixels are taken into account where the probability is high that the pixel actually contains a PoE. As a consequence, the MTF is higher for higher thresholds. For thresholds close to zero and close to one, the contrast and the statistic become worse, and as a consequence, the MTF cannot be calculated properly via the slanted edge method. For the presented results, these limits are 0.01 and 0.85. However, the ideal threshold depends on the application. A higher threshold leads to a higher resolution. However, the statistic of the measurement has to be high enough that it is acceptable to throw away pixels with a PoE where the probability for a PoE is lower.

Figure 8.16 shows the MTF and the multiplicity for a constant average primary particle rate of $0.01 \text{ e}^-/\text{pixel}/\text{frame}$. According to Figure B.3 on page 233, a rate of $0.01 \text{ e}^-/\text{pixel}/\text{frame}$ leads to a probability for a single pattern event of approximately 63 %¹.

Besides the applied threshold, another important parameter is the average primary particle rate. Ideally, the multiplicity is high and close to one over a wide range of average primary particle rates, and the MTF decreases only for a high average primary particle rate.

Figure 8.17 depicts the multiplicity and the MTF at 0.5 times the Nyquist frequency as a function of the primary particle rate. The multiplicity is one

¹Figure B.3 on page 233 shows the probability for a single pattern event as a function of the primary electron rate. Since the classical methods are only able to reconstruct single event patterns reasonably, the probability for single event patterns can be directly compared with the multiplicity for the neural network approach. As the classical approach can use only single event patterns, the MTF, and, therefore, the resolution is independent of the primary particle rate as long as enough single event patterns occur. However, even for a primary particle rate of $0.02 \text{ e}^-/\text{pixel}/\text{frame}$, the probability of getting a required single pattern event is below 50 %.

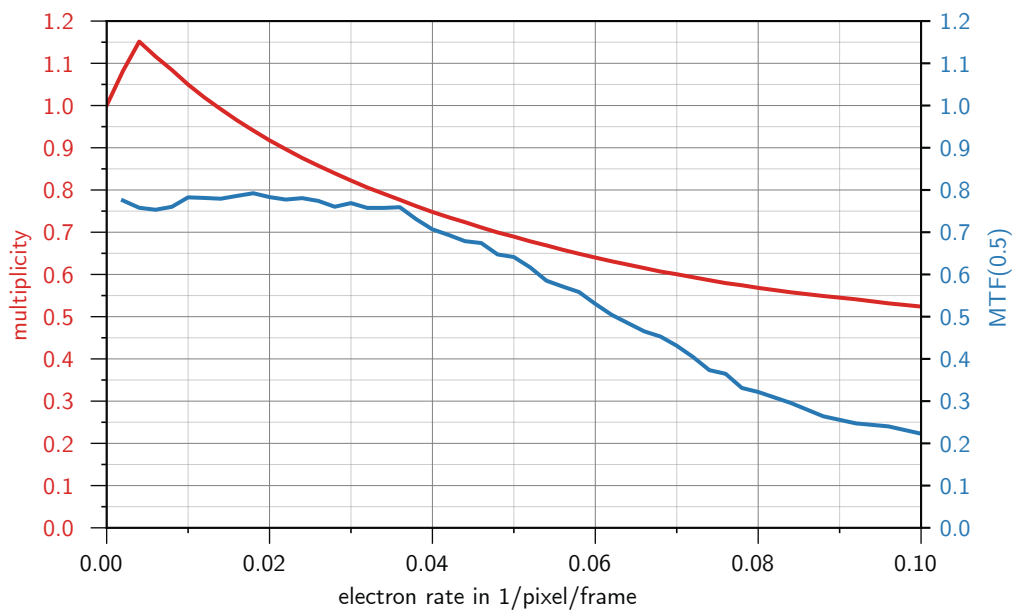


Figure 8.17: Multiplicity and MTF as a function of the primary electron rate for a primary energy of 300 keV and an applied threshold of 0.5. The MTF is obtained via the slanted edge method and graphed at 0.5 times the Nyquist frequency.

for very low particle rates since the event patterns are easily separable. With increasing rate, more overlapping events occur and, therefore, the ratio of pattern pile-up events increases. The probability for a pattern pile-up event is shown in Figure B.3 on page 233. The consequence of pattern pile-up events is a decreasing multiplicity. The multiplicity is for low rates higher than one to compensate for the decrease. This ensures a multiplicity of one at the average primary particle rate of the training data set. The shape of the multiplicity as a function of the primary particle rate looks similar for reasonably applied thresholds and changes only slightly. Reasonably in this context are thresholds between approximately 0.3 and 0.7. However, a higher threshold shifts the curve towards zero. The multiplicity decreases with increasing primary rate. However, in comparison to the classical approach, this decrease is slow. Even for a primary particle rate of 0.06 e^- /pixel/frame, the multiplicity is higher than the multiplicity of the classical approach at 0.01 e^- /pixel/frame. This means the neural network is able to handle a more than six times higher primary particle rate than the classical approach to provide the same multiplicity and resolution.

A multiplicity higher than the probability to get a single event pattern is only possible because the approach based on a neural network can handle pattern

pile-up events and reconstruct more than one PoE for a pattern pile-up event. The MTF is constant up to a rate of $0.04 e^-/\text{pixel}/\text{frame}$. Even for rates up to $0.08 e^-/\text{pixel}/\text{frame}$, the approach of the neural network leads to a better MTF than the classical approach, which uses only single event patterns and achieves an MTF at 0.5 times the Nyquist frequency in the order of 0.3 (Figure 10.13 on page 201). [13]

The validation of the accuracy of the PoENN with measured data can be found in Section 10.2.

8.5.2 Performance

The reconstruction rate of the used PoENN with 16 features and a depth of two (Appendix C.5) is presented in Table 8.2. To make the results best possible classifiable, different environments are used. A standard workstation with a CPU (Intel Core i5-8400 with 6 cores and a clock of 2.8 GHz), a NVIDIA[®] GeForce[®] GTX 960 [165], a server with 24 CPUs (Intel Xenon Gold 6128 with 6 cores and a clock of 3.4 GHz) is used. To get at the state-of-the-art limits, the performance is additionally tested in the environment of Google Colab [190] with TPU [191] hardware acceleration, respectively GPU (NVIDIA[®] Tesla[®] P100 [192]) hardware acceleration.

The performance of the presented PoENN strongly depends on the used hardware. Using GPUs and specially designed ASICs (TPUs) leads to the best performance. For the tested frame sizes, the scaling of the performance with the frame size is in the order of one.

Table 8.2: Reconstruction rate in Hertz with PoENN [13]. The batch size for 32×32 pixel is 1000 frames, and for 256×256 pixel 100 frames. The edge TPU is not supported since the depth to space operation can be currently not quantized [162]. Quantization is necessary for the deployment to the edge TPU. Since the parallelization of PoENN is limited, the difference between i5 and Xenon is marginal.

frame size	i5 (CPU)	GTX 960 (GPU)	Xenon (CPU)	TPU (TPU)	P100 (GPU)
32×32 pixel	3000	15 000	3000	17 000	43 000
256×256 pixel	50	250	60	250	900

Chapter 9

Super-Resolution of Intensity Images

For very high primary particle rates, the energy depositions of individual particles can not be separated anymore. In nearly every illuminated pixel and their neighbors, energy depositions of at least one primary particle can be detected. The resulting image is called an intensity image. This makes it impossible to reconstruct PoEs of individual primary particles even with convolutional neural networks. In the following, two approaches to handle intensity images will be discussed.

The first approach is called single image super-resolution (SISR). SISR increases the sharpness of edges and points and provides a higher resolution by introducing subpixels. The SISR also reduces the noise, a concept which is known as denoising. Single image super-resolution is widely used in photography and medical imaging analysis but can also be applied to electron imaging in TEM or X-ray applications. [193, 194]

In information theory, a theorem exists called data processing inequality. This theorem states that the content of information of a signal can not be increased by local physical operations [195].¹ A consequence of this is that a data analysis step can not generate information that was not already in the data set. However, this does not effect the single image super-resolution as this approach has an additional source of information to the image. The additional information originates from the training process with a large data set. Therefore, the data processing inequality is not violated by adding information to the image. However, there are limits due to the data processing

¹A global physical operation that takes more information into account can increase information content. An example of a global physical operation is the offset correction of the raw data obtained by the detector.

inequality [196]. For example, suppose the high-resolution image only contains uncorrelated noise in every pixel. In that case, it is obviously impossible to reconstruct the noise in every pixel by using its corresponding low-resolution image. Only certain correlated structures in the image enable the reconstruction of the high-resolution image from its low-resolution representation. Without this correlation, denoising is not possible too. There is no difference between the initial noise, which should be reconstructed and added noise that should be removed.

The second approach for intensity imaging is multi-image super-resolution (MISR). The idea behind MISR is to use more than one image of the same scene and feed them into the neural network in parallel. Ideally, the only difference between these multiple images is the noise of the detector system. The neural network can combine the individual images to reduce the noise, which leads to a better result. [197]

The classical analogy is combining many images and using the average. If the noise is Poisson distributed, the averaging process reduces the noise by a factor of $1/\sqrt{n}$ in comparison to the individual images. In this context, n is the number of averaged images. [38]

9.1 Single Image Super-Resolution

The first approach is a neural network for single image super-resolution (SISR). This means the neural network takes one image as input and returns the same image with a finer segmentation. A detailed survey of the different state-of-the-art approaches for single image super-resolution SISR can be found in [198].

9.1.1 Network Architecture

The architecture of the SISR network is adapted from the PoENN with super-resolution extension introduced in Chapter 8.

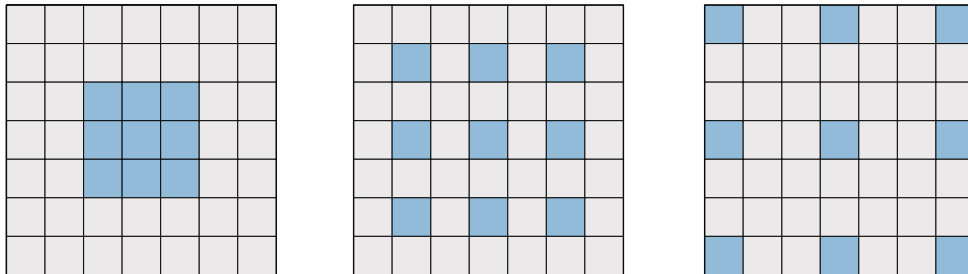
The main difference is the loss function applied during the training process. The loss function has to be modified because the neural network's output for SISR is not binary but represents different intensities. However, the output range is between zero and one due to normalization (eq. 6.7 on page 94). The normalization is no restriction because the neural network's output for SISR can be scaled with the input.

The modular architecture of the SISR is similar to the architecture of

PoENN with super-resolution extension and shown in Figure 8.8 on page 147. The input module (Figure 8.5a on page 145) does the data preparation, which takes place in the neural network and expands the dimension of the input tensor by one. The preamble (Figure 8.6a on page 145) contains a first convolution block to increase the number of channels. The next block in the SISR module is the u-net module (Section 8.1.1). A convolution block follows the u-net module in the postamble (Figure 8.6b on page 145) to do the adaption for the SR module (Section 8.1.4). The output module of the SISR module decodes the abstract feature maps of the SR module to a physical hit density map and reduces the dimension.

9.1.1.1 Residual network

The u-net module and the SR module are adopted from Section 8.1.1 respectively 8.1.4 and supplemented by the idea of residual networks. Residual networks contain skip connections also called shortcuts to provide the option to jump over some layers. These shortcuts have some advantages. Until the layer learns its weights, the activation of the previous layer can be reused to avoid the problem of vanishing gradients during the training process. Shortcuts effectively simplify deep neural networks and make them, therefore, simpler to optimize. This automatically leads to an acceleration of the training process. The optimization process gradually restores the skipped layers to learn more complex features. [129]



(a) Dilation rate = 1 (b) Dilation rate = 2 (c) Dilation rate = 3

Figure 9.1: Dilated convolution with a kernel size of three times and three and various dilation rates. The kernel is shown in blue. A higher dilation rate covers a larger spatial area without increasing the number of trainable parameters. A dilation rate of three is only shown for clarity but not used in this thesis’s framework.

The kernel size of the convolution is three for each spatial dimension.

Nevertheless, a combination of two or more convolutional layers indirectly can cover larger areas. The training data and the data that should be processed have patterns of very different sizes. The shortcuts may help the neural network to handle these different sizes easier since using them or not using them effectively changes the number of sequential convolutional layers.

Additionally, each convolution block is bypassed by a convolution block with a dilation rate of two for the spatial dimensions. The dilation rate defines the spacing between the values in a kernel (Figure 9.1). The kernel size stays the same. This enables the explicit handling of larger areas. The three times three kernel with the dilation rate of two has the same field of view as a five times five kernel but needs only nine (plus one) instead of 25 (plus one) parameters. The one extra parameter comes from the bias. A small number of parameters reduces the number of calculations in the neural network and, therefore, leads to a better performance in terms of reconstruction speed.

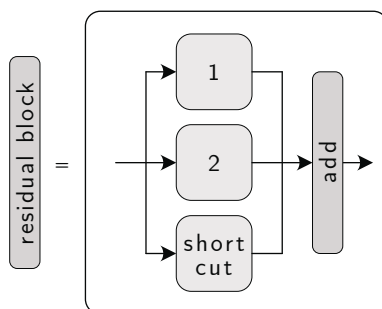


Figure 9.2: Residual block

The shortcuts and bypasses are combined by a layer that sums (adds) a list of inputs. The dimensions of the inputs in this list have to be equal. This summation is implemented in a residual block (Figure 9.2). The shortcuts should change the input as little as possible. However, the number of feature maps has to be adapted to enable the summation. This is achieved by a convolution layer with a kernel size of one in each dimension and linear activation.

9.2 Multi-Image Super-Resolution

A further development is the proposed network for multi-image super-resolution MISR. This neural network does not contain fully-connected layers, which allows large flexibility regarding the number of input images. This

neural network can combine any number of images from one scene. One scene means that the only difference between these multiple images is artifacts that should be removed originating, for example, from the noise of the detector system. The combination of many images of the same scene reduces the noise of the output because the neural network has more information about what is a real feature and what is detector noise. [197]

9.2.1 Network Architecture

Figure 9.3 shows the schematic arrangement of the modules used in the MISR neural network. The arrangement is similar to the SISR (Section 9.1) and consists of the same structure and modules, but an additional fusion module is added. The special feature here is that the u-net module and the SR module parameters are shared in parallel for multiple images. This concept is based on Siamese networks [199]. Technically, this is realized with three-dimensional convolutions with a kernel size of one in the third dimension. The fusion module described in Section 9.2.1.1 is inserted between the SR module and the output module. The fusion compares the multiple images.

In the case of MISR, the output module (Figure 8.5b on page 145) has to be supplemented by a global averaging pooling (Section 6.7.3) over all images in the image stack to combine the multiple images into one image. Global average pooling averages the entries over the pixels with the same spatial position. Figure 9.4 shows the modified output module.

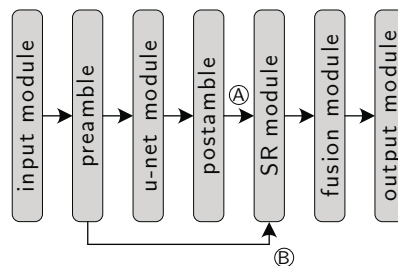


Figure 9.3: Architecture of the neural network for MISR. Compared to the SISR network, an additional submodule called fusion net is implemented between the SR net and the output. The SR module gets the output of the postamble (A) and the output of the preamble (B) as inputs.

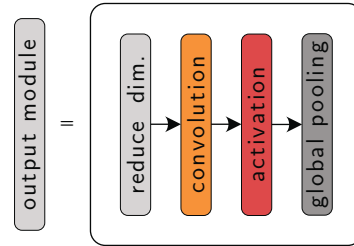


Figure 9.4: Architecture of the modified output module for MISR

9.2.1.1 Fusion Module

The fusion module (Figure 9.5) combines the high-resolution outputs of the SISR for the multiple images to one high-resolution output with lower noise. The individual images are step-wise combined via repetitive fusion submodules (Figure 9.5a).

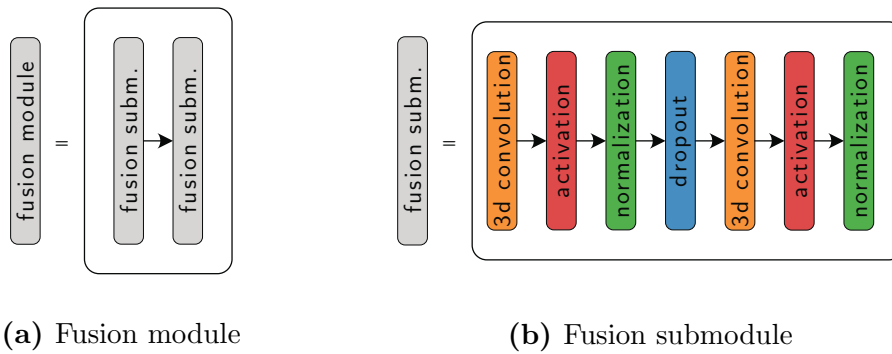


Figure 9.5: Architecture of the fusion module and the fusion submodule. In this example, two fusion submodules are repetitively applied to build the fusion module.

The fusion submodules (Figure 9.5b) consist of three-dimensional convolution layers with ReLU activation (Table 6.1 on page 93) by default. Optionally, batch normalization (Section 6.7.6) is applied. The third dimension of the kernel is in the direction of the image stack and has a size of two. The size of two results from physical considerations: A kernel size of one handles each image separately and has no combining effect. Kernel sizes that are three or larger would combine the individual images but can also introduce artifacts in terms of temporal correlation. A larger kernel is able to find a correlation between images and their relative position to each other in the image stack. This is not desired since the arrangement of the images in the image stack is purely

random and does not contain any physical information.

The stride in all dimensions is one. This ensures that not every second image of the stack is treated special and guarantees that each image is handled in the same way.

Like for the u-net module and the SR module, a residual network option is available (Section 9.1.1.1), which shortcuts the fusion blocks and adds convolutions with a dilation rate of two in the spatial dimensions. The dilation rate over the image stack remains one.

9.3 Training

Like the other presented approaches based on neural networks, the quality of the reconstruction by the neural network strongly depends on the features of the training data set and its understanding.

Generating the training data for intensity images from Monte Carlo simulations is very computational intensive as many individual primary particles contribute to one image. However, it is not necessary to generate the images by simulating individual particles because the individual particle's energy depositions are not of interest but the intensity distribution of many particle's energy depositions. The important parameters for the training data set are the blur at the edges of the structures in the image and pixel-wise noise.¹

The training data set is generated by producing images with random shapes and intensity distributions in high resolution. These high-resolution images are used as ground truth.

The low-resolution images to the neural network are created by down-sampling the high-resolution images by local averaging [110]. The blurring introduced by the character of the energy deposition is taken into account by applying a Gaussian filter.

For the Gaussian filter, the standard deviation is obtained by a slanted edge measurement of experimental data, described in detail in Chapter 10.2.

The pixel-wise noise represents the noise by the detector system, including the readout chain, and is assumed to be Gaussian distributed. Technically, a training data generator is used, which generates the training data by adding individual pixel-wise noise for each image directly at the time when the

¹The pixel-wise intensity distribution must be as general as possible to train the neural network for all possible intensity distributions. The amplitude can be arbitrary since the input data to the neural network are normalized.

training process requires the training sample. The training data generator uses one low-resolution training sample and generates multiple low-resolution images that only differ in their pixel-wise noise. The number of low-resolution images is typically between two and 100 for MISR.

This in-time generation has two advantages. First, the pixel-wise noise is different for every training sample, even over more epochs. Second, for MISR, the required memory is lower using the training data generator since only one low-resolution image for each training sample is required. The other frames that only differ in noise are generated directly by the training process. Additionally, the training generator enables flexible numbers of low-resolution images for every training step. Therefore, the MISR is not trained to a fixed number of input frames but to a variable one.

9.3.1 Loss Function

The loss function proposed in Section 8.3 is not applicable to super-resolution problems. Due to its design, it can only handle values of zero and one in the ground truth. Therefore, it is not capable to handle intensity images.

Additionally, compared to the data set for the PoE reconstruction, the data set for training the super-resolution networks has the advantage that they can be designed in a balanced data set.

This means that the intensity distribution over the training data set can be distributed equally, which allows to use a simpler standard loss function. The used loss functions are the mean square error and mean absolute errors [130]. The mean square error penalizes large errors more than the mean absolute error.

The optimization process focuses on reducing small errors. The mean square error could lead to the behavior that many small errors are acceptable if this reduces the number of large errors. [164]

Which loss function should be preferred depends on the training data set and the neural network's task.

Both loss functions compare only the values of individual pixels with each other but do not make a statement about the pattern or other structures. The introduced structure similarity loss is a perceptual loss function ¹ based on the

¹Perceptual loss functions are used when comparing two different images. The function is used to compare high level differences, like structures in the images.[200]

structure similarity SSIM [201]:

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \quad (9.1)$$

x and y denote the images or windows of the images that should be compared. μ_j is the average amplitude over the image $j \in \{x, y\}$, σ_j^2 is the variance of the amplitude over the image $j \in \{x, y\}$, σ_{xy} is the covariance between image x and image y , and c_i are variables to stabilize the division for small denominators. The variables c_i are defined for $i \in \{1, 2\}$:

$$c_i = (k_i \cdot L)^2 \quad (9.2)$$

k_1 is typically 0.01, k_2 is typically 0.03, and originally $L = 2^{\# \text{ bits per pixel}} - 1$ denotes the dynamic range for each pixel.

The structure similarity can have values between minus one and one [130]. A value of one corresponds to the best similarity. Therefore, a possible loss called structural similarity loss in the following can be defined as:

$$\mathcal{L}_{\text{SSIM}} = -\log\left(\frac{\text{SSIM}(x, y) + 1}{2}\right) \quad (9.3)$$

The argumentation for using the negative logarithm is analog to the argumentation in Section 8.3.

The structure similarity is more a perception-based model than a pure mathematical description. This leads to a visually better result but not necessarily to a better result in mathematical terms.

The best results are obtained from a loss which is a linear combination of the mean square error and the structural similarity loss. This linear combination optimizes the model in mathematical terms and visual properties.

9.3.2 Generative Adversarial Networks

For the training of SISR and MISR, the concept of generative adversarial networks (GAN) is used. GANs have achieved, among other things, great success in the creation of photorealistic images. GANs are predestined for producing a good visual result. This may lead to a worse result compared to a traditional loss function in physical terms because the focus of the training process is not on physical terms. GAN consist of two separate neural networks which perform a zero-sum game [202] also called a strictly competitive

game [203]. A zero-sum game means that one neural network's advantage automatically leads to a loss of the other one and vice versa. Figure 9.6 shows the schematic of one optimization step. [204]

The first neural network called **generator** creates images from a given input. The second neural network, which is the **discriminator**, tries to distinguish the generated image from the ground truth. In the training process, both networks iteratively learn. This arrangement makes a loss function for the generator network unnecessary. The loss function of the discriminator is the binary cross-entropy since its output is binary and represents the probability for a real image. Checkerboard artifacts and other artifacts make it easy for the discriminator to determine whether an image is fake. Therefore, this type of optimization leads to nearly artifact-free results.

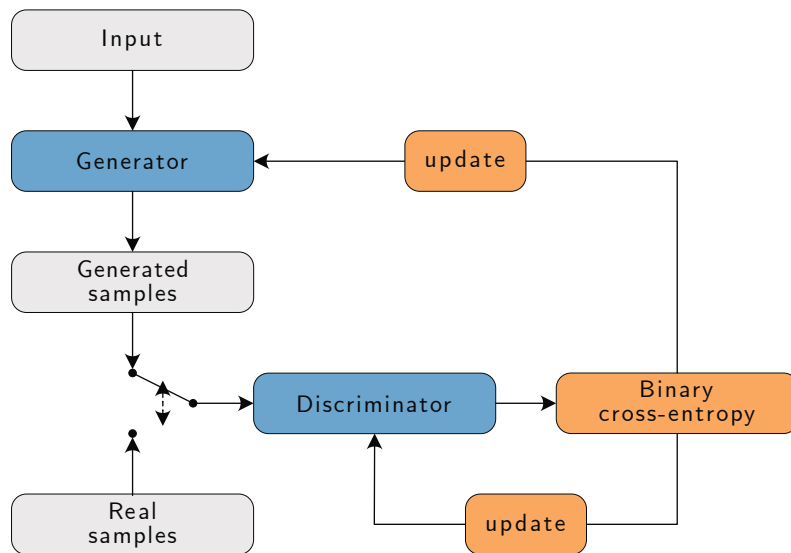


Figure 9.6: Working principles of generative adversarial networks (GANs). Blue boxes denote neural networks, grey boxes denote data, and orange boxes denote training mechanisms. The real samples represent the ground truth. The input of the discriminator toggles between the generated samples and the real samples.

Like the training process with a loss function, each training step requires input images and the ground truths. However, the ground truth does not have to be the corresponding ground truth to the input images. First, the generator creates its predictions for the input images. This output is then combined with the ground truths to one image stack. Matching this image stack, a vector is produced, containing a one at the indices of the ground truth

and containing a zero at the indices of the predicted images. The image stack and the vector are shuffled to avoid unwanted correlation and patterns. The image stack is the training data for the discriminator network. The vector containing the ones and zeros is the ground truth for the discriminator.¹

In the next step, the discriminator makes its prediction to the image stack. These predictions are values between zero (fake) and one (real) which describes the probability of the authenticity for each image of the stack. A common optimizer then updates the weights of the discriminator with a binary cross-entropy loss function (Section 6.5.3).

In a last step, the weights of the generator are updated. Therefore, only the predictions of the discriminator from the generator-generated images are used. The generator should be optimized in a way that the discriminator always predicts one (real). For this reason, the binary cross-entropy between the values predicted by the discriminator and a vector containing only the value one is calculated. The generator is optimized based on this loss function. The weights of the discriminator are not updated in this step. [204, 205]

Potentially, a small batch size leads to a better result as the discriminator does not get many examples in the first training step before initially optimizing the generator. Many examples in the first training step might overpower the discriminator over the generator, which leads to a worse quality of the accuracy of the neural networks. The best result is achieved for optimizing and learning in the same order for the discriminator and the generator. In most cases, the task of the discriminator is easier than the task of the generator. As a consequence, the discriminator learns and converges faster than the generator [206]. To compensate for this imbalance and to support the generator, the loss function for the generator is supplemented by the structure similarity loss and mean square error loss (Section 9.3.1).

9.4 Validation

In this Section, the results for the SISR neural network and the MISR neural network are presented and discussed. In the shown case, a quantitative analysis by the MTF of a slanted edge is only of limited significance because SISR and MISR are trained on edges. Instead, the structural similarity and the mean square error between the ground truth and the high-resolution output is calculated, and its average over ten thousand frames is taken. Like for the

¹It is a common trick to add additional noise to the binary entries of the vector to obtain better results. [204]

other neural network approaches in Chapter 7 and Chapter 8, the validation Section is split into accuracy and performance.

9.4.1 Accuracy of the Neural Network

The accuracy is separately shown for SISR and MISR. Figure 9.7 shows in the top row the low-resolution input to SISR, in the middle row the ground truth, and in the bottom row the reconstructed high-resolution prediction by SISR. The structures shown in the frames are random and not seen by the SISR during the training process. In the examples shown, SISR produces an 8×8 subpixel reconstruction. The structure of the data is described in detail in Section 9.3. A pixel-wise and Gaussian-shaped noise is assumed. The signal-to-noise ratio is chosen in a way that the signal-to-noise ratio for the pixel containing the highest intensity of the image intensity is one over 10 %. Since the input to SISR is an intensity image, a signal-to-noise ratio of 10 is very low for applications but shows the strength of the approach.

Each column represents one example frame. The left example contains coarse structures in comparison to the low-resolution pixel size. The right example also contains finer structures. The SISR reconstructs the coarse structures and the intensity distribution within the structures. Compared to the low-resolution input, the noise decreases, and the resolution increases. As shown in the right column, finer structures are partially reconstructed by SISR. For example, the round structure on the upper left side in Figure 9.7d is reconstructed in the high-resolution output (Figure 9.7f), even if it is not strongly pronounced in the low-resolution input (Figure 9.7b). Fine structures such as the separation of the two oval structures in the upper right side in Figure 9.7d are not reconstructable by SISR. Here, only a constriction is visible in the high-resolution output (Figure 9.7f). Overall, SISR is capable of performing denoising and increasing spatial resolution.

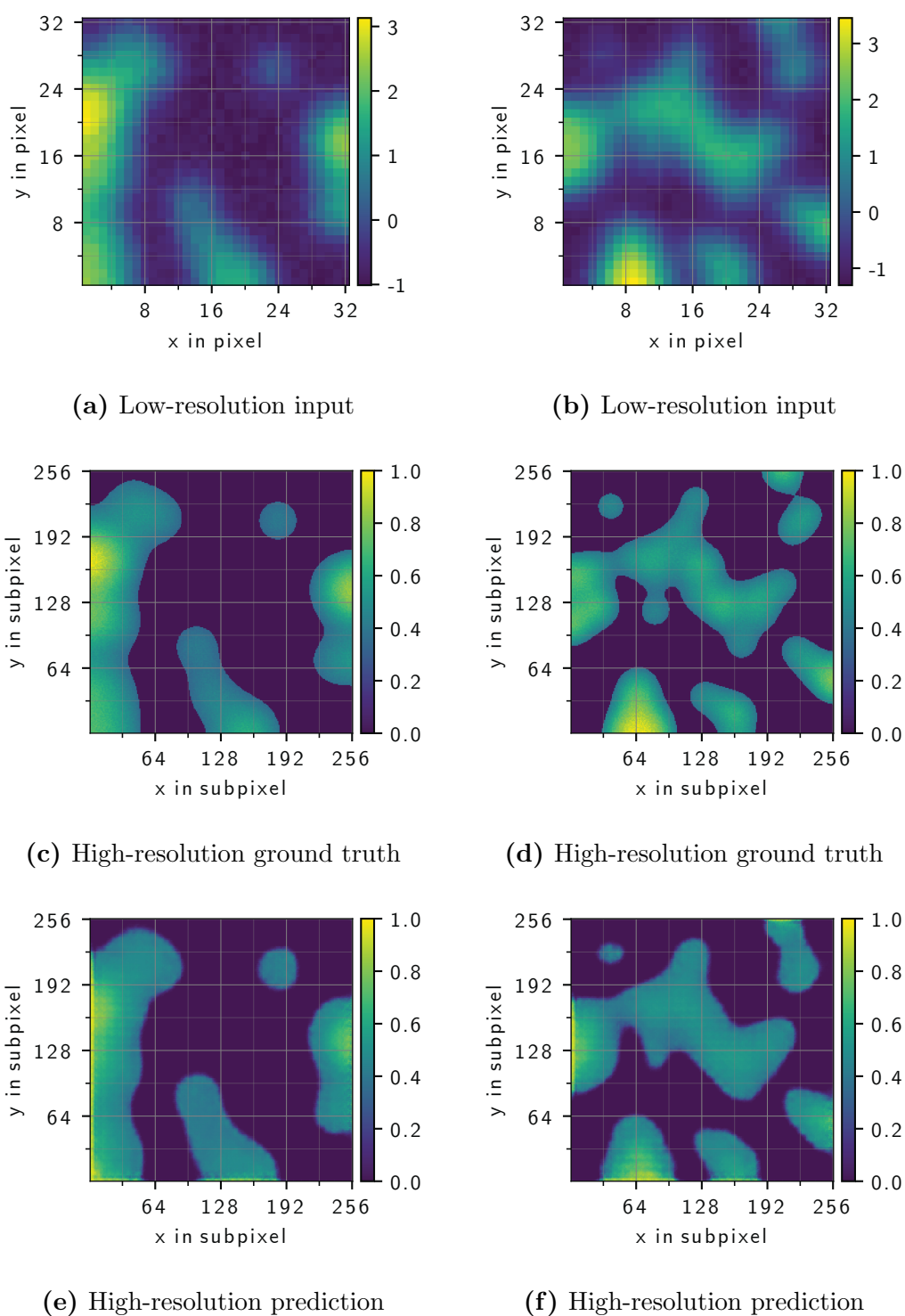


Figure 9.7: Two examples for SISR prediction. A pixel-wise and Gaussian-shaped noise is assumed. The signal-to-noise ratio is chosen in a way that the signal-to-noise ratio for the pixel containing the highest intensity of the image intensity is one over 10 %. One column depicts the same scene.

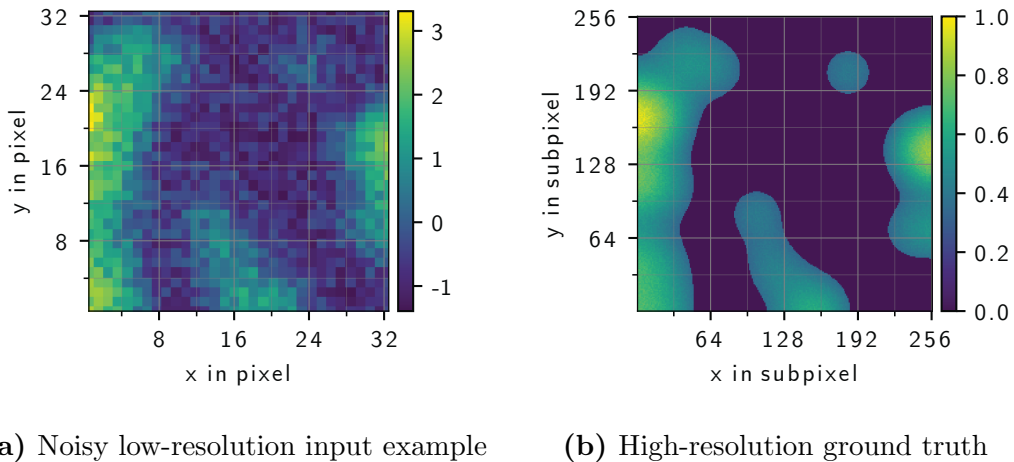


Figure 9.8: Input and ground truth examples for MISR prediction. The signal-to-noise ratio is three. **(a)** presents an example input frame. The other input frames contain the same structure but differ by noise. **(b)** presents the corresponding high-resolution ground truth.

Another more powerful approach is MISR. Here, many frames with the same intensity distribution but different noise contributions are fed to the neural network. Since the basic structure of the input frames is always the same, the neural network is equally able to reduce the noise like classical averaging over many images. The accuracy increases with the number of used frames. A very noisy input is used to show this behavior. Figure 9.8 shows the noisy low-resolution input and the high-resolution ground truth. The signal-to-noise ratio is 3 to show the impact of using multi images. Figure 9.9 presents the high-resolution output of MISR for various numbers of frames at the input of the neural network. In our case, MISR provides an 8×8 subpixel reconstruction. As expected, the quality of the reconstruction increases with the number of input frames. The quality increases very fast for a low number of frames and saturates for a larger number of frames. The improvement between one and ten frames is significant, whereas an improvement between 50 and 100 frames is neglectable for the used signal-to-noise ratio.

In Figure 9.9, it is clearly visible that MISR is able to reconstruct intensity distributions, and its accuracy is increased with the number of available frames fed into the neural network.

Because of missing information at the frame's borders, SISR and MISR introduces artifacts at the frame's borders. These artifacts are, for example, clearly visible in the form of a spatially repeating pattern at the center of the lower border of the reconstructions (Figure 9.9). These border effects can be avoided using only inner pixels. The validation of the accuracy of the MISR

with measured data can be found in Section 10.2.

For a quantitative analysis of SISR and MISR, the structural similarity (SSIM) [201] and the mean square error (MSE) [130] between the ground truth and the high-resolution output are calculated, and its average over 10000 frames is taken. The SSIM and MSE are calculated from normalized frames. A higher SSIM represents a better result. A lower MSE represents a better result. The SSIM and MSE between the low-resolution input and the ground truth are calculated as a reference value. Therefore, the low-resolution is upsampled by nearest-neighbor interpolation. For multiple images, the images are averaged before the interpolation step. SISR and MISR lead to a significant quality

Table 9.1: Accuracy of of SISR and MISR. The shown parameters are the structure similarity (SSIM) and the mean square error (MSE) between the ground truth and the result obtained by the neural network. The MSE is normalized to the maximal intensity. A detailed description of the reference value can be found in the text. The S/N refers to the signal to noise ratio of the pixel with the highest intensity in the image. For an S/N of 10, SISR is used, and for all other results MISR.

number of images	S/N	SISR/MISR		reference	
		SSIM	MSE	SSIM	MSE
1	10	0.87	0.7%	0.30	3.6%
1	3	0.75	4.1%	0.20	6.4%
2	3	0.76	4.0%	0.24	5.5%
5	3	0.77	3.9%	0.28	5.1%
10	3	0.77	3.8%	0.29	4.9%
50	3	0.77	3.9%	0.32	4.8%
100	3	0.78	3.7%	0.32	4.8%

improvement of the reconstructed images compared with the reference reconstruction and the raw images. SISR and MISR increase the images' spatial resolution and decrease the noise. Especially the SSIM is increased by using multiple images. The classical averaging process cannot reconstruct subpixel information. Consequently, the MSE decreases for the reference reconstruction slower than $1/\sqrt{n}$ for using many images.¹

¹The classical averaging process reduces the noise by a factor of $1/\sqrt{n}$ in comparison to the individual images if no subpixel resolution is applied. In this context, n is the number of averaged images. [38]

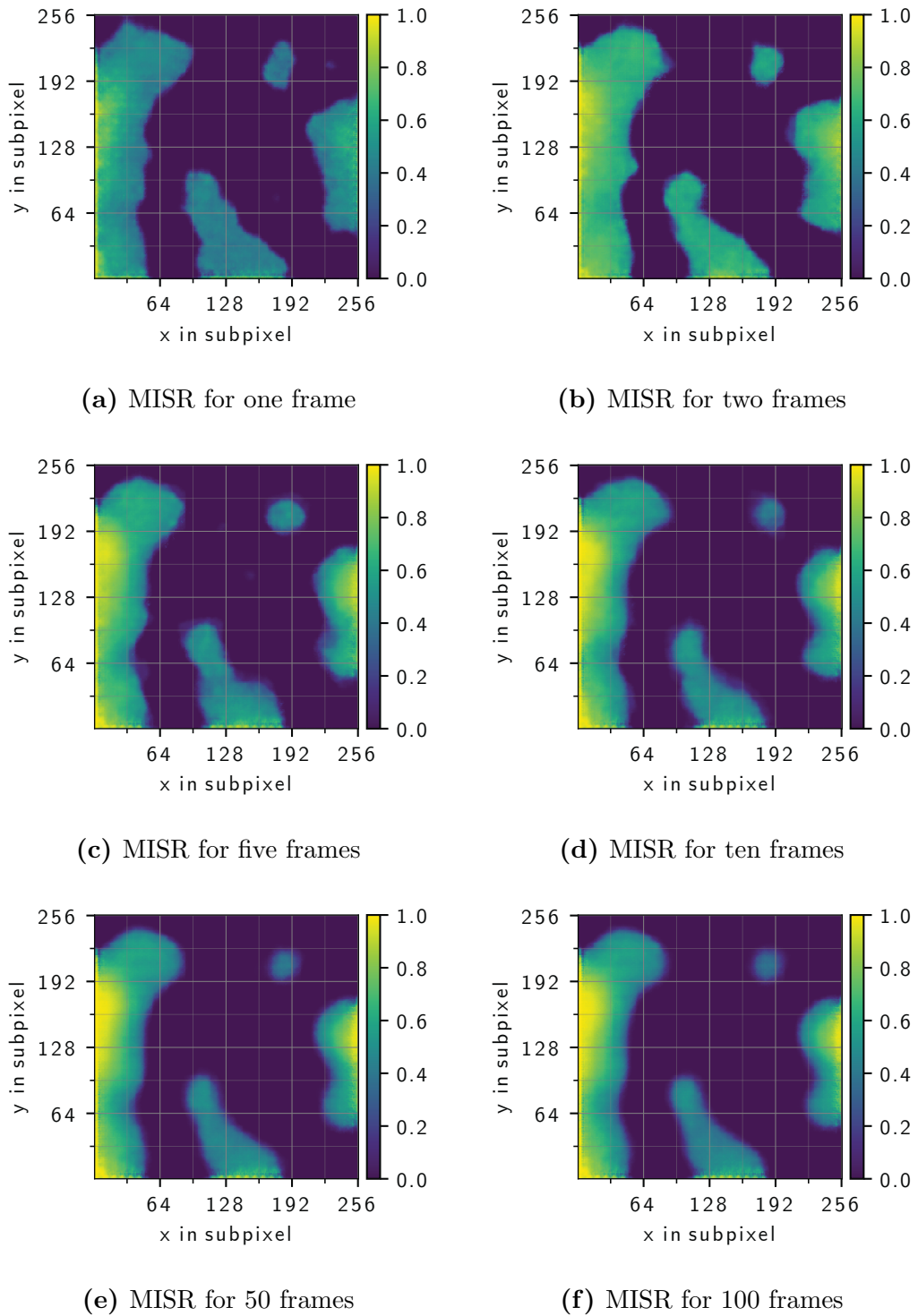


Figure 9.9: Example for MISR prediction. The number of used low-resolution input frames is varied between the figures. Input and ground truth are shown in Figure 9.8.

