Event reconstruction for ground-based γ-ray particle detectors with Transformer networks

Master‘s Thesis in Physics

Presented by
Markus Pirke
16.09.2024

Erlangen Centre for Astroparticle Physics (ECAP)
Friedrich-Alexander Universität Erlangen-Nürnberg

Supervisor: Prof. Dr. Christopher van Eldik

Abstract

Ground-based γ-ray particle detectors are dedicated observatories to observe large parts of the sky at the highest energies of the electro-magnetic spectrum. Such a detector is compromised out of an array of detection units, which directly detect secondary particle of air showers. Commonly, these detectors are based on the water-Cherenkov technique, where individual detection units are cylindrical tanks filled up with purified water, equipped with photo multiplier tubes.

A future planned ground-based particle detector array for γ-ray astronomy is called the Southern Wide-field Gamma-ray Observatory. As the name suggests, it will be located in the Southern Hemisphere, in Chile, and will operate in an energy range from 100 GeV up to PeV scale.

These detectors directly sample secondary particles within air showers. That means during the reconstruction of air showers, only information is available from one horizontal slice of the whole shower. For this reason the event reconstruction, in such an observatory, is an extremly challenging task.

This work introduces a method for event reconstruction based on Transformer networks. A Transformer is a special neural network, originally introduced within the field of natural language processing. Nowadays, Transformers are mainly used for large language models, but have also been successfully applied in other areas of research.

The performance of this approach will be evaluated for energy reconstruction, γ/hadron separation and direction reconstruction. In the end the these tasks are evaluated again, in a scenario where detector aging effects are taken into account during the reconstruction.

Contents

Chapter 1

Introduction

"[Noone] should escape our universities without knowing how little [they know]."

J. Robert Oppenheimer

Astronomy in the 21st century is one of the most active and fascinating fields in modern science. Over centuries, we have only known that there are stars and planets in the sky and their positions in the sky could be measured. We knew about lunar phases and eclipses. In 1842, the French philosophe Auguste Comte published a book called "Cours de Philosophie Positive", in which he wrote: "We can never learn their [(stars)] internal constitution, nor, in regard to some of them, how heat is absorbed by their atmosphere" [1][2].

Since the 20th century, we know why stars shine, and their composition can be inferred by studying their spectra. In 1929, Edwin Hubble [3] discovered that the universe is expanding, and in 1992 the first exoplanet was discovered [4].
Now, we also have methods to study the non-thermal universe, which refers to processes in the universe that emit radiation which is not produced by thermal mechanisms. We can measure photons with energies that are more than 10^{12} times higher than those of visible light.
At present, we have discovered more than 240 sources [5] "of Very High Energy (VHE, $\geq$ 100 GeV) or TeV gamma-radiation[, which] belongs to the most remarkable achievements of the last [few] decade[s] in astrophysics" [6].

The first dedicated ground-based observatory was the Whipple observatory, a so called Imaging Atmospheric Cherenkov Telescope (IACT) with a 10 m reflector and 37 pixel camera [7]. With that instrument is was possible, for the first time, to detect γ-rays from the Crab nebula in the VHE regime, with good statistical significance. Following Whipple, many more IACTs were built, which convincingly demonstrated the potential of this technique.

Complementary instruments (to IACTs) for ground based γ-ray astronomy are particle detector arrays. With the High-Altitude Water Cherenkov (HAWC) γ-ray observatory and the Large High Altitude Air Shower Observatory (LHAASO) we already have two such detectors successfully operating. Both are located in the Northern Hemisphere.

The Southern Wide-Field Gamma-Ray Observatory (SWGO) will be a future ground based particle detector array, based on the water-Cherenkov technique, which will be located in the Southern Hemisphere. The aim of SWGO will be to study "galactic and extragalactic particle accelerators, [monitor] the transient sky at very high energies, [probe] particle physics beyond the Standard Model, and the characterization of the cosmic ray flux" [8]. To achieve these science goals, we need efficient methods to analyze the complex data that is gathered by such an experiment in vast amounts.

In other scientific areas, where a lot of data is involved, machine learning and deep learning methods are already often state of the art (SOTA). For example in a long-standing problem in biology, called the protein folding problem, tremendous advances were possible with the help of a machine-learning algorithm called Alphafold [9]. But also in physics, machine learning algorithms are routinely used. For instance at the biggest particle collider in the world, the LHC, machine learning techniques are used for jet classification or for the generation of fast simulations [10]. Recently, machine learning also assisted in a search for high-energy astrophysical neutrinos from within the Milky Way [11].

Out of all possible machine learning algorithms, a certain subset, those that are based on neural networks, seem to be the most promising ones. A particular type of neural network was introduced in 2017 by a group at Google, called the Transformer [12]. The Transformer architecture has revolutionized Natural Language Processing (NLP) and is forming the backbone for every Large Language Model (LLM), like for example ChatGPT, where GPT stands for Generative Pretained Transformer [13]. It is not only successful in NLP. Transformer based networks are also routinely used in other fields of research with SOTA performance.

In this thesis we aim to investigate whether the Transformer architecture can be effectively used for a γ-ray event reconstruction in ground based particle detectors, using the example of SWGO. In the second chapter, we start with an overview of γ-ray astronomy and look at sources of TeV γ-rays. Following this, we discuss air showers and ground based detection methods with the aim of reconstructing such showers. We will provide detailed information of SWGO in the third chapter. In chapter four, we offer a general overview of the field of machine learning, beginning with a historical example in astronomy. We then explore supervised learning, gradient-based optimization, backpropagation, neural networks and how they are trained. The chapter concludes with an introduction to the Transformer neural network. In the fifth chapter, we discuss how the

Transformer architecture needs to be modified for ground-based particle arrays and introduce a efficient method for training Transformers. Equipped with this knowledge, we will examine the use of Transformer-based models for γ-ray event reconstruction. Finally, in the last chapter, we summarize the findings of this work and provide an outlook on future research directions.

Chapter 2

Gamma-ray astronomy

"The results of the present observations seem most likely to be explained by the assumption that radiation of very high penetrating power enters from above into our atmosphere"

Viktor Hess

"The main driving force for VHE gamma-ray astronomy was initially the search for the sources of the charged cosmic rays, while now, after the discovery of many sources, the interests have shifted to general astrophysics questions." [14].

Cosmic rays were first discovered by Viktor Hess in 1912 during his famous balloon experiments [15]. Following this discovery, many experiments, which had the aim of studying the cosmic radiation reaching us here on Earth, were set up. These observations led to the so-called all-particle cosmic ray energy spectrum, which is shown in Figure 2.1. It highlights that the (differential) flux of particles reaching us at Earth, follows roughly a power law behavior of $\approx E^{-\alpha}$ with $\alpha \approx 2.7$. The x-axis is spanning many orders of magnitude in energy, reaching up to the EeV scale.

A closer look at the spectrum reveals that at lower energies ($\approx$ MeV) the spectrum flattens due to the Earth's magnetic field, which effectively shields us at these energies. Up to the so-called "knee" at roughly 3 PeV, it is widely believed that these cosmic rays originate from within our galaxy. At the knee the spectrum steepens up the "ankle" $\approx$ 3 EeV. So all in all we can clearly see deviations from a simple power law behavior.

To determine the sources of cosmic rays, they need to be traced back to their origin. However, as cosmic rays possess an electric charge, they are deflected by ambient magnetic fields in the interstellar medium (ISM). The latter is traversed by cosmic rays on their way towards Earth. Consequently, their arrival direction

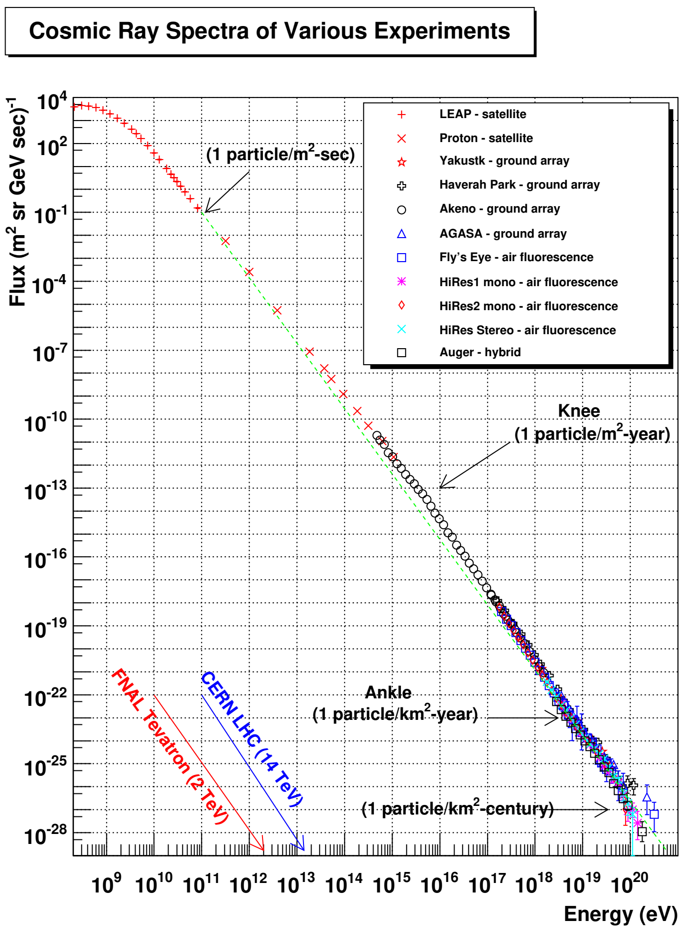

Figure 2.1: The all-particle cosmic ray energy spectrum from various experiments. The spectrum roughly follows a power law (shown in green). Taken from [16].

does no longer point back to their astronomical source.[1]

The field of γ-ray astronomy started in the 1950s, based on predictions of a diffuse γ-ray emission from the decay of π^0-mesons from cosmic ray interactions with the interstellar medium [17]. Experiments in γ-ray astronomy can be separated into two categories. The first experiments were space-based experiments. The satellite Explorer XI, which was launched on April 27 1961, was the first to detect 22 extraterrestrial gamma rays [18]. Such satellite instruments operate in the energy range from $\approx$ 20 MeV to $\approx$ 300 GeV, which is the energy range where for example the Fermi Large Area Telescope (LAT) instrument currently operates [19]. The LAT has a wide field of view (2.4 sr) and can observe the entire sky. In Figure 2.2 we can see the results of a five year all sky survey from Fermi-LAT. Sources of γ-rays above 1 GeV are highlighted in red to bright yellow. The drawback to the LAT, and to space-based instruments in general, is

[1]Unless at the ultra high energy regime.

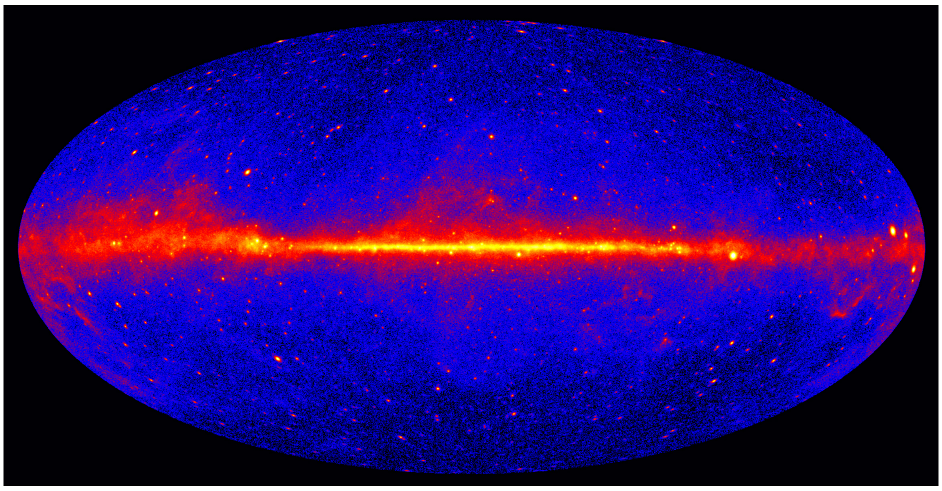

Figure 2.2: Sky map in γ-rays. The image contains data taken by the LAT telescope during five years of observations. Red to yellow bright spots are sources of γ-rays above 1 GeV. Taken from [20].

the lack in detection area. The LAT is composed out of 16 tracker modules, each with an area of $8.95 \times 8.95 \mathrm{cm}^2$, resulting together in an area smaller than $1\,\mathrm{m}^2$ [19].

The other category is ground-based γ-ray astronomy. At energies well beyond the GeV scale, the flux of γ-rays has dropped too much and space based experiments become impractical due to their limited detection area. On Earth we have large detection areas, but direct detection is no longer possible.
The gamma-rays need to traverse the atmosphere on their way to the ground-based detectors. At these energies we can use the atmosphere as a calorimeter, because particle cascades are initiated when these γ-rays interact with air nuclei. With instruments on the ground, we can observe these showers.

In the following, we will explore sources of γ-rays in the TeV regime and discuss how they emit photons at such high energies. Afterwards, we examine at the interaction of γ-rays with our atmosphere and at the resulting showers of secondary particles. At last, detection techniques will be introduced for the measurement of air showers.

2.1 Sources of γ-rays in TeV-regime

The Crab nebula was already mentioned in the introduction, as the first identified VHE γ-ray source. It first appeared as a guest star[2] to Chinese astronomers in the year 1054 [22]. Only later in 1942, astronomers presented studies which showed, that the Crab nebula is connected to this 1054 AD star explosion [23].

[2]So called by Chinese astronomers.

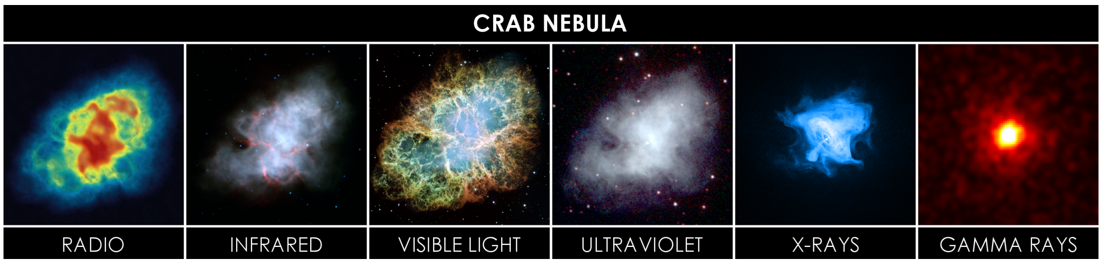

Figure 2.3: The Crab nebula seen in different parts of the electromagnetic spectrum. Taken from [21].

It was thus the first documented supernova in history [24].

The Crab is one of the most studied object in the sky and is observed throughout the whole electromagnetic spectrum as shown in Figure 2.3 [25]. Due to its high luminosity it is one of our prime laboratories for the study of non-thermal processes [24]. The Crab nebula is a highly complex object and has now been studied over centuries, with still many unanswered open questions. In general the Crab nebula is a Supernova Remnant (SNR)[3] with a pulsar, the Crab pulsar, sitting at its center. This pulsar is source of energy for the system. Inside the SNR, surrounding the pulsar, is a so-called Pulsar Wind Nebula (PWN). We will discuss these three types of objects, as sources of TeV γ-rays in the following.

Supernova remnants

When massive stars reach the end of their life cycle we have evidence that they are undergoing a supernova explosion. For a brief moment a Type II supernova can illuminate the universe with light of 100 billion suns, yet this radiative energy is only a small part. Most of the energy of the explosion itself is rapidly expelled into kinetic energy of the outer layers of the star being blasted out into space [22]. The resulting structure of this explosion is what we call a SNR. Shock waves in young supernova remnants are promising candidates for the acceleration of cosmic rays up to the "knee".
If SNRs indeed accelerate cosmic rays, we should also observe them as sources of non-thermal emission, when electrons and hadrons at high energies interact with the surrounding medium [26].

When electrons follow a circular orbit at relativistic speeds in a magnetic field **synchrotron radiation** is emitted.[4] It is polarized and for highly relativistic particles beamed in the direction of motion of the particle. In a SNR the spec-

[3]However as pointed out here [23], the Crab nebula should not be thought of as a SNR rather as a pulsar wind nebulae.

[4]Not only electrons will emit this radiation, also protons or other charged particles. However it is strongly suppressed for more massive particle.

trum of synchrotron radiation starts in the radio regime and extends into the x-ray band [26].

Higher energies can be reached when these energetic electrons interact with low energy photons from the cosmic microwave background (CMB) or stellar radiation fields via a process called **inverse Compton scattering**. Photons that are up-scattered by this process can reach energies in the γ-ray regime.
We have looked at the emission caused by electrons, but if also high energetic hadrons are present, an additional source of γ-radiation should be observable. In so called "pp interactions" these hadrons scatter inelastically with surrounding nuclei, resulting in spallation products X' and in pions π.
The charged pions will decay within 26 ns to muons and neutrinos, whereas the neutral pions will decay into two photons with energies in the γ-ray regime

$$\pi^0 \rightarrow \gamma + \gamma. \tag{2.1}$$

The exact ratio of gamma rays produced from a hadronic background via neutral pion decay or from inverse Compton scattering in SNRs is still an open question.

We have presently observed about thirty SNRs in γ-rays with different instruments. In the GeV regime with satellites, by IACTs in the TeV regime and with particle detector arrays like LHAASO and HAWC at even higher energies. Whether SNRs contribution to galactic cosmic ray acceleration plays a dominant part is still an open debate. New generation γ-ray instruments, like Cherenkov Telescope Array (CTA) and SWGO may give us new hints in this interesting discussion [26].

Pulsars
Following a type II supernova explosion, we will either get a neutron star or a black hole, depending on the mass of the exploded star. When it had a mass between 1.4 and 3 solar masses we get a neutron star. Neutron stars are objects where the quantum mechanical degeneracy pressure is just big enough to withstand the inward gravitational pressure.
The first neutron star was discovered in 1967 as an object that was emitting periodic radio pulses. Since then we have discovered nearly 3000 such sources, which we now call pulsars ("pulsating radio sources") [27].

A simple model, the so called "light house model" [27] can explain the observed radio pulses. If the magnetic dipole moment of the neutron star is misaligned with axis of rotation of the neutron star, we get a beam of particles and radiation in direction of the magnetic poles. As the neutron star rotates, this beam sweeps over the sky. If at some point this beam points towards Earth, we can observe it at regular intervals, for example pulses of radiation from the Crab pulsar reaching us every 33.6 ms [24]. Observations of the Crab pulsar, also show, in addition to radio pulses, pulsed VHE radiation [6].

Pulsar Wind Nebulae
The Crab Pulsar is slowing down and this loss of rotational energy is carried away by relativistic winds. Accelerated leptons within the pulsar wind nebula interact with magnetic fields and also with radiation fields, producing synchrotron radiation from radio to x-rays and high energy γ-rays [6]. Overall, one suspects that the energy output by the nebula is at least by an order of magnitude greater than that of the pulsar [24].
Pulsar wind nebulae are thought to be the most effective Galactic objects capable of producing VHE γ-rays [6].

A more exhaustive list of possible γ-ray sources can be found in [6].

2.2 Extensive air showers

Earth is surrounded by a thick layer of air, our atmosphere. It is a mixture of gases with 78 % nitrogen, 21 % oxygen, 0.9 % argon, 0.04 % carbon dioxide, and trace gases. Additionally, at an altitude between 15 km and 35 km in the so called stratosphere is a layer of ozone, which absorbs a big part of the ultraviolet radiation from the sun.

We already mentioned that for ground-based γ-ray astronomy we use the atmosphere as a calorimeter, because at energies starting at the GeV scale, γ-rays, similar to high energy cosmic rays, produce showers of secondary particles, during interactions with air nuclei. These showers are also called Extensive Air Showers (EAS) and were first discovered by Auger and Kohlhörster in the 1930s [28]. In the following we will introduce electro-magnetic showers induced by γ-rays and hadronic showers induced by cosmic rays.
Electro-magnetic showers.
To determine important parameters of these showers, a simplified model developed by Heitler [29] may be used. In Heitler's model a simple branching structure is assumed, where only photons, electrons e^- and positrons e^+ take part [30]. This branching structure is shown in Figure 2.4. In this model, photons only interact via **pair production** and the electrons and positrons via **bremsstrahlung**. At a energy above a few MeV, pair production is the most probable interaction mechanism of γ-rays with matter. If the γ-ray has an energy exceeding 1.022 MeV, twice the rest mass of the electron, a photon in the field of nuclei, can produce an e^+e^- pair. For charged particles, bremsstrahlung is the main energy loss mechanism for relativistic particles. At lower energies, energy losses due to collisions are favored [31].

The theory of bremstrahlung was first dicussed by Bethe and Heitler in 1934 [32]. They showed that the average energy radiated by an electron of energy E,

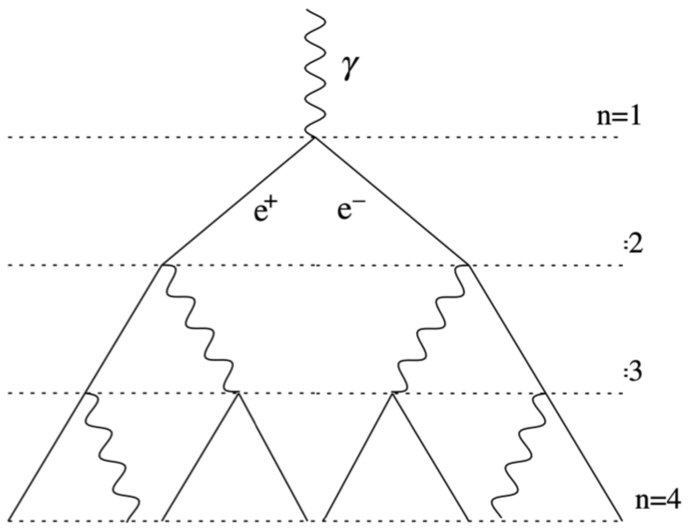

Figure 2.4: Sketch of the Heitler model for electronic showers. Here one assumes simple branching model, where after a fixed distance photons, electrons and positrons interact, via pair production and bremstrahlung. Taken from [30].

(if $E >> mc^2$) per unit length along its path is

$$-(dE/dx) \sim \frac{E}{m^2}, \tag{2.2}$$

where the energy loss depends linearly on the energy of the particle and is indirectly proportional to the mass squared. This means that for electrons the energy loss through bremstrahlung is greater than for protons or heavier nuclei.

If the density of the medium changes in which the electron travels, it is convenient to express lengths in units of the so called slant depth X. The slant depth tells us how much mass we have traversed per unit area. From the density ρ we can calculate it as

$$X = \int \rho\, ds. \tag{2.3}$$

In the integral we integrate along the path of the electron. If we simplify Equation 2.2 like this

$$\frac{\mathrm{d}E}{\mathrm{d}X} = -E/X_0, \tag{2.4}$$

we get the following solution for the energy loss

$$E(X) = E_0 e^{-E/X_0}. \tag{2.5}$$

This tells us that after a fixed distance X_0, called the attenuation length, the initial energy of the electron is reduced by a factor of $1/e$. For electrons in the atmosphere $X_0 = 37\,\mathrm{g\,cm^{-2}}$ (roughly 310 m at normal atmospheric conditions)

[31]. The following will be based on [30].

The Heitler model assumes, that the radiation lengths for both bremsstrahlung and electron positron pair production are equal. After certain fixed distances $d = \log(2)X_o$ particles undergo two-body splittings, resulting in a splitting structure as shown in Figure 2.4. Thus the energy is distributed equally at each splitting to the two outgoing particles. The total shower size, which is the total number of secondary particles, after n splitting lengths d is

$$N = 2^n = e^{n \ln 2} = e^{X_n/X_0} \text{ with } X_n = nX_0 \ln 2 \tag{2.6}$$

the total atmospheric depth that the shower traveled through the atmosphere. At the critical energy E_c (in air is 85 MeV), bremsstrahlung and pair production, are less favored. Instead energy losses due to collisions and ionization dominate. At this threshold particle production stops and the maximum number of particles in the cascade is reached. We call this point the penetration depth X_{max}, which we can write as a function of the initial energy E_0, like this

$$E_{\text{crit}} = E(X_{\text{max}}) = E_0 e^{-X_{\text{max}}/X_0} \Rightarrow X_{\text{max}} = X_0 \ln E_0/E_{\text{crit}}. \tag{2.7}$$

As a result of Equation 2.7, the atmospheric depth at which the shower maximum is reached is proportional to the logarithm of the primary γ-ray's energy. Even for high energy γ-rays X_{max} is still at very high altitudes in the atmosphere. For that reason particle detectors, which observe the secondary directly need to be located at high altitudes.

Hadronic showers.
Whereas in electro-magnetic showers the physics is well understood in terms of QED, hadronic showers are more complex, as the interactions in such a shower are governed by QCD and weak interactions. QCD, at such high energy scales, is not properly understood, which makes modeling of hadronic showers a complex endeavor.

However, we can still gain some insights by a simple branching model, called the Heitler-Matthew model [30]. Instead of an incoming photon, we assume an incoming proton. This proton will interact with some air molecule and produce pions and spallation products. The neutral pion will decay into photons and these will initiate an electronic shower. With that at each stage roughly one third of the energy is transitioned into an electromagnetic component of the shower.

The charged pions will decay into muons, which interact on very rarely. In principle their decay time is rather short, nevertheless, because they are produced at high energies, due to time dilation they are able to reach ground.

The third component of a hadronic shower are the secondary hadrons. These

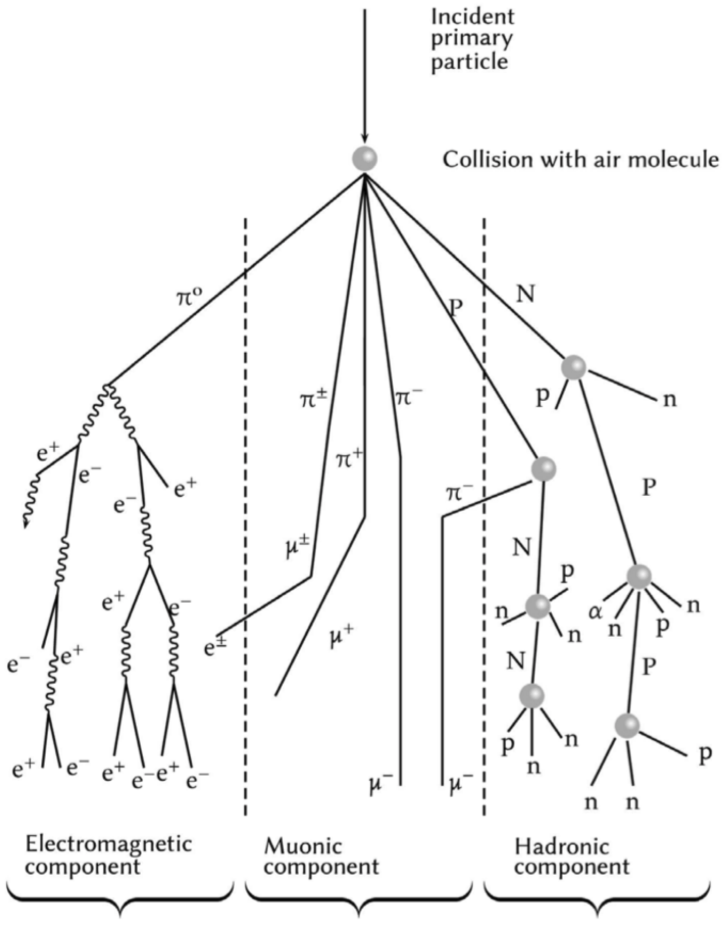
Incident primary particle
Collision with air molecule
π⁰
π±
π⁻
π⁺
μ±
μ⁺
μ⁻
e⁺
e⁻
e±
N
P
p
n
α
Electromagnetic component
Muonic component
Hadronic component

Figure 2.5: Sketch of a hadronic shower. On the left we can see the electromagnetic compoenent of the shower initiated by a neutral pion. In the middle we have the muonic component from the decay of charged pions. To the right the hadronic component is depicted. Taken from [34].

are typically produced with a transverse momentum leading to hadrons at large angles away from the shower axis [33]. In Figure 2.5 we see a sketch of a hadronic shower, with its three different components. The characteristic features of these three components will become important later, when we try to distinguish between electronic and hadronic showers.

The Heitler and Heitler-Matthew model are only a rough approximation, which are useful for getting a feel of what is happening. A more accurate approach is based on Monte Carlo simulations, where physical processes are simulated individually. An example of a simulated electronic-magnetic shower induced by a γ-ray is shown in Figure 2.6. Here we can also clearly see the different lateral profile of the electronic shower on the left, centered around the shower axis, and the hadronic shower on the right, which has a larger lateral extend.

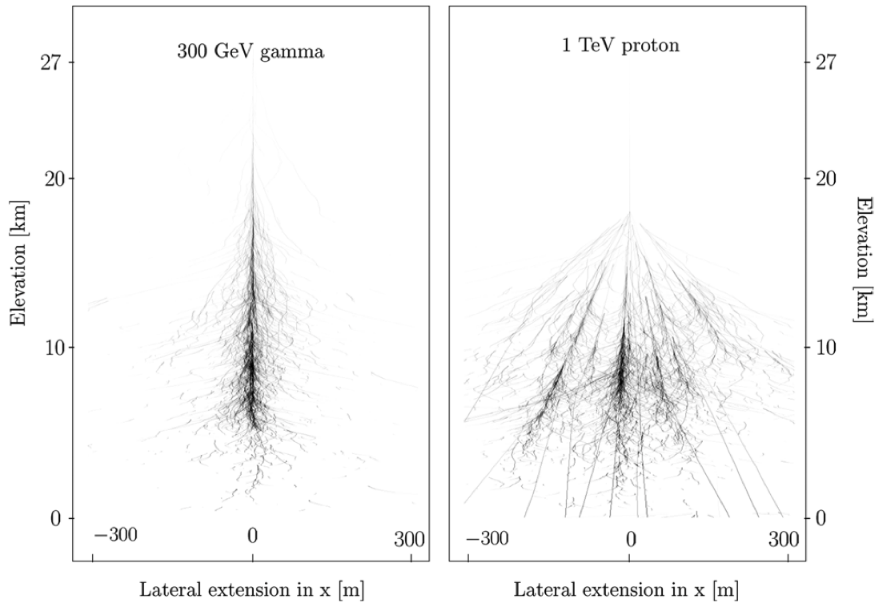

Figure 2.6: Simulated air showers. On the left a shower initiated by a 300 GeV γ-ray and on the right one from a proton at 1 TeV. Taken from [35].

2.3 Instruments for ground-based γ-ray astronomy

From the air shower we need to infer the direction and the energy of the primary particle and the type of particle which initiated the shower [36]. The last point is subtle. In satellite measurements we can directly detect the particles and we can suppress the background of cosmic rays with anticoincidence detectors. This is not possible for ground-based detection, as both cosmic rays and γ-rays initiate the showers. There we need methods to distinguish γ-ray induced showers from the overwhelming, by a factor of 10^4, background of cosmic rays.

We have two complementary methods, IACTs and particle detector arrays. Both of these methods are reconstructing the showers by detecting Cherenkov radiation.

Cherenkov radiation
It was well known, for a long time, that planes or projectiles, which move at velocities greater than the speed of sound produce conical waves, so-called Mach waves [37].
In 1934, Pavel Cherenkov, for the first time, observed light that was produced by a similar mechanism. A charged particle that moves through a dielectric medium, faster than the phase velocity of light of the medium, will produce a conical wave of light.

The Cherenkov angle is determined by the following equation

$$\cos(\theta) = \frac{1}{\beta n}, \tag{2.8}$$

where $\beta = {}^{v}/_{c_0}$, is the particle's speed over the speed of light in vacuum. In 1958, Pavel Cherenkov received the Nobel prize for the discovery together with Frank and Tam, who established the theory of Cherenkov radiation [38].