9.4.2 Performance

The reconstruction rate of the used SISR and MISR (using five frames¹) with 16 features and a depth of two (Appendix C.5) is presented in Table 9.2. Different environments are used to allow a classification of the results. A stan-

Table 9.2: Reconstruction rate in Hertz with SISR and MISR (using five frames). The batch size is limited by the available memory. A deployment to TPUs is not possible for SISR and MISR due to the architecture of the networks. Currently, TPUs do not support three-dimensional convolution and the merging required by the resnet option [190]. The batch size describes how many images are reconstructed in one step. A larger batch size increases the performance but requires more RAM. The GTX 960 has not enough RAM to perform a prediction for MISR with 256^2 pixels.

network	image size	batch size	i5 (CPU)	GTX 960 (GPU)	Xenon (CPU)	P100 (GPU)
SISR	32^2 pixels	100	30	140	30	1300
	256^2 pixels	5	0.3	2.1	0.3	20
MISR	32^2 pixels	10	1.2	4.5	1.2	25
	256^2 pixels	1	0.02		0.02	0.3

dard workstation with a CPU (Intel Core i5-8400 with 6 cores and a clock of 2.8 GHz), a GPU of type NVIDIA[®] GeForce[®] GTX 960 [165], a server with 24 CPUs (Intel Xenon Gold 6128 with 6 cores and a clock of 3.4 GHz) are compared. To get at the state-of-the-art limits, the performance is additionally tested in the environment of Google Colab [190] with GPU (NVIDIA[®] Tesla[®] P100 [192]) hardware acceleration.

The performance of the presented approaches strongly depends on the used hardware. Using GPUs leads to the best performance. The rapid progress in developing new technologies and basics will lead to a higher reconstruction rate in the next years. The performance for MISR is in comparison to SISR lower. However, this is not a bottleneck since the detector must capture many individual low-resolution frames for each image generated. If, for example, 100 low-resolution frames are taken for each reconstructed high-resolution image, the detector should have a frame rate a hundred times higher than the reconstruction rate by MISR.

¹The usage of five frames is a compromise between performance and accuracy.

9.5 Summary

With the help of SISR and MISR, the resolution of intensity images can be increased, and the pixel-wise noise can be decreased. SISR and MISR do not use the physical effects behind the signal generation of individual primary particles but mainly use statistical effects. Especially at borders of the depicted structures, the statistical behavior of many individual energy depositions can be used to reconstruct the border precisely. The other feature of SISR, particularly MISR, is denoising. MISR can significantly reduce noise by combining many images of the same scene.

Chapter 10

Analysis of Photon Detection and Electron Tracks

In this Chapter, the different approaches based on neural networks are compared with each other and with conventional methods. Additionally to simulated data, data measured with a pnCCD camera with 264×264 pixels is used (Section 3.3). Due to the structure of the neural network at each border, four pixel rows and columns are removed for the measurements with electrons presented in Section 10.2. Only the inner 256 (2^8 pixel) times 256 pixel are used as input to the neural network. For the previous simulations and the measurements, the pixels are quadratic with a size of $48 \times 48 \mu\text{m}^2$. The analysis is divided into applications for photon detection and applications for electron detection.

10.1 Photon Detection

In this Section, approaches based on neural networks and conventional approaches with photons as primary particles are compared. For the reconstruction of the data obtained with photons, the CoNN as a neural network is used (Chapter 7).

The measurements for photons are performed with two experimental setups. First, direct spot illumination of the detector surface, and, second, X-ray fluorescence of a copper grid aiming for a summed intensity image.

10.1.1 Low Energy X-ray Pencil Beam

The second experimental setup for X-ray photons is a spot illumination with photons of an energy of 1320 eV . The equivalent noise charge is 3.7

electrons. For the spot illumination, the spatial resolution can be directly derived from the distribution of the reconstructed PoE. The monochromatic spot illumination on the detector surface is achieved by a parallel X-ray beam that passes a monochromator and a Fresnel zone plate. Figure 10.1 shows the experimental setup. The diameter of the spot at the detector surface is about $0.8\ \mu\text{m}$ [12]. The reference to the used data can be found in Table D.2.

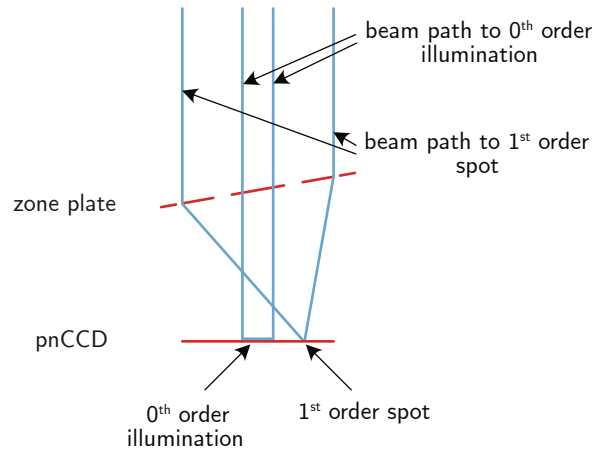


Figure 10.1: Experimental setup of the measurement with low energy X-ray. The beam is focused via a zone plate onto the pnCCD. The 0th order illumination is caused by the undiffracted part of the beam and illuminates a wide area. The 1th spot is due to the tilt of the zone plane next to the 0th order illumination. Figure adapted from [12].

The width of the center spot of the zone plate is limited due to requirements of a large working distance. As a consequence, the number of non-diffracted photons (0th order) is not zero. The zone plate is mounted slightly tilted to separate the 0th order and the focused 1th order photons spatially on the detector surface. The focused 1th order photons are used in the following for obtaining the spatial precision. [12]

To get different focal spot positions relative to the pixel structure, the spot position is moved on a two-dimensional grid over the pixel structure. The grid has 22 times 7 positions with a spacing of $3\ \mu\text{m}$ in the x-dimension and a spacing of $10\ \mu\text{m}$ in the y-dimension. The illumination intensity is chosen in a way that the number of single event patterns is maximized.

A detailed description of the experimental setup and the conventional reconstruction method can be found in [12].

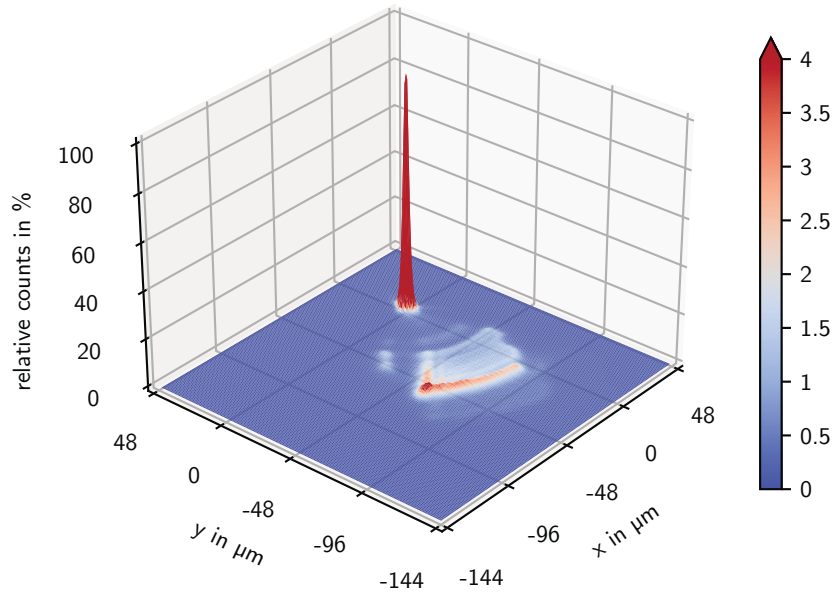


Figure 10.2: Two-dimensional histogram of the reconstructed PoEs. An area of 4×4 pixels is shown. The reconstructed 1th order spot position is at (0,0). The PoEs caused by 0th order photons are clearly separated from the 1th order spot. The z-axis and the color bar show the relative amount of reconstructed PoEs. The bins are quadratic and have a size of $1 \mu\text{m}^2$.

Figure 10.2 shows a two-dimensional histogram of the reconstructed PoEs with the CoNN (Chapter 7). The two-dimensional histogram shows all reconstructed PoEs for all beam spot positions. The registration and, therefore, the spatial alignment of the relative beam spot positions to each other is based on the reconstructed 1th order beam spot positions. As reconstructed 1th order beam spot position, the peak position of the individual measurements for the different scan positions within the pixel structure is used. Incorrect registrations of 1th order beam spots on the pixel border are neglectable since the introduced systematical error is only around $1 \mu\text{m}$ and, therefore, not visible in Figure 10.2.

The PoEs caused by 0th order photons are clearly spatially separated from the 1th order spot. Therefore, a distinction between the 0th order photons and the 1th order photons is possible. For calculating the spatial precision, a Gaussian shape of the 1th order peak is assumed, and the standard deviation

in each dimension is used.

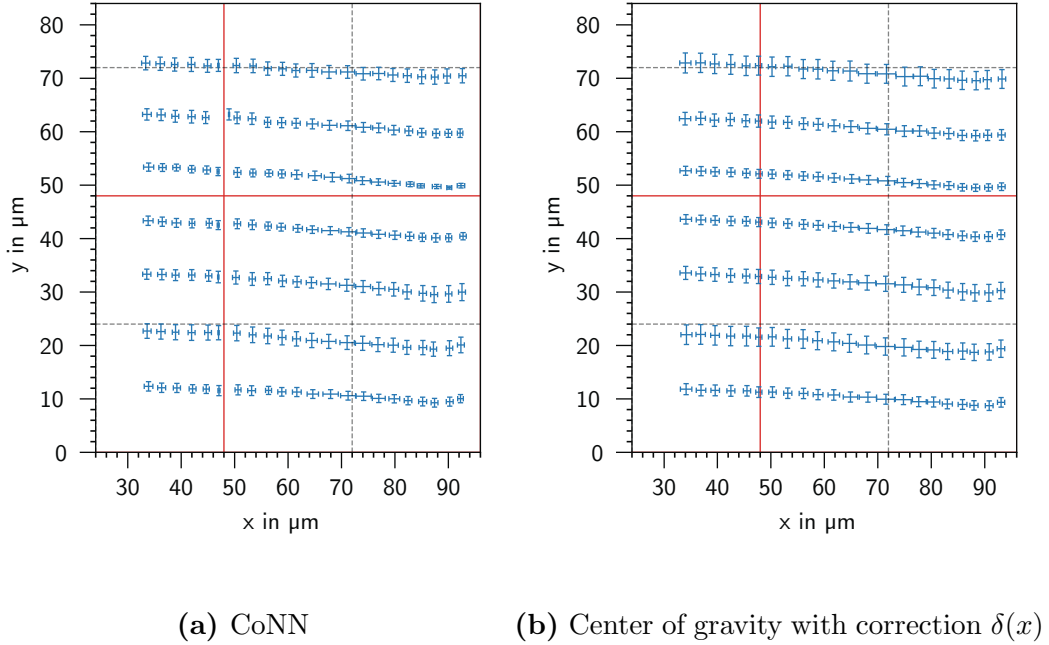


Figure 10.3: Reconstructed beam spot positions. The blue dots indicate the reconstructed beam spot positions. The error bars represent the standard deviation of the reconstructed positions down-scaled by a factor of two. The solid red lines mark the physical pixel borders and the gray dashed lines the physical pixel centers.

Figure 10.3 shows the reconstructed beam spot positions and their corresponding uncertainties for the CoNN (Chapter 7) and the center of gravity method with corrections (Appendix B.2). A two-dimensional Gaussian fit over the reconstructed PoEs obtains the position and the uncertainties for both methods.

Since the primary energy of the photons is only 1320 eV, the common-mode of the detector system plays a significant role and has to be corrected prior to the CoNN and the classical approach. Due to the readout scheme of the detector system (Section 3.4.1), the pixels of each row are read out simultaneously. Therefore, the common-mode mainly affects the spatial resolution in the y-direction and not in the x-direction.

The amount of contributing PoEs to the individual beam spot positions is in the order of 70 000 events.

As predicted by the theory (Appendix B.2.4) and obtained by the simulation

(Section 7.3.1), the spatial resolution depends on the position within the pixel structure and is the best near the physical pixel borders. The CoNN leads for all pencil beam positions to better spatial precision than the center of gravity method with corrections.

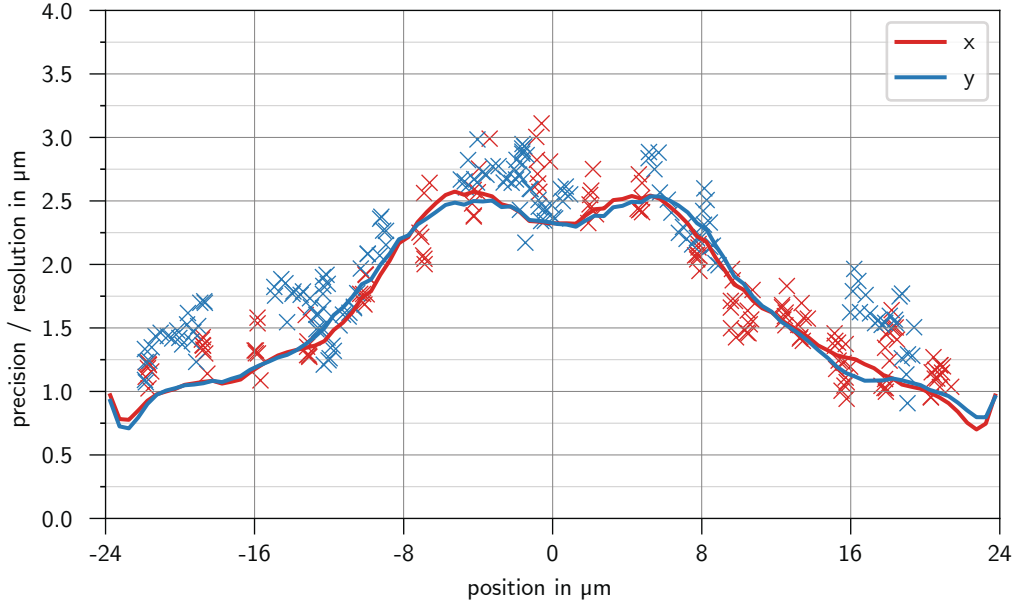


Figure 10.4: Spatial precision in x- and y-dimension obtained by the CoNN for a low energy X-ray pencil beam. The crosses show the reconstructed spatial precision (standard deviation) as a function of the x- and y-position. The pencil beam’s width of the measurement is $\sigma = 0.8 \mu\text{m}$ [12]. The solid line shows the resolution for the x- and y-dimension obtained from the Monte Carlo simulation. For the results obtained by measurement, due to the presence of common-mode, the resolution in the y-direction is slightly worse than the simulated results.

Table 10.1 shows the spatial precision in both spatial directions for the CoNN, the weighted centroid method with corrections, and an approach using a lookup table described in [12]. Basically, the approach using the lookup table calculates the PoE by using the weighted centroid method with corrections, but the selection which pixels of the event patterns are used to obtain the weighted centroid is not done by an event pattern analysis but by a lookup table. This lookup table is obtained by a Monte Carlo simulation. [12]

For all approaches, the spatial precision in the x-direction is better than the spatial precision in the y-direction. This behavior could be caused by a not-perfect common-mode correction applied before the event pattern analy-

Table 10.1: Spatial precision determined by a low-energy X-ray pencil beam. The units are micrometers. The spatial precision of the CoNN and the weighted centroid method with corrections (δ) is obtained by a two-dimensional fit. The pencil beam’s width is $\sigma = 0.8 \mu\text{m}$ [12]. The results for using the lookup table are adapted from Ihle et. al. [12].

	CoNN	δ	lookup table
x-direction	1.58	1.98	1.66
y-direction	2.00	2.44	1.94

sis. The CoNN leads to the best spatial precision in the x-direction and a comparable spatial precision to the lookup table in the y-direction. The data preparation can explain the slightly worse spatial precision of CoNN in the y-direction. In this thesis, a simplified common-mode correction which subtracts the median for each row (Appendix B) is used.

Figure 10.4 depicts the spatial precision, which is defined as the standard deviation for each pencil beam position and the spatial resolution that describes the Euclidean distance between the ground truth and the predicted PoE by the CoNN.¹ The precision is, like theoretically described, worst in the center of the pixel and gets better towards the pixel borders. Since the signal-to-noise ratio of the pencil beam is much lower than for X-rays created by copper as mentioned in Section 10.1.2, the noise term is not neglectable during the training process. This results in the two characteristic peaks in the center (Figure 7.2b on page 130 and Figure 7.2d on page 130). The spatial precision is obtained from the measurement since the resolution is not a directly accessible quantity as the ground truth PoEs are unknown. However, the spatial precision and the resolution are comparable if the accuracy is good. Due to the architecture of the CoNN, the neural network is not capable to reconstruct PoEs which are on the pixel border. The consequence is a slightly worse spatial accuracy on the pixel borders. The CoNN shifts PoEs on the pixel border towards the center (Figure 10.3). Therefore, the spatial resolution which is influenced by the spatial accuracy is worse than the spatial precision.

In comparison to the classical methods, CoNN has the advantage that it can be trained with systematically measured data by an experiment planned in a further step. The measured data describe all effects introduced by the detector system. Therefore, a training data set obtained by measured data

¹A detailed description of the spatial precision and the spatial resolution can be found in Appendix A.1.

contains all detector-specific effects, which is not possible for simulated data. The more accurate training data lead to better accuracy of the CoNN.

10.1.2 X-ray Fluorescence

In this Section, a simple experiment to show summed intensity images with CoNN is presented.

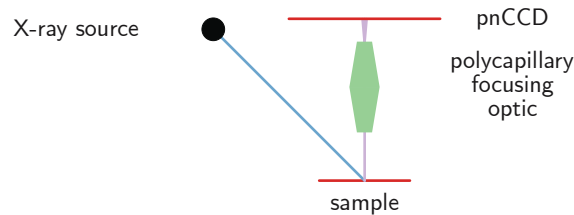


Figure 10.5: Experimental setup of the XRF measurement. A sample is excited by an X-ray source. Fluorescence X-ray photons emitted by the sample pass the 1:1 polycapillary optic with a capillary diameter of $22\ \mu\text{m}$ and a length of $38\ \text{mm}$ before the pnCCD detects them. The exit divergence $\Delta\theta$ of the polycapillary optic is between 0.2° and 0.3° [207] at $8048\ \text{eV}$. The distance between the polycapillary optic and the pnCCD is approximately $1.3\ \text{cm}$.

The schematic experimental setup for measuring the X-ray fluorescence (XRF) is shown in Figure 10.5. Figure 10.6 shows a light microscope image of the copper grid, which is used as a sample. The reference to the used data can be found in Table D.1.

The detailed processes in copper atoms can be found in Section 2.3.1. The copper grid emits characteristic photons with an energy of $E_{K\alpha_1} = 8048\ \text{eV}$ [40]. Only X-ray photons parallel to the beam axis pass the polycapillary optic and are detected by the pnCCD. The used 1:1 polycapillary optic has a capillary diameter of $22\ \mu\text{m}$ and a length of $38\ \text{mm}$. In the experimental setup, the polycapillary optic's properties are the spatial resolution's limiting factors. The measurement was recorded with a pnCCD backside voltage of $-230\ \text{V}$. The measured equivalent noise charge is 3.1 electrons.

Figure 10.7a shows the reconstruction via the maximum method (Appendix B.1.4) for comparison. For summing the maximum method, the visualized pixels are the physical pixels of the detector with a size of $48 \times 48\ \mu\text{m}^2$. Figure 10.7b presents the reconstruction with the CoNN presented in Chapter

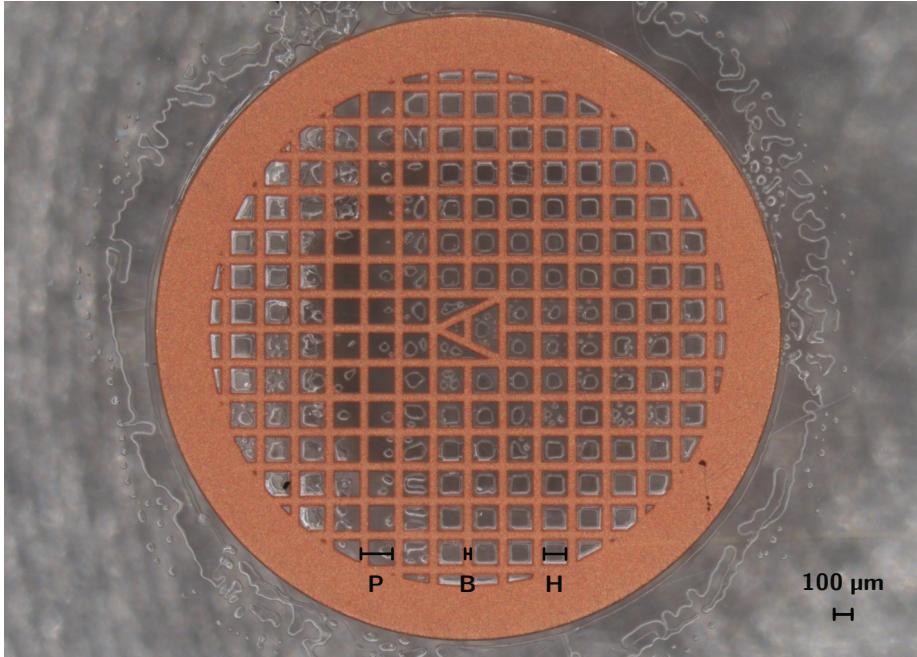
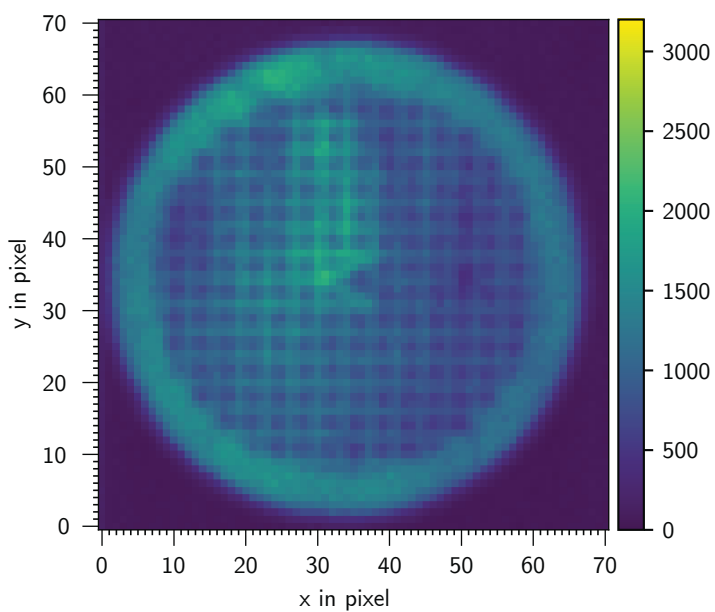


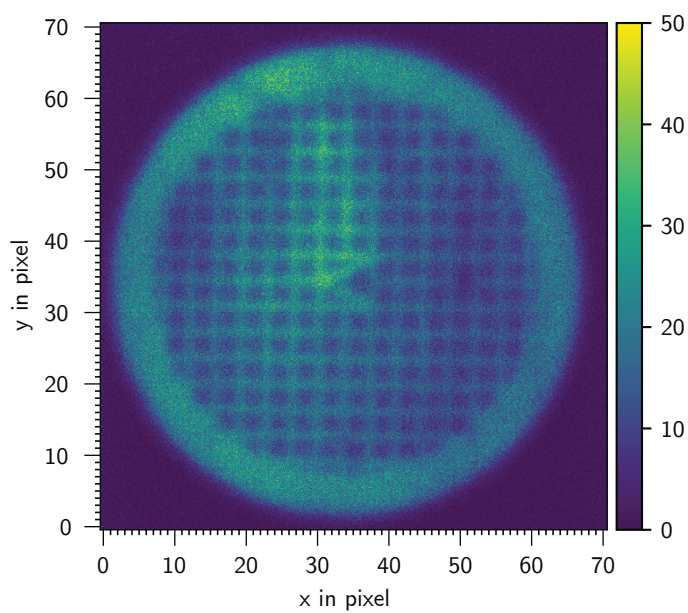
Figure 10.6: Light microscope image of the copper grid. The mesh size is 150 lines per inch. The pitch P is $165\ \mu\text{m}$, the bar width B is $40\ \mu\text{m}$, and the hole width H is $125\ \mu\text{m}$ [208].

7. A binning to a virtual pixel size of $6 \times 6\ \mu\text{m}^2$ is used. This means that each physical pixel is divided into 8×8 virtual subpixels. Only single event patterns selected via an energy filter are used. Since CoNN predicts exactly one PoE for each event pattern, the multiplicity that is the ratio between reconstructed PoEs and event patterns of both images has to be the same and is by a factor of 64 lower than the image using no subpixel reconstruction (Figure 10.7a). CoNN introduces no visible artifacts to the reconstructed copper grid. In comparison with the maximum method, the spatial resolution is increased.

Figure 10.8 depicts a line scan through the copper grid for the various methods. To increase the statistic, for the reconstructions on subpixel level, not a single line of pixels is depicted, but the sum over 12 physical pixels (96 virtual subpixels) between two horizontal grid lines is depicted. The spatial resolution of the experimental setup is limited by the used polycapillary optic and the distance between the polycapillary optic and the detector, and not limited by the properties of the detector system or the reconstruction method. The green dashed line shows the simulated signal distribution on the detector. Consequently, the classical η correction method leads to a similar



(a) Reconstructed image using the maximum method.



(b) Reconstructed copper grid in 8×8 subpixel regime with the CoNN.

Figure 10.7: Reconstructed image of the copper grid.

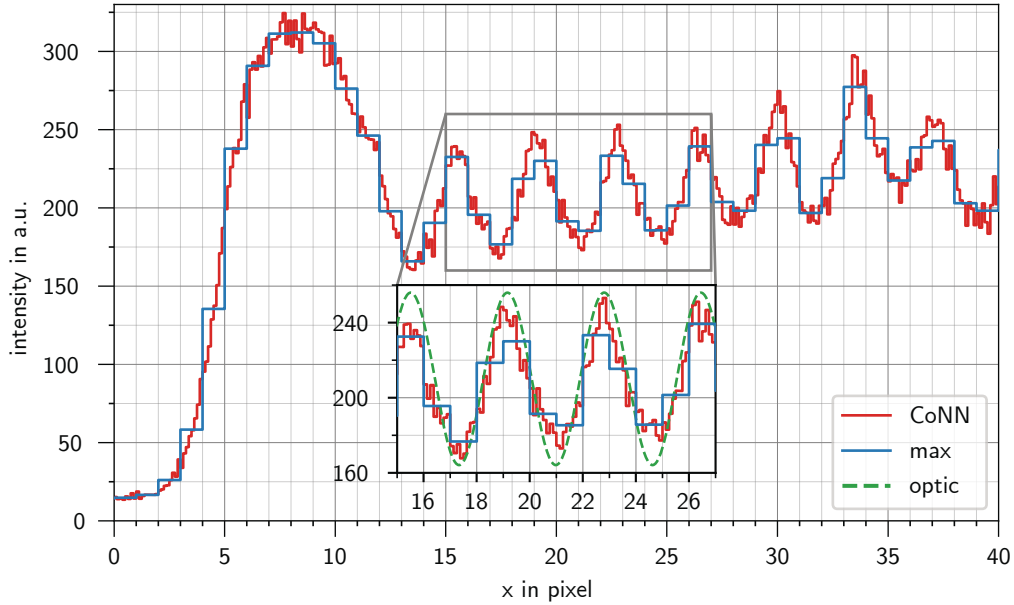


Figure 10.8: Line plot of the copper grid for different reconstruction methods. The line plot of max is divided by eight to get the same intensity as for subpixel resolution. In green, the approximated spatial resolution limit of the polycapillary optic and the experiment geometry is depicted.

spatial resolution.

The result obtained by CoNN improves the spatial resolution compared to the maximum method, and it is in good agreement with the simulated signal distribution on the detector surface. Due to the summation over many individual frames and, therefore, over many reconstructed PoEs, the intensity image is very sensitive to checkerboard artifacts. The small artifacts introduced by CoNN are on a sub-micrometer level (Section 7.3.1) and, therefore, in an 8×8 subpixel resolution representation not visible and neglectable.

CoNN provides reconstructed images in subpixel resolution. The reconstructed intensity images are checkerboard artifact-free.

10.2 Electron Detection

Electrons as primary particles are used as second example for comparison between the reconstruction methods. The reference to the used data can be found in Table D.3. All presented results created by approaches with neural networks in this Section require no previous event pattern analysis.

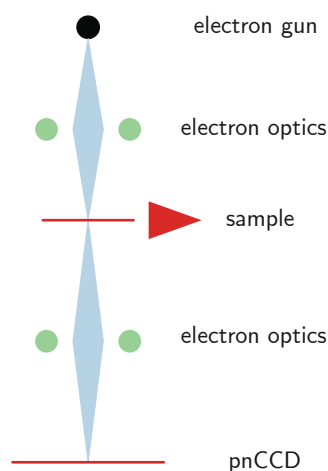


Figure 10.9: Experimental setup of the TEM measurement. The primary electrons are accelerated on a beam blanker which is used as a sample. The beam blanker has a triangle’s shape and totally absorbs primary electrons that hit it. The primary electrons which pass by the beam blanker are detected by a pnCCD.

The measurements for electrons are performed with a transmission electron microscope (TEM) [10] at an FEI Titan 80/300 G1 at the University of Bremen [209]. A schematic of the experimental setup is shown in Figure 10.9. The primary electrons are accelerated to various primary energies between 60 keV and 300 keV. A detailed list of the used data sets can be found in Table D.3. All measurements were recorded with a backside voltage of -420 V at the pnCCD (high charge handling capacity mode). An electron optics focuses the primary electrons on the sample. The sample of the measurements is a beam blanker that absorbs the primary electrons. The shape of the beam blanker is a triangle. The edges of the triangle are not completely smooth on a microscopic scale. Figure 10.10 depicts the fine structure of the beam blanker obtained by the weighted centroid method with η correction using an 8×8 subpixel resolution. It can be seen that the lower edge of the triangle is smoother than the upper edge. Therefore, the lower edge is used to obtain the MTF via the slanted edge method in this Section. The detailed procedure is described in Appendix D.1. Since the tracks of the electrons become longer and more complex with increasing primary energy, the obtained results are more useful for higher primary energies. Typically, measurements with transmission electron microscopes are performed up to 300 keV. The improvement of the spatial resolution by PoENN in comparison

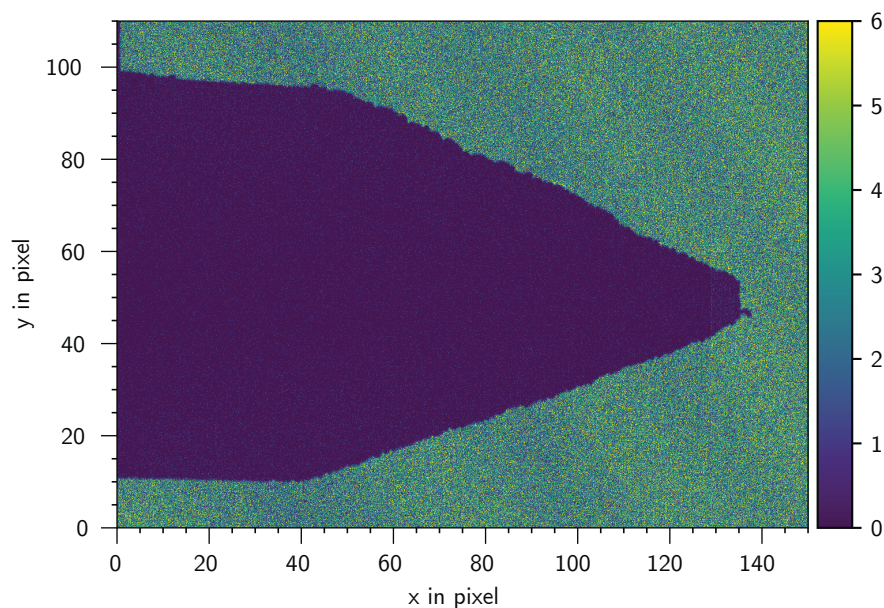


Figure 10.10: Fine structure of the beam blanker measured with a primary energy of 60 keV. An 8×8 subpixel resolution is applied using the η correction method. Not the whole image is shown, but only the region of interest. The reference to the used data is PID_e_060_1 (Table D.3).

to the classical methods increases with higher primary energies. Therefore, in the following, the results for electrons with a primary energy of 300 keV are presented in more detail.

Figure 10.11 shows the offset corrected raw data summed up over 150000 taken frames for a primary energy of 300 keV. The color scale denotes the energy deposition in arbitrary units. For clarity, not the whole image is shown but only the region of interest with 120×100 pixels containing the beam blanker. [11, 13] The input to the conventional reconstruction methods is offset corrected. Additionally, a gain correction with one value per octant is applied, corresponding to one value per ADC channel (Appendix B). The values for the gain correction are obtained from a homogeneous illumination with the same rate and energy of the primary electrons. The ratio between the common-mode and the signal generated by the primary electron is smaller than 0.1% and can be neglected. Consequently, the influence of the common-mode noise introduced by the column-parallel readout of the CAMEX can be neglected.

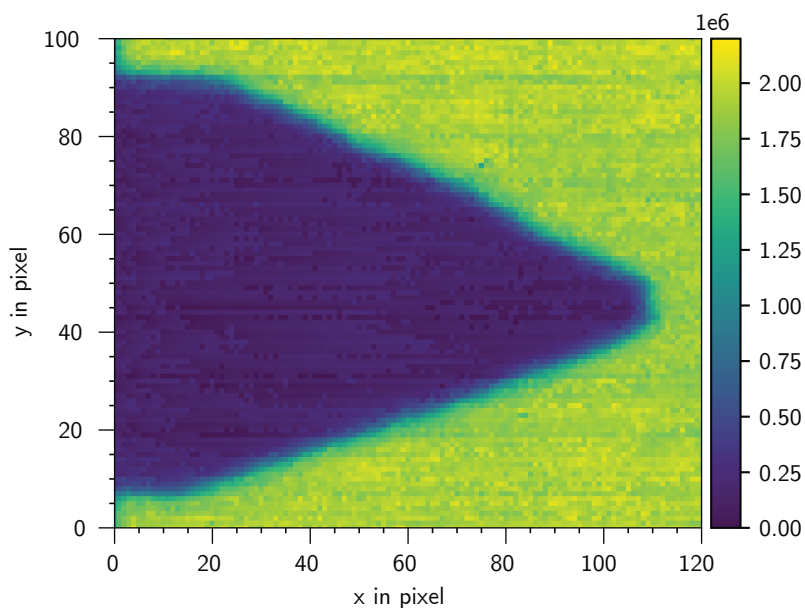


Figure 10.11: Offset corrected raw data for a primary energy of 300 keV summed up over all frames in Arbitrary Digital Units (ADU). The reference to the used data is PID_e.300_2 (Table D.3). Not the total field of view is shown, but only the region of interest (120×100 pixels). The readout nodes of the shown region of interest are located on the left side. Figure adapted from [13].

Figure 10.12 depicts the summation over the output of PoENN after applying a probability threshold of 0.5 (Chapter 8). The used data are the same as for Figure 10.11. The PoENN visibly improves the edges' sharpness of the beam blanker and reconstructs hidden features in the edges' structure. At the top of the beam blanker's tip, a bulge is clearly visible. Even the bulge's structure, invisible in the offset corrected raw data (Figure 10.11), is reconstructed by the PoENN. The beam blanker's upper edge is not straight and shows a structure in Figure 10.10. The PoENN can reconstruct the coarse curvature, whereas the curvature is hidden in the offset corrected raw data.

A quantitative measurement is the modulation transfer function (MTF). The MTF measures how much contrast of an object is transferred by the imaging process. This image process includes the behavior of the primary electrons in the silicon bulk of the pnCCD but, moreover, effects introduced by the electron optics and the electronics of the detector system. It can be obtained from the slanted edge produced by the beam blanker. A detailed description of the used process obtaining the MTF can be found in Appendix

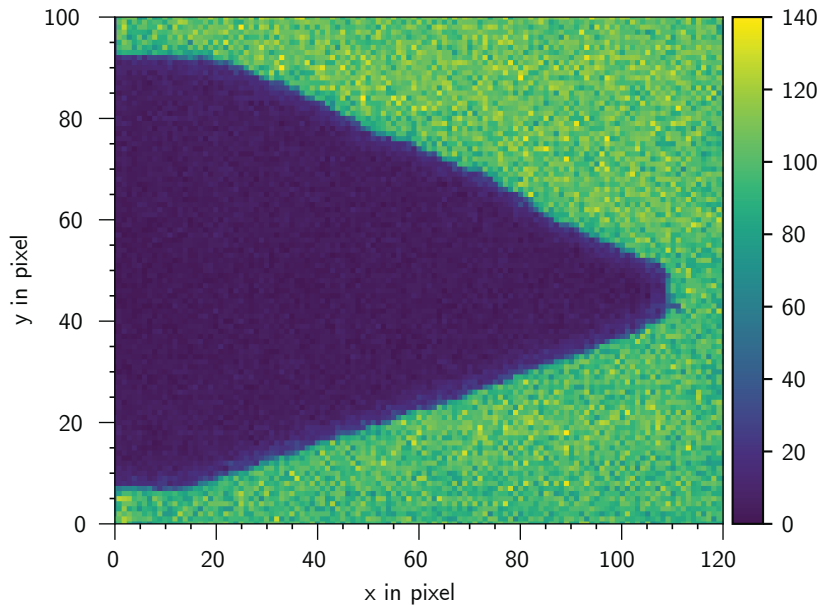


Figure 10.12: Summation over the binary output of the proposed neural network PoENN after applying the threshold of 0.5, which determines whether a pixel in the output contains a PoE. The primary energy is 300 keV. The reference to the used data is PID_e_300_2 (Table D.3). Not the total field of view is shown, but only the region of interest (120×100 pixels). The region of interest is the same as in Fig. 10.11. Figure adapted from [13].

D.1. Figure 10.13 shows the MTF for various reconstruction methods as a function of the spatial frequency. The basis of the conventional event pattern analysis is an event pattern detection with a threshold of 5σ of the pixel-wise noise. The simulated ground truth is the best achievable result of the MTF in a representation with a pixel size of $48 \times 48 \mu\text{m}^2$. A distinction is made between simulation and measurement. The simulation does not contain effects of the electron optics of the TEM or effects of the detector system's electronics further to the uncorrelated noise in the individual pixels. Therefore, the MTF of the measured data is always expected to be worse than the MTF obtained from the simulation data. However, the results are in a good agreement between the simulation and measurement. The best conventional method for 300 keV is the furthest away method (FAM). The MTF obtained by PoENN is between FAM and the ground truth and shows a large improvement of the spatial resolution.

Figure 10.14 depicts the reconstructed image for 60 keV using PoENN

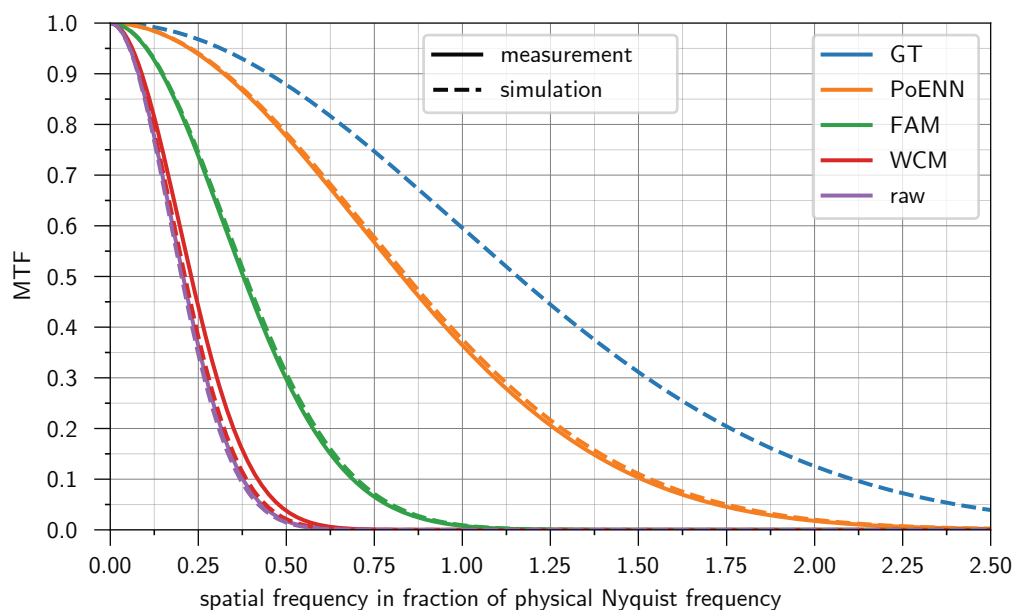


Figure 10.13: MTF based on the slanted edge method as a function of the spatial frequency for the ground truth (GT) which denotes the best achievable result, the proposed neural network (CNN), the furthest away method (FAM), the weighted centroid method (WCM), and the summed raw data (raw). The primary electron energy is 300 keV, and the pixel size is $48 \times 48 \mu\text{m}^2$. The reference to the used data is PID_e_300_2 (Table D.3). Figure adapted from [13].

with 4×4 subpixel resolution. The structure of the beam blanker’s edge is clearly visible. However, using PoENN with subpixel resolution leads to slightly pronounced checkerboard artifacts, which are one of the major open challenges for super-resolution. PoENN with the super-resolution extension can provide subpixel resolution without an event pattern analysis for low energetic electrons.

Figure 10.15 shows a reconstructed image by MISR (Chapter 9). For this, not individual frames are fed to the neural network but intensity images. These intensity images are obtained by summing over the individual frames with the reference PID_e_300_2 (Table D.3). Six intensity images containing 25000 individual frames each are used. An MTF measurement of the slanted edge is only limited meaningful and, therefore, not presented because the MISR was trained to find sharp edges. However, the edge’s structure of the beam blanker compared to Figure 10.11 is clearly visible. The fine structure such as the one in Figure 10.10 cannot be observed since this information is hidden by the statistical behavior of the primary electrons and the summation of the

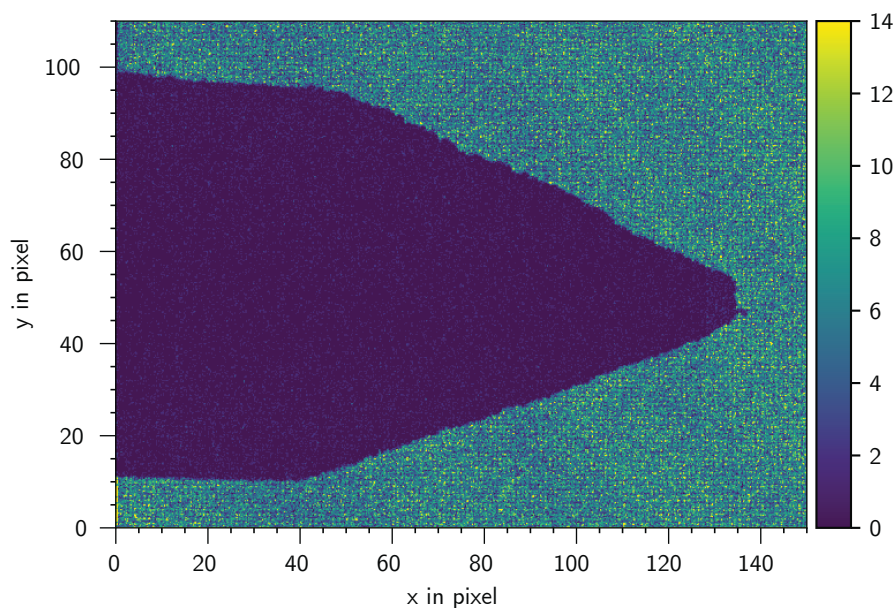


Figure 10.14: Fine structure of beam blanker measured with a primary energy of 60 keV. A 4×4 subpixel resolution using PoENN is presented. Not the whole image is shown, but only the region of interest. It is the same region of interest such as in Figure 10.10. The reference to the used data is PID_e_060_1 (Table D.3).

intensity image.

10.3 Summary

In this Chapter, the presented neural networks were applied to experimental data. For the presented measurements the three presented analyses based on neural networks CoNN (Chapter 7), PoENN (Chapter 8), and MISR (Chapter 9) improve the measured result significantly compared to classical methods. Especially for electrons with higher primary energies, PoENN and MISR show their full potential compared to the classical methods and increase the spatial resolution and the level of detail.

The choice of which neural network should be applied depends on the application. For low-dose applications with low primary electron energies or photons, the CoNN should be used. The CoNN requires a preceding event pattern analysis. This is no limitation compared to the classical methods since all

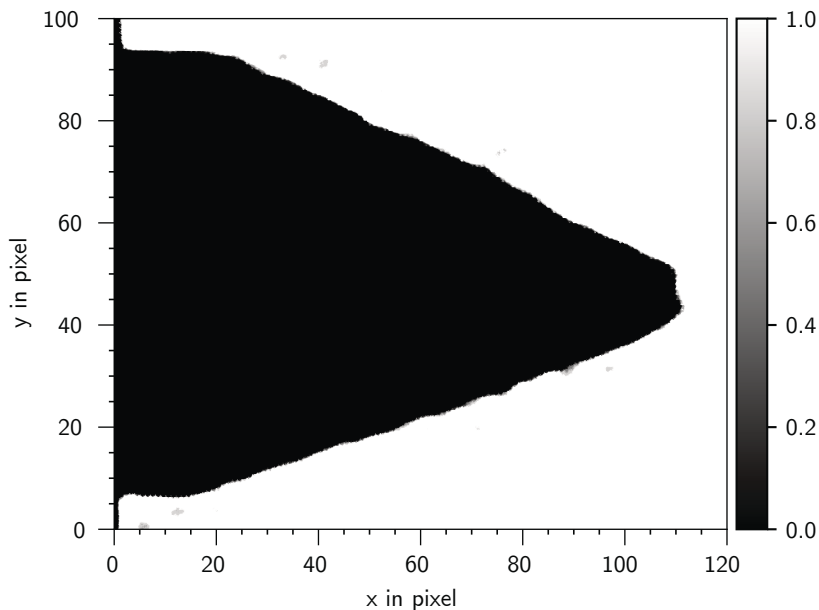


Figure 10.15: Output of the proposed neural network MISR. The primary energy is 300 keV. Six intensity images containing 25000 individual frames each are used. The reference to the used data is PID_e.300_2 (Table D.3). Not the total field of view is shown, but only the region of interest (120×100 pixels).

presented classical methods also require a prior event pattern analysis. If the computational time plays a major role PoENN with a super-resolution module can be used.

For electrons with higher primary energies, PoENN should be used. It was shown that PoENN increases the resolution in terms of the MTF in comparison to the presented classical approaches. Furthermore, PoENN requires no computational intensive event pattern analysis.

For higher particle rates, the energy depositions of individual particles are no longer distinguishable. As a consequence, the classical approaches are not applicable. PoENN can reconstruct the PoE of pattern pile-up events. For even higher primary particle rates resulting in intensity images, SISR, particularly MISR, can increase the spatial resolution and denoise the images where classical methods reach their limits.

Chapter 11

Conclusion and Outlook

For many applications in transmission electron microscopy and X-ray applications such as ptychography, diffraction, and fluorescence analysis, the precise **point of entry (PoE)** into the detector volume is of interest. However, modern detectors like the pnCCD do not detect the PoE but the energy deposited by the primary particles frame-wise. In the data frames, the energy deposition appears in the form of event patterns generated by signal electrons.

This thesis discusses primary particles that perpendicularly hit the detector surface. In this case, the energy deposition for photons is local, and the shapes of the event patterns are dominated by diffusion and repulsion. Consequently, except for noise, the obtained event patterns are similar for the same PoE. For electrons, especially with higher primary energies, the shapes of the event patterns are dominated by the energy deposition itself. The energy deposition caused by multiple scattering is randomly distributed along the particle tracks.

This thesis aims to develop a new analysis framework based on deep learning and neural networks to reconstruct the PoEs from the measured detector data and evaluate their potential and limits. Therefore, four different approaches based on the concept of convolutional neural networks were developed. Table 11.1 shows a comparison of the introduced convolutional neural networks. With these four approaches, a wide range of applications can be addressed. The neural networks were tested with data obtained by a simulation framework and several measurements. The measurements were performed with a detector system based on pnCCDs. However, the neural networks approach applies to all pixelated detectors and other kinds of particles such as protons or muons.

The **point of entry neural network (PoENN)** reconstructs the PoEs of individual primary electrons with energies above 100 keV on a physical pixel

Table 11.1: Overview of introduced neural networks. Single events are no pattern pile-up events, e.g., caused by intersecting particle traces. Individual pattern events are distinguishable for low rates, but pattern pile-up events are possible. In intensity images, no individual event patterns are identifiable. The obtained resolution varies from subpixel resolution to a resolution on pixel level, and the reconstruction speed varies for the different neural networks and used hardware. The requirements describe the previous corrections and analysis steps. The last column describes the training effort. The training effort depends on the used machine. It varies from several hours to several days.

name	Ch.	particle rate	resolution	speed	requirements	training
PoENN	8	low rate	medium	medium	offset map	medium
CoNN	7	single events	high	high	event analysis	low
SISR	9.1	intensity	low	medium	offset map	medium
MISR	9.2	intensity	medium	low	offset map	high

level. For primary electrons with energies below 100 keV and photons, the PoENN can reconstruct the PoE on subpixel level. For a primary energy of 300 keV, the resulting modulation transfer function (MTF) is 0.77 at a Nyquist frequency of 0.5 obtained from a slanted edge measurement. Compared to the best classical method used in the benchmark, the MTF at a Nyquist frequency of 0.5 is increased from 0.3 by a factor of 2.6.

Additionally, the PoENN can handle pile-up event patterns, e.g., intersecting particle traces, and reconstruct the PoEs of the contributing primary electrons. The MTF is constant up to a rate of $0.04 \text{ e}^-/\text{pixel}/\text{frame}$ ¹. Even for a primary electron rate increased by a factor of two, the neural network approach leads to a better MTF than the classical approach, which uses only single event patterns and achieves an MTF at 0.5 times the Nyquist frequency in the order of 0.3.

A reconstruction rate up to 900 Hz for a 256×256 pixel large detector was achieved. Due to its architecture, a computational intensive event pattern analysis that is required for all classical approaches is no longer necessary. Consequently, the speed of reconstruction is independent of the primary particle rate.

The **compact neural network** (CoNN) reconstructs PoEs on subpixel level for photons and low energetic primary electrons. The achieved subpixel resolution is less than 10% of the pixel dimensions. The precision for all

¹A primary particle rate of $0.04 \text{ e}^-/\text{pixel}/\text{frame}$ corresponds to a rate of $5 \cdot 10^6 \text{ e}^-/\text{s}$ for a full-frame readout (256×256 pixel) with 2000 Hz.

positions within the pixel structure was better or at least the same as the best compared conventional method. The CoNN can create artifact-free summed intensity images with super-resolution. The CoNN requires, as a previous step, an event pattern analysis. On the used GPU, an event pattern reconstruction rate of 10 kHz was demonstrated.

The presented **single image super-resolution (SISR)** approach and the **multi image super-resolution (MISR)** approach, that are based on the architecture of the PoENN, are presented for intensity images. In these intensity images, individual energy depositions of primary particles are no longer distinguishable due to the high primary particle rate. SISR and MISR provide, on the one hand, "denoising" and, on the other hand, extension into super-resolution. The structural similarity between the normalized ground truth and the normalized high-resolution output of SISR and MISR is calculated to measure the result's quality. A higher value represents a better result and a value of one perfect structural similarity. SISR provides a structural similarity of 0.87 for a signal-to-noise ratio of ten. The structural similarity between the low-resolution input to SISR and the ground truth is 0.3. This means SISR improves the image quality by a factor of 2.9.

The MISR combines multiple images of the same scenario to reduce noise effectively. A signal-to-noise ratio of only three was simulated to demonstrate the denoising process. The structural similarity increases with the number of used intensity images and is for using two images at the input 0.75 and for using 100 images 0.78. Therefore, it is sufficient for many applications to use only a few images of the same scenario to reconstruct a denoised high-resolution result.

The classical approach of denoising is averaging over multiple images. The structural similarity for averaging 100 frames is 0.32. This means MISR improves the image quality measured by the structural similarity by a factor of 2.4.

In the present work, the significant potential of deep learning and neural networks in the field of detector physics using pixelated semiconductor detectors was demonstrated. New data analysis methods for pixelated semiconductor detectors based on neural networks were developed and evaluated. These new methods provide a more precise and faster data analysis than conventional methods, improving resolution and enabling real-time analysis. Additionally, the presented work builds the bridge between data analysis for pixelated semiconductor detectors and machine learning. It provides the basis for a new era of data analysis for semiconductor detectors. This basis and the acquired understanding provide the starting point for future data analysis

methods.

One crucial parameter besides accuracy is the computation speed of the data analysis. The PoENN can reconstruct frames in real-time using GPUs for current detector systems. However, the next generation of detector systems provides faster readout rates. Therefore, the speed of future neural networks has to be higher. In addition, the transition from expensive and high power consumption servers or cloud computing toward edge computing should be expected. Here, specific hardware (ASICs) plays an important role. In the present thesis, the first test on a single **t**ensor **p**rocessing **u**nit (TPU) was performed. A future step will be to parallelize several TPUs on one host system. Another approach is to deploy the neural network to **f**ield-**p**rogrammable **g**ate **a**rrays (FPGAs) [210]. This hardware might be integrated to the sensor in the long term.

To further increase the accuracy of the neural network, detector-specific pixel-to-pixel variations can be added to the training process. This reduces the generality of the trained neural network but potentially increases the accuracy of the individual detector system.

A planned future step is to obtain the required training and validation data with a pencil beam realized by a laser scan relative to the pixel structure. Using experimental data instead of simulated data enables training of the neural network, which is entirely independent of simplifications and assumptions made during the Monte Carlo simulation. The result is a data set that perfectly describes the used detector system. The more precise data set increases the quality and accuracy of the neural networks trained by this data set.

Extending the developed approaches based on neural networks to photons of a broad energy band also is be a task of future works.

Furthermore, reinforcement learning offers the potential to train the required neural networks directly with measured data without requiring the exact PoE or simulated data. The result is a neural network perfectly tuned to the detector hardware. However, this requires a deep understanding of how neural networks act with data obtained from pixelated semiconductor detectors. This work provides the required knowledge for this purpose. A further extension of reinforcement learning could be optimizing detector parameters and operating conditions. The presented work presents the first step in this direction of a new era of data analysis and detector parameters optimization.

Appendix A

Physical Considerations

A.1 Spatial Precision, Spatial Accuracy, and Spatial Resolution

The spatial accuracy is defined as the average distance between the mean position of the individual reconstructed PoEs and their ground truth.

The spatial precision is defined as the standard deviation of the individual reconstructed PoEs.

The spatial resolution is defined as the average Euclidean distance between the individual reconstructed PoEs and their ground truth.

Figure A.1 illustrates the response in terms of precision, accuracy, and resolution for different reconstructions. The reconstructed PoEs are spatially close together for high precision and low accuracy. However, the distance between the mean position and the ground truth is high. For a low precision and a high accuracy, the mean position is spatially close to the ground truth. However, the reconstructed PoEs are widely spread. A good spatial resolution can only be obtained if the precision and accuracy are high. A low accuracy can, for example, be caused by a systematic error introduced by the reconstruction process. A low precision can be caused by the detector system's too high noise level.

A.2 Physical Limitations

Although modern computational approaches contribute to a significant improvement in data analysis, physical limits apply. Many of those physical limits apply to all data analysis methods and are not limited to approaches

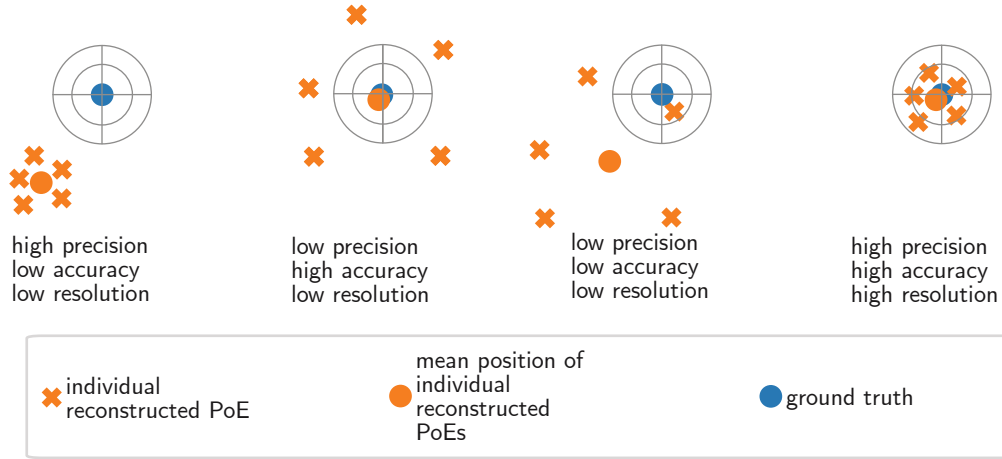


Figure A.1: Spatial precision, spatial accuracy, and spatial resolution. The ground truth and, therefore, the best reconstruction is shown as a blue circle in the center of the crosshair. The orange crosses indicate PoEs obtained by the reconstruction process. A high resolution can only be obtained if the precision and the accuracy are high.

based on neural networks. The limits of the parameters that data analysis methods can influence are investigated in this Chapter.

A.2.1 Position Resolution

One of the key features of pixelated detectors is the spatial resolution. The spatial resolution depends on the method of reconstruction, type of particle, and particle energy, but also detector properties such as the pixel size or operation conditions. In the following Section, the limitations of measurement resolution due to the pixel size are described, and a one-dimensional detector is investigated. The expansion to the spatial resolution σ_r of a two-dimensional pixelated detector with $r = \sqrt{x^2 + y^2}$ leads to $\sigma_r = \sqrt{2}\sigma_x$ if there is no preferential direction.

A.2.1.1 Binary Detector Response

A binary detector response (single event pattern) assumes that the generated charge cloud's spatial expansion is small compared to the pixel size. Only one pixel contains a signal. As reconstructed PoE, the center of the pixel is taken. Without restriction of generality, the origin of the coordinate system is at $x = 0$ is in the center of the pixel. The distance from center to center of the next pixel is the pixel size d . The position's uncertainty Δx is the difference

between the reconstructed PoE x_{rec} and the true PoE x_{truth} :

$$\Delta x = x_{\text{rec}} - x_{\text{truth}} \quad (\text{A.1})$$

Here, x_{rec} is the center of the pixel. If the true PoEs are distributed homogeneously over the pixel, the averaged uncertainty of the reconstruction method is zero $\langle \Delta x \rangle = 0$. The spatial resolution is defined as follows [38]:

$$\sigma_x = \sqrt{\langle \Delta x^2 \rangle - \langle \Delta x \rangle^2} = \sqrt{\langle \Delta x^2 \rangle} \quad (\text{A.2})$$

The uncertainty is maximal for a true PoE near the pixel's border and becomes smaller for more central hits. Assuming a homogeneous illumination, the density of hits $\rho(x)$ is given by [38]:

$$\rho(x)dx = \frac{1}{d}dx \quad (\text{A.3})$$

The variance of a random distribution is the second moment of this distribution [38]:

$$\sigma_x = \sqrt{\frac{1}{d} \int_{-d/2}^{d/2} \Delta x^2 d(\Delta x)} = \frac{d}{\sqrt{12}} \quad (\text{A.4})$$

The spatial resolution in this one-dimensional case is the size of the pixel times $1/\sqrt{12}$. This scenario is similar to the case where any number of pixels containing a signal by the primary particle and one pixel is reconstructed as PoE during data analysis, provided that the correct pixel is reconstructed. The spatial resolution calculated here, therefore, also indicates the maximum spatial resolution for the reconstruction method described in Chapter 8. The same limit applies for a reconstruction with subpixels (super-resolution), but the effective pixel size becomes smaller, and, thus, the best achievable spatial resolution becomes better. Table A.1 shows the resolution limits within different representations of the hit-maps.

A.2.1.2 Signal Split Over Multiple Pixels with Amplitude Sensitive Read-out

For Gaussian distributed charge clouds, a better estimate can be calculated using the center of gravity method than the center of the pixel. A Gaussian distributed charge cloud is a valid approximation for photons and low-energy electrons. Here, the spatial expansion of the charge cloud mainly results from diffusion and repulsion. The calculation for the center of gravity can be found in Appendix B.1.4.

For a perfect Gaussian distributed charge cloud and under the assumption of

Table A.1: Limit of the spatial resolution for binary detector response. All values are in μm . The physical limits with subpixel application can only be achieved if all PoEs are predicted correctly on subpixel level.

physical pixel size	no subpixel	2x2	4x4	8x8	16x16
25	7.2	3.6	1.8	0.9	0.5
48	13.9	7.0	3.5	1.7	0.9
100	28.9	14.4	7.2	3.6	1.8
150	43.3	21.7	10.8	5.4	2.7
200	57.7	28.9	14.4	7.2	3.6

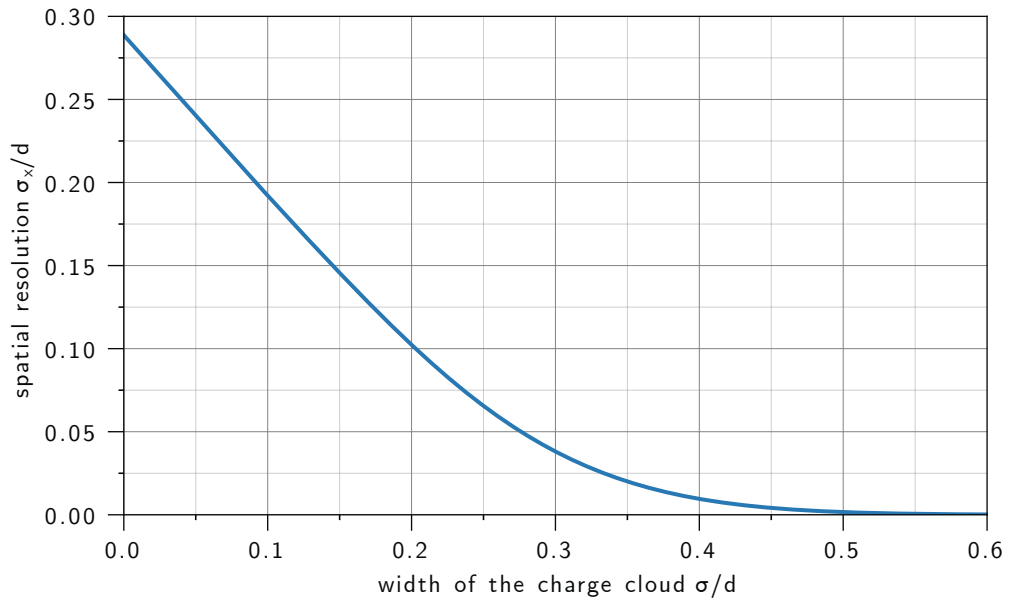


Figure A.2: Maximal spatial resolution for a Gaussian distributed charge cloud without additional noise. The spatial resolution is shown in units of the pixel width as a function of the charge cloud's width expressed in units of the pixel width. Figure adapted from [38].

no additional noise, the spatial resolution can be theoretically calculated by [38]:

$$\left(\frac{\sigma_x}{d}\right)^2 = \frac{1}{2\pi^2} \sum_{n=1}^{\infty} \frac{e^{-4\pi^2 n^2 (\sigma/d)^2}}{n^2} \quad (\text{A.5})$$

σ/d_x is the width of the charge cloud in units of the pixel size and n a summation index. Figure A.2 shows the spatial resolution as a function of

the charge cloud. For a perfect Gaussian distribution without noise, the uncertainty made by the reconstruction method is neglectable for $\sigma_x \gtrsim d/2$. In this case, the pixel size should be smaller than half the size of the standard deviation of the charge cloud's size.

The best achievable spatial resolution using the center of gravity method depends on the tuning of the pixel-width to the signal distribution's width and the signal-to-noise ratio. A large pixel-width leads to a resolution analog to the binary case. A too small pixel width leads to a small signal amplitude height per pixel and, thus, to a bad signal-to-noise ratio. A detailed view can be found in [38].

The influence of noise on the spatial resolution can be found in Appendix B.3. For the presence of noise, the spatial resolution depends on the position of the PoE within the physical pixel and is better near the borders of the physical pixels.

A.2.2 Energy Resolution

The energy resolution of the detector system is physically limited by the Fano statistics. The energy resolution of such a system can be described by the full width half maximum FWHM of the signal peak with a mean energy E [211]:

$$\text{FWHM} = 2\sqrt{2 \ln 2} \cdot w \cdot \sqrt{\text{ENC}^2 + \frac{F \cdot E}{w}} \quad (\text{A.6})$$

Here, w is the electron-hole pair generation energy, ENC is the equivalent noise charge of the readout chain, and $\frac{F \cdot E}{w}$ the Fano noise. The prefactor is derived from the conversion of the standard deviation of a Gaussian distribution to an expression of the full width half maximum [211]. Figure A.3 shows the energy resolution for different equivalent noise charges as a function of the signal's mean energy. In the physical limit, the ENC is equal to zero and does not contribute to the energy resolution.

A.2.3 Peak-to-background Ratio in the Energy Spectrum

The basic physical limit for the peak-to-background ratio in the energy spectrum is determined by the out-scattered particles, which partly deposit their energy in the detector volume. This leads to a characteristic background signal. To determine this physical limit, a silicon detector with no insensitive layers at the entrance window and a charge collection efficiency of 100% is assumed. A detector thickness of 450 μm is assumed. A thicker detector leads to

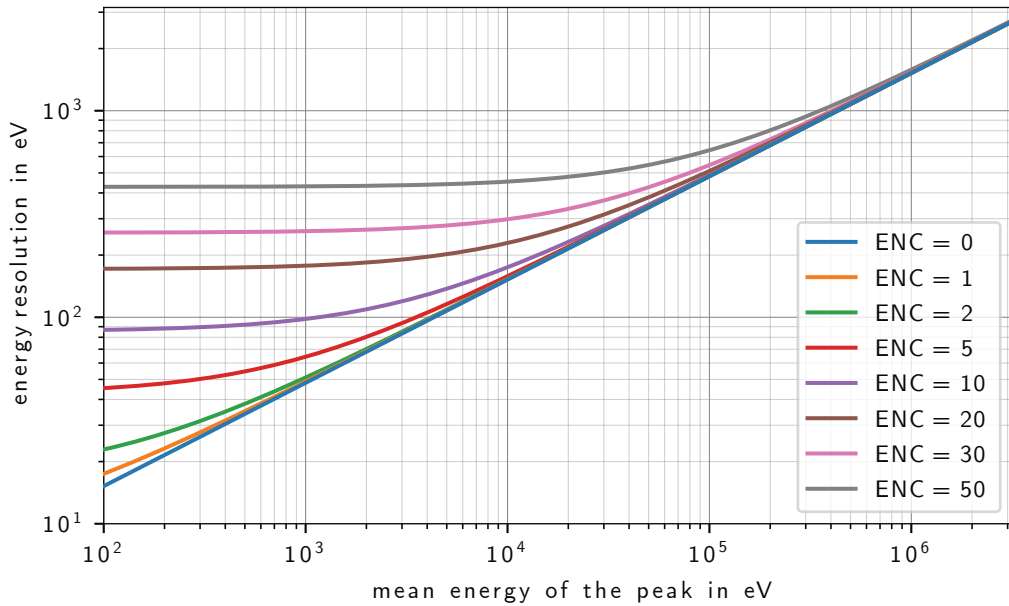


Figure A.3: Energy resolution as a function of the mean energy of the peak for different equivalent noise charges. The physical limit and, therefore, the best achievable energy resolution for a given mean energy of the peak is shown as the blue line with an equivalent noise charge of zero.

a smaller ratio of forward-scattered particles. If the primary particle or at least one secondary particle leaves the detector volume, an event is an out-scattered event. For each energy and each primary particle type, 10 million events were simulated. A detailed description of the spectra for photons and electrons can be found in Section 4.5, respectively, Section 4.6.

A.2.3.1 Peak-to-background Ratio in the Energy Spectrum for Photons

Figure A.4 shows the ratio of out-scattered events in silicon. The forward-scattered events are neglectable for low primary energies and increase with higher primary energies. The ratio of backscattered events is the highest at low energies and has a minimum at 11 keV. At higher energies, the probability of an escaping secondary particle increases.

Figure A.5 shows the peak-to-background ratio for photons as a function of the primary energy. The peak's amplitude is calculated by a Gaussian fit. The background is defined by a constant fit over a range of 1000 eV. The range is kept constant for all primary energies to make the results comparable for different primary energies. The higher end of the range is chosen at 1738 eV,

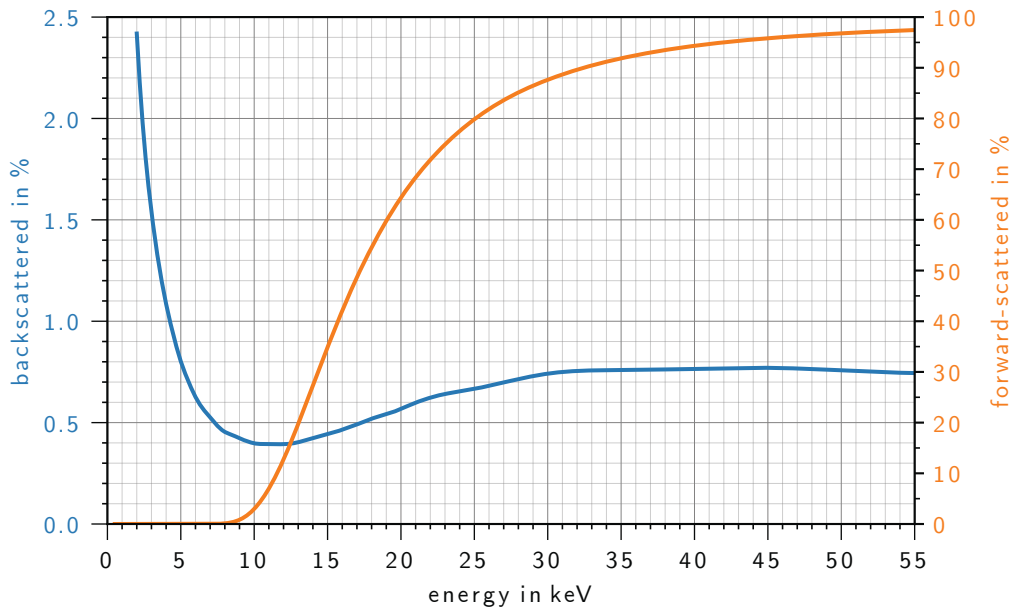


Figure A.4: Ratio of out-scattered events for photons as a function of the primary energy. The thickness of the silicon bulk is $450\ \mu\text{m}$. The blue curve shows the ratio of backscattered particles, and the orange curve the ratio of forward-scattered particles.

which is 100 eV below the K-edge.¹ The value of the peak-to-background ratio is very sensitive to the choice of the position and the range of the constant fit. However, the basic shape remains. Figure A.6 shows the Gaussian fit and the range for the background for a primary energy of 6 keV.

The peak-to-background ratio initially increases with increasing primary energy since fewer particles are backscattered. The ratio of forward-scattered particles becomes more significant for higher primary energies, and the peak-to-background ratio decreases. For the pnCCD, the entrance window limits the peak-to-background ratio to approximated 10000 at a primary photon energy of 7 keV [98].

¹Typically, the peak-to-background ratio is defined as the ratio between the manganese K_{α} signal peak and the mean value of the spectrum between 800 eV and 1200 eV. However, this definition is for the considerations made not applicable since we want to demonstrate the energy dependency of the peak-to-background ratio.

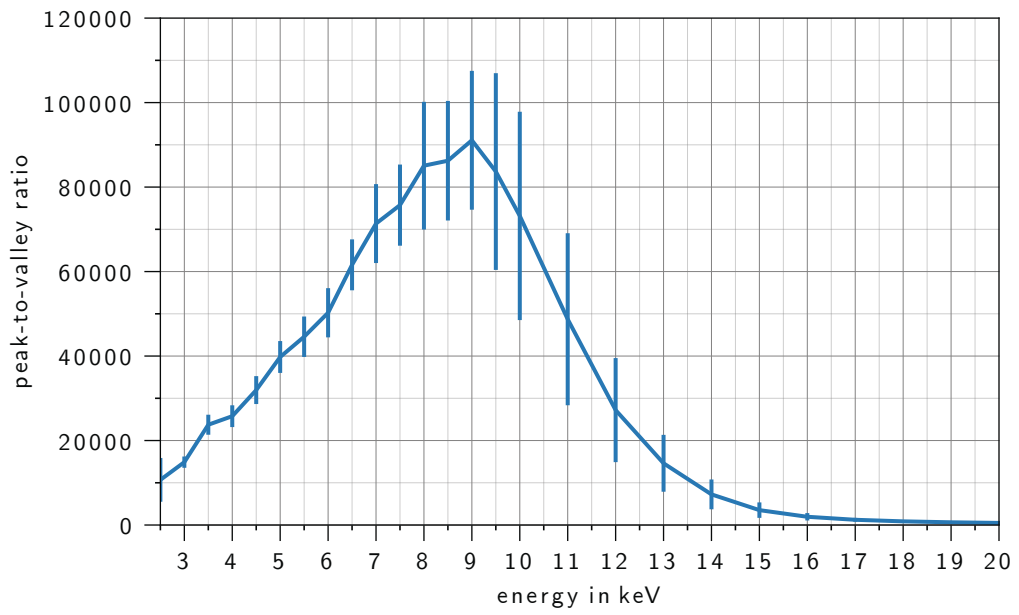


Figure A.5: Peak-to-background ratio for photons as a function of the primary energy. The error bars result from the uncertainties of the fits.

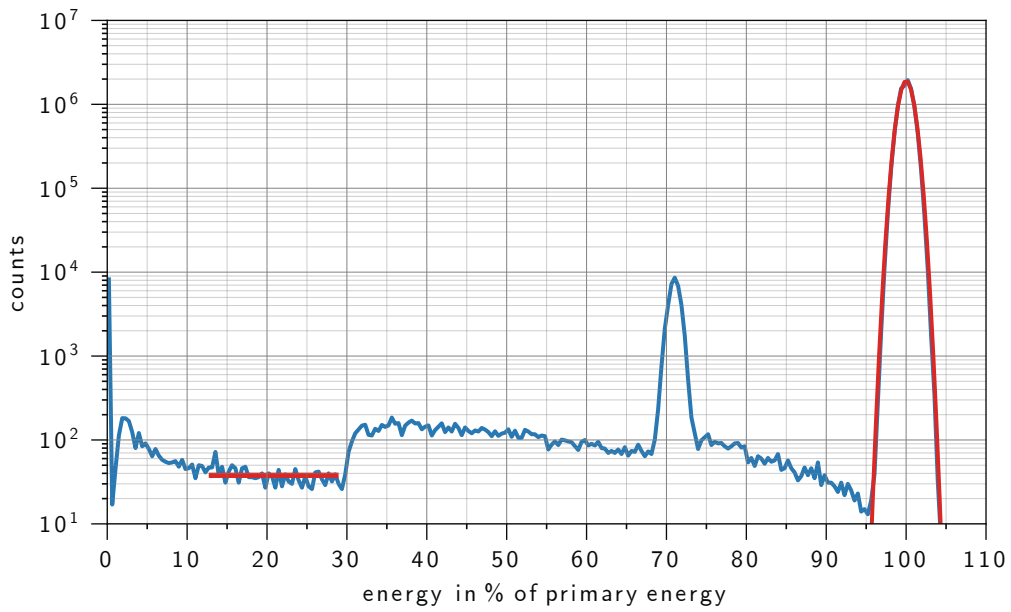


Figure A.6: Example of the fit for the peak-to-background ratio for photons with a primary energy of 6 keV. The spectrum is shown in blue, and the Gaussian fit of the main peak and constant fit of the background are shown in red. The peak at approximated 70% of the primary energy is the silicon escape peak.