Imaging Atmospheric Cherenkov Telescopes

IACTs are among the biggest optical telescopes on Earth. With these we try to collimate the Cherenkov radiation produced within the atmosphere from secondary particles in EAS onto a camera. This technique has been etablished as the most sensitive method for observing γ-rays above $\approx 50\,\text{GeV}$ [39]. In Figure 2.7

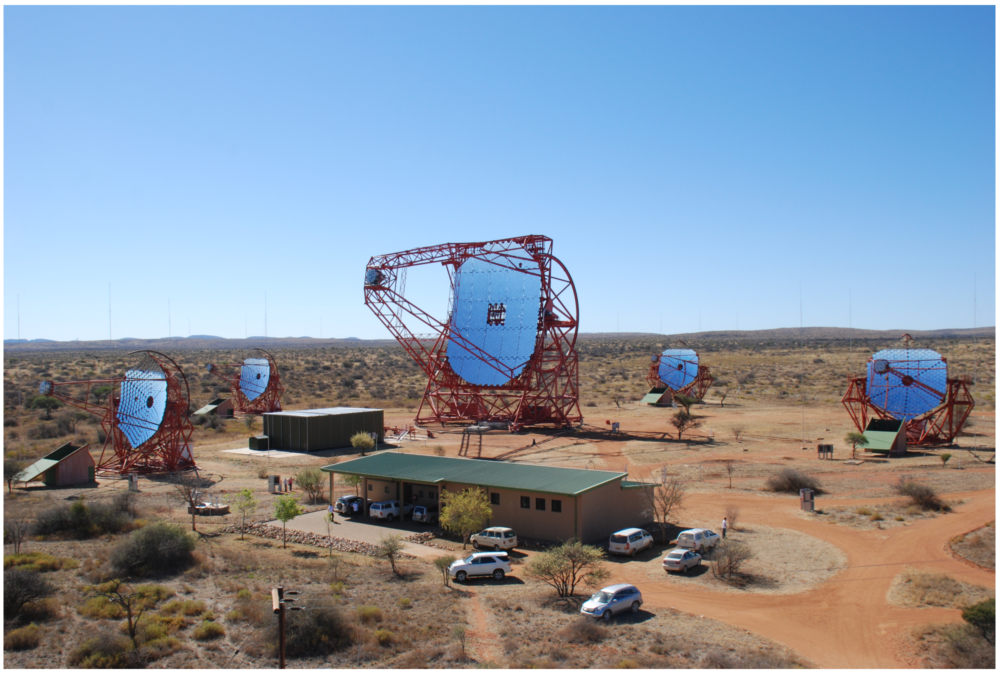

Figure 2.7: Photograph of the H.E.S.S. telescope array. In the middle the big telescope surrounded by four identical smaller ones. Taken from [40].

we can see the High Energy Stereoscopic System (H.E.S.S) telescopes, which are located in the Khomas Highland in Namibia at an altitude of 1800 m above sea level [41]. In the image we can see one central big telescope with 28 m diameter mirror and four identical small telescopes surrounding it. The mirrors collect the Cherenkov radiation and collimate it onto the camera which is mounted in the focal plane in front of the mirror. The camera consists of a tightly packed array of Photo Multiplier Tube (PMT)s, which allows the detection of the faint flashes of Cherenkov light. This also means that IACTs can only operate during nights. As a consequence, duty cycles of only roughly 10 % are possible [36].

Cherenkov light from γ-ray showers emitted will typically produce an elliptical image on the camera, from which the above mentioned properties need to be inferred. Cosmic rays, will produce images that are broader and more irregular and thus are rather easily differentiated from γ-rays, which leads to an excellent background rejection [36].

“IACT arrays are designed to observe point-like moderately extended ... objects ... . The potential of IACT arrays is limited for the search for very extended structures” [36] like diffuse emission or Fermi Bubbles. For such cases, a complentary approach which directly samples the secondary particles is used.

Ground-based particle detector arrays
Particle detector arrays have been used for a long time in cosmic rays research. In the beginning, Geiger-counters were used. Nowadays, mainly scintillator detectors or water-Cherenkov detectors are employed. In this work we will mainly be interested in water-Cherenkov detectors. The individual detection units are usually cylindrical tanks, with an outer layer that shields of any optical light. Inside, the tanks are filled with ultrapurified water and each tank hosts one or multiple PMTs. Secondary particles within showers will penetrate into the water tank. The secondary particles will produce Cherenkov radiation within the tank, which can be measured via the PMTs. The first γ-ray particle detector array based on the water-Cherenkov technique was Milagro. It was located near Los Alamos at an altitude of 2630 m. A large water pond with installed PMTs was surrounded by 175 small water tanks [42]. With the Milagro detector three new TeV γ-ray sources were confirmed during a survey of the galactic plane [14].

Following the success of Milagro a new array based on the water-Cherenkov technique was built in Mexico at 4100 m altitude, near the highest mountain of the country, Pico de Orizaba. The detector as well as Pico de Orizaba in the background can be seen in Figure 2.8. There are 300 steel water tanks, each equipped with four PMTs covering an area of 22 000 m^2 [44]. 900 of these PMTs were reused from the Milgaro experiment. In an upgrade additional 350 smaller water tanks were added surrounding the main array [45].

We can also see in Figure 2.8 that all tanks lie in one plane. That means with such an instrument the shower is sampled at a single depth, which leads to a higher degree of uncertainty in the reconstruction due to fluctuations [36]. This generally leads to a worse energy and angular resolution compared with IACTs. Also, the ability to distinguish between γ-rays and cosmic rays with traditional methods is rather limited.

But, such detector should be complementary to IACTs, thus it does not need to have the same performance in tasks where IACTs are already working ex-

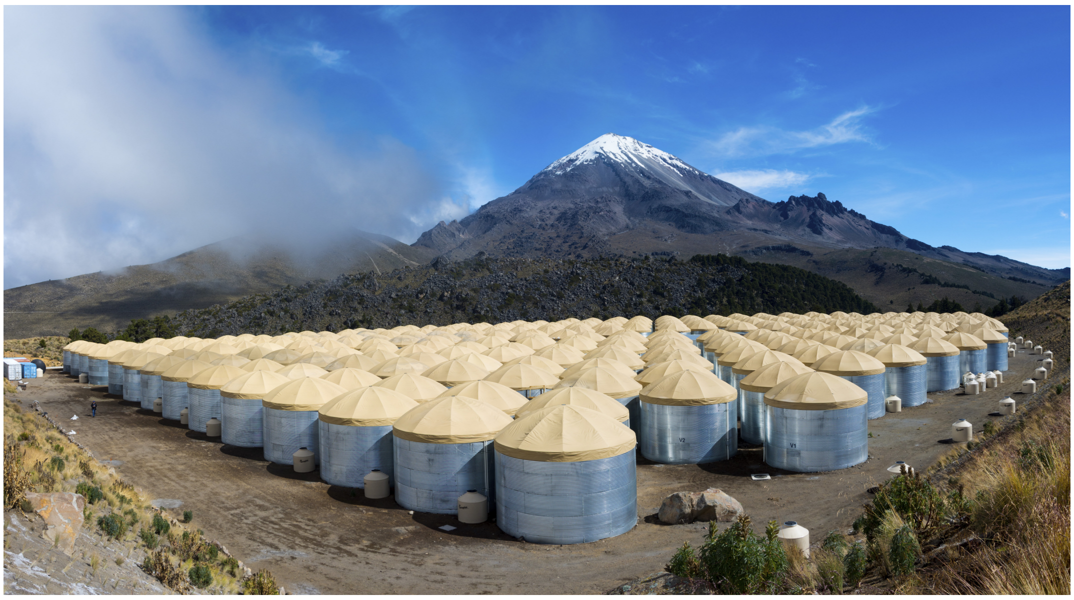

Figure 2.8: Photograph of the HAWC detector. We can see the array of steel water tanks in the middle. In the background the highest mountain in Mexico, the Pico de Orizaba. Taken from [43].

tremely well. These arrays have a large field of view ($\approx 2\,\mathrm{sr}$) and have an almost $100\,\%$ duty cycle, which makes them perfect as all-sky monitors [36].

Chapter 3

The Southern Wide-field Gamma-ray Observatory

Following the success of the HAWC γ-ray observatory, SWGO will be a future particle detector array for γ-ray astronomy in the VHE band, which will be located in the Southern Hemisphere [46]. SWGO, similar to HAWC, will be based on the water-Cherenkov technique. Recently it was announced that the detector "will be located in Pampa La Bola, at the Atacama Astronomical Park" [47] in Chile, on a mountain plateau in the Andes at an altitude of 4770 m.

This new observatory will be complementary to the CTA (an IACT array), also a new observatory for γ-ray astronomy in the VHE band [48]. In addition SWGO, together with particle detector arrays in the Northern Hemisphere, will provide monitoring capabilities of the whole sky.

3.1 Technical overview

The design of a particle detector array involves a number of compromises. Given a fixed budget, one needs to decide which design will be most appropriate for the planned science goals. At the moment multiple detector designs are being evaluated by the Collaboration. These differ in the number of water tanks, how these are arranged, their sizes, and how many PMTs they host. "[S]maller ... [water tanks] provide better spatial resolution and - usually - better time resolution for shower particles, but imply a larger number of sensor channels per unit area and hence - at fixed cost - a smaller detector area" [49]. A smaller detector area will reduce the number of high energy events which are observable. Spreading the tanks further apart will increase the area, but at the same time will increase the energy threshold[1]. These are just some of many considerations that need to be taken into account.

[1]The minimum energy a γ-ray needs to have to be registered

For this work, we will not take into account different detector layouts and focus on one particular configuration as a reference, which is shown in Figure 3.1. This configuration has 6588 water tanks. The array is nearly circular with a

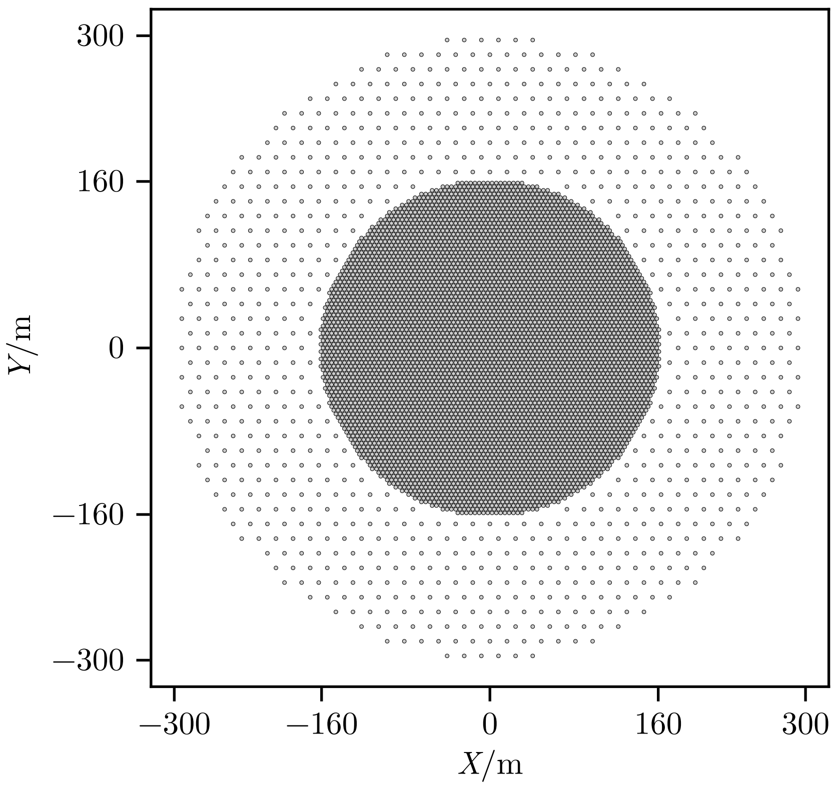

Figure 3.1: Sketch of the reference detector layout. The tanks are densely packed up to a radius of 160 m. Further out the tanks are more sparsely distributed up to a radius of 300 m.

radius of 300 m, which results in a physical area of $\approx 283\,000\,\mathrm{m}^2$. Close to the center and up to 160 m away from the center, is a dense arrangement of 5611 water tanks with a fill factor of $\approx 80\,\%$. From 160 m up to the border the arrangement is more sparse. With the dense inner array it will be possible to reach an energy threshold of ≈ 100 GeV. The outer array increases the area by a factor of ≈ 3.5 and thus the chance of capturing more high energy events.

One detection unit is a cylindrical water tank with a diameter of 3.82 m and a height of 3 m [50]. The tanks of the reference configuration have a double-layered design, as shown in Figure 3.2. Each chamber is filled separately with water and hosts one PMT. This double layer design allows us to identify muons [51], because muons within the shower, will most likely pass through the entire water tank and

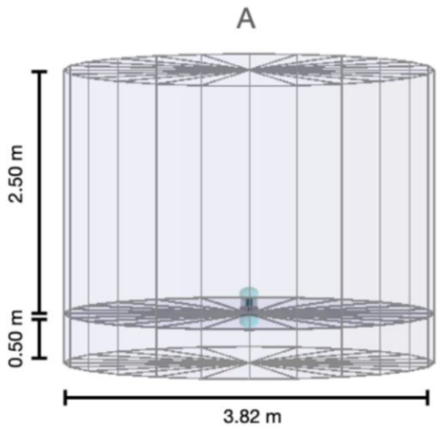

Figure 3.2: Sketch of the double layer tank design. Taken from [50].

produce Cherenkov light in both chambers. Electrons, Posititrons, and γ-rays on the other hand will most likely only produce Cherenkov light in the upper chamber. The upper chamber has an upward-facing PMT and the inner walls of the tank may be black or reflective. "The upward-facing PMT ensures that non-reflected Cherenkov photons with the smallest time dispersion are detected first" [51], whereas the lower chamber will have reflective walls and a downward-facing PMT. SWGO might be the first array with double layer tanks, however, already in Milgaro this principle was used within the water pond. Here they had two layers of PMTs in the pond, the upper layer was used for the direction reconstruction and the lower layer to discriminate against the background [14].

3.2 Monte Carlo simulations

Simulations of air shower
Currently, the experiment is only in the development phase and no measured data is available. However, we have access to simulations of air showers. These Monte Carlo simulations are done with a software package called COsmic Ray SImulations for KAscade (CORSIKA) (version 7.7410 [52]), which was initially developed for the KASCADE experiment [53].

There are simulations for primary photons and protons. The events generated with these simulations have four main characteristics: the energy E, the zenith angle θ, the azimuth angle ϕ, and the distance of the shower axis from the center of the array R. These parameters are chosen in the following way, similar for γ-rays and protons which is described in [54].

- **E.** Energies of the primary particles are drawn from a power law E^{α} with an exponent $\alpha = -2$, within an energy range of 31.6 GeV to 1 PeV (up to

10 PeV for protons) [52]. This is a harder spectrum than what we expect to measure, but it that way, we get more high energy events, thus we can evaluate the algorithms and the detector at these energies with better statistics.

- **R.** Events are simulated within a circle with a radius of 1500 m, where the detector is located in the center. The distance of the core of the shower, measured from the center follows a distribution, which ensures that per unit area the number of events is constant.

- $\boldsymbol{\theta}$. The zenith angle is sampled from this distribution $2\cos\theta\sin\theta$ with θ between 0 and 65°. The $\sin\theta$ term ensures a uniform distribution on the spherical cap, because more events need to be sampled for a given interval of $d\theta$ closer to the equator, compared to one at the pole. Additionally an $\cos\theta$ term is introduced because, viewed from the side, the detector area looks smaller, due to the projection effect.

- $\boldsymbol{\phi}$. The azimuth angles are drawn from a uniform distribution for angles between 0° and 360°.

In Figure 3.3 not directly E, R and θ, but instead E^{-1}, R^2 and $\sin^2\theta$ were plotted for 50000 simulated γ-ray air showers [2]. This is done because, if in this representation the distribution is uniform, one can infer how the parameters are distributed. This works because of a theorem from probability theory, which is sometimes called the **universality of the uniform** [55]. According to this theorem it is possible to generate any desired probability distribution from a uniform distribution. Consequently, if one knows how to sample values from a uniform distribution, it is possible to transform these values such that they follow some other distribution f.

As an example, let us take a look at the zenith angle, where we aim for values that are distributed according to $f(\theta) = 2\cos\theta\sin\theta$. The cumulative distribution function of f is $F(\theta) = \sin^2\theta$. The universality theorem states if we have values drawn from a uniform distribution and transform them with $F^{-1}(x) = \arcsin\sqrt{x}$, we get values that are distributed according to f. In reverse, this means that, given values distributed according to f, applying F to these values we get a uniform distribution. This shows why in the zenith angle histogram we use $\sin^2\theta$.

In Figure 3.4 the distributions of γ-ray air showers are shown, where at least one PMT has captured a signal. We can see that these distributions differ from the ones shown in Figure 3.4. First there are ≈ 16 times less events in the histogram and the distributions are no longer uniform. This means air showers in this subset are no longer distributed like they were sampled in the simulations. We have fewer events at high energies, fewer with great distance from the center, and also

[2]In Appendix A distributions for proton air showers are shown.