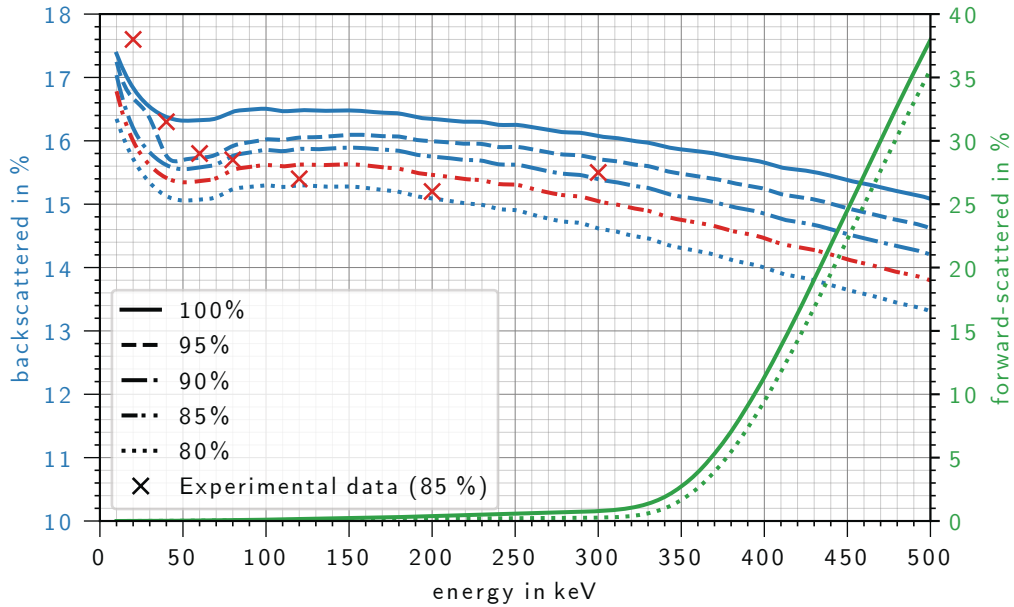


Figure A.7: Backscattering coefficient for electrons as a function of the primary energy. The thickness of the silicon bulk is $450\ \mu\text{m}$. The blue curves denote the ratio of events where the primary electron or a secondary particle is backscattered. The green curve denotes the ratio of events with a forward-scattered particle. The different line styles describe the maximal fraction of the primary energy which has to be deposited in the detector in order for the event to count as an out-scattered event. The line style is the same for backscattered and forward-scattered particles. This corresponds to a vertical line in the energy spectrum. Counts at lower energies than the line count as out-scattered events. Counts in an energetic higher bin than the vertical line do not count as out-scattered events. For a ratio of 100 %, every event, for that the primary particle or at least one secondary particle leaves the detector volume is counted as out-scatter event. The red crosses denote the backscattering coefficient extracted from measurements [69]. For the data obtained by the measurement, a fraction of the maximal energy deposition of 85 % was assumed. The red curve shows the backscattered ratio for a maximal energy deposition of 85 %.

A.2.3.2 Peak-to-background Ratio in the Energy Spectrum for Electrons

Figure A.7 shows the energy dependence of the backscattering and forward-scattering coefficient for the data obtained from the Monte Carlo simulation and experimental data from a pnCCD. The detector material is silicon with a thickness of 450 μm . For the measurement, an event counts as backscattered event if the primary energy was not totally deposited in the detector, i.e., the deposited energy is not in the spectrum's main peak. For higher energies, backscattering becomes less likely. The experimental data are obtained by indirect measurements. Every event which has less than 85 % of its primary energy deposited in the detector volume is counted as an out-scattered event. For 300 keV, the ratio of out-scattered events obtained from the measured data increases since, for this energy, the ratio of forward-scattered events can no longer be neglected. The primary energy-dependent behavior of the backscattered event ratio is also empirically described by Tabata et al. [212]. The empirical description agrees reasonably well with the measurement and the simulation.

Figure A.8 shows the peak-to-background ratio for electrons as a function of the primary energy. The peak's amplitude is calculated by a Gaussian fit. The background is defined by taking the average over the spectrum between 10 % and 80 % of the primary energy. The range is kept constant for all primary energies to make the results for different primary energies comparable.¹ The value of the peak-to-background ratio is sensitive to the background's choice and definition. However, the basic shape remains.

The peak-to-background ratio initially increases with increasing primary energy since fewer particles are backscattered. For higher primary energies above 350 keV, the ratio of forward-scattered particles becomes more significant, and the peak-to-background ratio decreases.

A.2.4 Particle Rate

The particle rate's upper limit depends on the readout speed of the detector and its dynamic range and is, therefore, specific for the detector system. Performing measurements with this upper limit particle rate results in intensity images.

If one wants to distinguish between single-particle tracks in the detector, the rate has to be smaller. In this case, the rate is limited by the range traveled by the particles in the detector. This range depends on the detector material, the particle type, and the particle energy.

¹The upper limit of 80 % also ensures for a primary energy of 10 keV that the photo escape peak is outside the background.

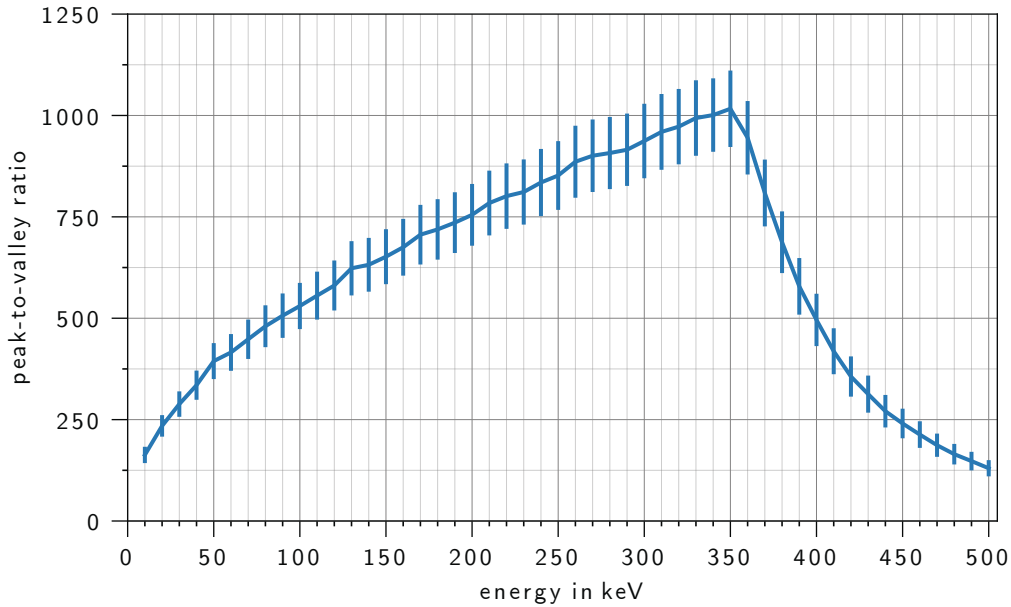


Figure A.8: Peak-to-background ratio for electrons as a function of the primary energy. The error bars result from the uncertainties of the fits.

The limiting factor is the maximum lateral expansion of the charge cloud and not the track length of a particle in the material. For photons as primary particles, the lateral expansion of the charge cloud is mainly dominated by the drift process and, therefore, strongly depends on the detector parameters such as the thickness or the back contact voltage (Section 4.8). The charge cloud can be approximated as radially symmetric.

For electrons as primary particles, the maximum lateral expansion is dominated by multiple scattering and, thus, strongly depends on the primary energy. The statistical character of multiple scattering allows only to make statements about the average over many particles and not about an individual particle.

A quantitative view of probability for pattern pile-up events for photons can be found in [213] and [214] and for electrons in Figure B.3 on page 233. Increasing the primary particle rate from this limit, a transition from single-particle tracks to intensity images happens.

A.3 Units of the Bethe Bloch Formula

The units of the Bethe Bloch equation are as follows:

$$\left[\frac{A^4 s^4 \frac{1}{\text{mol}} \frac{\text{kg}}{\text{m}^3}}{\frac{A^2 s^8}{\text{kg}^2 \text{m}^6} \frac{\text{kg}}{\text{mol}} \text{kg} \frac{\text{m}^2}{\text{s}^2}} \right] = \left[\frac{A^4 \cancel{s}^6 \text{kg}^3 \text{m}^6 \cancel{\text{mol}}}{A^4 \cancel{s}^6 \text{s}^2 \cancel{\text{kg}}^2 \cancel{\eta}^5 \cancel{\text{mol}}^5} \right] = \left[\frac{\text{kgm}}{\text{s}^2} \right] = \left[\frac{\text{J}}{\text{m}} \right] \quad (\text{A.7})$$

$$\left[\frac{\text{J}}{\text{m}} \right] = 10^{-6} \left[\frac{\text{J}}{\mu\text{m}} \right] = \frac{10^{-6}}{e} \left[\frac{\text{eV}}{\mu\text{m}} \right] = \frac{10^{-9}}{e} \left[\frac{\text{keV}}{\mu\text{m}} \right] \quad (\text{A.8})$$

Here, e is the elementary charge.

A.4 Landau Distribution Function

The distribution of the energy loss f_L of an electron passing a thin absorber is described by the Landau distribution [68]:

$$f_L = \frac{\theta(\lambda)}{\xi} \quad (\text{A.9})$$

$$\theta(\lambda) = \pi^{-1} \int_0^\infty e^{(-t \ln(t) - \lambda t)} \sin(\pi t) dt \quad (\text{A.10})$$

$$\lambda(x, \Delta) = \frac{\Delta - \langle \Delta \rangle}{\xi} - \beta^2 - \ln(k) - 1 - C_E \quad (\text{A.11})$$

$$\xi(x) = \frac{z^2 e^4}{4\pi \epsilon_0^2 m_e c^2} \frac{N_A \rho Z}{A} \beta^{-2} x \quad (\text{A.12})$$

$$k = \frac{\langle \Delta \rangle}{E_{\max}} \quad (\text{A.13})$$

$$\beta = \frac{v}{c} \quad (\text{A.14})$$

Here, Δ is the energy loss, $\langle \Delta \rangle$ is the mean energy loss, x is the thickness of the absorber, E_{\max} is the maximum transferable energy in one collision, Z is the atomic number, A is the atomic weight, ρ is the density of the absorber, and $C_E = 0.5772$ is the Euler constant. ξ is the prefactor of the Bethe Bloch formula (eq. 2.17 on page 24) times the thickness of the absorber. Note that the original nomenclature is used, and λ is not the wavelength. The mean energy loss can be obtained from the Bethe Bloch formula [38]:

$$\langle \Delta \rangle = \xi \left(\ln \left(\frac{2m_e c^2 \beta^2 \gamma^2}{I} \right) + \ln \left(\frac{\xi}{I} \right) + 0.2 - \beta^2 - \delta \right) \quad (\text{A.15})$$

Here, δ is a density correction, which becomes important at high energies, and I is the mean excitation energy. The maximum of the distribution is at $\lambda \approx -0.22$ and the full width at half maximum can be calculated to [38]:

$$W_{\text{FWHM}} \approx 4.018\xi \quad (\text{A.16})$$

A.5 Electric Potential and Field Approximation in the Silicon Detector Volume for pnCCD

In this Appendix, the equation for the electric potential and the electrostatic field for a model of the pnCCD is derived. In this model, no pixel structure and a homogeneous space charge density in the bulk and the n region are assumed. In addition, the thin p^+ region at the front- and the backside are neglected, and a totally depleted detector is assumed. Eq. A.17 shows the one-dimensional Poisson equation for the potential φ .

$$\nabla \varphi \stackrel{1 \text{ dim}}{=} \frac{\partial^2 \varphi}{\partial z^2} = -\frac{\rho}{\epsilon_0 \epsilon_{\text{Si}}} \quad (\text{A.17})$$

ϵ_0 is the electric permittivity, and ϵ_{Si} is the dielectric constant of silicon. The space charge density in the n -doped bulk is given by

$$\rho = e \cdot n_{n^-} \quad (\text{A.18})$$

and the space charge density in the n region near the frontside contact is

$$\rho = e \cdot n_n \quad (\text{A.19})$$

with the elementary charge e , the space density n_{n^-} of the n^- dopant and the space density n_n of the n dopant. A schematic drawing of the model and the choice of the coordinate system is shown in Figure 4.1 on page 49. The origin of the coordinate system is at the surface of the backside, and the z -axis goes toward the detector volume. The transition between the ranges is at $z = \delta$. The thickness of the detector is d . This leads to a general description of the space charge density:

$$\rho(z) = \begin{cases} e \cdot n_{n^-} & 0 < z \leq \delta \\ e \cdot n_n & \delta < z < d \\ 0 & \text{otherwise} \end{cases} \quad (\text{A.20})$$

The solution for the electric field can be parameterized as:

$$\varphi(z) = \begin{cases} \varphi_{n^-} = A_{n^-} \cdot z^2 + B_{n^-} \cdot z + C_{n^-} & 0 < z \leq \delta \\ \varphi_n = A_n \cdot z^2 + B_n \cdot z + C_n & \delta \leq z < d \end{cases} \quad (\text{A.21})$$

Applying the voltage V_b at the backside of the detector and V_f at the frontside, the solution to eq. A.17 must obey the following boundary conditions:

$$\begin{aligned}
 \varphi(z=0) &= \varphi_{n^-}(z=0) = V_b \\
 \varphi(z=d) &= \varphi_n(z=d) = V_f \\
 \varphi_{n^-}(z=\delta) &= \varphi_n(z=\delta) \\
 \frac{\partial}{\partial z}\varphi_{n^-}(z=\delta) &= \frac{\partial}{\partial z}\varphi_n(z=\delta)
 \end{aligned} \tag{A.22}$$

Solving the resulting system of equations leads to the solution for the coefficients A_i , B_i , and C_i :

$$\begin{aligned}
 A_{n^-} &= -\frac{\rho_{n^-}}{2\epsilon_0\epsilon_{Si}} \\
 B_{n^-} &= \frac{V_f - V_b}{d} + \left(\frac{\delta^2}{2d} - \delta\right) \cdot \frac{\rho_n - \rho_{n^-}}{2\epsilon_0\epsilon_{Si}} + \frac{d \cdot \rho_n}{2\epsilon_0\epsilon_{Si}} \\
 C_{n^-} &= V_b
 \end{aligned} \tag{A.23}$$

$$\begin{aligned}
 A_n &= -\frac{\rho_n}{2\epsilon_0\epsilon_{Si}} \\
 B_n &= \frac{V_f - V_b}{d} + \frac{\delta^2 \cdot (\rho_n - \rho_{n^-})}{2d \cdot \epsilon_0\epsilon_{Si}} + \frac{d \cdot \rho_n}{2\epsilon_0\epsilon_{Si}} \\
 C_n &= V_b + \frac{\delta^2 \cdot (\rho_{n^-} - \rho_n)}{2\epsilon_0\epsilon_{Si}}
 \end{aligned} \tag{A.24}$$

The electric field is determined by the negative gradient of the electric potential:

$$E(z) = -\frac{\partial\varphi(z)}{\partial z} = \begin{cases} -2A_{n^-} \cdot z - B_{n^-} & 0 < z \leq \delta \\ -2A_n \cdot z - B_n & \delta < z < d \end{cases} \tag{A.25}$$

Appendix B

Classical Data Analysis Methods

In the following Appendix, classical data analysis methods instead of neuronal networks are described. This understanding is needed for comparing the quality of the introduced analysis methods based on neural networks in this work.

Typically, the first step is a signal offset correction. The offset of the pixels varies, for example, because of slightly different gains and leakage currents. An offset map $o_{x,y}$ (eq. B.1) is determined and subtracted from the raw data. For this purpose, frames without illuminating the detector are recorded. These frames are called dark frames $D_{i,x,y}$ in the following. The pixel-wise average over a set of dark frames is the offset map $o_{x,y}$. Furthermore, the standard deviation over the dark frames is defined as noise map $n_{x,y}$ (eq. B.2). In this notation, i describes the frame's temporal index (frame id), x the frame's column index, and y the frame's row index. Both the offset map and the noise map are calculated for each pixel separately. [11]

$$o_{x,y} = \bar{D}_{i,x,y} = \frac{1}{n} \sum_i^n D_{i,x,y} \quad (\text{B.1})$$

$$n_{x,y} = \sigma_{D(i,x,y)} = \sqrt{\frac{1}{n} \sum_i^n [D_{i,x,y} - o_{x,y}]^2} \quad (\text{B.2})$$

An optional step at this point is the common-mode correction. So-called common-mode describes the temporal fluctuations in the supply voltages of the detector readout circuit. These temporal fluctuations affect all pixels that are readout simultaneously in the same manner. The common-mode correction can be realized in several ways. All these correction methods only consider pixels that do not contain any signal from an energy deposition but contain only noise. A simple common-mode correction is to subtract the

median of the common-mode affected pixel values, which is stable compared to the mean value against outliers of the simultaneously readout pixels. This method works if at least half of the simultaneously readout pixels are free of signal charges. [98]

The next step is an optional multiplication of a gain map $g_{x,y}$ (eq. B.3) to calibrate the amount of deposited energy. The gain map can be obtained through several ways.

One way is to obtain a relative gain map. With this relative gain map, the inhomogeneous amplifications of the pixels and the readout chain can be corrected. This is crucial to get well reconstructed PoEs of the incoming particles because the energy depositions in single pixels are compared with each other. The relative gain map is obtained by a reference measurement $H_{i,x,y}$ with spatially homogeneous illumination with a high particle rate. Under such conditions, the signal in every pixel should be the same. The numerator of eq. B.3 describes each pixel's mean signal for each frame of the reference measurement. The denominator is a normalization factor that can be chosen individually.[69]

$$\frac{1}{g_{x,y}} = \frac{\frac{1}{n} \sum_i^n H_{i,x,y}}{\frac{1}{n_y} \frac{1}{n_x} \frac{1}{n} \sum_y^{n_y} \sum_x^{n_x} \sum_i^n H_{i,x,y}} \quad (\text{B.3})$$

The parameters n_x and n_y specify the number of columns and rows of the detector, respectively. In the following, a gain map with one gain value per octant is applied for the conventional methods [69].

The other alternative way for determining the gain map, especially for low energies and low particle rates, is to calculate the gain map from single pattern events with an energy deposition in only one pixel. This method requires an event pattern analysis (Section B.1). From the event pattern analysis, a pixel-by-pixel spectrum can be created. The peak in the spectra and the known primary energy of the incoming particle can then be used as a basis for the calibration. [98]

The offset corrected frames $S_{i,x,y}^{(\text{oc})}$, and the gain corrected frames $S_{i,x,y}^{(\text{gc})}$ can finally be calculated from the raw frames $S_{i,x,y}^{(\text{raw})}$:

$$S_{i,x,y}^{(\text{oc})} = S_{i,x,y}^{(\text{raw})} \ominus o_{x,y} \quad (\text{B.4})$$

$$S_{i,x,y}^{(\text{gc})} = S_{i,x,y}^{(\text{oc})} \odot g_{x,y} \quad (\text{B.5})$$

Here, \ominus and \odot denote frame-wise difference and multiplication.

B.1 Event Pattern Analysis

An event pattern analysis searches for event patterns in the recorded frames. These event patterns are a group of adjacent pixels which exceed a certain threshold. Figure B.1 shows an offset corrected example frame on the left side and the extracted event patterns on the right side.

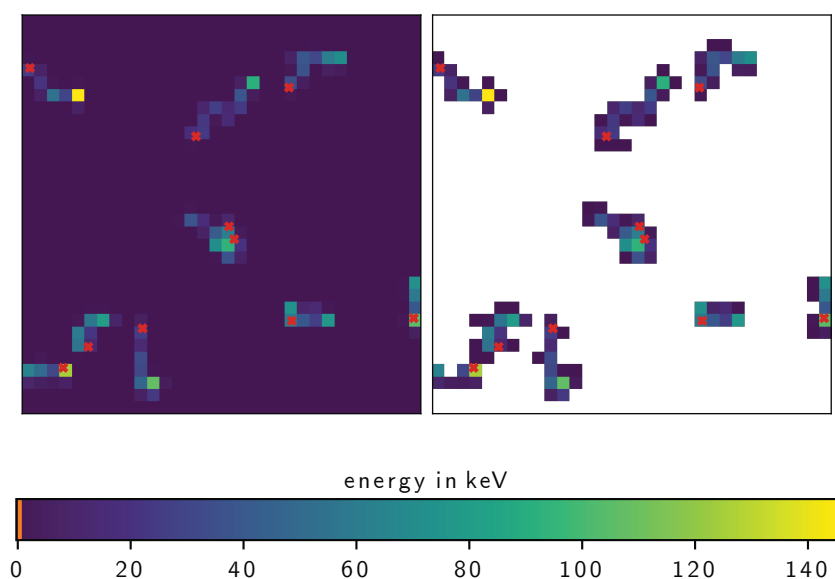


Figure B.1: Event Pattern Analysis. The left image shows a simulated example frame. The energy deposition into the individual pixels is color-coded. The red crosses label the PoEs of the primary electrons with an energy of 300 keV. The right image shows the remaining event pattern after the event pattern analysis. The primary threshold is at 5σ (orange line in the color scale), and the secondary threshold is at 2σ for an assumed equivalent noise charge of $20 e^-$. Event patterns with two PoEs denote a pattern pile-up event.

Here, the event pattern analysis can basically be structured into three parts.

The first step is the individual classification of all pixels for each frame. The second step is to cluster the pixels to events, and the third step is the housekeeping to ensure the correct assembling of the events after the

reconstruction of the PoE.

The individual classification of the pixel works with thresholds. Typically, the primary threshold is five times the noise value, and the secondary threshold is in the range of two to four times the noise map [98]. After the classification, the event clusters are built. All pixels that are higher than the secondary threshold and physically connected are combined to one cluster. This cluster is counted as an event if at least one pixel is higher than the primary threshold. This pixel-by-pixel classification is robust to the optional gain correction but not necessarily robust to a common-mode correction.

Typically, the event analysis provides information for housekeeping such as the frame index, the position, the bounding box which rectangularly surrounds the cluster, the signal of the pixels above the threshold, and their relative positions in the bounding box. Information such as the number of pixels above the threshold and the sum of the signals representing the total energy of the event can be provided optionally.

Since the frames are independent of each other, the event pattern analysis is perfectly parallelizable.¹ For the classical data analysis methods, the event pattern is the bottleneck of the performance but is inevitably required to create a spectrum or obtain the PoE of individual primary particles. The actual speed of the event pattern analysis strongly depends on the primary particle rate and is in the order of several minutes to hours for a typical measurement. For example, the event pattern analysis with a primary threshold of 5σ and a secondary threshold of 2σ of the noise and an electron rate of $0.92 \cdot 10^{-3}$ e⁻/pixel/frame (PID_e_300_1, Table D.3) on a standard workstation on six CPU cores [218] is performed with a frame rate of around 180 Hz. For an roughly 2.7 times higher event rate per frame (PID_e_300_2, Table D.3), the performance decreases to approximately 120 Hz.

B.1.1 Influence of Threshold Variations

Varying the primary threshold acts such as an event filter because the primary threshold is the criterion of whether an event cluster of an energy deposition is counted as an event pattern or not.

The choice of the secondary threshold impacts the ratio between the frequency of occurrence of the pattern with different sizes. The total number of detected patterns is not influenced by the secondary threshold as long as

¹The event analysis is performed with the help of methods *label()* and *find_objects()* from Scipy [215]. For the parallelization, the Python module Dask [216] is used [217].

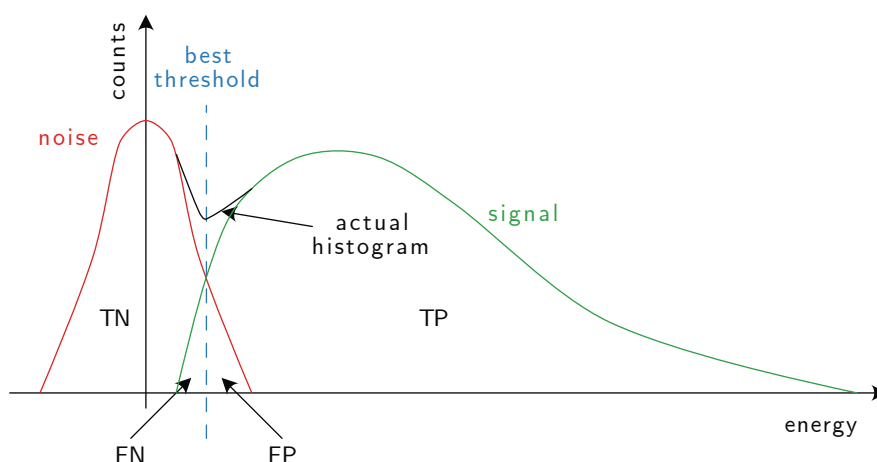


Figure B.2: Choice of the secondary threshold that decides whether a pixel contains a signal from an energy deposition or just noise. The shown spectrum is a so-called pixel spectrum. This means no event pattern recombination is performed, but the values of an individual pixel are used. TN denotes the true negative, TP the true positive, FN the false negative, and FP the false positive classified pixels. Figure adapted from [219].

two pixels that exceed the primary threshold are not connected. Thus, two events are merged into one.

Figure B.2 shows the best selection of the secondary threshold. Here, *best* is defined as the value where the number of false negatives and false positives is minimal. In this context, negative contributions are contributions by pixels labeled to contain only noise, and positive contributions are contributions by pixels labeled to contain an energy deposition. True and false describe the correctness of the generated label. Because of the noise contribution's statistical character, there is no threshold that labels the contributions with an accuracy of one hundred percent. In general, the selection criterion should be chosen at the energy value on which the contribution of the noise is of the same magnitude as the average contribution of a real energy deposition to the spectrum in one pixel (Figure B.2).

A lower secondary threshold than the best secondary threshold decreases the ratio of single pattern events. As a consequence, the false negative contribution decreases. For a single pattern event, it is less likely to miss an energy deposition in an adjacent pixel. Otherwise, the event would be a double pattern event. As a consequence, the energy resolution of the single pattern events increases. At the same time, the false positive contribution

to events where multiple pixels contribute increases. Therefore, the energy resolution of the multiple events decreases.

Other impacts on the pattern size and, therefore, on the ratio between patterns containing only one pixel and larger patterns are shown in Table B.1.

Table B.1: Dependencies of the single pattern events are obtained from an event pattern analysis using a noise-dependent primary and a secondary threshold. A detailed explanation can be found in the text.

property	behavior	single pattern events
secondary threshold	↗	↗
particle energy	↗	↘
back contact voltage	↗	↗
readout noise	↗	↗
pixel size	↗	↗

A higher energy of the primary particle leads to a stronger repulsion of the charge cloud due to the larger amount of generated charge carriers. For charged primary particles, there is also an additional effect. Charged particles randomly produce three-dimensional tracks in the detector volume. The length of those tracks becomes larger for higher primary energies. Both effects increase the event pattern size and, therefore, reduce the ratio of single pattern events.

A higher back contact voltage leads to a higher electric field in the silicon bulk and, therefore, to a shorter drift time of the charge cloud. The charge cloud expands less in this shorter time. As a consequence, the ratio of single pattern events increases with increasing back contact voltage.

A higher noise contribution of the readout chain leads to a higher secondary threshold. Therefore, more pixels are considered to contain just noise and no energy deposition from a primary particle. The number of pixels containing an energy deposition but are classified as pixels that only contain noise increases. Thus, the number of pixels of an event pattern decreases.

A larger pixel size yields for the same size and shape of the charge cloud to less pixels contributing to the pattern. Therefore, the patterns' size becomes smaller on average, and the ratio of single pattern events increases.

Depending on the application, a high ratio of single pattern events is desired or not desired.

A large ratio of single pattern events is desired for spectroscopy because every

additional pixel contributing to the event provides its individual noise. The more pixels contribute to the event pattern, the more noise terms contribute to the measured total energy deposition.

Large event patterns are advantageous for applications where the primary focus is the PoE's spatial resolution. In general, algorithms that determine the PoE on subpixel level need a spread of the energy deposition over several pixels (Figure B.10).¹ This statement applies to classical methods such as the center of gravity method (Section B.1.4) as well as for methods based on neural networks (Chapter 7).

B.1.2 Pattern Pile-up Events

The event pattern analysis works well for low particle rates. If the rate increases, more and more pattern pile-up events are detected. A pattern pile-up event is an event pattern where the energy deposition of two or more different primary particles is combined to one cluster and recognized as one event pattern by the event pattern analysis algorithm.

The classical methods for the reconstruction of the PoE for the primary particles, which are described in Section B.1.4, require event patterns to which only one primary particle contributes.

Basically, two types of pile-up events can occur, but these two types are only phenomenological and not distinguishable in measurements.

The first type is two or more physical events whose pixel boundaries touch each other, but at least theoretically, one physical event can be assigned to each pixel. Here, a physical event is the energy deposition of one primary particle. In principle, these types of events are separable. Every pixel contains at least the charge carriers from only one primary particle.

The second type is events where the charge clouds overlap, i.e., there is at least one pixel containing charge carriers from two different primary particles.

For even higher rates and intensity imaging, illuminated and adjacent pixels are counted as one event, making a proper event analysis impossible.

The ratio between events where only one primary particle contributes to the energy deposition and pattern pile-up events depends on the experimental conditions. Table B.2 show these dependencies. Because the event size

¹Not a precise PoE can be reconstructed for single pattern events, but the area centered at the pixel center in which the PoE has to be can be defined. The size of the area containing the PoE depends on the charge cloud size. A larger charge cloud leads to a smaller area.

influences the probability of a pattern pile-up event, most relationships are similar to the considerations for the single pattern events (Table B.1).

An increase of the readout rate acts similarly as a decrease of the particle rate and leads to a lower probability that two particles enter the detector in the same frame.

An increase in the pixel size has the same effect as smaller event pattern sizes. However, the probability for pixels of two events that are connected increases since the spatial dimensions of the charge clouds are the same. Therefore, the pattern pile-up probability increases.

The situation is different if one assumes not a constant overall particle rate but a constant particle rate per pixel. The larger pixel sizes lead to a smaller overall particle rate in this context. The dimensions of individual charge clouds stay the same since they are independent of the pixel size. Consequently, the probability of a pattern pile-up event decreases. [11]

Table B.2: Dependencies of the pattern pile-up probability. A detailed explanation can be found in the text. Table adapted from [69].

property	behavior	pattern pile-up
secondary threshold	↗	↘
readout rate	↗	↘
particle rate	↗	↗
particle energy	↗	↗
back contact voltage	↗	↘
readout noise	↗	↘
pixel size (const. particle rate)	↗	↗
pixel size (const. particle rate per pixel)	↗	↘

Figure B.3 shows the probability for a pattern pile-up event for different primary energies as a function of the primary electron rate. For small primary electron rates, the probability that an event pattern only contains the energy deposition of one primary electron is the highest. With increasing primary electron rate, the probability for single pattern events decreases, whereas the probability for multiple event patterns increases since it is more likely that two or more electron tracks overlap. The probability for event patterns where less than six primary electrons contribute is for low primary electron rates nearly one and decreases with higher primary electron rates since the number of event patterns with six or more primary electrons increases. The tracks of primary electrons with higher energies are larger than the tracks of primary electrons with low primary energies. Therefore, the probability for a single

pattern event as a function of the primary rate decreases faster with higher primary energy. For even higher rates, the contributions of single primary electrons are no longer separable, and intensity images are created. Since the classical approaches presented in Appendix B.1.4 require single pattern events and cannot handle pattern pile-up events, the multiplicity for the classical methods is related to the probability for single pattern events shown in Figure B.3. The multiplicity denotes the fraction between the number of reconstructed PoEs and actual PoEs. An energy filter applied to the event pattern can be used to separate single pattern events from the others.

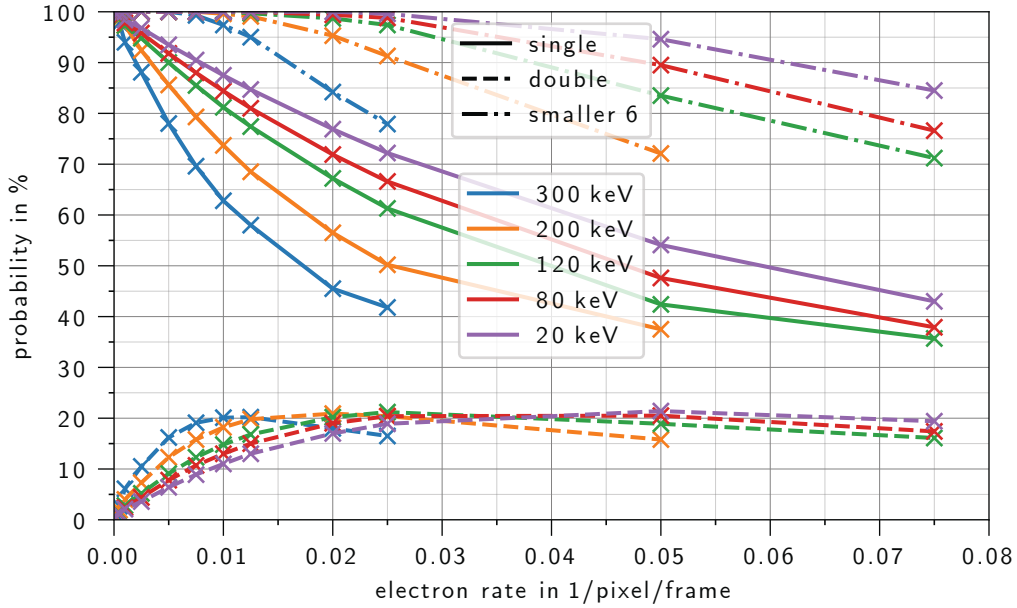


Figure B.3: Simulated pattern pile-up event probability for electrons for different primary energies as a function of the primary electron rate. The pixel size is $48 \times 48 \mu\text{m}^2$. The line style indicates the number of primary electrons contributing to the event pattern. "Single" denotes the probability for a single pattern event, and "double" denotes the probability for a double pattern event. "Smaller 6" denotes that less than six primary electrons contribute to the event pattern. Figure adapted from [69].

A quantitative view of probability for pattern pile-up events of photons can be found in [213] and [214], and for electrons in [69].

B.1.3 Energy Filter

Instead of using the primary threshold as a selection criterion whether a cluster should be counted as an event or not, another mechanism can be used. A detected cluster is counted as an event pattern if the total amount of deposited energy in the cluster exceeds a certain threshold or, alternatively, is in a certain range. The selection criterion works as an energy cut or as an energy window. In contrast to the event pattern detection with two thresholds, this energy filter depends on the quality of the gain correction.

This event filter is suitable, especially if the to-be-detected particles' energy is well known and monochromatic. A correctly chosen energy window reduces pattern pile-up events or only partial energy distributions of a primary particle in the detector.

B.1.4 Classical Methods for the Reconstruction of the Point of Entry

Several methods exist to reconstruct the PoE for each event obtained from the event pattern analysis. However, all these methods do not consider the statistical energy deposition behavior of charged particles in the detector volume and have in common that they cannot handle pile-up events. This has the consequence that the conventional methods are only applicable for low particle rates. As a criterion for a single event or pile-up event, the simple summation of the energy depositions can be used.

The reconstruction of the PoE is not necessarily robust to a gain correction or a common-mode correction. The following introduced methods to reconstruct the PoE for each event pattern assume that exactly one particle contributes to each event pattern.

- **Maximum:** The simplest method is to take the pixel with the maximal signal as PoE. This method is applicable for photons and electrons where the incident electrons do not create extended tracks and is used for so-called single pattern events where the number of pixels of the event is one. This method cannot provide any subpixel information.
- **Center of gravity:** The standard method to determine the PoE is to calculate the center of gravity \mathbf{x}_{CoG} of the pixels that contain a signal.

$$\mathbf{x}_{\text{CoG}} = \begin{pmatrix} x_{\text{CoG}} \\ y_{\text{CoG}} \end{pmatrix} = \frac{1}{\sum_{\text{pixels} \in \text{event}} S_{x,y}} \sum_{\text{pixels} \in \text{event}} S_{x,y} \begin{pmatrix} x \\ y \end{pmatrix} \quad (\text{B.6})$$

$S_{x,y}$ is the signal of the pixel with the center $(x, y)^T$. It is important to

note that the straightforward calculation leads to a systematical error that shifts the PoE towards the physical pixels' centers in comparison to the true PoE [12]. This systematical error can be partially corrected either with the δ correction method or the η correction method [220]. A detailed description and explanation can be found in Appendix B.2.

The idea behind the method is a symmetrically shaped charge cloud centered at the PoE. This means it is applicable to photons and electrons with low energies and can provide reconstructions of subpixel information.

- **Furthest away method:** The furthest away method is suitable for electrons with higher primary energies. They deposit most of their energy at the end of the track (Section 2.2). Therefore, the furthest away method assumes that the PoE is in the pixel with the largest distance to the pixel with the highest signal. This method cannot provide any subpixel information and only works well if the event shows a strong directional track. [11]

Figure B.4 shows the positions of the reconstructed PoE obtained by the different methods, for example, event pattern for a primary electron with an energy of 300 keV. The furthest away method achieves the best result. Best result in this context means the Euclidean distance between the true PoE and the reconstructed PoE is minimal. For this example, the worst of the described methods is using the pixel with the maximum energy deposition as PoE. The result obtained by the center of gravity method is slightly better. Figure 10.13 confirms this behaviour for large statistics.

B.2 Corrections to the Center of Gravity Method

In this Section, the systematical error made by the center of gravity method is mathematically described and analyzed.

B.2.1 Theoretical Derivation of the Corrections to the Center of Gravity Method

A detector system without noise and a vanishing electrostatic field component parallel to the detector surface is assumed for the following considerations. This simplification is valid because additional noise would only result in an additional statistical rather than a systematical error. An electrostatic field

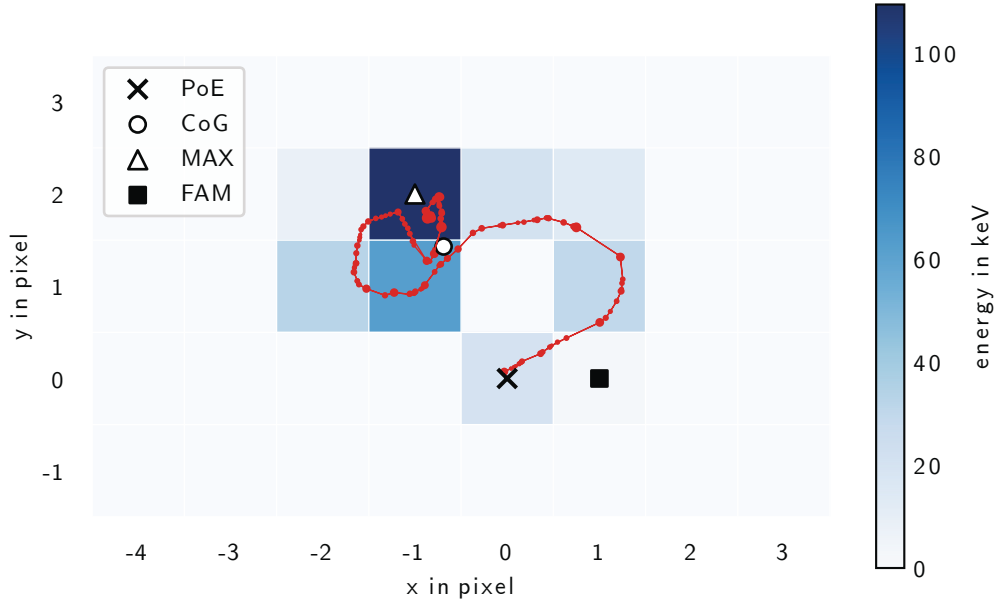


Figure B.4: Reconstructed event patterns by conventional methods. The color code indicates the energy deposition into the individual pixels. The primary electrons enter the detector volume at (0,0) with a primary energy of 300 keV and produce a three-dimensional track in the silicon shown in red. The size of the red dots indicates the amount of energy deposition along the track. This track is only shown for clarity but is not obtained by the measurement. The different marker shows the reconstructed PoE with the different conventional methods. PoE denotes the actual PoE, CoG the PoE obtained by the center of gravity method, MAX the PoE obtained by the maximum method, and FAM the PoE obtained by the furthest away method.

component perpendicular to the detector surface that is not zero is not considered in the following Section and would lead to a systematical shift of the PoE due to the lateral drift of the charge cloud.