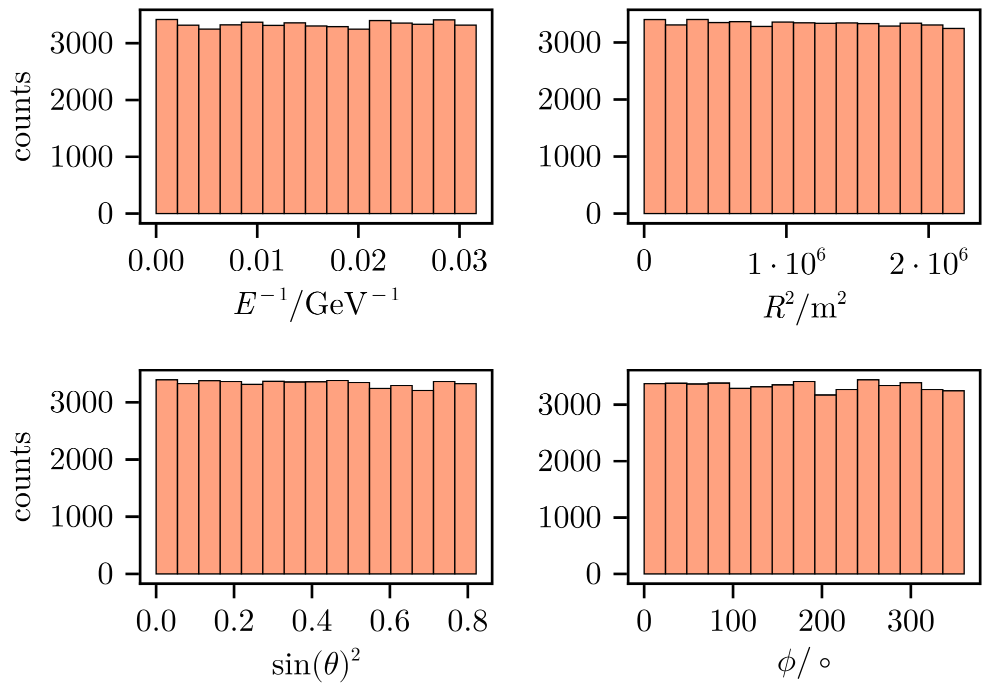

Figure 3.3: Distributions of input parameters of simulated γ-ray air showers.

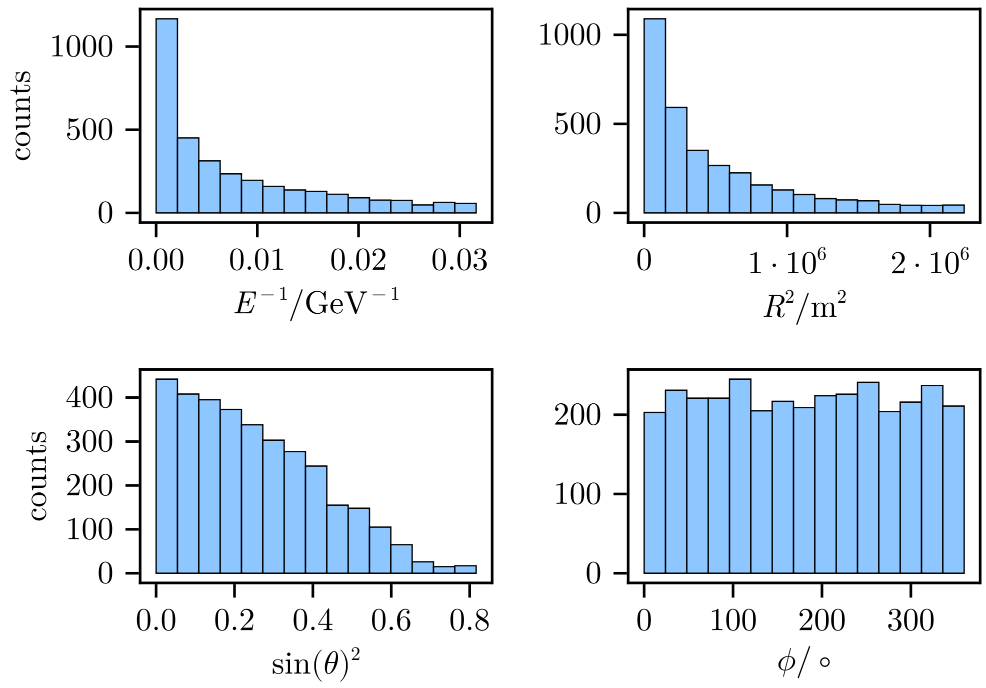

Figure 3.4: Distributions of input parameters of simulated γ-ray air showers, which were captured by at least one PMT.

fewer for large zenith angles.

Simulation of the detector response
The response of the detector for given air showers was simulated with a program called HAWCSim that is based on software package developed at CERN called GEANT [56].

3.3 Event reconstruction

As already mentioned in section 2.2 we need to extract the direction, the energy and the type of particle from the sampled footprint of a shower. We call this event reconstruction and by event we mean some observed air shower that triggered the array.
In the following, we want to take a look at an event reconstruction at the HAWC observatory and at methods already developed for SWGO.

Event reconstruction in HAWC

In a recent paper from the HAWC collaboration [57], an updated event reconstruction was discussed. In the following, we will summarize the main points of the event reconstruction.
Energy reconstruction
HAWC has two different energy estimators, one based on neural networks and the other based on the so-called Ground Parameter (GP). The latter one uses the reconstructed charge density at 40 m away from the core location, together with the reconstructed zenith angle for the energy estimate. The neural network is a shallow Multi Layer Perceptron (MLP) (section 4.5) with two hidden layers. As an input, handcrafted variables are used [58].

Gamma/hadron separation
In section 2.2 on EAS, we have learned that showers introduced by protons (hadrons) propagate differently through the atmosphere then a shower induced by a γ-ray. The signatures of a muonic or hadronic component can be used to differentiate between the primaries. The gamma/hadron separation in HAWC is mainly based on one parameter which is sensitive to the muon content, the LIC parameter, and one parameter, which is sensitive to the lateral development of the shower, the Parameter for IdeNtifying Cosmic rays (PINC) parameter [59]. The LIC parameter is an empirical parameter used in the Milagro experiment [42] and is defined as

$$\mathrm{LIC} = \log_{10} \frac{\mathrm{CxPE}_{40}}{\mathrm{nHit}}. \tag{3.1}$$

In this equation, CxPE_{40} is the largest amount of charge measured by a PMT that is more than 40 m away from the reconstructed shower core and the number of PMTs that measured some signal within 20 ns of the shower front is nHit. The idea is that γ-ray showers are characterized by a small value of the LIC parameter [59]. The PINC parameter is defined as

$$\text{PINC} = \frac{1}{N} \sum_{i=0} \frac{(\log_{10}(q_i) - \langle \log_{10}(q) \rangle)}{\sigma^2}, \tag{3.2}$$

where σ is the uncertainty in the charge measurements q_i based on data from the Crab nebula and $\langle \log_{10}(q) \rangle)$ is the mean value of all charges within an circle with 5 m radius around the measured charge q_i. The sum runs over all such possible circles. The Rectangular cuts based on these parameters are then used for the gamma/hadron separation [59].

Direction reconstruction
Originally, the direction reconstruction was done with a center-of-mass estimation, where the charges of hit PMTs are fitted to a lateral distribution function (LDF). Nowadays, a more sophisticated algorithm is used based on a maximum likelihood fit. A plane is fitted to the shower front, with information of the arrival times measured at different tanks. The shower front has a convex form and is not completely described by a plane. For that reason, additional curvature corrections are added during the fitting procedure [57].

Currently implemented methods in SWGO

Energy reconstruction
For the standard method for the energy estimation a **template-based reconstruction method** is used [60], which has already been successfully applied in other experiments [61]. Based on Monte Carlo (MC) simulations an expected probability distribution is generated and stored in the templates. During reconstruction one tries to minimize the negative log-likelihood of a given EAS using the predefined templates.
In our energy reconstruction (section 6.1) we compare to this method.

Gamma/hadron separation
The standard technique uses a shallow neural network as a classifier. The inputs to this network are similar parameters that are also used by HAWC, for example the before mentioned LIC and PINC parameters. So this approach uses a neural network, but with handcrafted variables. Another existing approach is based on so-called graph neural networks, where raw data is used as the input and no handcrafted variables are needed anymore. This method reduces the background over the entire energy range compared to the standard technique [62]. Later, in our gamma/hadron separation (section 6.2) we will compare to the graph neural network approach.

Direction reconstruction
Similar to the HAWC direction reconstruction, the method currently used as a

standard method is based on fitting a plane to the shower front. The fact that the shower front has a convex shape, is accounted for by including curvature corrections [63]. The reconstruction process begins with a preliminary estimate of the core position, based on the center-of-mass of the charge distribution. This is followed by an initial plane fit to the shower front. Then the core position is refined through a maximum likelihood method. The final planefit uses the likelihood core and the estimation of the first planefit, to produce a more accurate direction estimation [50]. In our direction reconstruction section 6.3 we will compare our method to this final planefit.

Specification	**Performance**		
	0.3 TeV	3 TeV	30 TeV
Angular resolution	1 deg	0.3 deg	0.15 deg
Energy resolution	100 %	50 %	25 %
Background rejection	50 %	99 %	99.9 %

Table 3.1: Performance goals for SWGO. Here we have listed the targeted angular and energy resolution and background rejection for different energies. Adopted from [46].

In Table 3.1 the expected performance goals for different reconstruction tasks in SWGO are shown. We can see that at an energy of 0.3 TeV one expects a angular resolution of 1 deg, an energy resolution of 100 % and a background rejection of 50 %. Compared to IACTs, the performance in these tasks is quite a bit worse, which again reflects the difficulty of an accurate event reconstruction in particle detector arrays.

3.4 Science goals

SWGO will be able to monitor big parts of the sky with access to the galactic plane. An **unbiased survey of the Galactic Plane** will be possible in the energy range of several tens of TeV [8]. Such a study may help to give new hints in the debate whether the TeV emission from young shell-type SNRs is of hadronic or leptonic nature and related to this whether SNRs are major contributors to Galactic cosmic rays up to the knee.

Extended regions in the sky will be studied, for example the Fermi Bubbles. These are large structures, which extend to 55° above and below the Galactic center. The angular resolution of SWGO should allow for more accurate characterization than is currently possible by the Fermi-LAT experiment [8].

SWGO aims for a lower energy threshold compared to HAWC. This together with the wide field of view and a nearly 100 % duty cycle, will make SWGO perfectly suited for **monitoring the transient sky** [8]. By transient sky, we mean VHE emitters, which show unexpected, mostly unpredictable flaring or exploding episodes at different timescales. An example are γ-ray bursts or gravitational waves. γ-ray bursts, like the name suggests, are intense flashes of γ-ray emission, happening over a timescale from milliseconds to seconds. 50 years after their discovery in 1969, the origin is still not properly understood. Candidates are stars at the end of their life cycle or compact binary systems that merge. With SWGO we expect more observations of γ-ray bursts in the VHE range and with the increase in statistics more precise discussions, of the different emission scenarios, may be possible [8].

Another science goal of SWGO is to search for physics beyond the Standard Model. SWGO could help in the search for dark matter particles by searching for γ-ray emission from the decay of such particles. In addition SWGO will search for so called primordial blackholes. These are hypothetical black holes formed during the early stages of the Universe [8].

As a last point, going back to the origins, SWGO will also be used for cosmic ray research. Besides composition and anisotropy studies, SWGO will be one of only a few γ-ray detectors in the TeV regime that can monitor the Sun. With measurements of the Suns “shadow” in the flux of Galactic cosmic rays, one can probe the magnetic fields surrounding the sun [64].

What it needs to achieve the science goals
In observations a high statistical significance is crucial, as this decides whether we can claim that we have discovered a new source or if we merely observed a statistical fluctuation of the background [36].

Idealized, we can write the significance as the ratio of the number signal counts over the square root of the number of background counts like this[3]

$$S = \frac{N_\gamma}{\sqrt{N_{\text{bkg}}}}. \tag{3.3}$$

Li and Ma [65] pointed out that this overestimates the significance, however for the following argument this is not important. For a given observation time we can then write S (for pointlike sources) in terms of a few important quantities like this [36]

$$S \propto \sqrt{A_{\text{eff}}} \cdot \frac{1}{\sigma_\theta} \cdot Q \tag{3.4}$$

where A_{eff} is the **effective area**, σ_θ the **angular resolution** and $Q = \frac{\epsilon_\gamma}{\sqrt{1-\epsilon_{\text{bkg}}}}$ the so-called **Q-factor**. In that last formula ϵ_γ is the gamma efficiency. This

[3]This considers only statistical fluctuations from the background and assumes the background follows a Poisson distribution.

value determines how many out of all γ-ray events are correctly detected as γ-ray. ϵ_{bkg} is the background efficiency, which states how many background events we correctly label as background. All the three quantities are greatly influenced by the performance of the reconstruction algorithms.

As an example let us take a look at the Q-factor, which is influenced by the performance of the gamma/hadron separation. We usually want to achieve a gamma efficiency of 80 %, thus $\epsilon_\gamma = 0.8$. If our algorithm would achieve a background efficiency of $\epsilon_{\text{bkg}} = 0.9$ we get $Q \approx 2.5$. If we would achieve a background efficiency of 0.99 at the same gamma efficiency we would get a Q-factor of $\approx$ 8. This alone would result in an approximately threefold increase in significance (compared to $Q \approx 2.5$).

The effective area can be defined as $A_{\text{eff}} = A_{\text{sim}}{}^{n_{rec}}/_{r_{sim}}$, where A_{sim} is the area on which air showers are simulated, and n_{sim} the number of simulated events and n_{rec} the number of reconstructed events. Often only a subset of triggered events is used for the event reconstruction, others are filtered by so-called cuts. For example one filters events, where the core did not land on the array or some quality criterion was not passed. If our event reconstruction algorithm is capable of reconstructing events, with as few cuts as possible, this would increase the effective area and thus the significance.

This shows the importance of powerful event reconstruction methods and current existing methods may benefit by new tools from the field of machine learning, which have already proven effective in other fields of research.

Chapter 4

Machine learning basics

4.1 Introduction and a first application in astronomy

The field of machine mearning (ML) searches for systems or procedures with “the ability to acquire their own knowledge, by extracting patterns from raw data” [66]. These general patterns should then also be useful for predictions on new unseen data.

One of the simplest and commonly used ML methods is called the method of **least squares**, which can be defined, in its most general case, as a technique to find an approximate solution $W \in \mathbb{R}^{\mathrm{m}}$ of a set of linear equations like shown in Equation 4.1 where $X \in \mathbb{R}^{\mathrm{m}\times\mathrm{n}}$ are independent input variables and $Y \in \mathbb{R}^{\mathrm{n}}$ dependent outputs.

$$XW = Y \tag{4.1}$$

Such a system has only an exact solution if $\mathrm{n} = \mathrm{m}$ and $\operatorname{rank} X = \mathrm{n}$, but in general only an approximation can be found. With the method of least squares we can find the “best” solution, which means a vector $\hat{W}$ is needed such that the **mean squared error**

$$L = ||XW - Y||^2 \tag{4.2}$$

is minimal. This equation defines a so-called objective function L. In the case of Equation 4.1 the vector $\hat{W}$ has an analytical solution given by $\hat{W} = \left(X^\top X\right)^{-1} X^\top Y$.

This method predates the era of computers, having been introduced in the early nineteenth century by Legendre and Gauss around the same time [67]. The question of who exactly was first is probably ”the most famous priority disputes in the history of science” [67] and remains an open debate.
However, the first major discovery in astronomy where least squares was used definitely belongs to Gauss. The rediscovery of **Ceres**. At 24 years of age, he

was able to calculate the orbit of the planet[1] Ceres in 1801, based on observations done by Italian astronomer Joseph Piazzi earlier in the same year. Based on the estimated trajectory of the planet, the planet was found again in December that same year [68].

The above mentioned techniques can be categorized in the class of so called **supervised learning**, which will be discussed in more detail in the next section.

4.2 Supervised learning

"Many practical problems can be formulated as requiring a computer to perform a mapping $f : X \to Y$, where X is an input space and Y is an output space" [69]. Speaking in this formulation, in image classification, all possible images would form the space X and the interval between $[0, 1]$ could form the space Y, corresponding to the probability of an object detected in the image [69]. In the context of astroparticle physics, the input space could be the set of measurements taken in a detector, for example a ground based gamma ray observatory, and the output space could be $\mathbb{R}^+$, if the goal is energy reconstruction of an extensive air shower.

The goal of **supervised learning** is given enough examples $(x, y) \in X \times Y$ find an approximation of the mapping f. In astroparticle physics correct labels are a priori not known, thus we rely on Monte Carlo simulations as data for supervised learning.