The total energy deposition or, more specific the generated amount of charges of one primary particle can be calculated by the following integral:

$$Q = \int \int \rho(x, y) dy dx \quad (\text{B.7})$$

Here, $\rho(x, y)$ is the projection of the charge cloud onto a plane parallel to the pixel structure of the detector.

The center of gravity method can only be used if the projection of the

charge cloud onto the plane parallel to the pixel structure is symmetric. This symmetry is ensured by the diffusion and the electrostatic repulsion since they are radially symmetric (Chapter 4.2.4).

The x-component of the theoretical PoE is the center of gravity and can be calculated by the expectation value of $\rho(x, y)$:

$$x_{\text{PoE}} = \frac{1}{Q} \int \int x \rho(x, y) dy dx = \frac{1}{Q} \int x \left[\int \rho(x, y) dy \right] dx = \frac{1}{Q} \int x \tilde{\rho}(x) dx \quad (\text{B.8})$$

Because the x-coordinate is independent of the coordinate y, the integral over y can be performed first. The result is the projection of the charge density on the x-axis, which is labeled as $\tilde{\rho}(x)$. The x-coordinate of the PoE is independent of the expansion of the charge cloud in the y-direction.

The calculated x-component of the PoE by the center of gravity method can be derived from eq. B.6:

$$\begin{aligned} x_{\text{CoG}} &= \frac{1}{\sum_{\text{pixels} \in \text{event}} S_{x,y}} \sum_{\text{pixels} \in \text{event}} S_{x,y} x_{\text{pixel}} \\ &= \frac{1}{Q} \sum_{\text{pixel}} x_{\text{pixel}} \int_{\text{width}} \int_{\text{length}} \rho(x, y) dy dx \\ &= \frac{1}{Q} \sum_{\text{columns}} \sum_{\text{rows}} x_{\text{pixel}} \int_{\text{width}} \int_{\text{length}} \rho(x, y) dy dx \\ &= \frac{1}{Q} \sum_{\text{columns}} x_{\text{pixel}} \int_{\text{width}} \left(\sum_{\text{row}} \int_{\text{length}} \rho(x, y) dy \right) dx \quad (\text{B.9}) \\ &= \frac{1}{Q} \sum_{\text{columns}} x_{\text{pixel}} \int_{\text{width}} \left(\int \rho(x, y) dy \right) dx \\ &= \frac{1}{Q} \sum_{\text{columns}} x_{\text{pixel}} \int_{\text{width}} \tilde{\rho}(x) dx \\ &= \sum_{\text{columns}} \frac{\int_{\text{width}} \tilde{\rho}(x) dx}{Q} x_{\text{pixel}} \end{aligned}$$

Here, x_{pixel} is the x-coordinate of the pixel center and $S_{x,y}$ is the signal in the pixel with the center $(x, y)^T$. The integral over the width has the limits

$x_{\text{pixel}} - \Delta x/2$ and $x_{\text{pixel}} + \Delta x/2$. Δx is the pixel's width. The integral over the length has the limits $y_{\text{pixel}} - \Delta y/2$ and $y_{\text{pixel}} + \Delta y/2$. Δy is the length of the pixel. Rows are in the x-direction, and columns are in the y-direction.

In the second line of eq. B.9, the signal in the pixel is substituted by eq. 4.20 on page 55. Here, $\rho(x, y)$ is the charge density created by the primary particle, which contributes to the event.

The summation over all pixels can be separated in the summation over the detector's rows and columns. Because the x-component of the pixel positions x_{pixel} is the same for all summands of the summation over the rows, the order of the summation over the rows and the integral over the width can be exchanged.

The summation over all rows of the integral over the length of the individual pixels can be simplified by the integral over the total detector length. The calculated x-component is the weighted summation over the x-positions of the pixels. The weighting factors are fractions of deposited energy in the corresponding detector columns.

The deviation $\delta(x)$ between the theoretical and the calculated PoE is the difference between x_{PoE} and x_{CoG} :

$$\begin{aligned}
 \delta(x) &= x_{\text{PoE}} - x_{\text{CoG}} \\
 &= \frac{1}{Q} \int x \tilde{\rho}(x) dx - \frac{1}{Q} \sum_{\text{column}} x_{\text{pixel}} \int_{\text{width}} \tilde{\rho}(x) dx \\
 &= \frac{1}{Q} \sum_{\text{column}} \left(\int_{\text{width}} x \tilde{\rho}(x) dx - x_{\text{pixel}} \int_{\text{width}} \tilde{\rho}(x) dx \right) \quad (\text{B.10}) \\
 &= \sum_{\text{column}} \frac{\int_{\text{width}} \tilde{\rho}(x) dx}{Q} \left(\frac{\int_{\text{width}} x \tilde{\rho}(x) dx}{\int_{\text{width}} \tilde{\rho}(x) dx} - x_{\text{pixel}} \right)
 \end{aligned}$$

In the second line, the integral over the detector's total width is split in the sum over the pixel-wise integrals. In eq. B.11, the projected charge density $\tilde{\rho}(x)$ is substituted with the charge density $\rho(x, y)$ to get a contribution of the

individual pixels to the deviation $\delta(x)$.

$$\begin{aligned} \delta(x) &= \sum_{\text{pixel}} \frac{\int_{\text{width}} \int_{\text{length}} \rho(x, y) dy dx}{Q} \left(\frac{\int_{\text{width}} \int_{\text{length}} x \rho(x, y) dy dx}{\int_{\text{width}} \int_{\text{length}} \rho(x, y) dy dx} - x_{\text{pixel}} \right) \\ &= \sum_{\text{pixel}} \frac{S_{x,y}}{\sum_{\text{pixel}} S_{x,y}} \left(\frac{\int_{\text{width}} \int_{\text{length}} x \rho(x, y) dy dx}{S_{x,y}} - x_{\text{pixel}} \right) \end{aligned} \quad (\text{B.11})$$

The deviation $\delta(x)$ is the weighted summation over the distance of the centers of gravity of the individual pixels and the positions of the pixels' centers x_{pixel} . The weights of the sum are the relative amount of charge in the individual pixels.

Figure B.5 shows the projection of the charge density on the x-axis and the corresponding representation in the pixel structure. The calculated PoE is shifted by $\delta(x)$ towards the center of the pixel structure. The integrated charge density in pixel 0 and pixel 3 can be neglected. $\delta(x)$ is the difference between the actual PoE x_{PoE} and the calculated PoE x_{CoG} . The dashed lines in the upper plot are the theoretical centers of gravity of the individual pixels. The dashed lines in the lower plot are the pixel-wise centers of gravity in the pixelated representation. Since the charge density is constant over one pixel in the pixelated representation, the center of gravity is at the same point as the pixel center. $\delta(x)$ is the weighted sum over the differences δ_i of the individual pixels.

Since in the theoretical and the pixelated representation, the charge density is constant for pixels 0 and 3, the center of gravity and the center of the pixel is the same. Therefore, the contribution of pixels 0 and 3 to $\delta(x)$ is zero.

The deviation of the y-component can be calculated in the same way. The calculated PoE has not to be corrected if either the center of gravity is the center of the pixel for all pixels or the contributions of the individual pixels cancel each other.

The first assumption is true if the charge density is constant over the individual pixels. In general, this is valid if either the charge density is constant or the pixel size is very small compared to the spatial change of the charge density.¹

¹These conditions also hold if the charge density is symmetric around the center of each pixel.

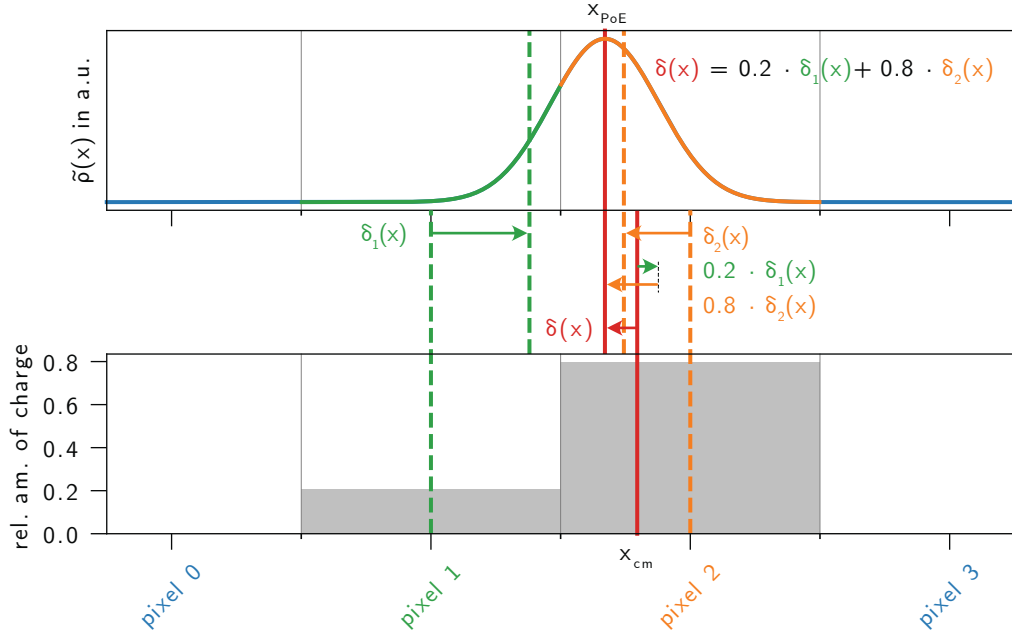


Figure B.5: Schematic of a correction to the center of gravity method. **Upper plot:** Gaussian distribution of the charge density as a function of the x-position in arbitrary units (a.u.). The solid blue line depicts the charge density in pixel 0 and pixel 3, the solid green line depicts the charge density in pixel 1, and the solid orange line the charge density in pixel 2. **Lower plot:** Relative amount of charge in the pixel structure. The relative amount of charge is the integral over the pixel boundaries divided by the total amount of charge.

The dashed colored lines depict the pixel-wise center of gravity for pixel 1 and pixel 2 in both shown representations. The arrows depict the derivations. The red lines show the center of gravity in both representations. The red line in the upper plot represents the theoretical PoE, and in the lower plot, the calculated PoE. The difference $\delta(x)$ (red arrow) is the deviation made by the reconstruction method and can be calculated by the pixel-wise derivations weighted with the relative amount of deposited charge.

The signals of the individual pixels cancel each other if the charge density is symmetric in the representation of the pixel structure. It is not sufficient that the charge density is symmetric. For example, this symmetry is accomplished if the theoretical PoE is at the border or the center of a pixel. In this case, the symmetry of the charge density is not broken by the representation of the pixel structure.

The systematic deviations $\delta(x)$ and $\delta(y)$ only depend on the projection on the x- and y-axis, respectively, and can, therefore, be treated independently.

The theoretical description of the deviation $\delta(x)$ is also valid if the primary particle only deposits its energy in one pixel (more precisely in one column and row, respectively). However, in this case, the deviation $\delta(x)$ is not bijective and, therefore, a mathematically unique reconstruction of the theoretical PoE is not possible.

Here, it must be noted that $\delta(x)$ is given as a function in the detector space.² In order to apply the correction to the calculated centers of gravity, a coordinate transformation into the analysis space defined by the calculated center of gravity must be performed. Analytically, this coordinate transformation depends on $\delta(x)$ and is given as follows:

$$x \rightarrow x - \delta(x) \tag{B.12}$$

B.2.2 Determination of the Corrections to the Center of Gravity Method by Simulation

Figure B.6 and Figure B.7 show the probability distribution and the cumulative probability over a pixel caused by the center of gravity method. The initial situation is a homogeneous distribution of PoEs over the pixel. Since the center of gravity method systematically shifts the calculated PoEs to the pixel center, the probability in the center is the highest. In a subpixel representation, this leads to so-called checkerboard artifacts. Checkerboard artifacts are repetitive artifacts that repeat themselves with a spatial frequency. Subpixels located in the center of physical pixels show a higher probability of being hit. A flat-field illumination leads to an inhomogeneous distribution in the reconstructed hit-map.

To correct the calculated center of gravity, $\delta(x)$ has to be added to the reconstructed center of gravity:

$$x_{\text{PoE}} = x_{\text{CoG}} + \delta(x) \tag{B.13}$$

The correction $\delta(x)$ depends on the charge cloud's shape and size and the pixel structure of the detector system.

²The detector space represents the direct physically accessible coordinates. Distances can be measured with a simple tape measure.

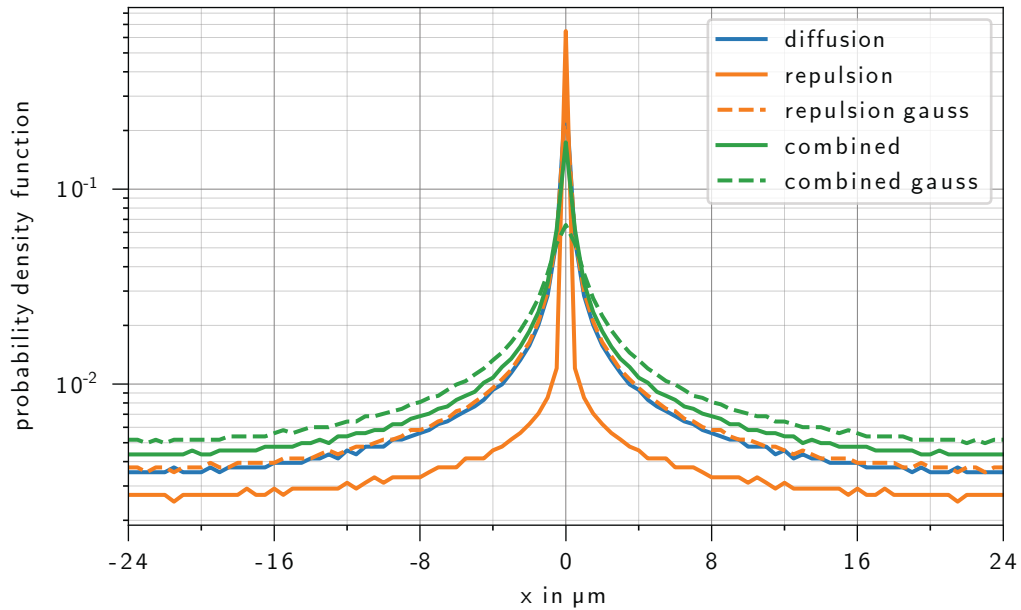


Figure B.6: Probability density function of the center of gravity for a homogeneous illumination. The pixel size is $48 \times 48 \mu\text{m}^2$, and the pixel center is at zero. A K_α photon emitted by copper generates the charge cloud. The dimensions and shape of the assumed charge clouds are shown in Figure 4.4 on page 55. The different colors refer to the different charge cloud models.

The following steps describe the procedure to create the correction function $\delta(x) = \delta(y)$ to the center of gravity method. The method is adapted from Belau [220] and Ryll [11] and supplemented by the different models of the charge cloud. The used data are generated by a numerical method described in Section 4.7.

- **Calculation of the center of gravity in the pixel domain:** In this step, the calculated PoE is obtained via the center of gravity method for each seven times seven pattern.¹ Pixel domain means that all centers of gravity are calculated relative to the central pixel of the seven times seven pattern.
- **Calculation of the shift between the actual PoE and the center**

¹The Gaussian distribution is never zero but for the range of the used parameters outside the seven times seven pattern smaller than one percent of the total charge and, therefore, neglectable.

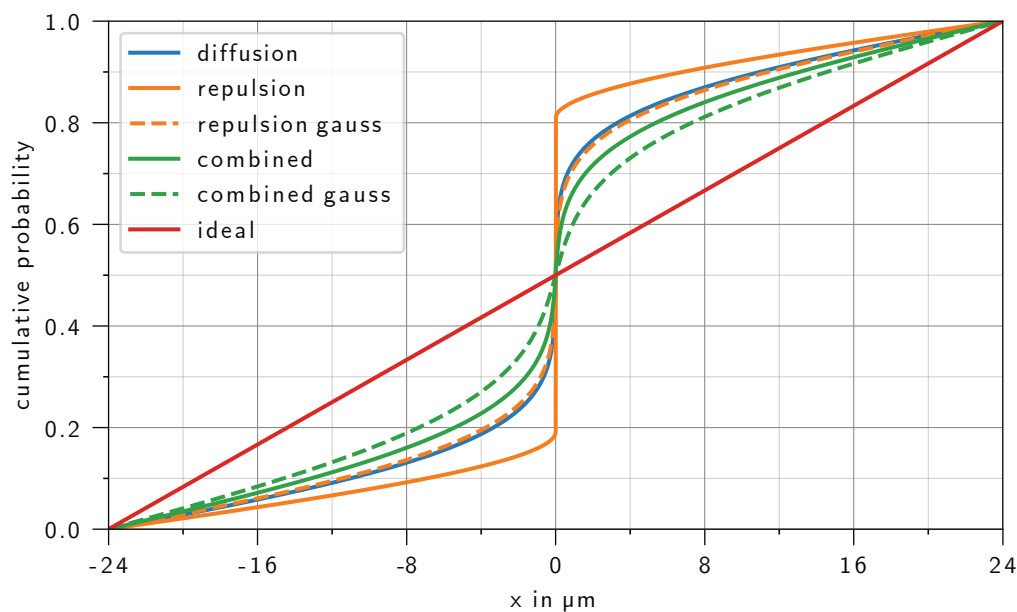


Figure B.7: Cumulative probability of the center of gravity for a homogeneous illumination. The pixel size is $48 \times 48 \mu\text{m}^2$, and the pixel center is at zero. A K_α photon emitted by copper generates the charge cloud. The dimensions and shape of the assumed charge clouds are shown in Figure 4.4 on page 55. The different colors refer to the different charge cloud models. For a homogeneous illumination, the cumulative probability should be a straight line.

of gravity in the detector space:² The deviation between the actual PoE and the calculated center of gravity is calculated for every event pattern and, therefore, for all different PoE positions within the central pixel of the pattern. The result is a map of the deviations or rather the shifts between the true PoE and the calculated center of gravity in the detector space. Figure B.8 shows the shift. As expected from the theoretical derivation, the shift at the pixel boundaries and in the center is zero. The calculated centers of gravity are shifted towards the pixel center.

- **Transfer the shift to the analysis space:**¹ The shifts obtained in the previous step contain all necessary information to perform the correction step. However, the entries of the shift map are a function of the coordi-

²The detector space represents the direct physically accessible coordinates. Distances can be measured with a simple tape measure.

¹The weighted centroids define the coordinate system of the analysis space.

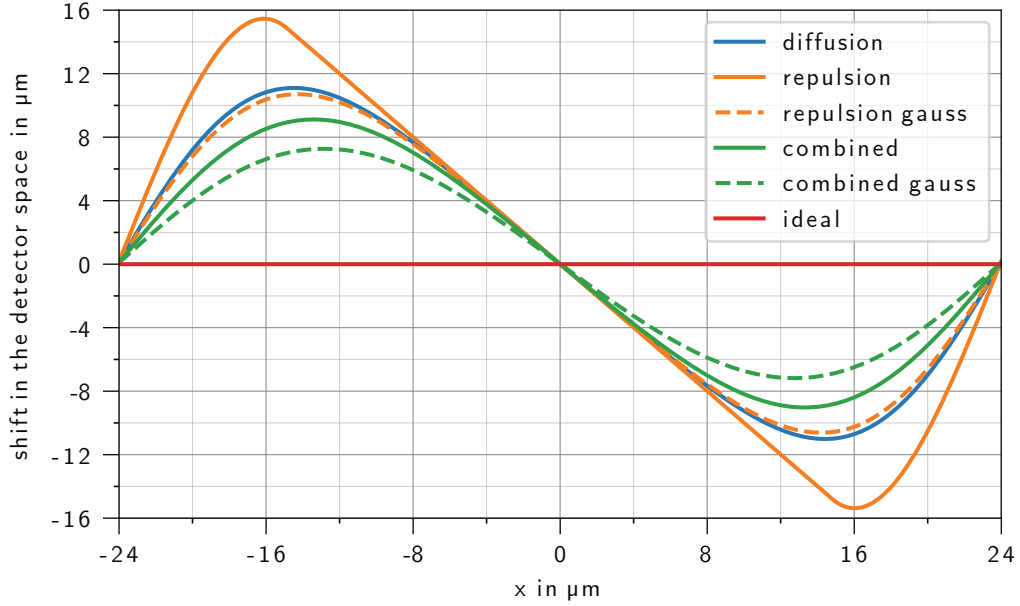


Figure B.8: Shift between the actual PoE and the center of gravity in the detector space. The different colors refer to the different charge cloud models. The shift between the actual PoE and the calculated PoE is zero for an ideal reconstruction.

mates in the detector space. This map must be transferred in the analysis space to find the corresponding correction to a reconstructed PoE by the center of gravity method. The analysis space is the coordinate system that results from the calculated PoE. Since the spacing in the coordinate system of the analysis space is not equidistant but depends on the coordinates itself, the coordinate transformation between the two reference systems is not linear. With the coordinate transform described by eq. B.12 follows:

$$\delta(x) \rightarrow \delta(x - \delta(x)) \quad (\text{B.14})$$

The correction map in the analysis space is obtained via interpolation of the correction map in the detector space. Figure B.9 shows the correction $\delta(x)$ as a function of the coordinates in the analysis space. Again, the shift at the pixel boundaries and in the center is zero. The bigger the charge clouds or the smaller the pixel sizes, the smaller is the correction $\delta(x)$.

If the actual PoE is located near the center of the pixel and the charge cloud is small compared to the pixel size, the charges are not spread over adjacent

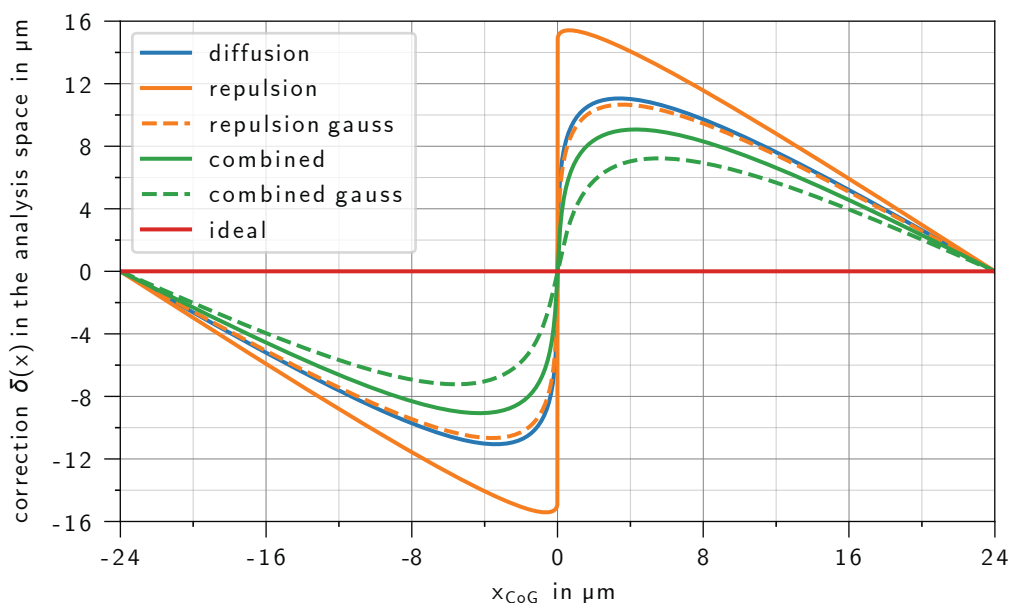


Figure B.9: Correction for the center of gravity in the analysis space to obtain the actual PoE. The different colors refer to the different charge cloud models. For an ideal reconstruction, the correction is zero.

pixels. As a consequence, a single pattern event occurs. In such a case, it is not possible to precisely reconstruct the actual PoE within a geometrical pixel due to the lack of information. Figure B.10 shows this exemplary for different charge clouds the area around the pixel center for which a single pattern event is created. This behavior is expressed in a vertical line in $\delta(x)$. An exact assignment of the weighted average and the actual PoE is not possible.

The same applies for double pattern events. Here, the actual position in one dimension can be reconstructed but not in the other dimension. For a horizontal double event pattern, the x-coordinate can be reconstructed, and for a vertical double event pattern, the y-coordinate. In general, to reconstruct the x- and the y-coordinate, the expansion of the event pattern in the x-, y-dimension has to be larger than one pixel, respectively.

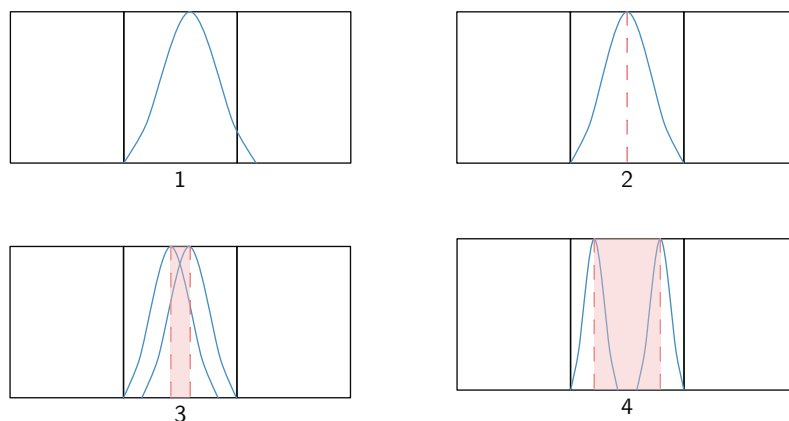
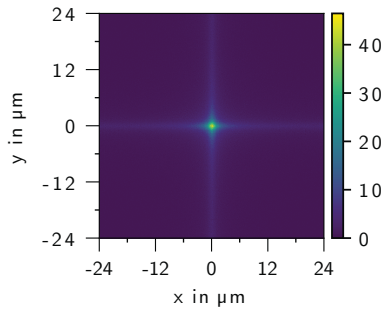
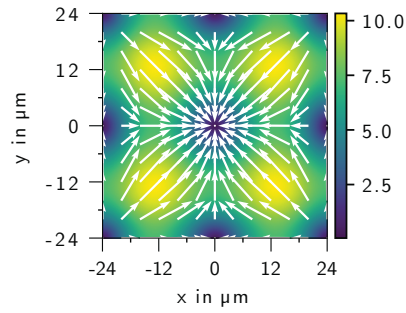


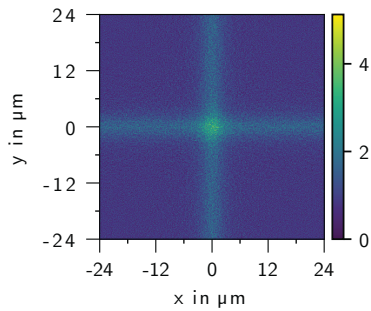
Figure B.10: Occurrence of single pattern events. A cut perpendicular to the detector surface is shown. For each example, the central pixel and its neighbors are shown. The black lines represent the pixel borders, and the blue curves represent the charge distributions for four different charge cloud sizes. The PoE is at the maximum of the charge distribution. **(1)** The charge cloud is larger than the pixel structure. As a consequence, no single pattern events occur, for all PoEs extend at least over two pixels. **(2)** The charge cloud has the same size as the pixel structure. For a PoE in the center of the pixel, a single pattern event occurs. This position is marked with the red dashed line. **(3)** The size of the charge cloud is slightly smaller than the pixel structure. Therefore, all events with a PoE within the red area produce a single pattern event. **(4)** For decreasing charge clouds compared to the pixel structure, the area in which a single pattern event is created increases.



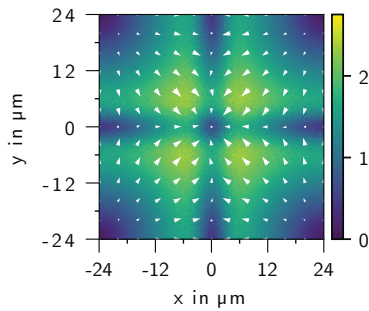
(a) Hit-map for no correction



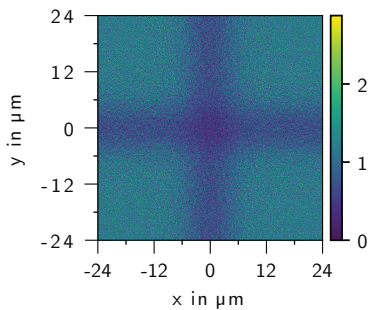
(b) Spatial accuracy map for no correction



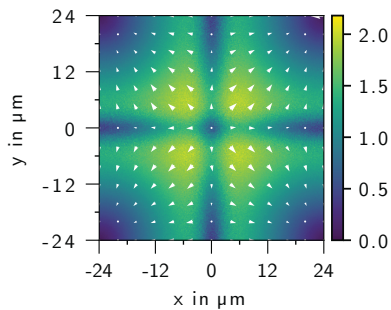
(c) Hit-map for too small correction



(d) Spatial accuracy map for too small correction



(e) Hit-map for too large correction



(f) Spatial accuracy map for too large correction

Figure B.11: Normalized hit-map and spatial accuracy map for the pure center of gravity, a too small correction, and a too large correction. The physical pixel size is $48 \times 48 \mu\text{m}^2$ with (0,0) in the center of the pixel, and the illumination is homogeneous with around two million simulated primary particles. The spacial accuracy map contains the average distance between the true PoE and the reconstructed PoE in μm , and the arrows describe the systematic displacement. The charge cloud is Gaussian shaped with an FWHM of $22 \mu\text{m}$. The pixel-wise noise is 0.01 % of the total signal. For the too small correction, an FWHM of $24 \mu\text{m}$, and for the too big correction, an FWHM of $20 \mu\text{m}$ of the charge cloud is used.

The accuracy of the correction $\delta(x)$ can be investigated by the generation of a two-dimensional histogram (Figure B.11). The two-dimensional histogram is created over all relative PoE positions within the individual pixels of the detector.¹ Ideally, this two-dimensional histogram should be homogeneous over the pixel area. It was seen that due to the large slope of $\delta(x)$ (Figure B.9) at the center of the pixels, inhomogeneities at the center of the two-dimensional histogram are most indicative of deviations between the true shape of the charge cloud and the assumed model of the charge cloud in Section 4.7.

Building the two-dimensional histogram over the results of the uncorrected center of gravity method leads to a density distribution with the maximum at the center of the pixel area (Figure B.11a and B.11b). This is caused by the fact that the calculated PoEs of the center of gravity method are shifted towards the center. This applies similarly to the x- and the y-component and results in a *plus*-shaped distribution.

A less pronounced *plus* shape suggests a too small correction $\delta(x)$ in the center of the pixel structure (Figure B.11c and B.11d). The too small correction in the center corresponds to a too flat slope in the center of the pixel structure. A too flat slope in the center is due to the fact that the charge cloud in the model was assumed to be too large. A too large charge cloud follows, for example, from a drift time that was assumed to be too long.

An inverse *plus* shape suggests a too large correction in the center of the pixel structure (Figure B.11e and B.11f). A too large correction in the center corresponds to a too large slope in the center of the pixel structure. A too large slope in the center is due to the fact that the charge cloud in the model was assumed to be too small. A too small charge cloud follows, for example, from a drift time that was assumed to be too short.

B.2.3 Determination of the Corrections to the Center of Gravity Method by Measurements

Between synthetically generated data and measurements, small deviations can be found. Those differences occur due to simplifications of the model. For example, the pixel structure of the detector leads to a small deviation between the simulated and the actual electric field. The deviation leads to a second, typically small, correction η :

¹A flat-field measurement is ideal. However, other measurements work well, too, as long as enough area of the detector is illuminated, that on average, the theoretically calculated two-dimensional histogram of the PoE is homogeneous.

$$\tilde{x}_{\text{PoE}} = x_{\text{PoE}} + \eta(x_{\text{PoE}}) = x_{\text{CoG}} + \delta(x) + \eta(x_{\text{CoG}} + \delta(x)) \quad (\text{B.15})$$

While $\delta(x) = \delta(y)$ is similar for x and y , this is not necessarily true for $\eta(x) \neq \eta(y)$. These differences can be caused by the symmetry breaking of the pixel structure.¹ The following steps can be performed to resolve differences represented by $\eta(x)$ between the synthetically generated results and the measurements. The construction of $\eta(y)$ works similarly. It is based on a method for silicon strip detectors [220, 221] following the steps:

- **Applying the $\delta(x)$ correction function (eq. B.13):** The first step is to perform a flat-field measurement with the desired settings. The measured data are analyzed by using the center of gravity method and applying the correction function $\delta(x)$ (eq. B.13). The obtained values are a list of all x_{PoE} .
- **Creating a histogram over a pixel:** The PoE x_{PoE} can be described as a sum of the position of the physical pixel x_{pixel} that contains the PoE and relative position ξ within the pixel.

$$\xi = x_{\text{PoE}} - x_{\text{pixel}} \quad (\text{B.16})$$

A histogram calculated over ξ can be interpreted as a stacking of all pixels of the detector.

- **Calculating the deviation from a homogeneous distribution:** The histogram over all ξ should be homogeneous for a homogeneous illumination if the synthetically generated data set describes the detector perfectly. The correction η , which has to be applied to the individual PoE to remove the inhomogeneities, is the difference between the actual cumulative distribution function $\text{cdf}(\xi)$ and the ideal cumulative distribution function $\text{cdf}(\text{const.})$:

$$\eta(\xi) = \text{cdf}(\xi) - \text{cdf}(\text{const.}) \quad (\text{B.17})$$

To correct inhomogeneities along the pixel structure, the second correction η can be individually calculated for each pixel of the detector. This pixel-wise second correction requires larger statistics. Moreover, for the pixel-wise correction, $\eta(\xi)$ corrects the systematic error introduced e.g. by a false gain calculation. The second correction η can also be used as an indirect validation

¹The storage and transfer of the signal charges towards the readout anode require a different electric field than the confinement of the signal charges perpendicular to the transfer direction.

of the simulated charge cloud size.

B.2.4 Influence of Noise to the Corrections to the Center of Gravity Method

The occurrence of noise in the individual pixels limits the spatial resolution. The non-linearity of the correction $\delta(x)$ leads to a location-dependent spatial resolution within the pixel structure. Figure B.12 shows the reconstructed PoEs for randomly chosen positions within the pixel structure. For each chosen position, around 4000 primary particles are simulated. The only difference between those event patterns is the contribution of the noise to the different pixels. Consequently, the histogram of the reconstructed PoE of the event pattern leads to a broad distribution and not a sharp spot for each PoE position. The shape of those distributions depends on the relative position of the PoE within the pixel structure.

Positions near the edges of the pixel structure are less sensitive to noise. Therefore, the spot created by many individual particles at the same PoE with different noise contributions is sharp (1 in Figure B.12). This behavior can be explained by the small slope of $\delta(x)$ (Figure B.9) near the pixel borders. A slight change of the center of gravity caused by noise impacts the reconstructed position only marginally.

Distributions of PoEs near the central axes are broad in the dimension in which they are near the central axis and sharp in the other dimension (2 in Figure B.12). This behavior can be explained by the large slope of $\delta(x)$ (Figure B.9) near the pixel center. A small change in the center of gravity leads to a large change in the position.

Distributions of PoEs in intermediate positions between the pixel border and the central axis cause a medium spread (3 in Figure B.12). Distributions of PoEs near both central axes are broad in both dimensions (4 in Figure B.12). In general, the distributions are broad in the dimension in which their position is near the central axis. The spreading in the x- and y-dimension can be treated separately.

Appendix B.3 contains a detailed theoretical derivation of the spatial resolution and its dependency on correlated and uncorrelated noise.

Figure B.13 shows the distribution of the reconstructed positions for five PoE's positions. The distribution of the center of gravity for many events at one PoE can be approximated symmetrically. Applying the non-linear correction $\delta(x)$ skews the symmetric distribution. Therefore, the recon-

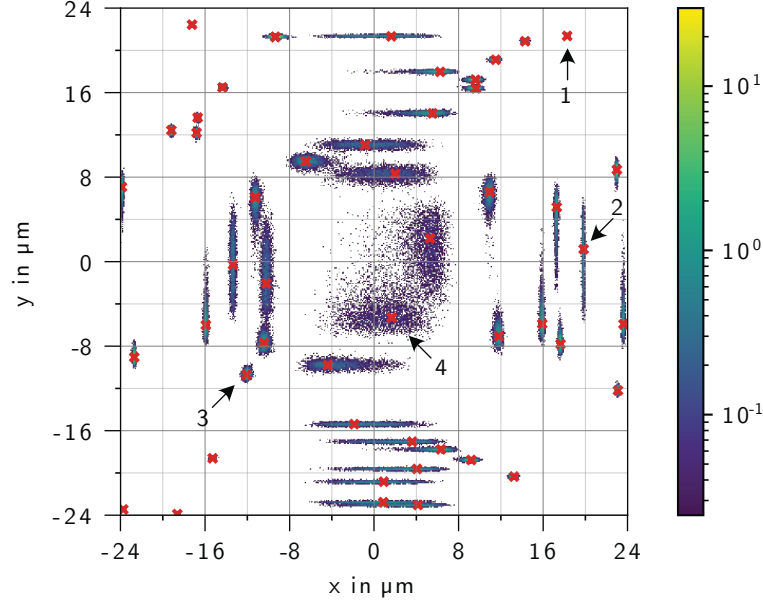


Figure B.12: Reconstruction with the center of gravity with correction of 50 randomly chosen PoEs within the pixel structure. The physical pixel size is $48 \times 48 \mu\text{m}^2$ with $(0,0)$ in the center of the pixel, and the energy of the primary photons is 8048 eV. A non-correlated Gaussian distributed pixel-wise noise with a mean of 11.3 eV is assumed. For each PoE's position, around 4000 primary particles are simulated. The red crosses indicate the position of those PoEs, and the color map shows the amount of individually reconstructed PoEs on a virtual grid with a spacing of $0.1 \mu\text{m}$ in percent. The structure of the reconstructed distribution is exemplarily explained with the help of the four numbered PoEs in the text.

structed distributions of the individual PoEs are, in general, not symmetric but skewed towards the center or, more specific the central axes of the pixel. Distributions of PoEs near the central axis are the most skewed due to the large slope of $\delta(x)$. Distributions of PoEs on the border and the central axis are symmetric due to the symmetry of $\delta(x)$. The skewness is symmetric along the central axis and becomes smaller for a larger signal-to-noise ratio. The sharpness of the distribution increases with the distance to the central axis of the pixel. The skewness of the distributions for individual PoEs leads to the effect that a flat-filed illumination which can be interpreted as a summation of many of these distributions for different PoEs does not lead to a homogeneous density distribution of PoEs. An example is shown in Figure 5.1a. To obtain a homogeneous response of a flat-filed illumination, the same $\eta(x) = \eta(y)$ such as described in Section B.2.3 can

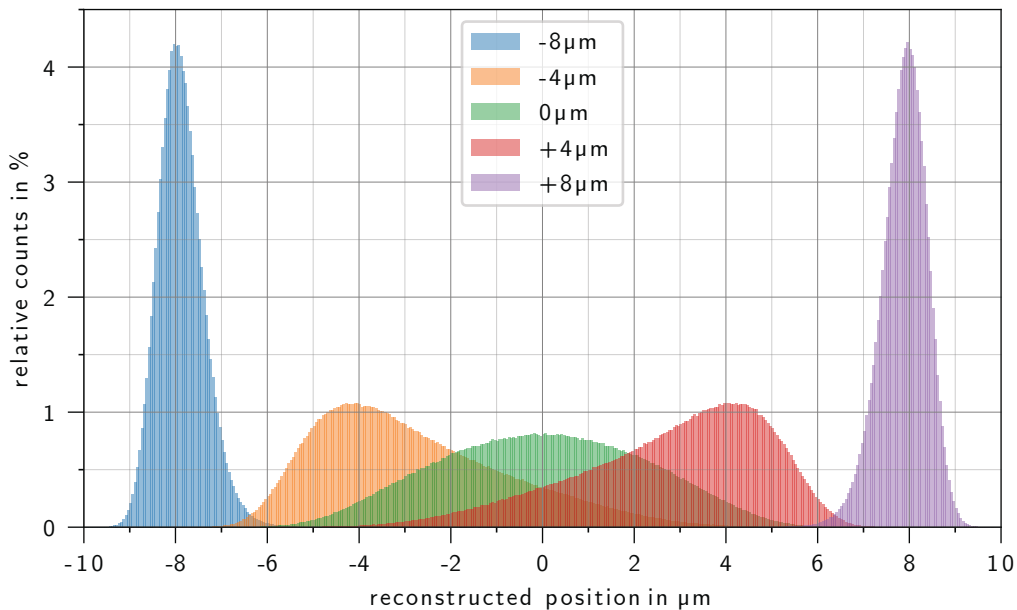


Figure B.13: Reconstruction with the center of gravity method with correction of five chosen PoEs within the pixel structure. The physical pixel size is $48 \times 48 \mu\text{m}^2$ with $(0,0)$ in the center of the pixel, and the energy of the primary photons is 8048 eV. A non-correlated Gaussian distributed pixel-wise noise with a mean of 11.3 eV is assumed. The color code refers to the different PoEs. For the sake of clarity, only the central area of the pixel is shown.

be introduced. However, instead of measured data, simulated data are used. This $\eta(x)$ corresponds to a systematic shift for each reconstructed PoE, which lowers the spatial accuracy of individual PoEs but leads for intensity images to images with fewer artifacts due to a more homogeneous normalized hit-map.

The presence of noise can be handled differently during the calculation of $\delta(x)$. The pixel with the maximal energy deposition and its next neighbors are used for all methods. The consideration of noise refers to the data set used to calculate the corrections $\delta(x)$. For the prediction and the calculation of $\eta(x)$, noise is always considered as it is also always present in real measurements. Due to the offset correction, negative values for the pixel's signal in the next neighbors can occur. However, they are not physically meaningful but can mathematically be used. Therefore, the methods, which are compared here, handle negative values in two different ways. On the one hand, ignoring negative values and, on the other hand, using them as valid values.

For neglecting the noise term for the simulated data set, which is used to

calculate the corrections, no difference between using or ignoring negative values appears since the values for all pixels are always positive or zero without the presence of noise.

To compare the different methods the normalized hit-map (Figure B.14a), the spatial precision in Figure B.14b, the accuracy in Figure B.14c, and the resolution in Figure B.14d, are used. The spatial precision is defined as the standard deviation of the individual reconstructed PoEs. The spatial accuracy is defined as the average distance between the mean position of the individual reconstructed PoEs and its ground truth. The resolution is defined as the average Euclidean distance between the individual reconstructed PoEs and their ground truth. A comparison between spatial precision, spatial accuracy, and resolution can be found in Appendix A.1. The line-style encoding is the same for all four plots: The color-codes correspond to the different handling of noise during calculation and the determination of which pixels are considered during the calculation and prediction. For the blue and the orange lines, no noise is assumed during the calculation process of $\delta(x)$. For the green and the red line, noise during the calculation is assumed. The blue and the green line originate using all neighboring pixels, whereas only pixels with a value above zero are used for the orange and red line.

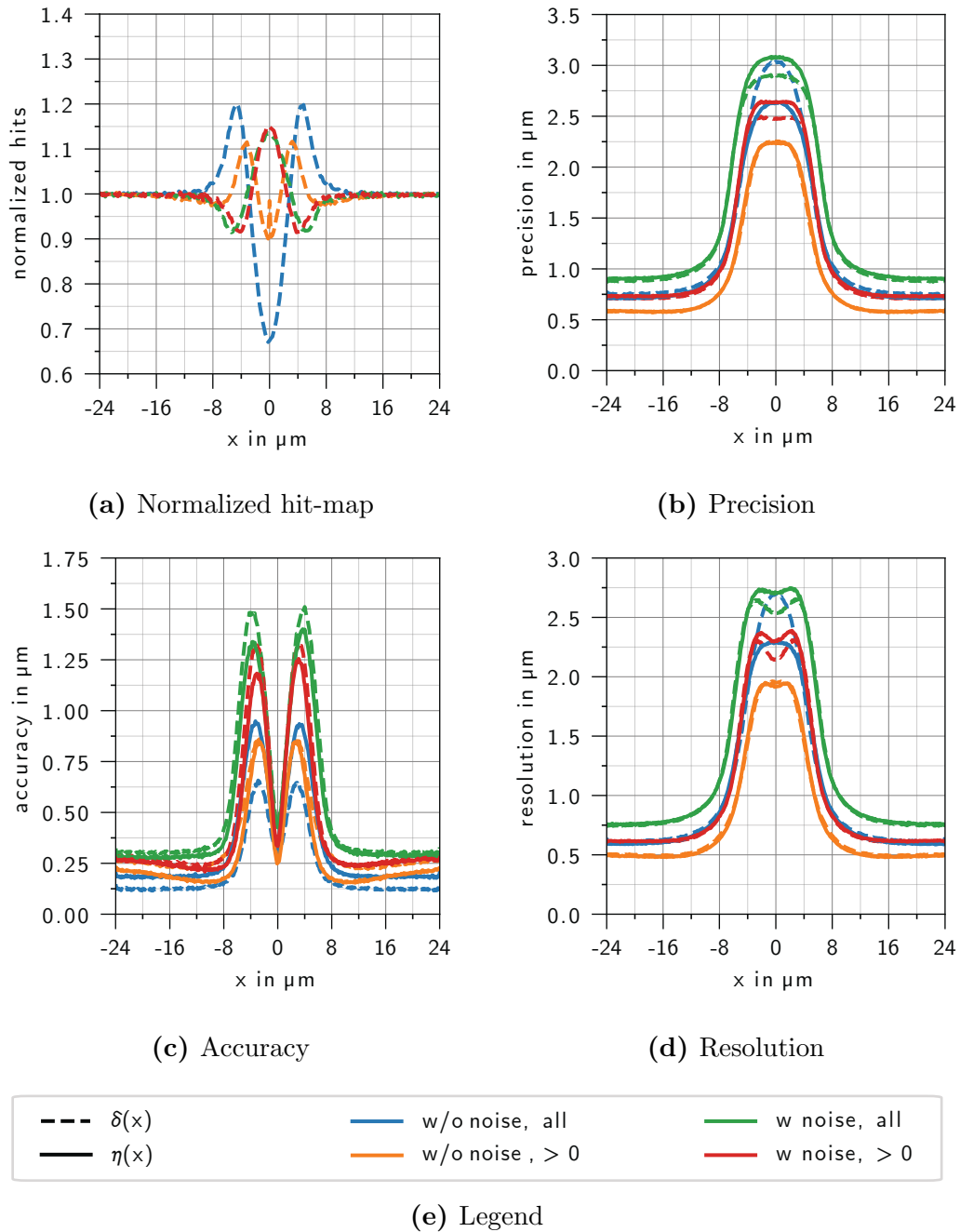


Figure B.14: Accuracy for conventional methods in terms of normalized hit-map, spatial precision, spatial accuracy, and resolution. The physical pixel size is $48 \times 48 \mu\text{m}^2$ with $(0,0)$ in the center of the pixel, and the energy of the primary photons is 8048 eV. A non-correlated Gaussian distributed pixel-wise noise with a mean of 11.3 eV is assumed. A detailed description of the legend can be found in the text.

Figure B.14a presents the normalized hit-map. The fluctuations of the curves are caused by the noise statistics. The increased number of hits of the orange curve in the center occurs by single pattern events. Due to noise, this feature is not seen in the blue curve.

Due to the non-linear response of the correction function $\delta(x)$ in the center of the pixel, the intensity image features artifacts at the pixel centers. These artifacts can be reduced by applying additionally the $\eta(x)$ correction, leading to a homogeneous response. The most homogeneous response is obtained by applying the $\eta(x)$ correction using all neighboring pixels and as well as noise during the composition of $\delta(x)$ values.

Another important parameter is the accuracy, which describes the average distance between the mean position of the individual reconstructed PoEs and their ground truth. Therefore, many patterns that differ only by noise are reconstructed for every PoE, and the average position is calculated and compared with the ground truth PoE. From Figure B.14c can be extracted that the accuracy is the best at the borders of the pixel and in the center of the pixel. The worst accuracy is at positions with a large change in slope of the correction functions. Here, little variations in the calculated weighted centroid have a significant influence on the PoE. The best accuracy is obtained using the $\delta(x)$ correction.

The spatial precision (Figure B.14b) describes the standard deviation of individual reconstructed PoEs that only differ by noise. The best spatial precision is obtained at the pixel borders.

As expected from the theoretical description, the resolution presented in Figure B.14d is the best at the pixel's borders. Using only neighboring pixel amplitudes larger than zero and assuming no noise during the calculation process of $\delta(x)$ leads to the best result. The resolution contains the contribution from the accuracy and the spatial precision. The dips in the resolution at the pixel centers are caused by the good accuracy in the pixel centers in comparison to positions near the pixel centers.

For the average distance between the individual reconstructed PoEs and its ground truth, the (linear) mean is used. The standard deviation, which uses the quadratic mean, is used for the spatial precision. For a good accuracy, which means the mean of the individual reconstructed PoEs can be approximated as the ground truth, the standard deviation is larger than the average distance (Appendix B.4). As a consequence of these definitions, the resolution can be better than the spatial precision if accuracy is high.

Depending on whether one is interested in the best results in terms of spacial accuracy and resolution for individual PoEs such as, for example, diffrac-

tion spots or a homogeneous response for intensity images, only the $\delta(x)$ correction or also the $\eta(x)$ correction should be applied. This different handling is caused by the asymmetric response (Figure B.13) due to the non-linearity of the correction functions.

Only the $\delta(x)$ correction should be used to get the best spatial accuracy and the best resolution. The $\eta(x)$ correction should be used to get the best response for intensity images. Individual PoEs are shifted by $\eta(x)$ in a way that the asymmetric response by the non-linearity of $\delta(x)$ is compensated. The individual PoEs shifted by $\eta(x)$ are very small in comparison to the $\delta(x)$ correction and in the sub-micrometer range but should still be applied. Since due to the summation over many individual PoEs, the intensity image is very sensitive to these systematic shifts. The shape of $\eta(x)$ strongly depends on the calculation method of $\delta(x)$ to shift the PoEs in a way that the multiplicity shown in Figure B.14a becomes constant across the pixel. The correction by $\eta(x)$ scales by the amount of pixel-wise noise and is zero in the noise-free limit.

B.3 Spatial Resolution in the Presence of Noise

In this Appendix, the influence of the noise on the spatial resolution for signals which are split over multiple pixels is described. The assumptions are only valid for the center of gravity method and the corresponding corrections described in Section B.2. Because the approaches based on neural networks use the information of the signal distribution of the x- and the y-axis, the made assumptions are not valid for this more abstract model of reconstruction. Without loss of generality, only one dimension (x) is assumed in the following. Technically, this can be achieved by summing over the other dimension (y). As a consequence, the index i denotes not a pixel but a column in the following.

The calculation of the spatial resolution is adapted from Kolanoski et al. [38] and expanded and generalized for an arbitrary detector response described by the function f . The noise is assumed to be constant for all pixels. The noise of column i is defined as the variance of the detector response q_{n_i} for no illumination (eq. B.2 on page 225):

$$\langle q_{n_i}^2 \rangle = \sigma_n^2 \tag{B.18}$$

The variance of the noise σ_n^2 is assumed to be constant over the measurement. Since the frames are offset corrected, the average over the detector response for no illumination is zero:

$$\langle q_{n_i} \rangle = 0 \tag{B.19}$$

The relative noise of each column is defined as the standard deviation of the noise divided by the total charge within one event:

$$n_i = \frac{\sqrt{\langle q_{n_i}^2 \rangle}}{\sum S_i} \quad (\text{B.20})$$

In the following, a distinction is made between statistically independent noise (eq. B.21) and fully correlated noise (eq. B.22). Fully correlated noise is, for example, noise produced by common mode.

$$\langle n_i n_j \rangle = \delta_{ij} \sigma_n^2 \quad (\text{B.21})$$

$$\langle n_i n_j \rangle = \sigma_n^2 \quad (\text{B.22})$$

In comparison to the total charge within one event, the noise has to be small [38]:

$$n_i \ll \sum_i S_i \quad (\text{B.23})$$

Without loss of generality, the origin of the coordinate system is set to the center of the event. Figure B.15 shows the choice of the coordinate system for an even and an uneven number of pixels of the event¹.

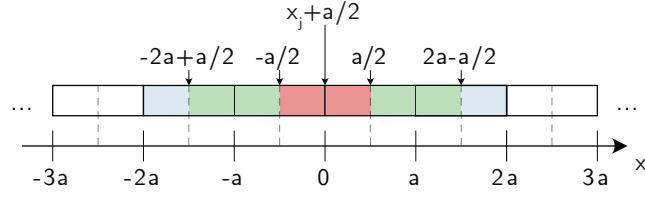
$$\sum_i x_i = 0 \quad (\text{B.24})$$

The reconstructed center of gravity $x_{\text{CoG}}^{\text{rec}}$ results from the true center of gravity \tilde{x}_{CoG} plus a deviation which is caused by the noise [38]:

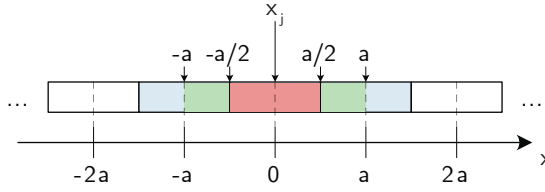
$$\begin{aligned} x_{\text{CoG}}^{\text{rec}} &= \frac{\sum_i (S_i + n_i) x_i}{\sum_i (S_i + n_i)} = \frac{\tilde{x}_{\text{CoG}} + \sum_i n_i x_i}{1 + \sum_i n_i} \\ &= \left(\tilde{x}_{\text{CoG}} + \sum_i n_i x_i \right) \left(1 - \sum_i n_i + \mathcal{O}(n_i^2) \right) \end{aligned} \quad (\text{B.25})$$

In the last step, the denominator is approximated by a Taylor expansion for small n_i .

¹It was assumed that the number of pixels in y-dimension that contribute to the event pattern is the same for every x-position. Otherwise, the origin of the coordinate system is the weighted average over the x-positions of the pixels that contribute to the event pattern. The weights of the averaging are the corresponding number of pixels in the y-dimension which contribute to the event pattern.



(a) N is even



(b) N is uneven

Figure B.15: Choice of the coordinate system for even (a) and uneven sizes (b) of the event in the x-dimension. The coordinate axis indicates the origin of the coordinate system. The dashed lines represent the pixels' centers with a pixel size of a , and the solid lines describe the pixel borders. The blue pixels denote pixels that contain a signal by the event plus noise. The white pixels only contain noise. N denotes the number of pixels in the event pattern in the x-dimension. The green-colored area denotes the mathematically possible positions of the PoE. However, the true PoE is in the red area due to the symmetry and the requirement to the noise (eq. B.23). The true PoE is between two pixel centers for the even case and within the central pixel for the uneven case.

The made uncertainty $\Delta_x(x)$ is the difference between the reconstructed center of gravity $x_{\text{CoG}}^{\text{rec}}$ and the true PoE x [38]:

$$\begin{aligned}
 \Delta_x(x) &= x_{\text{CoG}}^{\text{rec}} - x \\
 &= \sum_i n_i x_i - x \sum_i n_i - \left(\sum_i n_i x_i \right) \left(\sum_i n_i \right)^{-1} + \mathcal{O}(n_i^2) \\
 &= \sum_i n_i (x_i - x) + \mathcal{O}(n_i^2)
 \end{aligned} \tag{B.26}$$

In this case, it was assumed that the true center of gravity \tilde{x}_{CoG} is the true PoE x .

The variance can be calculated as follows [146] [38]:

$$\begin{aligned}
 \sigma_x^2(x) &= \langle \Delta_x^2 \rangle - \underbrace{\langle \Delta_x \rangle^2}_{=0} = \langle \Delta_x^2 \rangle \\
 &= \left\langle \sum_{i,j} n_i n_j (x_i - x)(x_j - x) \right\rangle + \mathcal{O}(\sigma_n^3) \\
 &= \sum_{i,j} \langle n_i n_j \rangle \langle (x_i - x)(x_j - x) \rangle + \mathcal{O}(\sigma_n^3)
 \end{aligned} \tag{B.27}$$

The bracket indicates an averaging over all $x_{\text{CoG}}^{\text{rec}} - x$, respectively, over the noise.

Again, statistical independent noise (eq. B.28) and fully correlated noise (eq. B.29) is assumed [38]:

$$\begin{aligned}
 \sigma_{x, \text{uncor}}^2(x) &= \sum_{i,j} \langle n_i n_j \rangle \langle (x_i - x)(x_j - x) \rangle + \mathcal{O}(\sigma_n^3) \\
 &= \sum_i \sigma_n^2 \langle (x_i - x)^2 \rangle + \mathcal{O}(\sigma_n^3) \\
 &= \sigma_n^2 \sum_i \langle x_i^2 - \underbrace{2x_i x}_{=0} + x^2 \rangle + \mathcal{O}(\sigma_n^3) \\
 &= \sigma_n^2 \left[\left(\sum_i x_i^2 \right) + N \langle x^2 \rangle \right] + \mathcal{O}(\sigma_n^3)
 \end{aligned} \tag{B.28}$$

Due to the definition of the coordinate system, the sum over x_i is zero.

$$\begin{aligned}
 \sigma_{x, \text{cor}}^2(x) &= \sum_{i,j} \langle n_i n_j \rangle \langle (x_i - x)(x_j - x) \rangle + \mathcal{O}(\sigma_n^3) \\
 &= \sum_{i,j} \sigma_n^2 \langle (x_i - x)(x_j - x) \rangle + \mathcal{O}(\sigma_n^3) \\
 &= \sigma_n^2 \sum_{i,j} \left\langle \underbrace{x_i x_j}_{=0} - \underbrace{x_i x}_{=0} - \underbrace{x_j x}_{=0} + x^2 \right\rangle + \mathcal{O}(\sigma_n^3) \\
 &= \sigma_n^2 N^2 \langle x^2 \rangle + \mathcal{O}(\sigma_n^3)
 \end{aligned} \tag{B.29}$$

In the following, a necessary correction like described in Section B.2 is assumed [38]:

$$x^{\text{rec}} = x_{\text{CoG}}^{\text{rec}} + \delta(x_{\text{CoG}}^{\text{rec}}) = f(x_{\text{CoG}}^{\text{rec}}) \tag{B.30}$$

For simplicity, the function $\delta(x_{\text{CoG}}^{\text{rec}})$ which describes the correction is redefined

as $f(x_{\text{CoG}}^{\text{rec}})$:

$$f(x) = x + \delta(x) \quad (\text{B.31})$$

The requirements for the function f are as follows:

- $0 \leq |f| \leq 1$: The corrected PoE is always in the same physical pixel as the reconstructed PoE by the center of gravity method.
- $\frac{\partial f(x)}{\partial x} > 0$: The function f is strictly monotonous. Otherwise, the assignment between the center of gravity and the reconstructed PoE are not unique.
- $\tilde{f}(x) = f(x) - 0.5 = -\tilde{f}(-x)$: The function f is anti-symmetrical around $f(0) = 0.5$.
- $f(-a/2) = 0$ and $f(a/2) = a$: The center of gravity and the reconstructed PoE are the same for the pixel borders due to symmetry.

Using the correction function f , eq. B.26 modifies to:

$$\begin{aligned} \Delta_x^{(f)}(x) &= f(x_{\text{CoG}}^{\text{rec}}) - x \\ &= f\left(\tilde{x}_{\text{CoG}} + \sum_i n_i(x_i - \tilde{x}_{\text{CoG}}) + \mathcal{O}(n_i^2)\right) - x \\ &\approx \underbrace{f(\tilde{x}_{\text{CoG}})}_{=x} + \left.\frac{df(\xi)}{d\xi}\right|_{f^{-1}(x)} \cdot \left(\sum_i n_i(x_i - x)\right) - x \\ &= \left.\frac{df(\xi)}{d\xi}\right|_{f^{-1}(x)} \cdot \left(\sum_i n_i(x_i - x)\right) \\ &\approx \left.\frac{df(\xi)}{d\xi}\right|_{f^{-1}(x)} \cdot \Delta_x(x) \end{aligned} \quad (\text{B.32})$$

Here, the function f is approximated using the Taylor expansion at the point ξ .

$$\xi = \tilde{x}_{\text{CoG}} + \sum_i n_i(x_i - x) \quad (\text{B.33})$$

The made uncertainty under the assumption of the correction f results analogous to eq. B.28, respectively, eq. B.29:

$$\left(\sigma_{x, \text{uncor}}^{(f)}\right)^2(x) \approx \sigma_n^2 \left\langle \left(\left.\frac{df(\xi)}{d\xi}\right|_{f^{-1}(x)} \right)^2 \right\rangle \left[\left(\sum_i x_i^2 \right) + N \langle x^2 \rangle \right] \quad (\text{B.34})$$

$$(\sigma_{x, \text{cor}}^{(f)})^2(x) \approx \sigma_n^2 N^2 \left\langle \left(\left. \frac{df(\xi)}{d\xi} \right|_{f^{-1}(x)} \right)^2 \right\rangle \langle x^2 \rangle \quad (\text{B.35})$$

The summations and integrals can be simplified for an even number of columns in the event pattern (Figure B.15a) and an uneven number of columns in the event pattern (Figure B.15b):

For an even number of columns, the summation simplifies to:

$$\sum_{i=1}^N x_i^2 = 2 \sum_{i=1}^{N/2} x_i^2 = 2a^2 \sum_{i=1}^{N/2} (i - 0.5)^2 \quad (\text{B.36})$$

For $N = 2$, the sum is $\frac{a^2}{2}$ and for $N = 4$, the sum is $5a^2$.

For an uneven number of columns, the summation simplifies to:

$$\sum_{i=1}^N x_i^2 = 2 \sum_{i=1}^{N/2-0.5} x_i^2 = 2a^2 \sum_{i=1}^{N/2-0.5} i^2 \quad (\text{B.37})$$

For $N = 3$, the sum is $2a^2$.

An averaging over noise leads to

$$\langle x^2 \rangle = x^2 \quad (\text{B.38})$$

and an averaging over space to

$$\begin{aligned} \langle x^2 \rangle &= \frac{1}{a} \int_{-\frac{1}{2}a}^{\frac{1}{2}a} x^2 dx \\ &= \frac{1}{a} \left[\frac{x^3}{3} \right]_{-\frac{1}{2}a}^{\frac{1}{2}a} = \frac{a^2}{12}. \end{aligned} \quad (\text{B.39})$$

Since it is actually an averaging over all possible $f(x_{\text{CoG}}^{\text{rec}}) - x$ and since $f(x_{\text{CoG}}^{\text{rec}})$ describes without the contribution of noise the exact PoE, the integral limits are defined by the red area in Figure B.15 [38]. The limit of the integral is independent of the number of columns that contribute to the event pattern.

It should be noted that an incorrect event pattern analysis leads to an additional error and that, in general, σ_n^2 depends on the number of

contributing rows. Since the derivative of $f(x)$ is the largest in the center of the pixels, the spatial resolution is in the pixel center the lowest and increases towards the pixel's borders. The simulation shown in Figure B.12 on page 251 and the measurement presented in Section 10.1.1 confirms this behavior.

B.4 Average Distance from the Mean in Comparison to the Standard Deviation

The average distance from the mean is defined as:

$$\overline{\text{dist}} = \frac{1}{N} \sum_{i=1}^N \text{dist}_i = \frac{1}{N} \sum_{i=1}^N \sqrt{\sum_{j=1}^2 (x_{i,j} - \langle x \rangle_j)^2} \quad (\text{B.40})$$

Here, i describes the index for the individual contributions to the mean, j describes the index of the dimension, and $\langle x \rangle_j$ is the mean of the j^{th} dimension.

The standard deviation σ_x is defined as:

$$\begin{aligned} \sigma_x &= \sqrt{\sum_{j=1}^2 \sigma_j^2} \\ &= \sqrt{\sum_{j=1}^2 \left(\frac{1}{N} \sum_{i=1}^N (x_{i,j} - \langle x \rangle_j)^2 \right)} \\ &= \sqrt{\sum_{i=1}^N \left(\frac{1}{N} \sum_{j=1}^2 (x_{i,j} - \langle x \rangle_j)^2 \right)} \\ &\geq \frac{1}{N} \sum_{i=1}^N \sqrt{\sum_{j=1}^2 (x_{i,j} - \langle x \rangle_j)^2} = \overline{\text{dist}} \end{aligned} \quad (\text{B.41})$$

Here, i describes the index for the individual contributions to the mean, j describes the index of the dimension, and $\langle x \rangle_j$ is the mean of the j^{th} dimension. The approximation in the last row is done by using a modification of the Jensen's inequality [222], the facts that the square root is a concave function, $1/N > 0$, and $\sum_{i=1}^N 1/N = 1$.

Appendix C

Artificial Neural Networks

C.1 Pseudo-Code of the Adam Optimization Algorithm

In this Appendix, a pseudo-code implementation of the Adam optimizer is shown.

Algorithm 1: Adam optimization algorithm [140]

Require: α : Step size
 $\beta_1, \beta_2 \in [0, 1)$: Decay rate for the moment estimations
 $f(\theta)$: Function with parameters θ
 θ_0 : Initial parameter vector

Return : θ_t : Resulting parameter vector

```
1  $m_0 \leftarrow 0$ 
   #Initialize first moment vector
2  $v_0 \leftarrow 0$ 
   #Initialize second moment vector
3  $t \leftarrow 0$ 
   #Initialize time steps

   #while not converged
4 while  $\|\theta_t - \theta_{t-1}\| > \text{precision}$  do
5    $t \leftarrow t + 1$ 
   #Increase time step
6    $g_t \leftarrow \nabla_{\theta} f_t(\theta_{t-1})$ 
   #Get gradient w.r.t. parameters at timestep t
7    $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ 
   #Update biased first moment estimate
8    $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ 
   #Update biased second raw moment estimate
9    $\hat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ 
   #Update bias-corrected first moment estimate
10   $\hat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ 
   #Update bias-corrected second raw moment estimate
11   $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$ 
   #Update parameters
12 end
```

C.2 Gradient of the Loss Function with Respect to the Weights and the Biases of the Neural Network

This Appendix demonstrates an explicit calculation of the gradient of the loss function with respect to the biases and weights according to [113].

Figure C.1 shows an example network with two hidden layers.

The change in loss is related to the change in the weight w_{jk}^i and can be approximated as:

$$\Delta\mathcal{L} \approx \frac{\partial\mathcal{L}}{\partial w_{jk}^i} \Delta w_{jk}^i \quad (\text{C.1})$$

This means a small change in the weight w_{jk}^i causes a small change in the loss function. The partial derivative describes how this small change propagates through the neural network. The idea is to look at this propagation in detail and get an expression for the partial derivative.

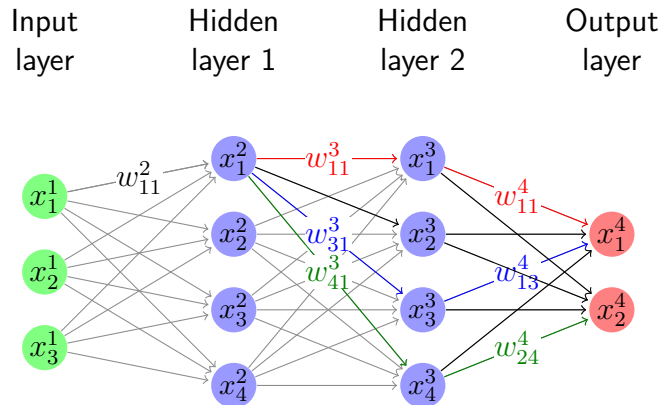


Figure C.1: Concept of backpropagation. The loss function is $\mathcal{L} = \mathcal{L}(x_1^4, x_2^4)$ and depends only on the activation of the output neurons. The goal is to find how the weight w_{11}^2 influences the loss function. A change of w_{11}^2 propagates the red path along with the neural network. But there are many more paths like, for example, the blue or green path. To consider the overall influence, all possible paths have to be considered. These paths contain all combinations of the colored and black connections between the different layers.

A change of the weight w_{jk}^i causes a change in the activation x_j^i of the j^{th} neuron in the i^{th} layer:

$$\Delta x_j^i \approx \frac{\partial x_j^i}{\partial w_{jk}^i} \Delta w_{jk}^i \quad (\text{C.2})$$

The change of the activation x_j^i changes the activation x_m^{i+1} of the next layer:

$$\Delta x_m^{i+1} \approx \frac{\partial x_m^{i+1}}{\partial x_j^i} \Delta x_j^i \quad (\text{C.3})$$

This change propagates along with the layers of the neural network. Mathematically, this can be achieved by applying eq. C.3 recursively, resulting in chain rule. Combining eq. C.2 and eq. C.3 leads to an expression for the change of the activation x_r^I in the last layer of the neural network:

$$\Delta x_r^I \approx \frac{\partial x_r^I}{\partial x_q^{I-1}} \frac{\partial x_q^{I-1}}{\partial x_p^{I-2}} \cdots \frac{\partial x_m^{i+1}}{\partial x_j^i} \frac{\partial x_j^i}{\partial w_{jk}^i} \Delta w_{jk}^i \quad (\text{C.4})$$

Eq. C.4 considers only one path through the neurons of the neural network. This path can be, for example, the red path in Figure C.1. But there are many more paths on which the change Δw_{jk}^i can propagate to the input of the q^{th} neuron of the last layer I. Other possible paths are the blue or the green path in Figure C.1. To consider all possible paths through the layers between layer i and layer I, one has to sum over all possibilities:

$$\Delta x_r^I \approx \sum_{q,p,o \dots n,m} \frac{\partial x_r^I}{\partial x_q^{I-1}} \frac{\partial x_q^{I-1}}{\partial x_p^{I-2}} \frac{\partial x_p^{I-2}}{\partial x_o^{I-3}} \cdots \frac{\partial x_n^{i+2}}{\partial x_m^{i+1}} \frac{\partial x_m^{i+1}}{\partial x_j^i} \frac{\partial x_j^i}{\partial w_{jk}^i} \Delta w_{jk}^i \quad (\text{C.5})$$

The loss function can be written as a function of the output of the neural network, which is nothing else than the activation of the neurons of the last layer:

$$\Delta \mathcal{L} \approx \sum_{r,q,p,o \dots n,m} \frac{\partial \mathcal{L}}{\partial x_r^I} \frac{\partial x_r^I}{\partial x_q^{I-1}} \frac{\partial x_q^{I-1}}{\partial x_p^{I-2}} \frac{\partial x_p^{I-2}}{\partial x_o^{I-3}} \cdots \frac{\partial x_n^{i+2}}{\partial x_m^{i+1}} \frac{\partial x_m^{i+1}}{\partial x_j^i} \frac{\partial x_j^i}{\partial w_{jk}^i} \Delta w_{jk}^i \quad (\text{C.6})$$

The comparison of eq. C.1 and C.6 leads to an expression for the derivative of the loss function with respect to w_{jk}^i :

$$\frac{\partial \mathcal{L}}{\partial w_{jk}^i} = \sum_{r,q,p,o \dots n,m} \frac{\partial \mathcal{L}}{\partial x_r^I} \frac{\partial x_r^I}{\partial x_q^{I-1}} \frac{\partial x_q^{I-1}}{\partial x_p^{I-2}} \frac{\partial x_p^{I-2}}{\partial x_o^{I-3}} \cdots \frac{\partial x_n^{i+2}}{\partial x_m^{i+1}} \frac{\partial x_m^{i+1}}{\partial x_j^i} \frac{\partial x_j^i}{\partial w_{jk}^i} \quad (\text{C.7})$$

Eq. C.7 has an intuitive interpretation. The change in weight w_{jk}^i of j^{th} neurons in the i^{th} layer propagates through all neurons of the following layers. Doing this, each neuron modifies this change by a factor which is simply the partial derivative of the activation with respect to the activation of a neuron of the previous layer. The total change is the sum of all possible paths through the

C.2. Gradient of the Loss Function with Respect to the Weights and the Biases of the Neural Network

neural network. The derivation for the influence of the change of the biases to the loss function works in the same way and leads to a similar result:

$$\frac{\partial \mathcal{L}}{\partial b_j^i} = \sum_{r,q,p,o \dots n,m} \frac{\partial \mathcal{L}}{\partial x_r^I} \frac{\partial x_r^I}{\partial x_q^{I-1}} \frac{\partial x_q^{I-1}}{\partial x_p^{I-2}} \frac{\partial x_p^{I-2}}{\partial x_o^{I-3}} \dots \frac{\partial x_n^{i+2}}{\partial x_m^{i+1}} \frac{\partial x_m^{i+1}}{\partial x_j^i} \frac{\partial x_j^i}{\partial b_j^i} \quad (\text{C.8})$$

Eq. C.7 and C.8 can be simplified by looking at one of the partial derivatives and one of the sums in detail. The derivatives of the last layer with respect to the second last layer can be rewritten as a matrix:

$$M^I := \frac{\partial x^I}{\partial x^{I-1}} = \begin{bmatrix} \frac{\partial x_1^I}{\partial x_1^{I-1}} & \frac{\partial x_1^I}{\partial x_2^{I-1}} & \dots \\ \vdots & \ddots & \\ \frac{\partial x_R^I}{\partial x_1^{I-1}} & & \frac{\partial x_R^I}{\partial x_Q^{I-1}} \end{bmatrix} \quad (\text{C.9})$$

Here, the capital letters R and Q denote the number of neurons in the last, respectively, the second last layer. The definition of the matrices of the other layers is similar. Using the matrix multiplication, the sum over the second last layers simplifies¹:

$$\sum_q \frac{\partial x_r^I}{\partial x_q^{I-1}} \frac{\partial x_q^{I-1}}{\partial x_p^{I-2}} = M^I \cdot M^{I-1} \quad (\text{C.10})$$

Therefore, eq. C.7 and C.8 simplify basically to a sequence of matrix multiplications:

$$\frac{\partial \mathcal{L}}{\partial w_{jk}^i} = \frac{\partial \mathcal{L}}{\partial x_r^I} M^I M^{I-1} M^{I-2} \dots M^{i+2} M^{i+1} \frac{\partial x_j^i}{\partial w_{jk}^i} \quad (\text{C.11})$$

$$\frac{\partial \mathcal{L}}{\partial b_j^i} = \frac{\partial \mathcal{L}}{\partial x_r^I} M^I M^{I-1} M^{I-2} \dots M^{i+2} M^{i+1} \frac{\partial x_j^i}{\partial b_j^i} \quad (\text{C.12})$$

In eq. C.11 and C.12, the first derivative cannot be simplified since the loss function can be an arbitrary function. The last derivatives with respect to the weights and the biases can be simplified using the explicit formulation of the activation of a neuron defined by eq. 6.4 on page 91:

$$\frac{\partial x_j^i}{\partial w_{jk}^i} = x_k^{i-1} \quad (\text{C.13})$$

¹The sum can be performed directly since the other terms in eq. C.7 and C.8 are independent of q.

$$\frac{\partial x_j^i}{\partial b_j^i} = 1 \quad (\text{C.14})$$

By using again eq. 6.4 on page 91, the entries of the individual matrices can be written as:

$$\frac{\partial x_j^i}{\partial x_k^{i-1}} = \frac{\partial f(z_j^i)}{\partial z_j^i} \frac{\partial z_j^i}{\partial x_k^{i-1}} = \frac{\partial f(z_j^i)}{\partial z_j^i} \frac{\partial (b_j^i + \sum_k w_{jk}^i x_k^{i-1})}{\partial x_k^{i-1}} = \frac{\partial f(z_j^i)}{\partial z_j^i} w_{jk}^i \quad (\text{C.15})$$

The derivative of the activation function $f(z_j^i)$ with respect to the weighted input z_j^i cannot be simplified since the activation function can be an arbitrary function.

This means the matrices M are special because they depend on the values of the neurons and, therefore, on the input. Therefore, their entries change for every training sample.

C.3 Pseudo-Code of the Backpropagation Algorithm

In this Appendix, a pseudo-code implementation of the backpropagation optimizer is shown.

Algorithm 2: Backpropagation algorithm

Require: x^0 : Input training examples in batch
 x^I : Ground truth of the examples in the training batch
 w_{jk}^i : Current weights
 b_{jk}^i : Current biases
 Neural network topology

Return : $\frac{\partial \mathcal{L}}{\partial b_j^i}$: Gradient with respect to the biases
 $\frac{\partial \mathcal{L}}{\partial w_{jk}^i}$: Gradient with respect to the weights

```

#for each training example in batch
1 for all  $x^0$  do
    #Feed forward
2   for  $i = 0, 1, \dots, I$  do
3      $z^i \leftarrow w^i x^{i-1}$ 
        #Calculate weighted input
4      $x^i \leftarrow f(z^i)$ 
        #Calculate activation
5   end
6    $\delta^I \leftarrow \nabla_x \mathcal{L} \odot \frac{df(z_j^I)}{dz_j^I}$ 
        #Calculate errors of last layer
    #Backpropagation of the error
7   for  $i = I-1, I-2, \dots, 0$  do
8      $\delta^i \leftarrow \left( (w^{i+1})^T \right) \delta^{i+1} \odot \frac{df(z_j^i)}{dz_j^i}$ 
        #Calculate errors of the hidden and input layer
9   end
10   $\frac{\partial \mathcal{L}}{\partial b_j^i} \leftarrow \delta_j^i$ 
        #Calculate the gradient with respect to the biases
11   $\frac{\partial \mathcal{L}}{\partial w_{jk}^i} \leftarrow x_k^{i-1} \delta_j^i$ 
        #Calculate the gradient with respect to the weights
12 end

```

C.4 Hyper-Parameters of the Special Layers Used in Neural Networks

The following Tables show the used layers and their corresponding hyper-parameters.