Finding the mapping
To find an approximate mapping f, such that for given examples $(x_1, y_1), (x_2, y_2), ...$ made up of independent and identically distributed samples from a data generating distribution D, i.e. $(x_i, y_i) \sim D$ for all i, an objective, or loss function is minimized. Mathematically speaking a scalar-valued function $L(\hat{y}, y)$ is chosen that calculates the difference between the predicted label $\hat{y}$ and the true label y [69] to measure the quality of f. A common loss functions is the **mean squared error** loss, already mentioned before, often used in regression problems. This loss is defined as

$$L = ||\hat{y} - y||^2. \tag{4.3}$$

In classification problems the **cross entropy loss** is frequently used, which is defined as follows

$$L = -\sum_{i=1}^{N_C} y^i \log(\hat{y}^i), \tag{4.4}$$

where the sum runs over all possible classes.

[1]Nowdays we call Ceres a dwarf-planet. It is an asteroid located in the asteroid belt. However back then they first thought of Ceres as a planet.

The **objective in supervised learning** is to find a function $f \in \mathcal{F}$, where $\mathcal{F}$ is the space of all possible functions which are considered, such that the following is satisfied [69]

$$f = \arg\min_{f \in \mathcal{F}} \mathbb{E}_{(x,y)\sim D} L(f(x), y) \tag{4.5}$$

Here $\mathbb{E}_{(x,y)\sim D} L(f(x), y)$ means, the expectation value out of all possible samples from the distribution D. Once this function f is found, it can be used to map a given element of X to the "correct" element in Y. In reality this is not how it works. Finding a global minimum of Equation 4.5 is not feasable, because not all possible samples of the distribution D are available. Thus Equation 4.5 is restricted to all elements from a given dataset with N examples [69]

$$f = \arg\min_{f \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} L(f(x_i), y_i). \tag{4.6}$$

Given a solution to Equation 4.6 it is expected that this solution will also work for new elements $(\tilde{x}, \tilde{y})$ outside of the dataset.

In astroparticle physics the learned mapping does not only need to work for new elements outside of a new dataset generated via Monte Carlo simulations, but in additions has to be applicable to real world experimental data from a given detector. We cannot evaluate this point for SWGO yet, but it is important to keep in mind.

4.3 Gradient-based optimization

In general, we will have no analytical solution at hand which minimizes Equation 4.6 as we have it for the linear least squares problem. The problem of linear least squares differs from the more general problem defined in Equation 4.6. In the linear least squares problem, we have restricted ourselves to a set of linear equations with unknown parameters W, whereas in Equation 4.6 no particular form of the function f is mentioned. Here, we are searching for a function out of all possible functions in an infinite dimensional space F. This is in general a much harder task. It is also not possible to find the global minimum, if f is a high dimensional function. However, by restricting ourselves to only a particular set of functions with tunable parameters $\vec{\theta}$, we are back to finding the minimum of a function.

If f is a differentiable function we can rely on gradient based methods to find a minimum. The negative of the local gradient will point in the direction of a local minimum. By updating the current position by a small margin in direction of the negative gradient we will get closer to the local minima. This essentially is the essence of **gradient descent**. Mathematically the update of the parameters looks like this

$$\vec{\theta}' = \vec{\theta} - \epsilon \cdot \frac{1}{N} \sum_{i=1}^{N} \vec{\nabla}_\theta L(f(\vec{x_i}; \vec{\theta}), y_i), \tag{4.7}$$

where ϵ is the so-called learning rate. Usually the learning rate is set to a small value and one performs Equation 4.7 iteratively. When $\nabla f = 0$ we have reached a local minimum. Gradient descent is a first order method, meaning it makes only use of the first derivative.

There are two challenges with this approach. Equation 4.7 is defined as an average over the whole dataset, which is computationally too expensive to calculate repeatedly. Secondly the loss function depends on many parameters θ. LLMs for example have billions of tunable parameters. Thus we need an efficient method for calculating gradients [70]. This point will be discussed in the next section 4.6.

Stochastic Gradient Descent
We can solve the first issue by adapting Equation 4.7. Instead of averaging over the whole dataset, we are randomly selecting a small number of examples and taking the average over this small subset, also called a **batch**. This method is called Stoachastic Gradient Descent (SGD). We expect that the introduced stochasticities (noise) will average out after sufficiently many small steps [70].

ADAM
A very popular alternative, or extension, to SGD is called adaptive moment estimation (ADAM), which was introduced in 2014 [71], which is used as a reference in the following. Similar to SGD, ADAM is also a first-order gradient based method. But unlike SGD, where individual updates of the parameters are independent of what has happened before, ADAM takes into account previously calculated gradients.

In addition to the gradient $g_t = \vec{\nabla}_\theta f(x; \theta)$ at the current step, we also store a weighted sum of previous gradients $(g_t, g_{t-1}, ...)$ and gradients squared $(g_t^2, g_{t-1}^2, ...)$ from all past iterations t_i. The weights are chosen in a way that more recent gradients have the most impact and gradients far in the past are exponentially suppressed. This exponential decay is controlled by two parameters β_1 and β_2 $(\in [0, 1))$. Nowadays, ADAM is the most frequently used algorithm for the gradient update, especially in the realm of transformer research.

4.4 Backpropagtion

In the previous section we learned how parameters of differentiable functions can get updated such that some loss function L gets minimized. What we have not discussed yet is how the gradients get calculated. We have already mentioned

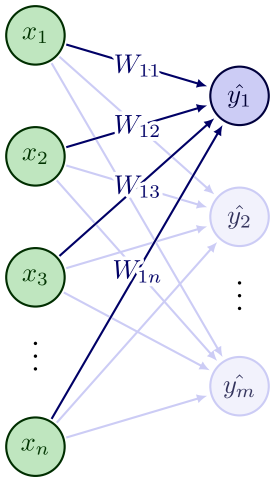

Figure 4.1: Visualization of a linear function $y = Wx$. Adopted from [72].

that this needs to be done in an efficient way when the number of parameters is large. With the backprogation technique we can calculate gradients of some scalar valued function F at the same cost as evaluating the function F.

In our case the scalared values function will be a loss function L. Let us illustrate the method with a simple example. Lets assume we have a linear function $f(x; W) = y = Wx$ with $x \in \mathbb{R}^m$, $y \in \mathbb{R}^n$ and $W \in \mathbb{R}^{n,m}$. We can visualize such a function as shown in Figure 4.1. Entries in the input vector x_i and in the output vector y_i are represented as nodes. Output nodes, for example y_1 are calculated from a weighted sum of every input node, where the weights are the entries W_{i1} of the matrix W. In addition, let us assume we are using a mean squared error loss:

$$L = \frac{1}{2} \sum_k (\hat{y}_k - y_k)^2 \tag{4.8}$$

Now we want to calculate the gradient $\nabla_W L(f(x; W), y)$ by making use of the chain rule

$$\frac{\partial L}{\partial W_{ij}} = \sum_k \frac{\partial L}{\partial \hat{y}_k} \frac{\partial \hat{y}_k}{\partial W_{ij}} = \sum_{k,l} \Delta_k \frac{\partial W_{kl} x_l}{\partial W_{ij}} = \sum_{k,l} \Delta_k \delta_{ki} \delta_{lj} x_i = \Delta_i x_j. \tag{4.9}$$

Here we defined $\Delta_k = \hat{y}_k - y_k$. We can write this in matrix notation as an outer product

$$\nabla_W L = x \Delta^T. \tag{4.10}$$

If we look again at Figure 4.1 and in particular at W_{11}, the parameter update of W_{11} is influenced by the difference at $\hat{y_1}$ and by the input node x_1.

We can calculate how many operations the evaluation of f, the forward pass, took and also calculate how many operations the calculation of gradients, the backward pass, took. The vector matrix multiplication in the forward pass has $m \cdot n$ operations, which is equal to the number of entries N_W in W. So, the forward pass has $\mathcal{O}(N_W)$ operations. In the outer product we need the same amount of calculations $m \cdot n$, which shows that also the backward pass has $\mathcal{O}(N_W)$ operations.

If we compare this to a finite differences method

$$\frac{\partial L}{\partial W_i} = \frac{L(f(x, W_{ij} + \epsilon)), y) - L(f(x, W_{ij}), y)}{\epsilon} + \mathcal{O}(\epsilon), \tag{4.11}$$

this needs $\mathcal{O}(N_W^2)$ operations, as each evaluation of $f(x, W)$ needs W operations and we need to calculate N_W finite differences [73].

Now this method is not restricted to only linear functions, it can be applied to an arbitrary function. Lets illustrate this with one more complicated function, which will also make clear why this is called the backward pass. Let assume we have the following function

$$f(x; W) = y = W^2 z(x; W^1) \tag{4.12}$$

where we added an additional function z which also contains adjustable weights in a matrix W^1, but can also include nonlinearities. Assuming again Equation 4.8 than gradients with respect to W^2 are just $z\Delta^\top$. Gradients with respect to W^1 look like this

$$\frac{\partial L}{\partial W_{ij}^1} = \sum_{k,l} \Delta_k W_{kl}^2 \frac{\partial z_l}{\partial W_{ij}^1}. \tag{4.13}$$

Here we can see that the differences (errors) Δ get passed back with W^2 and then only the local gradient $\frac{\partial z}{\partial W^1}$ needs to be calculated. By this procedure gradients for an arbitrary composition of functions can be calculated efficiently.

4.5 Neural networks and deep learning

With the tools from the last section we can find some minimum of a given function f with tunable parameters $\vec{\theta}$ and also know how to efficiently calculate gradients. The question is now, which representation do we choose for f. From approximation theory, we know that there exist many families of basis functions, which can approximate an arbitrary function to arbitrary precision [74].

An example we all know from our first course in calculus is Taylor's theorem, which states that any function can be approximated by a sum of polynominals, like this in one dimension

$$f(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \mathcal{O}(x^3). \tag{4.14}$$

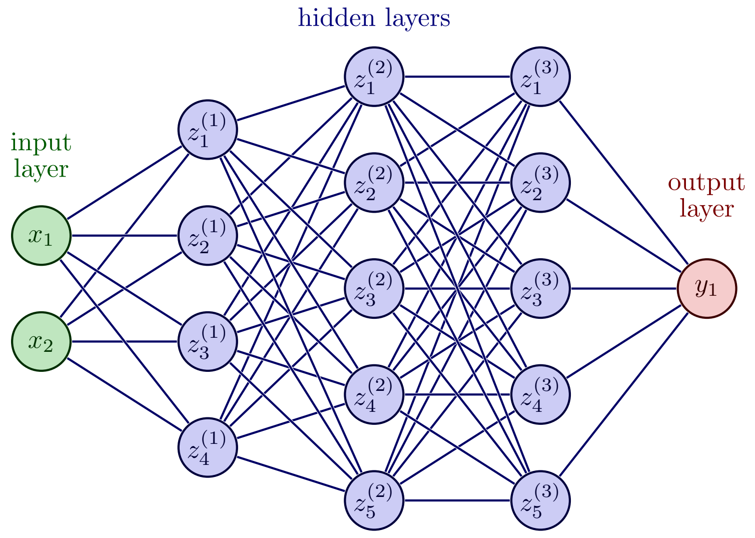

Figure 4.2: Multilayer perceptron with 3 hidden layers. The input x is two dimensional and is mapped to one number. The hidden layers have a width of 4, 5 and 5. Adopted from [72].

Taylors theorem gives us even an symbolic representation for all θ_i,

$$f(x) = f(0) + f'(0)x + \frac{f''(0)}{2!}x^2 + \mathcal{O}(x^3). \tag{4.15}$$

However, we often encounter problems in machine learning where the number of dimensions in the data is high. In such situations problems become exceedingly difficult and the number of terms for example in a Taylor expansion will explode. This phenomenon is known as the **curse of dimensionality** [66]. Empirically, and in some situations also shown mathematically [75], deep learning with neural networks does not suffer as much from the curse of dimensionality. A particular neural network is the MLP, which we now define.

4.5.1 Multi layer perceptron

The idea of Multi Layer Perceptrons (MLPs) was introduced by Frank Rosenblatt in 1958 [76]. His theory of the perceptron was based upon ideas and assumptions by Hebb and Hayek. For example that physical connections of the nervous system of an organism at birth are largely random, and that through exposure to a large sample of stimuli these connections will be changed [76]. These assumption hold true for the modern theory of neural networks.

Mathematically a MLP defines a mapping $f : \mathbb{R}^m \to \mathbb{R}^n$ with $y = f(x; \theta)$ where θ are learnable parameters . This mapping can be visualized, similar to the linear function, as a network (Figure 4.2) consisting of multiple layers. The functional form of f can be written as

$$f(x) = f^{(N)}(f^{(N-1)}...(f^{(1)}(x))) \tag{4.16}$$

where each function $f^{(i)} : \mathbb{R}^{d_{i-1}} \to \mathbb{R}^{d_i}$ is a mapping from the $i-1$ layer with width (or dimension) d_{i-1} to the i layer with width d_i. $f^{(1)}$ is the first layer, also called the input layer, mapping inputs x to the first layer. In that respect the final layer $f^{(N)}$ is called the output layer. Intermediate layers are termed hidden layers. The functional form of individual layers can be written as an affine function together with an elementwise nonlinear function. The nonlinear function g is called an **activation function**. The output of a given layer $f^{(i)}$ are called the **activations** of the i-th layer, which will be written as $z^{(i)}$. Thus[2]

$$z^{(i)} = f^{(i)}(z^{(i-1)}) = g(z^{(i-1)}W^{(i)} + b^{(i)}), \tag{4.17}$$

where $z^{(0)} = x$. The entries in the matrix $W^{(i)}$ and in the vector $b^{(i)}$ are called weights and biases, the tunable parameters of the MLP during training. If the MLP has multiple hidden layers it is called a deep neural network. As already mentioned, the **width** of a given layer is given by the dimension of the particular activation in that layer. Thus, all in all the MLP can be classified by the number of layers and their widths and the chosen activation function.

MLPs are also known by the names **feedforward** (FNN) or **fully connected** neural networks. Feedforward, because the flow of information is in one direction. A layer $f^{(i)}$ cannot be influenced by a later layer $f^{(j)}$ $(j > i)$ in the chain. They are called fully connected, because, viewing the neural network as a graph, nodes in neighboring layers are all connected to each other, which can be seen in Figure 4.2.

4.5.2 Activation functions and universal approximators.

Without activation functions between individual layers, the whole neural network would consist of multiple matrix multiplications (affine transformations) and we know from Linear Algebra, that the concatenation of linear functions is again a linear function. Thus our whole network would be a linear function. Linear functions have nice properties and are well understood however they lack in representation power. By adding **nonlinear** functions between layers, we can in principle approximate any function arbitrarily well. This is the so-called **universal approximation theorem**, which states that a MLP with only one hidden layer with arbitrary width is a universal function approximator.

[2]From now on we will think about vectors as row vectors, as this notation is used in Transformer literature.

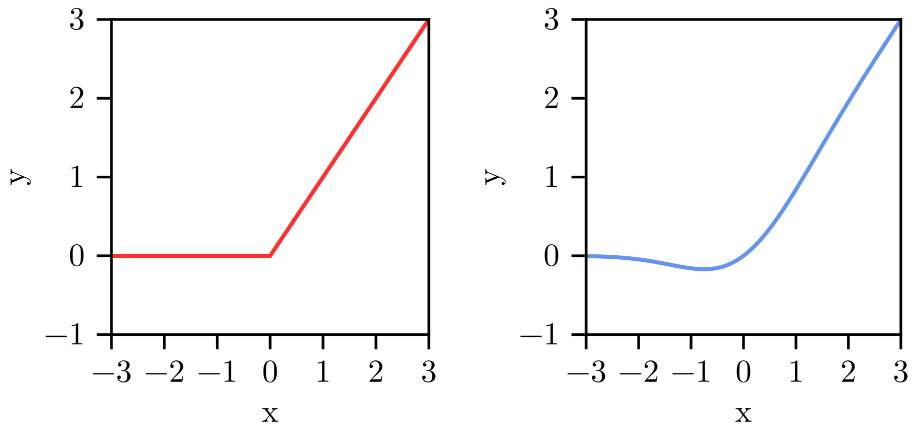

Figure 4.3: Two common activation functions. One the left side the RELU function and on the right side the GELU function.