Table C.1: Hyper-parameters of the special layers used in neural networks. Based on [130].

layer	arguments	type	description
dense	units	integer	Size of output space
activation	activation	function	Definition of the activation function
n-dimensional convolution	channels	integer	Number of channels at the output
	kernel size	list of n integers	Size of the kernel
	strides	list of n integers	Step size of moving
	padding	string	Can be "valid" or "same"; Behavior at the edges
n-dimensional pooling	dilation rate	list of n integers	Dilation rate to use for dilated convolution
	pool size	list of n integers	Size of the pooling window
	strides	list of n integers	Step size of moving
dropout	padding	string	Can be "valid" or "same"; Behavior at the edges
	rate	float between 0 and 1	Fraction of the input units to drop

Table C.2: Hyper-parameters of the special layers used in neural networks. Based on [130].

layer	arguments	type	description
2-dimensional upsampling	size interpolation	list of 2 integers string	Upsampling factor for columns and rows Can be "bilinear", "nearest", "bicubic" "lanczos3", "lanczos5", "gaussian", "area", or "mitchellcubic"
n-dimensional separable convolution	channels kernel size strides padding dilation rate	integer list of n integers list of n integers string list of n integers	Number of channels at the output Size of the kernel Step size of moving Can be "valid" or "same"; Behavior at the edges Dilation rate to use for dilated convolution
n-dimensional transposed convolution	channels kernel size strides padding dilation rate	integer list of n integers list of n integers string list of n integers	Number of channels at the output Size of the kernel Step size of moving Can be "valid" or "same"; Behavior at the edges Dilation rate to use for dilated convolution
batch normalization	axis	integer	Axis that should be normalized; Typically the channel axis

C.5 Topology and Number of Parameters of Used Neural Networks

Table C.3: Parameters of the compact neural network (Chapter 7). None denotes the batch size. Dense and convolution layers are always followed by an activation layer.

layer	output shape	parameter
input	(None, 3, 3)	0
expand dimensions	(None, 3, 3, 1)	0
2-d convolution	(None, 3, 3, 64)	640
2-d convolution	(None, 3, 3, 64)	36928
dropout	(None, 3, 3, 64)	0
flatten	(None, 576)	0
dense	(None, 100)	57700
dense	(None, 2)	202
total parameters:		95470
trainable parameters:		95470
non-trainable parameters:		0

Table C.4: Hyper-parameters of the convolutional neural network PoENN and the SR neural networks. The column ref denotes the Section in the thesis which provides more information.

arguments	type	default	description	ref
activation	function	Leaky ReLU	Applied activation function (The activation function of the last layer is sigmoid.)	6.4
kernelSize	integer	3	Size of the kernel	6.7.1
separable	boolean	False	Should separable convolu- tions be used?	6.7.2
batchNorm	boolean	False	Is batch normalization ac- tive?	8.1
upsampling	string	"subpixel"	Can be "subpixel", "trans- posed", or "resize" and de- notes the upsampling type.	8.1.5.2
preamble	boolean	False	Should there be an addi- tional convolution block be- fore and after the u-net?	8.1
features	integer	16	Number of channels of the preamble, SR module and fusion module	8.1
factor	integer	16	Factor for the channels of the u-net. It represents the number of channels of the first and last u-net block.	8.1
depth	integer	2	Depth of the u-net.	8.1
middleBlocks	integer	1	Number of blocks in the bot- tleneck of the u-net	8.1

Table C.5: Additional hyper-parameters for the SR neural networks. The hyper-parameters in Table C.4 are also used for the SR neural networks. The column ref denotes the section in the thesis which provides more information.

arguments	type	default	description	ref
subPixelFactor	integer	3	Expansion into the subpixel regime. Every physical pixel is divided in each dimension into $2^{\text{subPixelFactor}}$ subpixels.	8.1.4
srBlock	integer	2	Number of convolution blocks after upsampling in the SR-net.	8.1.4
stepWiseSR	boolean	True	Is super-resolution achieved in one step or iterative with a stepsize of two?	8.1.4
resnet	boolean	False	Is a residual neural network used?	9.1.1.1
threeDim	boolean	False	Is it SISR or MISR?	9.2.1
fusionBlock	integer	4	Number of convolution blocks of the fusion net.	9.2.1.1

Table C.6: Parameters of the convolutional neural network PoENN (Chapter 8) (Part 1 of 2). The encoder and bottleneck are shown. The decoder is in Table C.7 None denotes the batch size. The frame size is 256×256 pixel. However, due to the architecture, the number of parameters is independent. Dense and convolution layers are always followed by an activation layer.

id	layer	output shape	parameter	connected to
1	input	(None, 256, 256)	0	
2	expand dimensions	(None, 256, 256, 1)	0	1
3	2-d convolution	(None, 256, 256, 16)	160	2
4	dropout	(None, 256, 256, 16)	0	3
5	2-d convolution	(None, 256, 256, 16)	2320	4
6	max pooling	(None, 128, 128, 16)	0	5
7	2-d convolution	(None, 128, 128, 32)	4640	6
8	dropout	(None, 128, 128, 16)	0	7
9	2-d convolution	(None, 128, 128, 32)	9248	8
10	max pooling	(None, 64, 64, 32)	0	9
11	2-d convolution	(None, 64, 64, 64)	18496	10
12	dropout	(None, 64, 64, 64)	0	11
13	2-d convolution	(None, 64, 64, 64)	36928	12

Table C.7: Parameters of the convolutional neural network PoENN (Chapter 8) (Part 2 of 2). The decoder is shown. The encoder and the bottleneck are in Table C.6. None denotes the batch size. The frame size is 256×256 pixel. However, due to the architecture, the number of parameters is independent. Dense and convolution layers are always followed by an activation layer.

id	layer	output shape	parameter	connected to
14	2-d convolution	(None, 64, 64, 128)	73856	13
15	depth to space	(None, 128, 128, 32)	0	14
16	concatenate	(None, 128, 128, 64)	0	9, 15
17	2-d convolution	(None, 128, 128, 32)	18464	16
18	dropout	(None, 128, 128, 32)	0	17
19	2-d convolution	(None, 128, 128, 32)	9248	18
20	2-d convolution	(None, 128, 128, 64)	18496	19
21	depth to space	(None, 256, 256, 16)	0	20
22	concatenate	(None, 256, 256, 32)	0	5, 21
23	2-d convolution	(None, 256, 256, 16)	4624	22
24	dropout	(None, 256, 256, 16)	0	23
25	2-d convolution	(None, 256, 256, 16)	2320	24
26	2-d convolution	(None, 256, 256, 1)	145	25
27	reduce dimensions	(None, 256, 256)	0	26
total parameters:				198945
trainable parameters:				198945
non-trainable parameters:				0

Table C.8: Hyper-parameters of the SR neural network. The used SISR has 273537 parameters, of which 270705 are trainable and 2832 are non-trainable. The MISR has 282265 parameters, of which 279241 are trainable and 3024 are non-trainable. The detailed parameters are not shown since the layer structure would lead to a very long list. As a consequence, only the hyper-parameters are shown. The abstract design is described in Chapter 9.

hyper-parameters	value
activation	Leaky ReLu
kernelSize	3
separable	False
batchNorm	True
upsampling	"transposed"
preamble	True
features	16
factor	16
depth	2
middleBlocks	1
subPixelFactor	3
srBlock	True
stepWiseSR	True
resnet	True
threeDim	True
fusionBlock	True

Appendix D

Experiments

D.1 Modulation Transfer Function

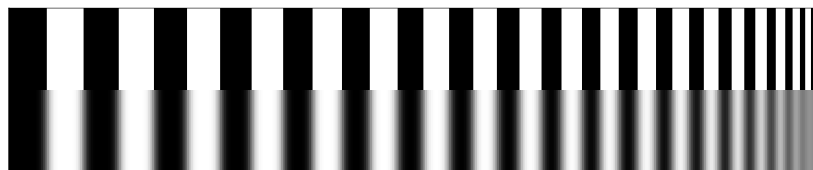
The modulation transfer function (MTF) describes how much contrast of the object function is transferred by the imaging process [223]. The upper part of Figure D.1a shows alternating stripes of illuminated and not illuminated areas. The spatial frequency of the alternation increases from left to right. The lower part of Figure D.1a shows the response of an image process. The image process blurs the sharp edges of the stripes and smooths them. In the raw data, the main contribution to this blurring is due to the random energy depositions in the form of tracks and not a point-wise energy deposition. In the analyzed data, the main contribution occurs from false reconstructed positions.

Figure D.1b shows the intensity and the contrast of the image process in Figure D.1a. The magnitude of the extremes of the intensity decreases with an increasing spatial frequency of the stripes. The contrast is a function of the spatial frequency ω and can be calculated from the intensity of the illuminated areas I_{\max} and the not illuminated areas I_{\min} [224]:

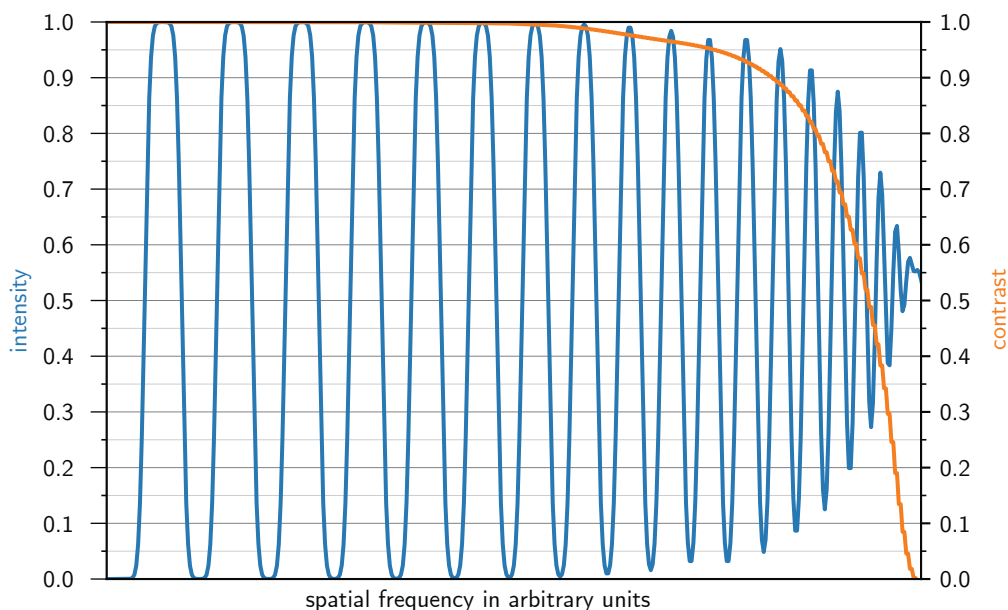
$$\text{contrast}(\omega) = \frac{I_{\max} - I_{\min}}{I_{\max} + I_{\min}} \quad (\text{D.1})$$

The MTF can be obtained from a slanted edge. It is essential that the sample is as sharp and the line is as straight as possible. Otherwise, the MTF is affected by the sample and not only by the imaging process. The edge should be tilted about 6 to 8 degrees against the pixel structure and should span over at least 20 rows, respectively, columns of the frame. [225]

With the following approach, the MTF (Figure D.2) can be determined [223]:



(a) The upper line shows the object function of increasingly finer stripes, and the lower line the image function. The image function shows fewer details because of effects created by the imaging process.



(b) Contrast and intensity for the increasingly finer stripes shown in Figure D.1a. Due to the resolution limit introduced by the imaging process, the contrast and the envelope of the intensity decrease with higher spatial frequencies of the stripes.

Figure D.1: Visual concept of the MTF. Figure adapted from [69].

- Find a suitable edge.
- Define a region of interest of the image containing the edge (Figure D.2a).
- Find the edge for each row in this region of interest.
Therefore, fit the edge spread function (ESF) for each row and use the center position b as position of the edge (Figure D.2b).

$$\text{ESF}(x) = a \cdot \operatorname{erf}\left(\frac{b-x}{c}\right) + d \quad (\text{D.2})$$

- Fit the edge approximated as line (Figure D.2c).

$$f(x) = m \cdot x + t \quad (\text{D.3})$$

- Create an oversampled ESF by shifting each line so that the position of the edge is the same for all lines (Figure D.2d).
- Fit ESF to oversampled edge (Figure D.2d).
- Use the relationship between ESF (eq. D.6) and MTF (eq. D.7) to obtain the MTF. In particular, use the fit parameter λ from the ESF in the Gaussian approximation to obtain the MTF in the Gaussian approximation.¹

Figure D.3 shows the analytical relationship between the optical transfer functions. The following definitions are valid if the smearing is Gaussian. The point spread function (PSF) is defined as [228]:

$$\text{PSF}_G(r) = \frac{1}{\pi\lambda^2} e^{\left(\frac{-r^2}{\lambda^2}\right)} \quad (\text{D.4})$$

The line spread function (LSF) can be written as the PSF in one dimension [228]:

$$\text{LSF}_G(x) = \frac{1}{\pi\lambda} e^{\left(\frac{-x^2}{\lambda^2}\right)} \quad (\text{D.5})$$

The ESF is the integral of the LSF [228]:

$$\text{ESF}_G(x) = 0.5 \cdot \operatorname{erf}\left(\frac{-x}{\lambda}\right) \quad (\text{D.6})$$

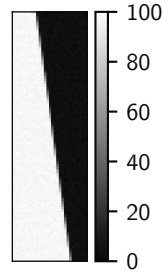
In this framework, the MTF is defined as [228]:

$$\text{MTF}_G(\omega) = e^{\left(\frac{-\pi^2\lambda^2\omega^2}{4}\right)} \quad (\text{D.7})$$

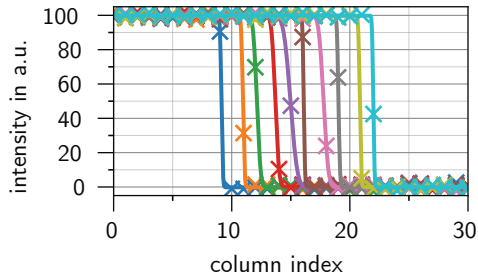
¹The MTF is the Fourier transformation of the LSF. The LSF can be obtained by differentiating the ESF, which can be calculated with the above described routines (Figure D.3). However, the derivative of the ESF can be very noisy, and it turns out that fitting equation D.6 and extracting the parameter λ leads to better results.

The relation between the parameter λ in eq. D.6 and the physical accessible full width at half maximum (FWHM) is described by the following relation:

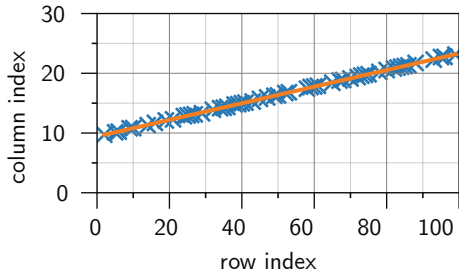
$$\text{FWHM}_G = 2\sqrt{\ln(2)}\lambda \quad (\text{D.8})$$



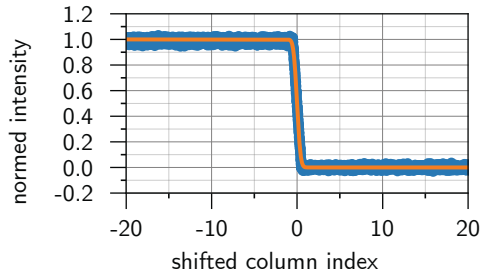
(a) Region of interest (ROI) to obtain the MTF



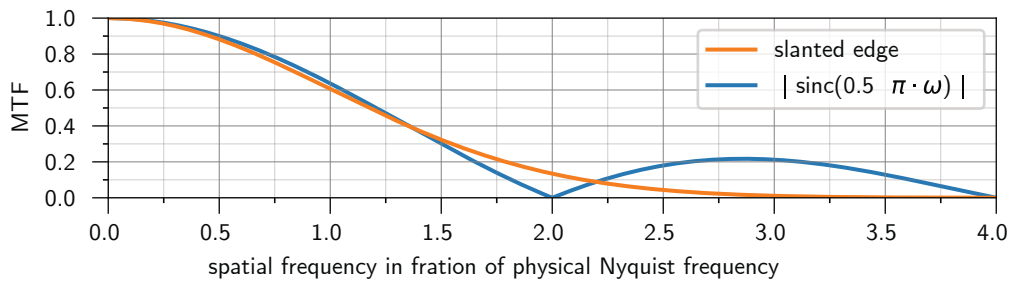
(b) ESFs of the individual rows of the ROI. For clarity, only every tenth row is plotted.



(c) Linear fit to the edge positions obtained from individual ESFs of Figure D.2b.



(d) Oversampled ESF of the overlaid ESF of the individual rows (Figure D.2b) shifted by the displacement obtained from the linear fit in Figure D.2c



(e) MTF in the Gaussian approximation obtained from the slanted edge in Figure D.2a. For comparison, the physical limit of the MTF, which is expressed by the absolute value of the sinc function, is shown [226].

Figure D.2: Approach to obtain the MTF from slanted edge

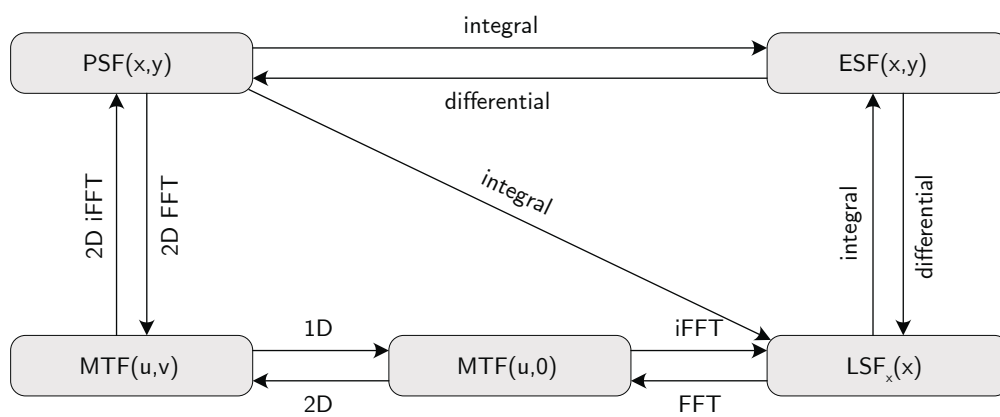


Figure D.3: Relationship between the optical transfer functions. Connection between the point spread function (PSF), error spread function (ESF), line spread function (LSF), and the modulation transfer function (MTF) as a function of one or two spatial frequencies. FFT denotes a Fourier transformation, and iFFT denotes the inverse Fourier transformation. Figure adapted from [227].

D.2 Data Reference

Table D.1: Reference of experimental data for measurements with photons used in Section 10.1.2. In the thesis, the data is referenced by the processing identifier (PID) in the first column. The original data is referenced by the measurement identifier (MID).

PID	MID	energy eV
PID_γ_XRF_1	C16_18_64_210114_001	8048

Table D.2: Reference of experimental data for measurements with photons used in Section 10.1.1. The index from 1 to 154 denotes the different selected positions of the PoE. All other parameters remain unchanged. In the thesis, the data is referenced by the processing identifier (PID) in the first column. The original data is referenced by the measurement identifier (MID).

PID	MID	index	energy eV
PID_γ_SCN_1	MAXYMUS_140720	1-154	1320

Table D.3: Reference of experimental data for measurements with electrons used in Section 10.2. The sample is for all measurements a beam blinker. The frame size of all measurements is 264×264 pixels. In the thesis, the data is referenced by the processing identifier (PID) in the first column. The original data is referenced by the measurement identifier (MID).

PID	MID	energy keV	rate $\cdot 10^{-3} e^-/\text{pix}/\text{frame}$
PID_e_020_1	C16_15_24_140212_40	20	1.01
PID_e_040_1	C16_15_24_140212_34	40	2.05
PID_e_060_1	C16_15_24_140212_30	60	1.33
PID_e_080_1	C16_15_24_140212_27	80	2.47
PID_e_120_1	C16_15_24_140212_23	120	1.32
PID_e_200_1	C16_15_24_140212_18	200	1.16
PID_e_300_1	C16_15_24_140212_13	300	0.92
PID_e_300_2	C16_15_24_140212_14	300	2.44

List of Figures

2.1	Cross-section for the photon in silicon	14
2.2	Schematic view of the photoelectric effect	15
2.3	Feynman diagram of the Compton effect	16
2.4	Klein–Nishina distribution as function of the scattering angle . .	17
2.5	Transferred energy from incoming photon to electron during the Compton effect	19
2.6	Schematic view of the pair production	19
2.7	Absorption for photons in silicon	21
2.8	Schematic view of the interactions between the electron and matter	22
2.9	Energy deposition as a function of the energy	26
2.10	Trajectory length for electrons as a function of the primary energy	27
2.11	Atomic relaxation	28
3.1	Doping of silicon	33
3.2	The p-n junction	34
3.3	Sideward depletion	35
3.4	Schematic view of the pnCCD	37
3.5	Schematic view of the pnCCD readout	38
3.6	Comparison of counting and integrating mode	40
3.7	Energy deposition in counting detector mode	42
4.1	Schematic cut through the approximated model used to calcu- late the electric field and electrostatic potential of the pnCCD .	49
4.2	Electric field and electric potential in a pnCCD as a function of z .	50
4.3	Drift time of the charge cloud into the pixel structure	52
4.4	Shape of the charge cloud as a function of the radius.	55
4.5	Simulated spectrum of deposited energy in silicon for photons. .	57
4.6	Energy distribution of the backscattered particles leaving the silicon.	58

4.7	Averaged single event pattern size as a function of the primary energy.	60
4.8	Projection of the energy deposition of ten randomly selected primary electrons into the x-z-plane	61
4.9	Projection of the energy deposition of ten randomly selected primary electrons into the x-y-plane.	62
4.10	Relative energy deposition as a function of the depth z for various energies of the primary electrons	64
4.11	Relative energy deposition as a function of the lateral coordinate x for various energies of the primary electrons	65
4.12	Lateral range and depth as a function of the energy of the primary electron	66
4.13	Simulated spectrum of deposited energy in silicon for primary electrons	67
4.14	Energy distribution of the backscattered particles leaving the silicon.	68
4.15	Averaged single event pattern size as a function of the primary energy.	70
4.16	Multiplicity for electrons with a primary energy of 300 keV and a pixel size of $48 \times 48 \mu\text{m}^2$	71
4.17	Penetration depth of photons as a function of the primary energy	72
4.18	Mean deposited energy for photons with a low primary energy in a silicon bulk with a thickness of $450 \mu\text{m}$	76
4.19	Size of the charge cloud as a function of the primary photon's energy	77
5.1	Normalized hit-map and resolution map for the center of gravity with correction obtained from a homogeneous illumination . . .	83
5.2	Euclidean distance between the reconstructed PoE and the true PoE as a function of the primary energy of the electron	84
6.1	Conceptual approach of machine learning	86
6.2	Conceptual approach of reinforcement learning	87
6.3	Types of image analysis	88
6.4	Schematic of a neuron	89
6.5	Architecture of a simple neural network	90
6.6	Ideal amount of epochs	97
6.7	Principle of one-hot encoding	98
6.8	Types of classification	99
6.9	Representation of the training data set in the feature space . . .	109
6.10	Optimization process in the parameter space	110

6.11	Computing the output values of a discrete convolution	112
6.12	Two-dimensional convolution	113
6.13	Two-dimensional separable convolution	116
6.14	Maximal pooling	117
6.15	Computing the output values of a transposed convolution	118
6.16	Typical workflow to provide a problem-related neural network	122
7.1	Schematic of the CoNN	127
7.2	Accuracy of CoNN	130
7.3	Normalized hit-map and resolution map for the CoNN obtained from a homogeneous illumination	133
7.4	Reconstruction with CoNN of five chosen PoEs within the pixel structure.	134
7.5	Reconstruction by CoNN with correction of 50 randomly chosen PoEs within the pixel structure	135
7.6	Normalized hit-map and resolution map for the CoNN obtained from a homogeneous illumination	136
8.1	Architecture of the PoENN that contains the u-net module and the embedding	140
8.2	Schematic of the u-net	141
8.3	Architecture of the u-net module	143
8.4	Architecture of the u-net's submodules	144
8.5	Architecture of the necessary embedding modules	145
8.6	Architecture of the optional embedding modules	145
8.7	Schematic of the convolution block	146
8.8	Architecture of the neural network for PoENN with super- resolution extension	147
8.9	Architecture of the SR module and the SR submodule	147
8.10	Pixel structure and grids for different subpixel levels	149
8.11	Schematic viewing of the different upsampling methods in the context of neural networks	152
8.12	Schematic view of a subpixel convolution	152
8.13	Behavior of logarithmic function	156
8.14	Loss function based on the confusion matrix for two pixels	158
8.15	Various statistical validation parameters as a function of the applied threshold	161
8.16	Multiplicity and MTF as a function of the applied threshold	162
8.17	Multiplicity and MTF as a function of the primary electron rate	166
9.1	Dilated convolution	171
9.2	Residual block	172

9.3	Architecture of the neural network for MISR	173
9.4	Architecture of the modified output module for MISR	174
9.5	Architecture of the fusion module and the fusion submodule . . .	174
9.6	Working principles of generative adversarial networks	178
9.7	Two examples for SISR prediction	181
9.8	Input and ground truth examples for MISR prediction	182
9.9	Example for MISR prediction	184
10.1	Experimental setup of the measurement with low energy X-ray .	188
10.2	Two-dimensional histogram of the reconstructed PoEs	189
10.3	Reconstructed beam spot positions	190
10.4	Spatial precision obtained by the CoNN for a low-energy X-ray pencil beam	191
10.5	Experimental setup of the XRF measurement	193
10.6	Light microscope image of the copper grid	194
10.7	Reconstructed image of the copper grid.	195
10.8	Line plot of the copper grid for different reconstruction methods.	196
10.9	Experimental setup of the TEM measurement	197
10.10	Fine structure of the beam blanker	198
10.11	Offset corrected raw data for a primary energy of 300 keV	199
10.12	Summation over the binary output of the proposed neural net- work PoENN	200
10.13	MTF based on the slanted edge method as a function of the spatial frequency	201
10.14	Fine structure of beam blanker	202
10.15	Output of the proposed neural network MISR	203
A.1	Spatial precision, spatial accuracy, and spatial resolution	210
A.2	Maximal spatial resolution for a Gaussian distributed charge cloud	212
A.3	Energy resolution as a function of the mean energy of the peak .	214
A.4	Ratio of out-scattered events for photons as a function of the primary energy	215
A.5	Peak-to-background ratio for photons as a function of the pri- mary energy	216
A.6	Example of the fit for the peak-to-background ratio for photons with a primary energy of 6 keV	217
A.7	Backscattering coefficient for electrons as a function of the pri- mary energy	218
A.8	Peak-to-background ratio for electrons as a function of the pri- mary energy	220
B.1	Event Pattern Analysis	227

B.2	Choice of the secondary threshold that decides whether a pixel contains a signal from an energy deposition or just noise	229
B.3	Simulated pattern pile-up event probability for electrons	233
B.4	Reconstructed event patterns by conventional methods	236
B.5	Schematic view of a correction to the center of gravity method	240
B.6	Probability density function of the center of gravity	242
B.7	Cumulative probability of the center of gravity	243
B.8	Shift between the actual PoE and the center of gravity in the detector space.	244
B.9	Correction for the center of gravity in the analysis space to obtain the actual PoE.	245
B.10	Occurrence of single pattern events	246
B.11	Normalized hit-map and spatial accuracy map for the pure center of gravity method, a too small correction, and a too big correction	247
B.12	Reconstruction with the center of gravity with correction of 50 randomly chosen PoEs within the pixel structure	251
B.13	Reconstruction with the center of gravity method with correction of five chosen PoEs within the pixel structure.	252
B.14	Accuracy for conventional methods	254
B.15	Choice of the coordinate system	258
C.1	Concept of backpropagation	267
D.1	Visual concept of the MTF	282
D.2	Approach to obtain the MTF from slanted edge	285
D.3	Relationship between the optical transfer functions	286

List of Tables

4.1	Average single event pattern size obtained from the conventional event analysis for various primary energies. The ENC is determined by the measured data and used for the simulation. Due to different detector settings, the ENC varies for different primary energies. The data reference is shown in Table D.3. . . .	69
6.1	Commonly used activation functions.	93
8.1	Confusion matrix	154
8.2	Reconstruction rate in Hertz with PoENN [13]. The batch size for 32×32 pixel is 1000 frames, and for 256×256 pixel 100 frames. The edge TPU is not supported since the depth to space operation can be currently not quantized [162]. Quantization is necessary for the deployment to the edge TPU. Since the parallelization of PoENN is limited, the difference between i5 and Xenon is marginal.	167
9.1	Accuracy of SISR and MISR	183
9.2	Reconstruction rate in Hertz with SISR and MISR (using five frames). The batch size is limited by the available memory. A deployment to TPUs is not possible for SISR and MISR due to the architecture of the networks. Currently, TPUs do not support three-dimensional convolution and the merging required by the resnet option [190]. The batch size describes how many images are reconstructed in one step. A larger batch size increases the performance but requires more RAM. The GTX 960 has not enough RAM to perform a prediction for MISR with 256^2 pixels.	185
10.1	Spatial precision determined by a low-energy X-ray pencil beam	192
11.1	Overview of introduced neural networks	206
A.1	Limit of the spatial resolution for binary detector response . . .	212

B.1	Dependencies of the single pattern events	230
B.2	Dependencies of the pattern pile-up probability	232
C.1	Hyper-parameters of the special layers used in neural networks .	272
C.2	Hyper-parameters of the special layers used in neural networks .	273
C.3	Parameters of the compact neural network	274
C.4	Hyper-parameters of the convolutional neural network PoENN and the SR neural networks	275
C.5	Additional hyper-parameters for the SR neural networks	276
C.6	Parameters of the convolutional neural network PoENN (Part 1)	277
C.7	Parameters of the convolutional neural network PoENN (Part 2)	278
C.8	Hyper-parameters of the SR neural network	279
D.1	Reference of experimental data for measurements with photons .	287
D.2	Reference of experimental data for measurements with photons .	287
D.3	Reference of experimental data for measurements with electrons	287

Bibliography

- [1] TSMC. *Process size: 7 nm Technology*. 2021. URL: https://www.tsmc.com/english/dedicatedFoundry/technology/logic/1_7nm (visited on 03/30/2021) (cit. on p. 7).
- [2] Esra Ozcesmeci. *LHC: pushing computing to the limits*. 2021. URL: <https://home.cern/news/news/computing/lhc-pushing-computing-limits> (visited on 03/30/2021) (cit. on p. 7).
- [3] CERN. *Storage: What data to record?* 2020. URL: <https://home.cern/science/computing/storage> (visited on 01/27/2021) (cit. on p. 7).
- [4] Tekin Bicer et al. “Real-Time Data Analysis and Autonomous Steering of Synchrotron Light Source Experiments.” In: *2017 IEEE 13th International Conference on e-Science (e-Science)*. 2017, pp. 59–68. DOI: 10.1109/eScience.2017.53 (cit. on pp. 7, 8).
- [5] Deep Space Network. “DSN Telecommunications Link Design Handbook.” In: *Pasadena, CA: JPL* (2010) (cit. on p. 7).
- [6] Gerhard Lutz et al. *Semiconductor radiation detectors*. Springer, 2007. ISBN: 9783540648598 (cit. on pp. 8, 31–34, 36–39, 51, 52).
- [7] Leonardo Rossi et al. *Pixel detectors: From fundamentals to applications*. Springer, 2006. ISBN: 9783540283331 (cit. on p. 8).
- [8] Andreas Maier et al. *Medical Imaging Systems: An Introductory Guide*. Springer, 2018. ISBN: 9781013271038 (cit. on p. 8).
- [9] Eric Lifshin and Eric Lifshin. *X-ray Characterization of Materials*. Vol. 38. Wiley Online Library, 1999. ISBN: 9783527296576 (cit. on p. 8).
- [10] C Barry Carter and David B Williams. *Transmission electron microscopy: Diffraction, imaging, and spectrometry*. Springer, 2016. ISBN: 9783319266510 (cit. on pp. 8, 29, 197).
- [11] H Ryll et al. “A pnCCD-based, fast direct single electron imaging camera for TEM and STEM.” In: *Journal of Instrumentation* 11.04 (2016), P04006 (cit. on pp. 9, 198, 225, 232, 235, 242).

- [12] S Ihle et al. “Direct measurement of the position accuracy for low energy X-ray photons with a pnCCD.” In: *Journal of Instrumentation* 12.02 (2017), P02005 (cit. on pp. 9, 188, 189, 191, 192, 235).
- [13] Björn Eckert et al. “Electron Imaging Reconstruction for Pixelated Semiconductor Tracking Detectors Using the Approach of Convolutional Neural Networks.” submitted to IEEE, Transaction on Nuclear Science. 2020 (cit. on pp. 9, 141, 153–155, 167, 198–201).
- [14] Ulrich Eberl. “Die Revolution der smarten Maschinen: Künstliche Intelligenz.” In: *Physik Journal April 2020* (2020) (cit. on pp. 9, 10).
- [15] Siri Team. “Hey Siri: An On-device DNN-powered Voice Trigger for Apple’s Personal Assistant.” In: *Apple Machine Learning Journal* 1.6 (2017) (cit. on pp. 9, 85).
- [16] Chi-Feng Wang. *Alexa Voice Service (AVS)*. 2021. URL: <https://developer.amazon.com/en-US/docs/alexa/alexa-voice-service/get-started-with-alexa-voice-service.html> (visited on 01/26/2021) (cit. on p. 9).
- [17] Chenyang Lei and Qifeng Chen. “Fully Automatic Video Colorization With Self-Regularization and Diversity.” In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 3753–3761 (cit. on p. 9).
- [18] Yonghui Wu et al. “Google’s Neural Machine Translation System: Bridging the Gap between Human and Machine Translation.” In: *arXiv preprint arXiv:1609.08144* (2016) (cit. on p. 9).
- [19] Chi-Feng Wang. *DeepL Translator*. 2020. URL: <https://www.deepl.com/translator> (visited on 08/26/2020) (cit. on p. 9).
- [20] Lex Fridman et al. “MIT Advanced Vehicle Technology Study: Large-Scale Naturalistic Driving Study of Driver Behavior and Interaction With Automation.” In: *IEEE Access* 7 (2019), pp. 102021–102038 (cit. on pp. 9, 85).
- [21] John McCarthy et al. “A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence, August 31, 1955.” In: *AI magazine* 27.4 (2006), pp. 12–12 (cit. on p. 9).
- [22] Gordon E Moore et al. *Cramming More Components onto Integrated Circuits*. 1965 (cit. on p. 10).

-
- [23] “Googles Quantenprozessor.” In: *Physik in unserer Zeit* 52.1 (2021). DOI: <https://doi.org/10.1002/piuz.202170101>. eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/piuz.202170101>. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1002/piuz.202170101> (cit. on p. 10).
- [24] TensorFlow Quantum. *TensorFlow*. 2020. URL: <https://www.tensorflow.org/quantum> (visited on 01/26/2021) (cit. on p. 10).
- [25] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep learning.” In: *nature* 521.7553 (2015), pp. 436–444 (cit. on p. 10).
- [26] Pankaj Mehta et al. “A high-bias, low-variance introduction to Machine Learning for physicists.” In: *Physics reports* 810 (2019), pp. 1–124 (cit. on pp. 10, 85).
- [27] Yann LeCun and Corinna Cortes. “MNIST handwritten digit database.” In: (2010). URL: <http://yann.lecun.com/exdb/mnist/> (cit. on p. 10).
- [28] Alina Kuznetsova et al. “The Open Images Dataset V4: Unified image classification, object detection, and visual relationship detection at scale.” In: *International Journal of Computer Vision* (2020), pp. 1–26 (cit. on p. 10).
- [29] Alexander Selvikvåg Lundervold and Arvid Lundervold. “An overview of deep learning in medical imaging focusing on MRI.” In: *Zeitschrift für Medizinische Physik* 29.2 (2019), pp. 102–127 (cit. on p. 10).
- [30] Junyoung Park et al. “Computed tomography super-resolution using deep convolutional neural network.” In: *Physics in Medicine & Biology* 63.14 (2018), p. 145011 (cit. on p. 10).
- [31] Ji He, Yongbo Wang, and Jianhua Ma. “Radon Inversion via Deep Learning.” In: *IEEE transactions on medical imaging* 39.6 (2020), pp. 2076–2087 (cit. on p. 10).
- [32] Özgün Çiçek et al. “3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation.” In: *International conference on medical image computing and computer-assisted intervention*. Springer. 2016, pp. 424–432 (cit. on p. 10).
- [33] Todd C Hollon et al. “Near Real-Time Intraoperative Brain Tumor Diagnosis Using Stimulated Raman Histology and Deep Neural Networks.” In: *Nature medicine* 26.1 (2020), pp. 52–58 (cit. on p. 10).
- [34] Martin Erdmann. “Erkenntnisgewinn durch moderne datengetriebene Methoden: Deep Learning.” In: *Physik Journal April 2020* (2020) (cit. on p. 11).

- [35] Jonathan Tompson et al. “Accelerating Eulerian Fluid Simulation With Convolutional Networks.” In: *International Conference on Machine Learning*. PMLR. 2017, pp. 3424–3433 (cit. on p. 11).
- [36] Dan Guest, Kyle Cranmer, and Daniel Whiteson. “Deep Learning and Its Application to LHC Physics.” In: *Annual Review of Nuclear and Particle Science* 68 (2018), pp. 161–181 (cit. on p. 11).
- [37] Dimitri Bourilkov. “Machine and deep learning applications in particle physics.” In: *International Journal of Modern Physics A* 34.35 (2019), p. 1930019 (cit. on p. 11).
- [38] H. Kolanoski and N. Wermes. *Teilchendetektoren: Grundlagen und Anwendungen*. Springer Berlin Heidelberg, 2016. ISBN: 9783662453506. URL: <https://books.google.de/books?id=L5uFCwAAQBAJ> (cit. on pp. 13, 15–24, 28, 29, 170, 183, 211–213, 222, 256–259, 261).
- [39] Glenn F Knoll. *Radiation Detection and Measurement*. John Wiley & Sons, 2010. ISBN: 9780470131480 (cit. on pp. 13–16, 23, 25, 29).
- [40] Albert C Thompson, Douglas Vaughan, et al. *X-ray data booklet*. Vol. 8. 4. Lawrence Berkeley National Laboratory, University of California Berkeley, CA, 2001 (cit. on pp. 13, 29, 55, 58, 59, 67, 193).
- [41] D E Cullen, J H Hubbell, and L Kissel. “EPDL97: the evaluated photo data library ‘97 version.” In: (Sept. 1997). DOI: 10.2172/295438. URL: <https://www.osti.gov/biblio/295438> (cit. on pp. 14, 21, 45).
- [42] International Atomic Energy Agency - Nuclear Data Section. “EPICS2014: Electron photon interaction cross sections (version 2014), rev. 1.” In: *Vienna, Austria, Tech. Rep. IAEA-NDS-218* (2015) (cit. on pp. 14, 21, 45).
- [43] J. D. Hunter. “Matplotlib: A 2D Graphics Environment.” In: *Computing in Science & Engineering* 9.3 (2007), pp. 90–95. DOI: 10.1109/MCSE.2007.55 (cit. on p. 14).
- [44] Albert Einstein. “Über einem die Erzeugung und Verwandlung des Lichtes betreffenden heuristischen Gesichtspunkt.” In: *Annalen der physik* 4 (1905) (cit. on p. 14).
- [45] R.C. Alig, S Bloom, and C.W. Struck. “Scattering by ionization and phonon emission in semiconductors.” In: *Physical Review B* 22.12 (1980), p. 5565 (cit. on pp. 15, 36).
- [46] F Scholze and M Procop. “Modelling the response function of energy dispersive X-ray spectrometers with silicon detectors.” In: *X-Ray Spectrometry: An International Journal* 38.4 (2009), pp. 312–321 (cit. on pp. 15, 36).