A common activation function is the Rectified Linear Unit (RELU) function, which is shown on the left side of Figure 4.3. It is defined as

$$\mathrm{RELU}(x) = \max(0, x). \tag{4.18}$$

It is a nearly linear function and guarantees faster computation since no exponentials and divisions are needed. The RELU function was proposed in 2010, and has been the most widely used activation function in deep learning. RELU offers better performance and generalization compared to the Sigmoid and tanh activation functions [77]. Building on the successes of RELU, slight modifications where proposed for example the so-called ELU. In this work we will mainly use the GELU activation function, which was introduced in 2016 [78] and is defined as

$$\mathrm{GELU}(x) = x \cdot \frac{1}{2}[1 + \mathrm{erf}\,({}^{x}\!/\!\sqrt{2})]. \tag{4.19}$$

RELU and GELU are functions which operate pointwise on activations in a network. That means if $z^i = \left[z_1^i, z_2^i, ..., z_{d_i}^i\right]$ are the activations in layer i, the RELU activation function will be applied pointwise like this
$\mathrm{RELU}(z^i) = \left[\mathrm{RELU}(z_1^i), \mathrm{RELU}(z_2^i), ..., \mathrm{RELU}(z_{d_i}^i)\right]$.

A third activation function that is frequently used is called the **softmax function**. The softmax function can be used to normalize a given set of activations. It is defined as

$$\mathrm{softmax}(z_j) = \frac{\exp(z_j)}{\sum_k \exp(z_k)}. \tag{4.20}$$

From this equation we can see that unlike for GELU and RELU activation functions, softmax is a multivariate function that transforms individual activations

z_j based values of all other activations $z_1, z_2, ...$ in the same layer. The softmax function is often used in combination with the cross-entropy loss Equation 4.4, as the cross-entropy loss expects values that are in an interval between 0 and 1.

4.6 Training neural networks

In the last two chapters we have introduced optimization techniques and the first neural network architecture. Optimizing a neural network, adjusting the weights by gradient descent, is called training a neural network. In the following we will discuss individual steps during the training procedure.

Data loading and preprocessing
First, we need a dataset consisting of inputs $x_1, x_2, ...$ and labels $y_1, y_2, ...$ The inputs, also called features, are usually preprocessed. The standard procedure is to make sure the input dataset has mean zero and standard deviation one. If the values are distributed over a wide spectrum, logarithmic rescaling is frequently used.

Choosing the network structure and setting the hyperparameters
We need to decide which network architecture is suitable for the given task. Lets assume we take a MLP. Then, the so-called hyperparameters need to be chosen. For a MLP we need to decide on the number of hidden layers, the width of the hidden layers and which activation function we use in the individual layers.

Further, hyperparameters are the **learning rate** (and the **learning rate decay**), **batch size**, **number of epochs**. If the learning rate decay is used, the learning will get adjusted (get smaller) during the training process. The batch size determines how many training examples are used to calculate the gradients during one iteration of gradient descent. Commonly, batch sizes greater than 32 are used. Lastly, we need to decide how often we want to iterate over the dataset. One iteration is called an epoch.

Initialization of the network
Next, we need to initialize our network, which refers to setting the tunable parameters (weights), at the start. The initial weights are usually chosen to be random values around zero and often in such a way that the activations afterwards have again zero mean and standard deviation one. We will not go into the details, as this is usually done for us by the neural network software framework, but a detailed study of how initialization and activations functions influence the training of neural networks can be found here [79].

Training, validation and test set
"The central challenge in machine learning is that we must perform well on new, previously unseen inputs–not just those on which our model was trained" [66]. This ability to perform well on data outside the set of known examples is called

generalization. Thus, before we start the training procedure, we will split the dataset into three parts. The training set is actively used during the learning algorithm, meaning gradients are calculated from these inputs and labels in this set, and the model parameters are updated based on these gradients. Then, we have a validation set, which is used throughout the training, but only to observe the error on this set, the **validation error** or **validation loss**. Examples out of the validation set are not used to calculate gradients. We hold back a test set, and only once we are finished with all the training, we evaluate the model on the test set and calculate the **test error** or **test loss**. A typical split uses 70 % of data for training, 10 % for validation and 20 % for testing. The model

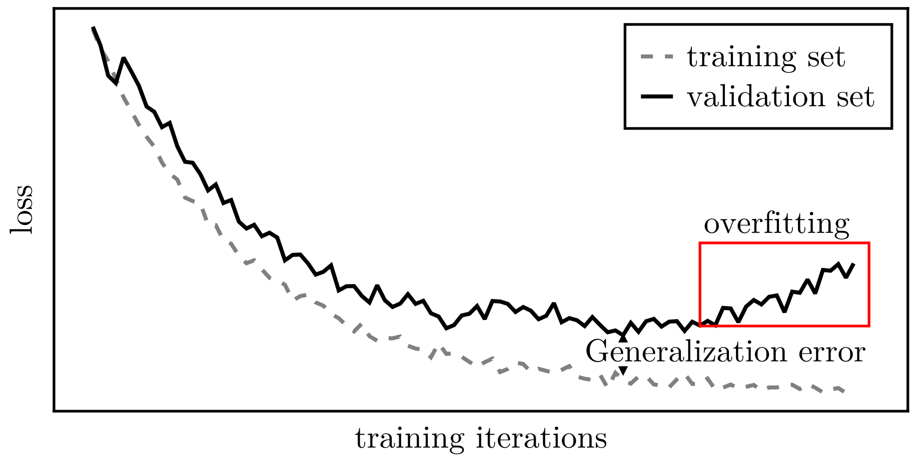

Figure 4.4: Training and validation loss curves. Adopted from [80].

has a good generalization, if the training error is small and the gap between the validation error and training error is low. The size of this gap is called the **generalization error**, which can be seen in Figure 4.4. In this figure common training and validation loss curves are shown. In the beginning, the validation and training loss are both getting smaller. After a few epochs, the validation loss stagnates, and after more epochs, the validation even increases again. This increase at the end is a sign that the neural network memorizes the training set and is called **overfitting**. If the training error is still high, the model is **underfitting**.

Which of these occurs is determined by the **capacity** of the model. Put simply, a small network with only a few parameters has a low capacity, whereas a large network with many parameters has a high capacity. A too small network, may lack the capacity to perform well on complex tasks, whereas a too big network may memorize properties of the training set and will not perform well on the test set. To figure out which set of hyperparameters works best, multiple trainings are done, where hyperparameters are changed and by monitoring the validation loss a good set of hyperparameters can be found, such that a low training error and also a low generalization error is achieved.

To reduce the amount of overfitting regularization methods can be used, which we will discuss in the next paragraph.

Regularization
In order to control the capacity we can add techniques which inhibit, at least to some part, overfitting. Ideally, with regularization we reduce the generalization error, but not the training error. Two common methods are **dropout** and **weight decay**.

In weight decay we introduce weight penalties. This means we add an additional term to the loss function, which penalizes large weights. Usually, one uses L^2 parameter norm penalty which is commonly referred to as weight decay which looks like this

$$L' = L + \frac{\lambda}{2} w^\top w. \tag{4.21}$$

Here, in w all weights of the neural network are contained and λ is the so-called regularization coefficient. It is an additional hyperparameter which determines the relative importance of the original error term L and the weight decay term. By calculating the gradient with respect to w we get:

$$\nabla_w L' = \nabla_w L + \lambda w. \tag{4.22}$$

Then, the update of the weights looks like this

$$w \leftarrow (1 - \epsilon\lambda) w - \epsilon \nabla_w L. \tag{4.23}$$

Comparing this to Equation 4.7 we see that the update rule got modified by the $-\epsilon\lambda w$ term. Equation 4.23 is equivalent to weight decay as it was orignally introduced [81].[3] In Equation 4.23 also highlights the name weight decay. The introduced term "encourages weight values to decay towards zero, unless supported by data" [73]. For example if the gradient with respect to L is zero, weight decay still draws w closer to zero.

In addition or individually we can add dropout. If dropout gets applied we are "dropping" out units (nodes) in our neural network. The means, we temporarily remove some nodes of the network, togehter with all incoming and outgoing connections. The nodes which are dropped are randomly selected [85]. This can be seen in Figure 4.5, where on the left we have the network without dropout and on the right side dropout was applied. The red crosses emphasize the dropped

[3]Here we argue, and show by calculations, that adding a L^2 penalty is the same as weight decay. This is only true of SGD, for adaptive methods like ADAM this is no longer true. This is why in most deep learning frameworks, one needs to use the AdamW optimizer instead of Adam, as the latter one has a wrongly implemented weight decay [82][83].

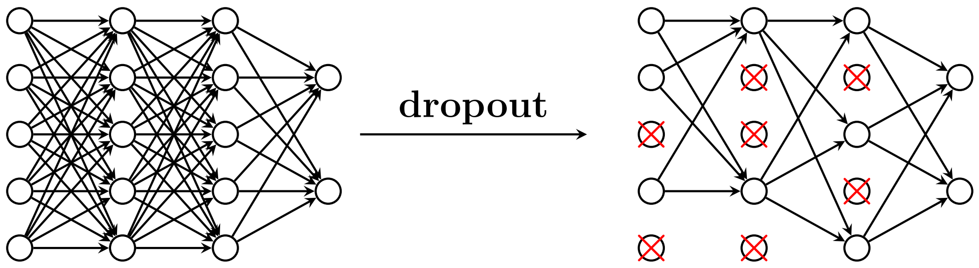

Figure 4.5: Visualization of Dropout regularization. On the left a network with applied dropout. On the right red crosses mark nodes which were randomly dropped and where all incoming and outgoing connections were removed. Adopted from [84].

nodes. Dropout is only applied during training and is turned off during inference.

Trainings loop
A typical training loop may look like the following (Algorithm 1). The trainings

Algorithm 1 A typical training loop.

```
for $i$ in $1, \ldots, n_{\text{epochs}}$ do
    for $(X, Y)$ in trainset do
        Y' = model(X)
        $\mathcal{L}$ = loss(Y, $\hat{Y}$)
        $\nabla_\theta$ = gradients($\mathcal{L}$)
        update!(model, $\nabla_\theta$)
    end for
end for
```

loop starts by iterating n_{epochs} times over the entire training set. During the next loop, we randomly sample batches of inputs X and labels Y. We then calculate the predictions $\hat{Y}$ and in a next step calculate the error or the loss between the prediction $\hat{Y}$ and the true labels Y. From this the gradients are calculated with the **backpropagation** algorithm. In the end, we update the parameters, via some form of gradient descent.[4]

Skip connections and layer norms.
As a last point, we want to mention skip connections and layer norms, as we will make use of them later. These techniques address some of the difficulties that arise in the training of neural networks, helping to decrease training time and also improve performance.

[4]What we left out in this example is the tracking of performance metrics, like storing the training loss.

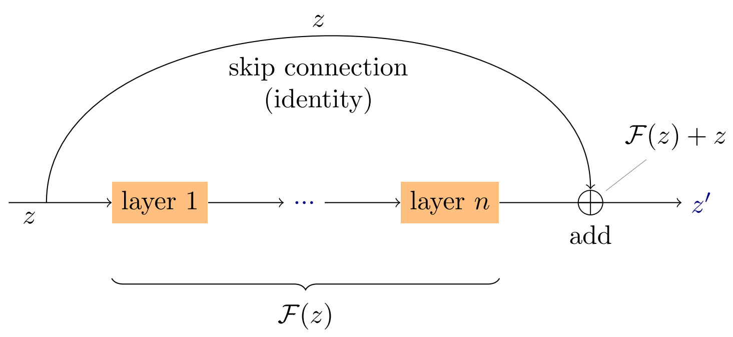

Figure 4.6: Visualization of skip connections. In parallel to the layers in orange an additional pathway is added. Adopted from [86].

Skip connections, also called residual connections, can be thought of as a shortcuts that skip layers. Here, the activations z before a particular layer are added directly to the output z at some point later in the network. Let us assume $\mathcal{F}$ is a function, comprised out of all layers or maybe multiple layers. Without skip connections we would have

$$z' = \mathcal{F}(z). \tag{4.24}$$

If we add skip connection the transformation would change to

$$z' = \mathcal{F}(z) + z \tag{4.25}$$

In Figure 4.6 this is visualized. Empirically, it was observed that deeper models suffered from degeneration in performance, and adding skip connections helped with optimization [87].

Vanishing gradients are another problem in deep neural networks. We can reduce this problem by normalizing the activations of the neurons. We have already mentioned this in the discussion on initialization of neural networks. However the correct initialization only makes sure activations are normalized before training. A so called **layernorm** makes sure that the activations stay normalized through out the training. From all activations a^j within one layer one calculates the mean and variance like this

$$\mu^{(j)} = \frac{1}{H}\sum_{k=1}^{H} z_k^{(j)} \quad \sigma^{(j)} = \sqrt{\frac{1}{H}\sum_{k=1}^{H}(z_k^{(j)} - \mu^{(j)})}. \tag{4.26}$$

Then, the activations get scaled like this

$$\bar{z}_k^{(j)} = \frac{g^{(j)}}{\sigma_k^{(j)}}(z_k^{()} - \mu_k^{(j)}) \tag{4.27}$$

where g_k are a gain parameters that scale the normalized activations [88].

4.7 The Transformer

The Transformer architecture was introduced in 2017 in the paper ”Attention is all you need” [12], which was published by a group at Google. In that paper the Transformer was applied in the context of natural language processing (NLP). In particular the authors were concerned with machine translation. Previously recurrent neural networks (RRNs) were state of the art approaches in this field. However, they are intrinsically sequential systems and this ”sequential nature precludes parallelization within training examples” [12], making them inefficient to train. With the Transformer, Vaswani et all, proposed a model architecture, which ”[eschewed] recurrence and instead [relied] entirely on an attention mechanism to draw global dependencies between input and output” [12]. Ever since the Transformer became the dominant approach for all subfields of NLP. Nowadays, Transformer based models are most known for text generation. ChatGPT was the first so called Large Language model (LLM), which caught the attention of the public.

The architecture is composed out of five parts. An **embedding layer**, the **self attention**, a **MLP**, a **positional encoding** and a **task specific final layer**. This structure can be seen in Figure 5.2. In the context of NLP an additional layer is needed which is called the **tokenization step**. The Transformer was introduced as a network mapping sequences of words to sequences of words. We will first explain the individual parts, in the context of NLP, and later we explain the changes that are being made for using the Transformer in SWGO. Here we mainly follow the original paper [12].

4.7.1 Tokenization

In the tokenization step, a sentence, will be transformed into a numerical representation. One common way of doing this is a so called word tokenization. Here one has a dictionary of words, and each word in the dictionary relates to a unique number. For example the sentence “Physics is interesting.” could look like ”34999, 374, 7185, 13”, meaning “Physics” = 34999, “ is” = 374, “ interesting” = 7185 and “.”=13. These numbers are not randomly drawn, but this the representation of our sentence that would be used in the open source LLM LLAMA-3-8B from Meta [89]. [5]

4.7.2 Embedding

With the tokenization step the sequence of words was transformed into a sequence of integers. A different view would be to interpret the integers as unit vectors in

[5]That in that example each word was mapped to a individual number was only by accident. The Llama-3 model uses a subword tokenization, where not necessary complete words will be mapped to a number, but also subword parts.

the space of all words of our vocabular. Thus the sequence can be interpreted as a sequence of unit vectors $\{x_1, x_2, x_3, ...\}$ where $x_i \in \mathbb{R}^{N_v}$ and N_v is the length of the vocabulary. The sequence can contain an arbitrary amount of tokens. The number of tokens in the sequence is called the **sequence length**. From now on we should think about sequences as matrices, where each row represents one token in a sequence, this is the representation used during computation, because matrix multiplications can be done efficiently and in parallel on the computer. The sequence of unit vectors is then a matrix X with L rows, each row being the corresponding unit vector, thus $X \in \mathbb{R}^{L,N_v}$. The embedding layer is a linear mapping from the

$$\mathbb{R}^{N_v} \to \mathbb{R}^{C}, \tag{4.28}$$

where C is called the channel (or embedding) dimension. Thus, the embedding layer can be written as a matrix $U_{\text{emb}} \in \mathbb{R}^{N_v,C}$. Performing the matrix multiplication, which is computationally cheap, gives us the following embedded sequence,

$$X' = XU_{\text{emb}} \in \mathbb{R}^{L,C}. \tag{4.29}$$

4.7.3 Self attention

At this point individual tokens were processed only with respect to themselves. The self attention layer lets different tokens "communicate" with each other. To make it more precise we will now define the self attention layer first and explain the individual parts afterwards. The attention matrix A is defined as [12]

$$A = \text{softmax}\left(\frac{QK^\top}{d_k}\right) \tag{4.30}$$

where Q are the so-called queries defined as

$$Q = YU_Q,\ U_Q \in \mathbb{R}^{C,C} \tag{4.31}$$

and K are the so-called keys defined as

$$K = YU_K,\ U_K \in \mathbb{R}^{C,C}. \tag{4.32}$$

An entry A_{ij} is a scalar product with one query Q_i (a row in Q) and key K_j (a row in K). As we calculate the scalar product between every two input tokens, this is a global operation. The scalar product get rescaled by d_k, the dimension of the keys, to make sure that the products are not too big before the softmax function is applied. Large values in a softmax are numerically unstable and also the calculated gradients become extremely small. If we would not scale the scalar products by d_k their variance would be d_k. This can be seen with the following argument. If we assume that entries in Q and K are independent random variables with mean 0 and variance 1, the scalar product $Q_i \cdot K_j$ will have mean 0 and variance d_k. The softmax function is applied rowwise [12].

With A we calculate the outputs of the self attention layer like the following

$$Z = AV \tag{4.33}$$

where V are the so-called values defined as

$$V = YU_V,\ U_V \in \mathbb{R}^{C,C}. \tag{4.34}$$

$Z \in \mathbb{R}^{L,C}$ is a newly transformed sequence, where individual tokens contain information from each of the input tokens.

We already want to highlight one property of the attention matrix, which will be become import later, namely that A scales **quadratically** with the lenght of the input, as $A \in \mathbb{R}^{L,L}$.

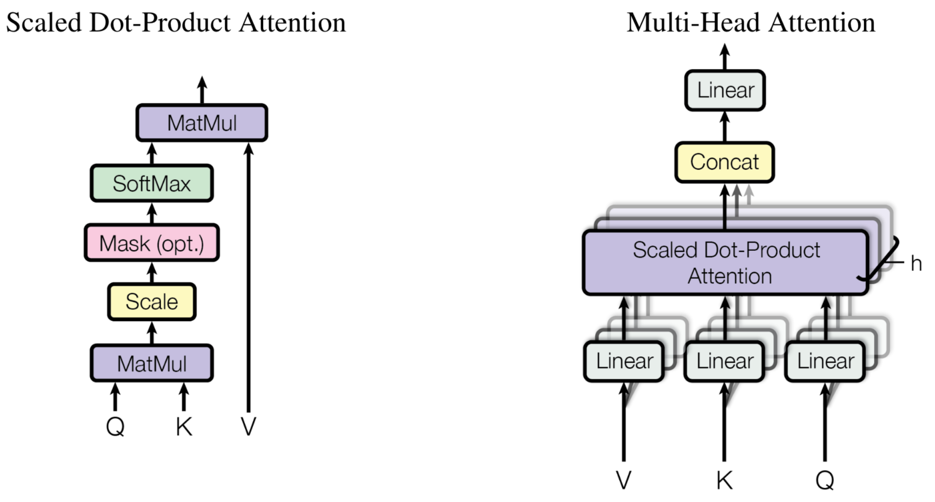

Figure 4.7: Visualization of the attention mechanism. On the left the standard self attention mechanism or scaled dot-product attention is shown. On the right the multi-head self attention. Taken from [12].

On the left side in Figure 4.7 the individual steps we have discussed are visualized again, starting in the bottom with the queries, keys and values.

Figure 4.7 shows one detail that we left out. Before applying the softmax function an optional masking operation can be applied. This is called **masked attention**. This mask determines which scalar products are evaluated. In LLMs one usually uses a triangular mask, which makes sure that no scalar products are calculated from queries which come before keys in the sequence.

Multi-head self attention

Usually **multi-head self attention** is used instead of self attention as introduced. What changes is that, instead of calculating one matrix A, multiple so-called heads of attention are calculated. A sketch of this operation is shown on the right side of Figure 4.7. Queries, keys and values will get projected into a space of dimension ${}^{C}\!/_{h}$ where h is the number of attention heads. This means the channel dimension needs to be some multiple of h. One attention matrix is then defined as

$$A_i = \text{softmax}\left(\frac{(QW_i^Q)(KW_i^K)^\top}{{}^{C}\!/_{h}}\right) \tag{4.35}$$

and

$$Z_i = A(VW_i^V) \tag{4.36}$$

with $W_i^Q, W_i^K, W_i^Q \in \mathbb{R}^{C,C/h}$. In the end we concatenate the Z_i from different heads and apply a further linear transformation $W \in \mathbb{R}^{C,C}$

$$Z = \text{Concat}(Z_1, ..., Z_h)W. \tag{4.37}$$

4.7.4 Token-wise multilayer perceptron

Following the attention block, individual tokens (rows of Z) are further processed in a shallow MLP, usually with one hidden layer, like this

$$Z' = \text{RELU}(ZW^{(1)} + b^{(1)})W^{(2)} + b^{(2)} \tag{4.38}$$

In the original paper [12] the dimension of the hidden layer was set to $4C$.

4.7.5 The Transformer block

The combination of Self attention together with the MLP, is often called the **Transformer block**. In bigger networks, multiple of these blocks are stacked sequentially, which is indicated as $N\times$ in Figure 5.2. If no mask is applied and scalar products from all keys and queries are calculated this is usually called an **encoder**. If scalar products are masked out with an triangular mask, it is called a **decoder**.

Skip connections are added in parallel both to the self attention and to the MLP. Originally, each of these two operations were followed by **layer normalization**, which normalizes tokens along the channel dimension. Nowadays, the layer norms are added before the attention operation and the MLP [90]. Mathematically the Transformer block then looks like this

$$Z = (Y + \text{attention}(\text{Layernorm}(Y))), \tag{4.39}$$

$$Z' = (Z + \text{MLP}(\text{Layernorm}(Z))). \tag{4.40}$$

4.7.6 Positional encoding

Through the tokenization, embedding, self attention and MLP, we have transformed a sequence of words, to a sequence of tokens in a vector space. Apart from the self attention layer, tokens where processed individually. One can show that a permutation in the order of the tokens in the original input sequence X results in the permuation of the tokens in the output sequence Z'. That means tokens are processed without regard to their position in the sequence. However, in language we know that the order of words in a sentence makes a big difference. To make sure this spatial ordering is also relevant in the transformer a positional encoding is added after the embedding layer. In the original paper, a constant positional encoding was used, consisting of sinus and cosine functions with different frequencies. The positional information is added after the embedding layer, like this

$$X' = XU_{\text{emb}} + \text{PE} \tag{4.41}$$

where $\text{PE} = \text{const} \in \mathbb{R}^{C,C}$.
In the paper, they also experimented with learnable positional embeddings and reported that they did nearly produce identical results [12].

4.7.7 Final head

After the Transformer block we have a sequence of tokens, where each token is a vector $\in \mathbb{R}^C$. This sequence is processed in the so-called final head to the desired outputs. In NLP the goal is often to transform this sequence of vectors into a sequence of probabilities that determine which words (or subwords) are most likely. This is usually done by a simple linear layer followed by a tokenwise softmax function, like this

$$Y = \text{softmax}(Z'U_{\text{final}}), \tag{4.42}$$

where $U_{final} \in \mathbb{R}^{C,A}$ maps the tokens back to the dimension of the vocabulary. In LLMs this sequence should resemble the input sequence shifted to one word. This is usually called next word prediction, because by shifting the sequence to the right, the first token is dropped and a new token is generated in the end.

Chapter 5

A Transformer for particle detector arrays

Inspired by the success of Transformers in the field of NLP, a group at Google published a paper called "An Image is Worth 16x16 words", were they applied a Transformer directly to images, with as few as possible modifications to the original architecture [91].
We will follow their architecture and modify it, such that it is suitable for SWGO[1]. Before we go into the details of the architecture we want to motivate why a Transformer is a fitting neural network for particle detector arrays.

5.1 Why a Transformer?

In particle detector arrays, we are conflicted with two challenging aspects. In Figure 5.1 we can see two footprints of γ-ray air showers. On the left, an event where a large fraction of tanks have measured a signal is shown. On the right, a sparse event is depicted, where only a small fraction of tanks triggered. The latter example is the more typical case. An efficient event reconstruction would make use of this sparseness, but this leads to variable sized inputs. A MLP cannot accommodate for variable-sized inputs, as the input layer can only process data with a given fixed size.

A second challenge is the non-uniform spacing of tanks, as shown in Figure 3.1. These two aspects limit the choice of possible neural network architectures.

GNNs can deal with data that is irregularly distributed in space and also with variable-sized inputs. These networks are designed to work for non-Euclidean domains. They have already been used in different physical contexts. For example in particle physics [92], weather predictions [93] or in neutrino astronomy

[1]And in principal to other particle detector arrays, maybe with minor modifications in the embedding layers.

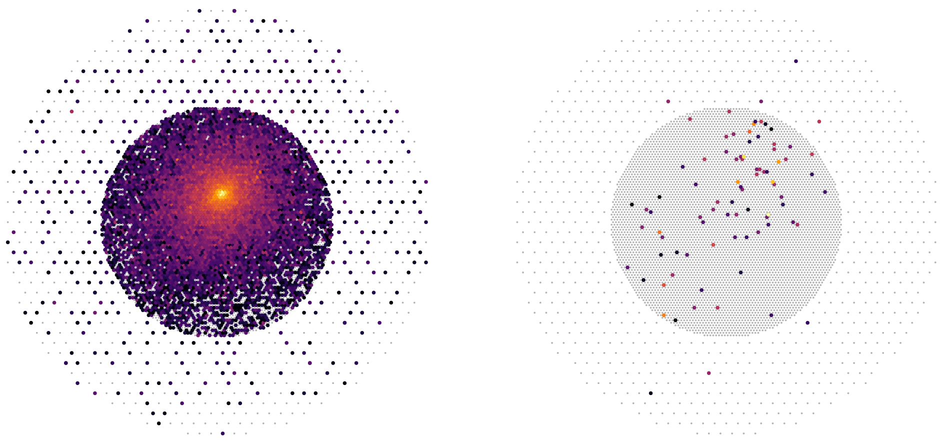
Figure 5.1: Two example footprints of air shower. On the left a γ-ray event with an energy of 47 TeV with 5370 triggered tanks. On the right a more sparse γ-ray event with an energy of 2 TeV and only 85 triggered tanks. The color scale signals the induced charge in individual tanks.

[94]. Recently they have also been successfully applied within the field of γ-ray astronomy [95]. As already mentioned in section 3.3 these networks are already being used in SWGO.

A second alternative architecture, capable of handling non-uniform spacing and variable-sized inputs, is the Transformer architecture section 4.7. For the HAWC observatory, a Transformer based architecture was already investigated, however without utilizing the full self attention mechanism [96].

5.2 Input data

5.2.1 Structure of the data

A footprint of an air shower, as shown in Figure 5.1, can be naturally thought of as a set of four vectors, each looking like $x = \left[t^u, t^d, q^u, q^d\right]$. Each vector contains an arrival time and an integrated charge signal per PMT.

In addition to this measurement information, we know the location of the tanks, giving us a set of positions. We can interpret each item in these sets as a row in a matrix. The two matrices look like the following

$$X = \begin{pmatrix} t_1^{(u)} \, t_1^{(d)} \, q_1^{(u)} \, q_1^{(d)} \\ t_2^{(u)} \, t_2^{(d)} \, q_2^{(u)} \, q_2^{(d)} \\ \cdots \\ t_n^{(u)} \, t_n^{(d)} \, q_n^{(u)} \, q_n^{(d)} \end{pmatrix} \text{ and } R = \begin{pmatrix} x_{(1)} \, y_{(1)} \\ x_{(2)} \, y_{(2)} \\ \cdots \\ x_{(n)} \, y_{(n)} \end{pmatrix} \tag{5.1}$$

where the i-th row in X has the measured time t_i^u and charge q_i^u of the upper PMT (and t_i^d, q_i^d for the lower PMT) and i-th row in R the corresponding x_i and y_i coordinates.

5.2.2 Normalization

Normalization of arrival times
First, we center the data by subtracting the average arrival time (of all events in the dataset) and then we divide by the standard deviation of arrival times σ_T

$$\tilde{T}_i = \frac{T_i - \bar{T}}{\sigma_T}. \tag{5.2}$$

Normalization of signals
For the charge information a logarithmic rescaling is used. First charges get incremented by one, then the logarithm is applied. The added offset ensures that a zero value in the signal translates to zero after the transformation. Then we divide by the standard deviation of these logarithmic charges.

$$\tilde{Q}_i = \frac{\log{(Q_i + 1)}}{\sigma(\log(Q + 1))} \tag{5.3}$$

Tank positions
The tank positions already have roughly a mean of zero, because they are radially centered around (0,0). Thus we only divide by the standard deviation σ_P of the distances of tank positions relative to the center.

$$\tilde{P}_i = \frac{P_i}{\sigma_P} \tag{5.4}$$

5.3 Modifications in the architecture

A Transformer used in the context of computer vision is called a Vision Transformer (ViT) [91]. We have adopted the ViT architecture to handle footprints of air showers. A sketch of our architecture is shown in Figure 5.2. In the following we discuss the modifications of our architecture compared to the one discussed

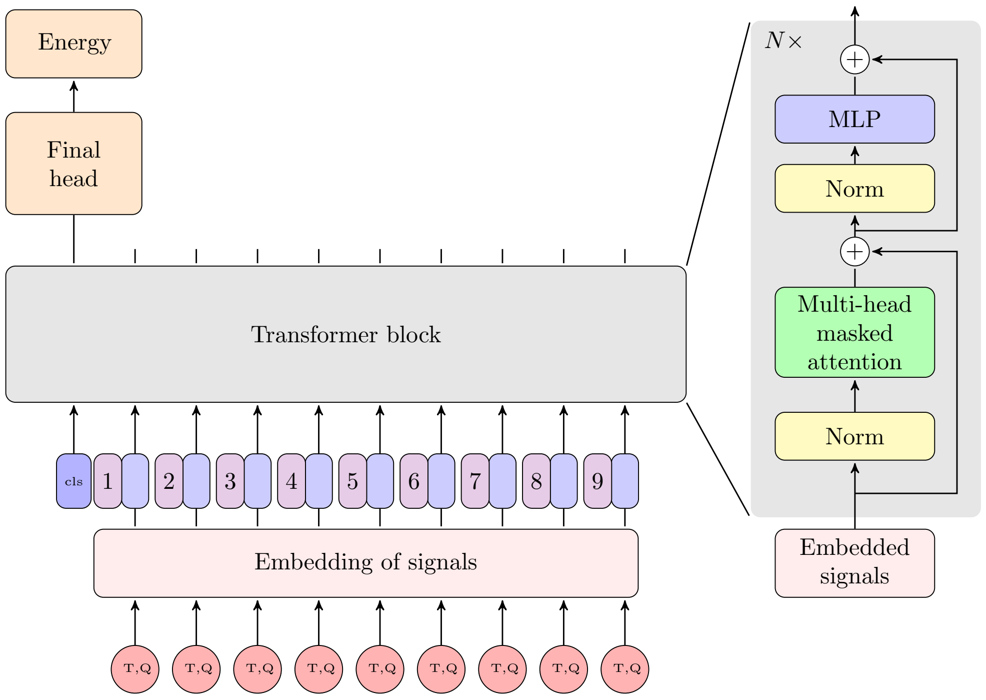

Figure 5.2: A Transformer architecture for SWGO. Starting at the bottom a sequence containing charge and time information is linearly embedded and positional embeddings are added. At this stage an additional CLS-token is added. The resulting sequence is fed to the Transformer block. In the end only the first token is used in the final layer. The illustration was inspired by [91][12]

in section 4.7.

Embedding and positional encoding
The tokenization step, which is needed to transform a sequence of words, into a continuous representation, is not needed, because the signals are already vectors. Thus, the first layer in our architecture is the embedding layer, where the four vectors of signals are linearly transformed into a higher-dimensional space.

As a positional encoding, we have chosen a learnable positional encoding. This is just a simple linear transformation $U_{\text{pe}} \in \mathbb{R}^{2,C}$, which takes position vectors from the matrix P and maps them into the embedding vector space. This positional information is then added to the embedded inputs like this

$$X' = XU_{\text{emb}} + PU_{\text{pe}}. \tag{5.5}$$

CLS token
Before feeding this sequence into the Transformer block, an additional token is added to the beginning of the sequence, the so-called CLS-token (classification token), as shown in Figure 5.2. The entries of this token are additional learnable parameters. Thus, the input to the Transformer block, will be a matrix of size $(L + 1,\, C)$.

Transformer block
The Transformer block in it's structure is unchanged from the one we have introduced in the previous chapter. We only change the activation function inside the MLP to a GELU activation function, as this was also used in the original ViT [91].

What changes is the **masking operation**. During training, we need to construct batches out of randomly sampled events from the training set. These events will, in general, have different sequence lengths (different numbers of triggered tanks). The Transformer can deal with different sequence lengths, however within one batch, the sequence lengths need to match. To ensure this within the batch, smaller events will be padded with zeros, such that each event has the same sequence length as the largest event. To ensure that the added zeros do not change the results of the calculations we need to make sure these tokens are masked out during the self attention operation.