-
- [47] Arthur H. Compton. “A Quantum Theory of the Scattering of X-rays by Light Elements.” In: *Phys. Rev.* 21 (5 May 1923), pp. 483–502. DOI: 10.1103/PhysRev.21.483. URL: <https://link.aps.org/doi/10.1103/PhysRev.21.483> (cit. on p. 16).
- [48] Oskar Klein and Yoshio Nishina. “The Scattering of Light by Free Electrons according to Dirac’s New Relativistic Dynamics.” In: *Nature* 122.3072 (1928), pp. 398–399 (cit. on p. 16).
- [49] Paul Adrien Maurice Dirac. “The quantum theory of the electron.” In: *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character* 117.778 (1928), pp. 610–624 (cit. on p. 18).
- [50] Hanno Krieger. *Grundlagen der Strahlungsphysik und des Strahlenschutzes*. Vol. 6. Springer, 2019. ISBN: 9783662605844 (cit. on pp. 21, 22, 29).
- [51] Ernest Rutherford. “LXXIX. The scattering of α and β particles by matter and the structure of the atom.” In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 21.125 (1911), pp. 669–688 (cit. on p. 23).
- [52] Don Groom. *Critical energy for electrons and positrons*. URL: https://pdg.lbl.gov/2017/AtomicNuclearProperties/critical_energy.html (visited on 03/13/2021) (cit. on p. 23).
- [53] John David Jackson. *Classical Electrodynamics*. 1998. ISBN: 978-0471309321 (cit. on p. 23).
- [54] Stephen M Seltzer and Martin J Berger. “Bremsstrahlung spectra from electron interactions with screened atomic nuclei and orbital electrons.” In: *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms* 12.1 (1985), pp. 95–134 (cit. on p. 23).
- [55] H.W. Koch and J.W. Motz. “Bremsstrahlung Cross-Section Formulas and Related Data.” In: *Reviews of modern physics* 31.4 (1959), p. 920 (cit. on p. 23).
- [56] Walter Heitler. *The quantum theory of radiation*. Courier Corporation, 1984. ISBN: 9780486645582 (cit. on p. 24).
- [57] GEANT Collaboration, S Agostinelli, et al. “GEANT4—a simulation toolkit.” In: *Nucl. Instrum. Meth. A* 506.25 (2003) (cit. on pp. 24, 44).

- [58] Claude Leroy and Pier-Giorgio Rancoita. “Particle interaction and displacement damage in silicon devices operated in radiation environments.” In: *Reports on Progress in Physics* 70.4 (2007), p. 493 (cit. on p. 24).
- [59] GEANT Collaboration et al. “Physics reference manual.” In: *Version: geant4* 10.9 (2019) (cit. on pp. 24, 45).
- [60] Gert Moliere. “Theorie der Streuung schneller geladener Teilchen I. Einzelstreuung am abgeschirmten Coulomb-Feld.” In: *Zeitschrift für Naturforschung A* 2.3 (1947), pp. 133–145 (cit. on p. 24).
- [61] Gert Moliere. “Theorie der Streuung schneller geladener Teilchen II Mehrfach- und Vielfachstreuung.” In: *Zeitschrift für Naturforschung A* 3.2 (1948), pp. 78–97 (cit. on p. 24).
- [62] HW Lewis. “Multiple Scattering in an Infinite Medium.” In: *Physical review* 78.5 (1950), p. 526 (cit. on pp. 24, 45).
- [63] Hans Bethe. “Zur Theorie des Durchgangs schneller Korpuskularstrahlen durch Materie.” In: *Annalen der Physik* 397.3 (1930), pp. 325–400 (cit. on p. 24).
- [64] Hans Bethe. “Bremsformel für Elektronen relativistischer Geschwindigkeit.” In: *Zeitschrift für Physik* 76.5-6 (1932), pp. 293–299 (cit. on p. 24).
- [65] Martin J Berger, J S Coursey, and M A Zucker. *ESTAR, PSTAR, and ASTAR: computer programs for calculating stopping-power and range tables for electrons, protons, and helium ions (version 1.21)*. Tech. rep. 1999 (cit. on p. 25).
- [66] DC Joy and S Luo. “An empirical stopping power relationship for low-energy electrons.” In: *Scanning* 11.4 (1989), pp. 176–180 (cit. on p. 25).
- [67] Suicho Luo, Xiao Zhang, and David C Joy. “Experimental determinations of electron stopping power at low energies.” In: *Radiation Effects and Defects in Solids* 117.1-3 (1991), pp. 235–242 (cit. on p. 25).
- [68] Lev Davidovich Landau. “On the energy loss of fast particles by ionization.” In: *J. Phys.* 8 (1944), pp. 201–205 (cit. on pp. 27, 222).
- [69] Henning Ryll. “Direct detection of electrons with the pnCCD for applications in transmission electron microscopy.” PhD thesis. University of Siegen, 2018 (cit. on pp. 28, 37, 38, 40, 66, 67, 218, 226, 232, 233, 282).
- [70] Egon Wiberg. *Lehrbuch der anorganischen Chemie*. Walter de Gruyter GmbH & Co KG, 2019. ISBN: 9780123526519 (cit. on p. 31).

-
- [71] Siegfried Hunklinger. *Festkörperphysik*. Oldenbourg Verlag, 2009. ISBN: 9783486755589 (cit. on p. 32).
- [72] Massimo Rudan. *Physics of Semiconductor Devices*. Springer, 2015. ISBN: 9781493911516 (cit. on p. 33).
- [73] Emilio Gatti and Pavel Rehak. “Semiconductor drift chamber — An application of a novel charge transport scheme.” In: *Nuclear Instruments and Methods in Physics Research* 225.3 (1984), pp. 608–614 (cit. on pp. 35, 36).
- [74] Ugo Fano. “Ionization Yield of Radiations. II. The Fluctuations of the Number of Ions.” In: *Physical Review* 72.1 (1947), p. 26 (cit. on p. 36).
- [75] L Strüder et al. “The European photon imaging camera on XMM-Newton: the pn-CCD camera.” In: *Astronomy & Astrophysics* 365.1 (2001), pp. L18–L26 (cit. on p. 37).
- [76] Julia Schmidt. “A study of the charge collection, storage and processing in pixelated semiconductor detectors.” PhD thesis. University of Siegen, 2017 (cit. on p. 37).
- [77] Sven Herrmann et al. “CAMEX readout ASICs for pnCCDs.” In: *2008 IEEE Nuclear Science Symposium Conference Record*. 2008, pp. 2952–2957. DOI: 10.1109/NSSMIC.2008.4774983 (cit. on p. 40).
- [78] J Kemmer and Gerhard Lutz. “New detector concepts.” In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 253.3 (1987), pp. 365–377 (cit. on p. 41).
- [79] Shuai Leng et al. “Photon-counting Detector CT: System Design and Clinical Applications of an Emerging Technolog.” In: *Radiographics* 39.3 (2019), pp. 729–743 (cit. on p. 41).
- [80] Xavier Llopart et al. “Timepix, a 65k programmable pixel readout chip for arrival time, energy and/or photon counting measurements.” In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 581.1-2 (2007), pp. 485–494 (cit. on p. 41).
- [81] Piotr Otfinowski. “Spatial resolution and detection efficiency of algorithms for charge sharing compensation in single photon counting hybrid pixel detectors.” In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 882 (2018), pp. 91–95 (cit. on p. 41).
- [82] G McMullan, AR Faruqi, and R Henderson. “Direct Electron Detectors.” In: *Methods in Enzymology* 579 (2016), pp. 1–17 (cit. on p. 41).

- [83] Christopher Booth. *Detection Technologies for Cryo-Electron Microscopy*. S2C2 Workshop – Cryo-EM Training. 2020 (cit. on pp. 41, 42).
- [84] *C++*. URL: <https://en.cppreference.com> (visited on 01/28/2022) (cit. on p. 44).
- [85] *Python*. URL: www.python.org (visited on 01/28/2022) (cit. on p. 44).
- [86] The pandas development team. *pandas-dev/pandas: Pandas*. Version latest. Feb. 2020. DOI: 10.5281/zenodo.3509134. URL: <https://doi.org/10.5281/zenodo.3509134> (cit. on p. 44).
- [87] Charles R. Harris et al. “Array programming with NumPy.” In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. DOI: 10.1038/s41586-020-2649-2. URL: <https://doi.org/10.1038/s41586-020-2649-2> (cit. on p. 44).
- [88] GEANT4 Collaboration et al. “Guide For Physics Lists.” In: *Version: geant4, Release 10.5 9.0* (2019) (cit. on p. 44).
- [89] *Low Energy Electromagnetic Physics - Livermore*. URL: <https://geant4.web.cern.ch/node/1619> (visited on 06/26/2021) (cit. on p. 44).
- [90] S. T. Perkins et al. “Tables and graphs of atomic subshell and relaxation data derived from the LLNL Evaluated Atomic Data Library (EADL), Z = 1–100.” In: (Oct. 1991). DOI: 10.2172/10121422. URL: <https://www.osti.gov/biblio/10121422> (cit. on p. 45).
- [91] J.H. Scofield. “Radiative Transitions.” In: *Atomic Inner-Shell Processes 1* (1975), pp. 265–292 (cit. on p. 45).
- [92] *Low Energy Electromagnetic Physics - Atomic Deexcitation*. URL: <https://geant4.web.cern.ch/node/1620> (visited on 06/26/2021) (cit. on p. 45).
- [93] GEANT4 Collaboration et al. “Physics Reference Manual.” In: *Version: geant4, Release 10.5 9.0* (2019) (cit. on p. 45).
- [94] Gergely Soti et al. “Performance of Geant4 in simulating semiconductor particle detector response in the energy range below 1 MeV.” In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 728 (2013), pp. 11–22 (cit. on p. 46).
- [95] Nils Kimmel. “Analysis of the charge collection process in solid state X-ray detectors.” PhD thesis. University of Siegen, 2009 (cit. on pp. 46, 48, 49, 53).

-
- [96] G. Weidenspointner. *Documentation for HLLDetSim*. 2011 (cit. on pp. 46, 49, 51, 53, 54).
- [97] Shunji Goto. “Response functions of a Si (Li) detector for photon energies from 1 to 10 keV.” In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 333.2-3 (1993), pp. 452–457 (cit. on p. 47).
- [98] Stefanie Granato. “The response of silicon PNCCD sensors with aluminum on-chip filter to visible light, UV-and X-ray radiation.” PhD thesis. University of Siegen, 2012 (cit. on pp. 47, 49, 52, 59, 215, 226, 228).
- [99] Leslie Greengard. “The Numerical Solution of the N-Body Problem.” In: *Computers in physics* 4.2 (1990), pp. 142–152 (cit. on p. 47).
- [100] Daniel Müllner. “Modern hierarchical, agglomerative clustering algorithms.” In: *arXiv preprint arXiv:1109.2378* (2011) (cit. on p. 48).
- [101] “Scikit-learn: Machine Learning in Python.” In: () (cit. on pp. 48, 98).
- [102] C Canali et al. “Electron and hole drift velocity measurements in silicon and their empirical relation to electric field and temperature.” In: *IEEE Transactions on Electron Devices* 22.11 (1975), pp. 1045–1047 (cit. on p. 51).
- [103] Giampiero Ottaviani. “Correction to” Electron and hole drift velocity measurements in silicon and their empirical relation to electric field and temperatures.” In: *IEEE Transactions on Electron Devices* 23.9 (1976), pp. 1113–1113 (cit. on p. 51).
- [104] Canali Jacoboni et al. “A review of some charge transport properties of silicon.” In: *Solid-State Electronics* 20.2 (1977), pp. 77–89 (cit. on p. 51).
- [105] Emilio Gatti et al. “Dynamics of electrons in drift detectors.” In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 253.3 (1987), pp. 393–399 (cit. on pp. 52, 53).
- [106] Ansgar Steland. *Basiswissen Statistik*. Springer, 2016. ISBN: 9783642372018 (cit. on p. 54).
- [107] EJ Schioppa et al. “Study of Charge Diffusion in a Silicon Detector Using an Energy Sensitive Pixel Readout Chip.” In: *IEEE transactions on Nuclear Science* 62.5 (2015), pp. 2349–2359 (cit. on p. 55).

- [108] Ludwig Reimer. *Transmission Electron Microscopy: Physics of Image Formation and Microanalysis*. Vol. 36. Springer, 2013. ISBN: 9780387347585 (cit. on pp. 65, 67).
- [109] Thilo Michel et al. “A fundamental method to determine the signal-to-noise ratio (SNR) and detective quantum efficiency (DQE) for a photon counting pixel detector.” In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 568.2 (2006), pp. 799–802 (cit. on p. 70).
- [110] Stéfán van der Walt et al. “scikit-image: image processing in Python.” In: *PeerJ* 2 (June 2014), e453. ISSN: 2167-8359. DOI: 10.7717/peerj.453. URL: <https://doi.org/10.7717/peerj.453> (cit. on pp. 78, 175).
- [111] Ansgar Steland. *Basiswissen Statistik*. Springer, 2007. ISBN: 9783642026669 (cit. on p. 82).
- [112] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. <http://www.deeplearningbook.org>. MIT Press, 2016 (cit. on pp. 85, 105, 115).
- [113] Michael Nielsen. *Neural Networks and Deep Learning*. Vol. 25. Determination Press San Francisco, CA, 2015 (cit. on pp. 85, 90, 91, 105, 267).
- [114] J Hurwitz and D Kirsch. *Machine Learning Machine Learning For Dummies®*, IBM Limited Edition. 2018. ISBN: 9781119454953 (cit. on pp. 85, 87, 88).
- [115] François Chollet et al. *Keras*. <https://keras.io>. 2015 (cit. on p. 85).
- [116] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from [tensorflow.org](https://www.tensorflow.org). 2015. URL: <https://www.tensorflow.org/> (cit. on p. 85).
- [117] Florian Schroff, Dmitry Kalenichenko, and James Philbin. “Facenet: A unified embedding for face recognition and clustering.” In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 815–823 (cit. on p. 85).
- [118] IBM Cloud Education. *Artificial Intelligence (AI)* (cit. on p. 86).
- [119] Gopinath Rebala, Ajay Ravi, and Sanjay Churiwala. *An Introduction to Machine Learning*. Springer, 2019. ISBN: 9783030157296 (cit. on pp. 86, 89).
- [120] Charles W Anderson. “Learning to control an inverted pendulum using neural networks.” In: *IEEE Control Systems Magazine* 9.3 (1989), pp. 31–37 (cit. on p. 88).

-
- [121] Linda Shapiro. *Computer Vision and Image Processing*. Academic Press, 1992. ISBN: 9780323141567 (cit. on p. 89).
- [122] Zhong-Qiu Zhao et al. “Object Detection With Deep Learning: A Review.” In: *IEEE transactions on neural networks and learning systems* 30.11 (2019), pp. 3212–3232 (cit. on p. 89).
- [123] Nikhil R Pal and Sankar K Pal. “A review on image segmentation techniques.” In: *Pattern recognition* 26.9 (1993), pp. 1277–1294 (cit. on p. 89).
- [124] Ben Kröse et al. “An introduction to Neural Networks.” In: (1993) (cit. on p. 89).
- [125] V Zhou. *Machine Learning for Beginners: An Introduction to Neural Networks*. 2019. URL: <https://towardsdatascience.com/machine-learning-for-beginners-an-introduction-to-neural-networks-d49f22d238f9> (visited on 06/19/2021) (cit. on p. 89).
- [126] Jürgen Schmidhuber. “Deep learning in neural networks: An overview.” In: *Neural networks* 61 (2015), pp. 85–117 (cit. on p. 91).
- [127] Li Deng and Dong Yu. *Deep Learning: Methods and Applications*. Tech. rep. MSR-TR-2014-21. Microsoft, May 2014. URL: <https://www.microsoft.com/en-us/research/publication/deep-learning-methods-and-applications/> (cit. on p. 91).
- [128] W.J. Zhang et al. “On Definition of Deep Learning.” In: *2018 World Automation Congress (WAC)*. 2018, pp. 1–5. DOI: 10.23919/WAC.2018.8430387 (cit. on p. 91).
- [129] Kaiming He et al. “Deep Residual Learning for Image Recognition.” In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778 (cit. on pp. 91, 171).
- [130] TensorFlow. *TensorFlow: API Documentation*. 2020. URL: https://www.tensorflow.org/api_docs (visited on 09/18/2020) (cit. on pp. 91, 93, 96, 102, 103, 111, 119, 150, 156, 176, 177, 183, 272, 273).
- [131] Sachin Kumar. *Deep Learning Made Easy: Part 3: Activation Functions, Parameters and Hyperparameters and Weight Initialization*. 2020. URL: <https://towardsdatascience.com/deep-learning-made-easy-activation-functions-parameters-and-hyperparameters-and-weight-c7bcfeb9af24> (visited on 10/19/2020) (cit. on p. 92).
- [132] Ludwig Fahrmeir et al. *Statistik: Der Weg zur Datenanalyse*. Springer-Verlag, 2016. ISBN: 9783540212324 (cit. on p. 94).

- [133] Lothar Sachs. *Angewandte Statistik: Statistische Methoden und ihre Anwendungen*. Springer-Verlag, 2018. ISBN: 9780387088136 (cit. on p. 94).
- [134] TensorFlow. *Keras: API Documentation*. 2020. URL: <https://keras.io/api/> (visited on 09/18/2020) (cit. on pp. 96, 99, 100, 102, 119, 160).
- [135] *Three Type Of Classification Tasks*. URL: <https://www.microsoft.com/en-us/research/uploads/prod/2017/12/40250.jpg> (visited on 02/25/2021) (cit. on p. 99).
- [136] Prakhar Mishra Mishra. *Multi-Label classification with One-Vs-Rest strategy*. URL: <https://prakhartechviz.blogspot.com/2019/02/multi-label-classification-python.html> (visited on 02/25/2021) (cit. on p. 99).
- [137] Emilio Soria Olivas et al. *Handbook of research on machine learning applications and trends: Algorithms, methods, and techniques: Algorithms, methods, and techniques*. IGI Global, 2009. ISBN: 9781605667669 (cit. on p. 101).
- [138] TensorFlow. *TensorFlow: Transfer learning and fine-tuning*. 2020. URL: https://www.tensorflow.org/tutorials/images/transfer_learning?hl=en (visited on 09/18/2020) (cit. on p. 101).
- [139] Sebastian Ruder. “An overview of gradient descent optimization algorithms.” In: *arXiv preprint arXiv:1609.04747* (2016) (cit. on p. 102).
- [140] Diederik P Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization.” In: *arXiv preprint arXiv:1412.6980* (2014) (cit. on pp. 102–105, 266).
- [141] Léon Bottou. “Large-Scale Machine Learning with Stochastic Gradient Descent.” In: *Proceedings of COMPSTAT’2010*. Springer, 2010, pp. 177–186 (cit. on p. 103).
- [142] David Saad. “Online Algorithms and Stochastic Approximations.” In: *Online Learning* 5 (1998), pp. 6–3 (cit. on p. 103).
- [143] Léon Bottou. “Stochastic Gradient Descent Tricks.” In: *Neural networks: Tricks of the trade*. Springer, 2012, pp. 421–436 (cit. on p. 103).
- [144] Herbert Robbins and David Siegmund. “A convergence theorem for non negative almost supermartingales and some applications.” In: *Optimizing methods in statistics*. Elsevier, 1971, pp. 233–257 (cit. on p. 103).
- [145] Vitaly Bushaev. *Adam — latest trends in deep learning optimization*. 2018. URL: <https://towardsdatascience.com/adam-latest-trends-in-deep-learning-optimization-6be9a291375c> (visited on 10/01/2020) (cit. on p. 104).

-
- [146] Ilja N Bronstein et al. *Taschenbuch der Mathematik*. Vol. 1. Springer-Verlag, 2012. ISBN: 9783817120086 (cit. on pp. 104, 105, 259).
- [147] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. “Learning representations by back-propagating errors.” In: *nature* 323.6088 (1986), pp. 533–536 (cit. on p. 106).
- [148] Roger A Horn. “The Hadamard Product.” In: *Proc. Symp. Appl. Math.* Vol. 40. 1990, pp. 87–169 (cit. on pp. 106, 155, 156).
- [149] Leslie N. Smith. “Cyclical Learning Rates for Training Neural Networks.” In: *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2017, pp. 464–472. DOI: 10.1109/WACV.2017.58 (cit. on p. 108).
- [150] Keiron O’Shea and Ryan Nash. “An Introduction to Convolutional Neural Networks.” In: *arXiv preprint arXiv:1511.08458* (2015) (cit. on pp. 109, 110).
- [151] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. “Understanding of a convolutional neural network.” In: *2017 International Conference on Engineering and Technology (ICET)*. 2017, pp. 1–6. DOI: 10.1109/ICEngTechnol.2017.8308186 (cit. on p. 111).
- [152] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-Net: Convolutional Networks for Biomedical Image Segmentation.” In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Ed. by Nassir Navab et al. Cham: Springer International Publishing, 2015, pp. 234–241. ISBN: 978-3-319-24574-4 (cit. on pp. 111, 140–143).
- [153] Vincent Dumoulin and Francesco Visin. “A guide to convolution arithmetic for deep learning.” In: *arXiv preprint arXiv:1603.07285* (2016) (cit. on p. 111).
- [154] Chris. *Understanding separable convolutions*. 2020. URL: <https://www.machinecurve.com/index.php/2019/09/23/understanding-separable-convolutions> (visited on 08/26/2020) (cit. on p. 114).
- [155] Chi-Feng Wang. *A Basic Introduction to Separable Convolutions*. 2020. URL: <https://towardsdatascience.com/a-basic-introduction-to-separable-convolutions-b99ec3102728> (visited on 09/28/2020) (cit. on p. 114).
- [156] Jianxin Wu. “Introduction to Convolutional Neural Networks.” In: *National Key Lab for Novel Software Technology. Nanjing University. China* 5 (2017), p. 23 (cit. on p. 115).

- [157] Nitish Srivastava et al. “Dropout: A Simple Way to Prevent Neural Networks from Overfitting.” In: *Journal of Machine Learning Research* 15 (2014), pp. 1929–1958. URL: <http://jmlr.org/papers/v15/srivastava14a.html> (cit. on p. 116).
- [158] Matthew D. Zeiler et al. “Deconvolutional networks.” In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 2010, pp. 2528–2535. DOI: 10.1109/CVPR.2010.5539957 (cit. on p. 117).
- [159] Thom Lane. *Transposed Convolutions explained with... MS Excel!* 2020. URL: <https://medium.com/apache-mxnet/transposed-convolutions-explained-with-ms-excel-52d13030c7e8> (visited on 08/26/2020) (cit. on p. 117).
- [160] Tim Salimans and Diederik P Kingma. “Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks.” In: *arXiv preprint arXiv:1602.07868* (2016) (cit. on p. 119).
- [161] Sergey Ioffe and Christian Szegedy. “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.” In: *International conference on machine learning*. PMLR. 2015, pp. 448–456 (cit. on pp. 119, 120).
- [162] TensorFlow. *TensorFlow Lite API Reference*. 2020. URL: https://www.tensorflow.org/lite/api_docs?hl=en (visited on 09/18/2020) (cit. on pp. 120, 122, 137, 167).
- [163] Coral. *Coral: Documentation*. 2020. URL: <https://coral.ai/docs/> (visited on 09/18/2020) (cit. on pp. 121–123, 137).
- [164] Tianfeng Chai and Roland R Draxler. “Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature.” In: *Geoscientific model development* 7.3 (2014), pp. 1247–1250 (cit. on pp. 129, 176).
- [165] NVIDIA® GeForce® GTX 960. NVIDIA Corporation. 2015 (cit. on pp. 137, 167, 185).
- [166] Raspberry Pi foundation. *Raspberry Pi Documentation*. 2020. URL: <https://www.raspberrypi.org/documentation/> (visited on 10/23/2020) (cit. on p. 137).
- [167] Andreas Maier et al. “A gentle introduction to deep learning in medical image processing.” In: *Zeitschrift für Medizinische Physik* 29.2 (2019), pp. 86–101 (cit. on p. 139).

-
- [168] Shruti Jadon. “A survey of loss functions for semantic segmentation.” In: *2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*. 2020, pp. 1–7. DOI: 10.1109/CIBCB48159.2020.9277638 (cit. on p. 140).
- [169] Björn Eckert et al. “Super-resolution for x-ray applications with pixelated semiconductor tracking detectors using convolutional neural networks.” In: *Software and Cyberinfrastructure for Astronomy VI*. Vol. 11452. International Society for Optics and Photonics. SPIE, 2020, pp. 733–741. DOI: 10.1117/12.2576067. URL: <https://doi.org/10.1117/12.2576067> (cit. on pp. 144, 148, 152).
- [170] Chao Dong et al. “Learning a Deep Convolutional Network for Image Super-Resolution.” In: *Computer Vision – ECCV 2014*. Ed. by David Fleet et al. Cham: Springer International Publishing, 2014, pp. 184–199. ISBN: 9783319105932 (cit. on p. 146).
- [171] Dianyuan Han. “Comparison of Commonly Used Image Interpolation Methods.” In: *Proceedings of the 2nd international conference on computer science and electronics engineering*. Atlantis Press. 2013. DOI: <https://doi.org/10.2991/iccsee.2013.391> (cit. on p. 149).
- [172] Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically based rendering: From theory to implementation*. Morgan Kaufmann, 2016 (cit. on p. 149).
- [173] Don P Mitchell and Arun N Netravali. “Reconstruction filters in computer-graphics.” In: *ACM Siggraph Computer Graphics* 22.4 (1988), pp. 221–228 (cit. on p. 150).
- [174] Philip Mayne Woodward. *Probability and Information Theory, with Applications to Radar: International Series of Monographs on Electronics and Instrumentation*. Vol. 3. Elsevier, 2014. ISBN: 9781483169644 (cit. on p. 150).
- [175] Claude E Duchon. “Lanczos filtering in one and two dimensions.” In: *Journal of applied meteorology* 18.8 (1979), pp. 1016–1022 (cit. on p. 150).
- [176] BN Madhukar and R Narendra. “Lanczos Resampling for the Digital Processing of Remotely Sensed Images.” In: *Proceedings of International Conference on VLSI, Communication, Advanced Devices, Signals & Systems and Networking (VCASAN-2013)*. Springer. 2013, pp. 403–411. ISBN: 9788132215240 (cit. on p. 150).

- [177] Augustus Odena, Vincent Dumoulin, and Chris Olah. “Deconvolution and Checkerboard Artifacts.” In: *Distill* (2016). DOI: 10.23915/distill.00003. URL: <http://distill.pub/2016/deconv-checkerboard> (cit. on pp. 150, 151).
- [178] Andrew Aitken et al. “Checkerboard artifact free sub-pixel convolution: A note on sub-pixel convolution, resize convolution and convolution resize.” In: *arXiv preprint arXiv:1707.02937* (2017) (cit. on p. 151).
- [179] Wenzhe Shi et al. “Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network.” In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 1874–1883 (cit. on p. 151).
- [180] Yusuke Sugawara, Sayaka Shiota, and Hitoshi Kiya. “Super-Resolution Using Convolutional Neural Networks Without Any Checkerboard Artifacts.” In: *2018 25th IEEE International Conference on Image Processing (ICIP)*. 2018, pp. 66–70. DOI: 10.1109/ICIP.2018.8451141 (cit. on p. 151).
- [181] Wenzhe Shi et al. “Is the deconvolution layer the same as a convolutional layer?” In: *arXiv preprint arXiv:1609.07009* (2016) (cit. on p. 151).
- [182] Augustus Odena. *Deconvolution and Checkerboard Artifacts*. 2020. URL: <https://distill.pub/2016/deconv-checkerboard/> (visited on 08/31/2020) (cit. on pp. 153, 163).
- [183] Yuma Kinoshita and Hitoshi Kiya. “Checkerboard-Artifact-Free Image-Enhancement Network Considering Local and Global Features.” In: *2020 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE. 2020, pp. 1139–1144 (cit. on p. 153).
- [184] Yuri Sousa Aurelio et al. “Learning from Imbalanced Data Sets with Weighted Cross-Entropy Function.” In: *Neural Processing Letters* 50.2 (2019), pp. 1937–1949 (cit. on p. 155).
- [185] Francisco J Valverde-Albacete and Carmen Peláez-Moreno. “100 % Classification Accuracy Considered Harmful: The Normalized Information Transfer Factor Explains the Accuracy Paradox.” In: *PloS one* 9.1 (2014), e84217 (cit. on p. 160).
- [186] Tejumade Afonja. *Accuracy Paradox*. URL: <https://towardsdatascience.com/accuracy-paradox-897a69e2dd9b> (visited on 03/15/2021) (cit. on p. 160).

-
- [187] David M.W. Powers. “Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation.” In: *arXiv preprint arXiv:2010.16061* (2020) (cit. on p. 160).
- [188] Hao Huang et al. “Maximum F1-Score Discriminative Training Criterion for Automatic Mispronunciation Detection.” In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 23.4 (2015), pp. 787–797 (cit. on p. 160).
- [189] Jeremy West, Dan Ventura, and Sean Warnick. “Spring research presentation: A theoretical foundation for inductive transfer.” In: *Brigham Young University, College of Physical and Mathematical Sciences* 1.08 (2007) (cit. on p. 164).
- [190] *Google Colab*. Google LLC. URL: <https://colab.research.google.com/notebooks/intro.ipynb> (cit. on pp. 167, 185).
- [191] *Tensor Processing Units*. Google LLC. 2021. URL: <https://cloud.google.com/tpu/docs/system-architecture?hl=en> (cit. on p. 167).
- [192] *NVIDIA® Tesla® P100*. NVIDIA Corporation. 2016 (cit. on pp. 167, 185).
- [193] Wenming Yang et al. “Deep Learning for Single Image Super-Resolution: A Brief Review.” In: *IEEE Transactions on Multimedia* 21.12 (2019), pp. 3106–3121 (cit. on p. 169).
- [194] Weisheng Dong et al. “Denoising Prior Driven Deep Neural Network for Image Restoration.” In: *IEEE transactions on pattern analysis and machine intelligence* 41.10 (2018), pp. 2305–2318 (cit. on p. 169).
- [195] Normand J Beaudry and Renato Renner. “An intuitive proof of the data processing inequality.” In: *arXiv preprint arXiv:1107.0740* (2011) (cit. on p. 169).
- [196] Jingwei Zhang, Tongliang Liu, and Dacheng Tao. “An Information-Theoretic View for Deep Learning.” In: *arXiv preprint arXiv:1804.09060* (2018) (cit. on p. 170).
- [197] Michal Kawulok et al. “Deep Learning for Multiple-Image Super-Resolution.” In: *IEEE Geoscience and Remote Sensing Letters* 17.6 (2019), pp. 1062–1066 (cit. on pp. 170, 173).
- [198] Zhihao Wang, Jian Chen, and Steven C.H. Hoi. “Deep Learning for Image Super-resolution: A Survey.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2020), pp. 1–1. DOI: 10.1109/TPAMI.2020.2982166 (cit. on p. 170).

- [199] Jane Bromley et al. “Signature Verification using a ”Siamese” Time Delay Neural Network.” In: *Advances in neural information processing systems*. Vol. 6. 1994, pp. 737–744 (cit. on p. 173).
- [200] *Perceptual Loss Functions*. URL: <https://deepai.org/machine-learning-glossary-and-terms/perceptual-loss-function> (visited on 05/20/2022) (cit. on p. 176).
- [201] Zhou Wang et al. “Image quality assessment: from error visibility to structural similarity.” In: *IEEE transactions on image processing* 13.4 (2004), pp. 600–612 (cit. on pp. 177, 183).
- [202] Roz Combley. *Cambridge business English dictionary*. Cambridge University Press, 2011. ISBN: 9780521122504 (cit. on p. 177).
- [203] Manfred J Holler, Gerhard Illing, and Stefan Napel. *Einführung in die Spieltheorie*. Vol. 8. Springer, 2019 (cit. on p. 178).
- [204] Ian J Goodfellow et al. “Generative Adversarial Networks.” In: *arXiv preprint arXiv:1406.2661* (2014) (cit. on pp. 178, 179).
- [205] David Warde-Farley and Ian Goodfellow. “Adversarial Perturbations of Deep Neural Networks.” In: *Perturbations, Optimization, and Statistics* 311 (2016) (cit. on p. 179).
- [206] Hyungrok Ham, Tae Joon Jun, and Daeyoung Kim. “Unbalanced GANs: Pre-training the Generator of Generative Adversarial Network using Variational Autoencoder.” In: *arXiv preprint arXiv:2002.02112* (2020) (cit. on p. 179).
- [207] *Collimating Polycapillary*. Fischer Technology Inc. 2022 (cit. on p. 193).
- [208] Inc. Structure Probe. *SPI Supplies Brand TEM Grids G150*. 2021. URL: <https://www.2spi.com/item/2011c-xa/grids-cu-square> (visited on 06/01/2021) (cit. on p. 194).
- [209] *University of Bremen: Electron Microscopy, Germany, Bremen*. (Accessed on: 2021, May 27). URL: <https://www.uni-bremen.de/ifp/arbeitsgruppen/elektronenmikroskopie-ag-rosenauer/laboratorien-und-geraete> (cit. on p. 197).
- [210] Teng Wang et al. “An Overview of FPGA Based Deep Learning Accelerators: Challenges and Opportunities.” In: *2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*. 2019, pp. 1674–1681. DOI: 10.1109/HPCC/SmartCity/DSS.2019.00229 (cit. on p. 208).

-
- [211] Stefan Aschauer. “There is never enough dynamic range-DEPFET active pixel sensors with analog signal compression.” PhD thesis. Technische Universität München, 2014 (cit. on p. 213).
- [212] T Tabata, R Ito, and S Okabe. “An empirical equation for the backscattering coefficient of electrons.” In: *Nuclear instruments and methods* 94.3 (1971), pp. 509–513 (cit. on p. 219).
- [213] P Jethwa et al. “When is pile-up important in the XMM-Newton EPIC cameras?” In: *Astronomy & Astrophysics* 581 (2015), A104 (cit. on pp. 220, 233).
- [214] J Ballet. “Pile-up on X-ray CCD instruments.” In: *Astronomy and Astrophysics Supplement Series* 135.2 (1999), pp. 371–381 (cit. on pp. 220, 233).
- [215] Pauli Virtanen et al. “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python.” In: *Nature Methods* 17 (2020), pp. 261–272. DOI: <https://doi.org/10.1038/s41592-019-0686-2> (cit. on p. 228).
- [216] Dask Development Team. *Dask: Library for dynamic task scheduling*. 2016. URL: <https://dask.org> (cit. on p. 228).
- [217] Robert Ritz. Private Communication. Munich, Germany, 2021 (cit. on p. 228).
- [218] Intel[®]. *Intel[®] Core[™]i5-8400 Prozessor*. 2017. URL: <https://ark.intel.com/content/www/de/de/ark/products/126687/intel-core-i5-8400-processor-9m-cache-up-to-4-00-ghz.html> (cit. on p. 228).
- [219] Paul Mooney. *Getting the most out of a counting direct detector*. EMBO Croy School. 2019 (cit. on p. 229).
- [220] E Belau et al. “Charge collection in silicon strip detectors.” In: *Nuclear Instruments and Methods in Physics Research* 214.2-3 (1983), pp. 253–260 (cit. on pp. 235, 242, 249).
- [221] R Turchetta. “Spatial resolution of silicon microstrip detectors.” In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 335.1-2 (1993), pp. 44–58 (cit. on p. 249).
- [222] Johan Ludwig William Valdemar Jensen. “Sur les fonctions convexes et les inégalités entre les valeurs moyennes.” In: *Acta mathematica* 30.1 (1906), pp. 175–193 (cit. on p. 263).

- [223] *Photography — Electronic still picture imaging — Resolution and spatial frequency responses*. Standard. Geneva, CH: International Organization for Standardization, 2017 (cit. on p. 281).
- [224] Albert Abraham Michelson. *Studies in Optics*. Courier Corporation, 1995. ISBN: 9780226523880 (cit. on p. 281).
- [225] Françoise Viallefont-Robinet et al. “Comparison of MTF measurements using edge method: towards reference data set.” In: *Optics express* 26.26 (2018), pp. 33625–33648 (cit. on p. 281).
- [226] Daniel J Schroeder. *Astronomical optics*. Elsevier, 1999. ISBN: 9780126298109 (cit. on p. 285).
- [227] Ying Cheng, Hongwei Yi, and Xinlong Liu. “Lunar-edge based on-orbit modulation transfer function (MTF) measurement.” In: *AOPC 2017: Space Optics and Earth Imaging and Space Navigation*. Vol. 10463. International Society for Optics and Photonics. 2017, 104631U (cit. on p. 286).
- [228] G McMullan et al. “Detective quantum efficiency of electron area detectors in electron microscopy.” In: *Ultramicroscopy* 109.9 (2009), pp. 1126–1143 (cit. on p. 283).

Acknowledgements

I would like to express my gratitude and appreciation to all the people who contributed to the success of this PhD thesis and supported me during the journey. Special thanks goes to

- Lothar Strüder, for mentoring and providing me the opportunity to work on this fantastic topic and this PhD thesis.
- Stefan Aschauer, Thomas Zabel, and Petra Majewski for always having an open ear, the stimulating discussions and being such great colleagues.
- Peter Holl for supporting me at IT issues and the interesting discussions.
- Barbara Tietze and Daniela Fischer for the organization and taking care of the issues around the work.
- Robert Ritz and Martin Huth for supporting me with their knowledge of pnCCD.
- Alois Bechteler for support in the lab.
- Jeffrey Davis for supporting and embolding me to begin using artificial intelligence.
- Peter Nellist and Emma Hedley with the University of Oxford for supporting me with their knowledge of TEM instrumentation and for the various measured data.
- and all coworkers from PNSensor and PNDetector I have not mentioned by name.

and especially to

- my wife Laura Eckert, for keeping me up and supporting me during the hardest time of my PhD.

- my parents Ute and Guido Eckert, for supporting me my whole life unconditionally in everything I did. A special thanks to Guido for waking my interest in physics, explaining me so much and the many talks at the kitchen table.
- my sister Tina Eckert, for showing me how important a compensation through sport is and being such a great sister.
- my family in law Thauer, who included me immediately in their family and where I am always welcome.
- my friends for taking care of distraction, the great off-work projects and being always there for me.
- my grandparents for the many phone calls, especially during the Covid-19 lockdown.