Task specific final head
The output of the Transformer block is a new sequence of tokens $Z' \in \mathbb{R}^{L+1,C}$. In the final head only the first token in this sequence is used and mapped to the

desired output.[2] In general this layer will look like this

$$Y = \text{Layernorm}(Z'_0)U_{\text{final}}. \tag{5.6}$$

We normalize the first token and then apply a linear transformation U_{final}. The exact shape of U_{final} depends on the specific task.

An alternative to only using the first token, would be to compute the mean of each column in Z' and then apply layer normalization and the linear transformation. In the appendix D.3 the authors of [91] showed that both methods perform equally well if the learning rate is properly adjusted. We also experimented with this and did not observe any difference.

5.4 Computational requirements

One aspect of the computational cost of an algorithm, or in this case, the training of neural networks, is the amount of memory that is needed. For neural networks, this amount depends on several factors, including the size of the model, the optimizer, the batch size and the precision of the computations. In addition, there are different memory requirements during inference and training. All these points will be covered in the following, with a special focus on the Transformer architecture.

Floating-point numbers
Neural networks are usually trained using single precision floating point numbers. Thus for each number we need to store four bytes of memory. With the advent of LLMs half precision, i.e. Float16, numbers are used quite frequently. We will use single precision floating point numbers (Float32) as this is the default type used in the field if models are not too large.

Memory requirements during training
The memory requirements for training a neural network can be divided into 3 main categories, which we will call the model memory, activation memory and the gradient memory.
The **model memory** is the amount of memory needed to store the weights of the neural network.

During the forward pass, the activations of intermediate layers need to be stored,

[2]At first this seems to be wrong, why would we only use the first token. However what one needs to remember is that in the sequence Z' every token is a product of multiple attention operations and thus contains information of all input tokens.

Layer	Parameters	$\sim$ Memory / MB
Embedding	786	0.003
Pos. Encoding	384	0.001
Q,K,V	590592	2.4
MLP	1179648	4.7
Total	1771410	7.1

Table 5.1: Overview of the number of parameters and the model memory.

because we need to evaluate gradients with regard to the activations of the forward pass. This is what we call **activation memory**.[3]

After each training step, the parameters get updated via some form of gradient descent, thus for each parameter, the gradient has to be stored. We call this the **gradient memory**. The Gradient memory is typically the same as the model memory. We are using the ADAM optimizer, so the gradient memory is increased by a factor of three as we have to store past gradients, as explained in the section 4.3.

Memory requirements during inference
The memory cost for the activations is drastically reduced during inference compared to training. On the one side, because we do not need the backward pass, which reduces the memory because gradients do not need to be calculated (and evaluated), and thus we do not need to store activations from intermediate layers during the forward pass. This comes in handy because this mean that the computational requirements for the event reconstruction on site are not that big compared to the computational requirements needed for training.

5.4.1 Memory requirements of the Transformer

We will now calculate the model memory and activation memory for individual layers of the Transformer. For the model memory, we need to count the number of weights in each layer, which was done in table 5.1 (based on the hyperparameters from Table 5.3).

As an example let us calculate the number of weights in the MLP. We have two weight matrices of shape $\mathbb{R}^{C,4C}$ and $\mathbb{R}^{4C,C}$. Thus in total $2 \cdot 4C^2 \cdot N_{\text{blocks}}$ parameters. For $C = 192$ and 4 Transformer blocks this equals 1179648 parameters. We can see that the MLP needs the most amount of memory, but overall the memory requirements of the model is with $\approx 7\,\text{MB}$ rather small.

[3]The activation memory depends on the batch size.

Activations	$\sim$ Memory L=100 / MB	L=6588 / MB
Embedding	0.0768	5
Pos. Encoding	0.0768	5
Attention matrix	2.8	4400
MLP	1.8	120
Total	4.6	4530

Table 5.2: Overview of the activation memory for an event with 100 triggered tanks and for 6588 triggered tanks.

For the activation memory we need to count the number of activations in each layer, which we have summarized in Table 5.2. We can see that for a sequence length of 100 the activation memory is only 4.6 MB, however for a sequence length of 6588 (the number of tanks in our SWGO config) the activation memory is 4.5 GB.[4] Most of the memory is needed to store the attention matrix which scales quadratically with the sequence length. For that reason we need to think about an efficient training pipeline for our transformer networks, the subject in the next section.

5.5 Efficient training of transformer networks

The array configuration for SWGO has 6588 water tanks, as shown in Figure 3.1 earlier. This means that the maximum sequence length can be 6588 if every tank registers a signal. As a comparison, the LLAMA2 LLM has a context length of only 4096 [98]. This means that the computational complexity, with respect to the length of the input, for the reconstruction of the largest events is larger than for a SOTA LLM from 2023.

We can take advantage of the fact that extremely energetic, and thus often large, events are rare. The histogram in Figure 5.3 shows the distribution of the number of triggered tanks for all simulated air showers. Here we can see that most events have only a small number of triggered tanks, and there are only a handful of events that trigger the whole detector. This point is marked by the red vertical line.
The blue line shows roughly where the trigger is set. Roughly because we require that more than 25 PMTs have measured a signal, but due to the double layer tank design, we can have events passing the trigger with less than 25 tanks because both PMTs in some tanks measured a signal. To efficiently train a Transformer

[4]In the actual implementation of attention an algorithm is used that is called flash attention [97]. This lowers the computational complexity slightly, resulting a memory requirement slightly less than what is given here.

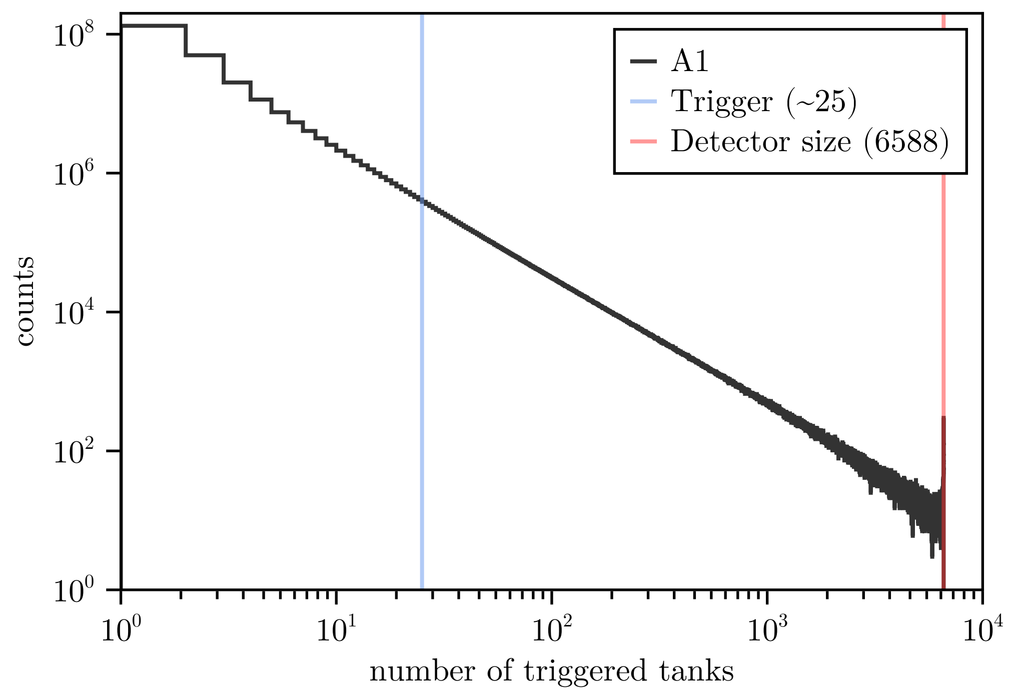

Figure 5.3: Distribution of the number of tanks triggered in γ-ray events.

model a tailored training pipeline is needed which exploits the sparseness.

If we just use the same training's algorithm (Algorithm 1) this would be extremely inefficient, because we have already mentioned that inside one batch of random samples, the sequence length needs to be matched to the largest events. That means if most of the events within the batch have about 25 triggered tanks, but one event has 6000 triggered tanks, each of the small ones need to be matched to the largest event. So the largest event inside a batch will determine the computational complexity of the training [99].

5.5.1 Sequence bucketing

A method for training, that accounts for inputs with large variations in their sequence length, is called **bucketing**. We found this idea in a kaggle competition that was called "IceCube – Neutrinos in Deep Ice" [100]. Besides this, there is not much literature on this topic. We have found some papers concerned with the training of recurrent neural networks [99] and some on **curriculum learning**, which is a familiar concept [101].

The main idea is as follows. Instead of padding every event within a batch with zeros to match the largest event, one first sorts the events by their sequence length. Then, these events are organized into different buckets. To make this more concrete, let's consider the following example.

In the case of SWGO the spectrum of sequence lengths ranges from about 25 to 6588. Let's say we use four different buckets. We fill the first bucket with events that have a maximum sequence length of 100. The next bucket contains events with a sequence length of 100 to 1000. The third bucket holds events from 1000 to 4000 and the last one contain the remaining events. Then, within each bucket, events are padded to the length of the largest event in that bucket. Thus, an event with a sequence length of 25 will only be padded to a length 100, instead of possibly 6588.

Once everything is sorted, we loop over the different buckets, calculate gradients for each bucket, and store them. The parameter update is performed using a weighted average over all gradients. The weighted average ensures that each event contributes equally in the update. This procedure is summarized in pseudocode in algorithm 2. In this example, static bucket boundaries are used (100,

Algorithm 2 A training loop with bucketing.

```
for $i$ in $1, \dots, n_{\text{epochs}}$ do
    for buckets in trainset do
        for $(X, Y)$ in buckets do
            $\hat{Y}$ = model(X)
            $\mathcal{L} = \frac{L_{\text{bucket}}}{B}$ loss(Y, $\hat{Y}$)
            $\nabla_\theta$ += gradients($\mathcal{L}$)
        end for
        update!(model, $\nabla_\theta$)
    end for
end for
```

1000, 4000, 6588). In principle this can be optimized by not setting fixed boundaries for each randomly drawn batch. In [99] a dynamic bucketing strategy is discussed, where they present an algorithm that finds the optimal bucket sizes. For this work, static boundaries were used, but in future work, dynamic bucketing could be a good option to further reduce the computation required for training.

5.6 Hyperparameters

In Table 5.3 we have listed the basic hyperparameters for the Transformer architecture. Many of them were used throughout all different training tasks, like the batch size, the embedding dimension, the number of heads, and the number of Transformer blocks.

Parameter	Value
Batch size B	128
Learning rate	0.0003
Embedding dimension C	192
Number of heads	6
Number of Transformer blocks	4
Weight decay	0.1

Table 5.3: Chosen hyperparameter for our models.

We have used a linear warm up and cosine learning rate decay. The linear warm up assists in stabilizing the training [102].

For regularization, a high weight decay of 0.1 was used. Dropout, if used, was applied after the linear layers in the MLP and before matrix multiplication (AV) during the attention operation.

Chapter 6

Evaluation of a transformer based event reconstruction

In this chapter we will evaluate the use of a Transformer for energy reconstruction, γ/hadron separation, and direction reconstruction. We will compare our method to performance goals set in the SWGO science case and to techniques that were introduced in chapter 3.
The training of the Transformer neural networks was done on ≈ 1 millions events. For validation 10 %, relative to the number of training events, were used and in the end, the performance was evaluated on $\approx 400{,}000$ events that were never used during training.

Events were only used for training and testing if more than 25 PMTs measured a signal. The evaluation of direction and energy reconstruction was done with the following cut: only events where the core landed on the array ($R < 300\,\mathrm{m}$) and where the zenith angle angles was below 45° were used. [1] For γ/hadron separation proton and γ-ray simulations, at a ratio of $\approx 50/50$, were used. Here no cuts on the data were applied.

Resources for training transformer networks
We want to address a critical aspect of using Transformer networks: their energy consumption. The transformer architecture is build in a way that supports parallelism and scalability. This allows the training of extremely large models, where the number of parameters is typically in the billions. For example GPT-3, the predecessor of ChatGPT, has 175 billion parameters [13].

However, at the same time, Transformers "lack some of the inductive biases inherent to CNNs, such as translation equivariance and locality, and therefore do not generalize well when trained on insufficient amounts of data."[91] So-called scaling laws show that increasing the model size, i.e. the number of parameters, along with the size of the training set, improves the model's performance. This

[1]Here the true core and true zenith angle are used.

performance increase follows a saturating power law with respect to the amount of compute, meaning that achieving further performance gains becomes increasingly expensive [103].
To set the energy consumption into perspective, let's take a look at the LLAMA2 model. The LLAMA2 LLM (70 billion parameters) released in 2023, was trained on a cluster using 2000 Graphical Processing Unit (GPU)s in parallel, resulting in an energy consumption (for the GPUs only) of 0.7 GW h for one training run [98]. The successor LLM LLAMA3 (70 billion parameters) needed 4.5 GW h of electrical energy for one training run, which is more than a sixfold increase in only a year[104].

Our models, with roughly 1 million parameters, can be trained, due to the bucketing strategy, on a single A40 GPU. We have tracked all training runs that we did, together with system information like for example the GPU power usage. From this we estimated the total energy consumption for all our training runs to 564 kW h (GPU only).

6.1 Energy reconstruction

For the energy reconstruction, a final head was utilized, which maps the cls-token to a scalar value $\log_{10}(E)/\mathrm{GeV}$. Mathematically the final layer looks like this,

$$Y = \mathrm{Layernorm}(Z_0')U_E + \langle \log_{10}(E_{mc}) \rangle, \tag{6.1}$$

where $U_{\mathrm{E}} \in \mathbb{R}^{C,1}$ and $\langle \log_{10}(E_{mc}) \rangle$ is the mean logarithmic energy of the training set.

For the loss function, a mean squared error was used

$$L = \frac{1}{N} \sum_{n}^{N} || \log_{10}(E_n/\mathrm{GeV}) - \log_{10}(E_{\mathrm{mc},n}/\mathrm{GeV}) ||^2, \tag{6.2}$$

where the sum runs over the number of examples inside the batch.
The training was done for 105 epochs, where the learning rate was reduced from 0.003 down to 0.0001 at epoch 100. In Figure 6.1 the trainings and validation losses during the training are shown.

The energy construction was evaluated in the energy range from 100 GeV up to 1 PeV and in energy bins that are equidistant on a logarithmic axis. To evaluate the performance of the energy reconstruction, we calculate the so-called **energy bias** and the **energy resolution**.

These are defined as follows: if we take a look at Figure 6.2, we can see the distribution of the differences between $\log_{10} E$ and $\log_{10} E_{\mathrm{mc}}$ for energies E_{mc} between 1 TeV and ≈3.16 TeV. So basically, the differences between the prediction

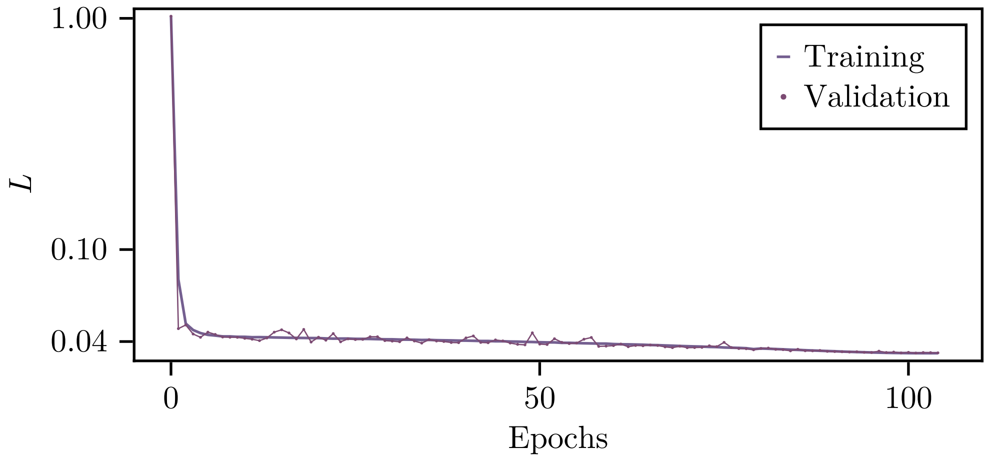

Figure 6.1: Training (solid line) and validation (dotted line) losses of the Transformer trained for energy reconstruction.

of the Transformer and the true Monte Carlo values. The energies bias is defined as the mean of this distribution and the energy resolution as the standard deviation. Ideally, the bias should be zero, and the energy resolution should be as small as possible.

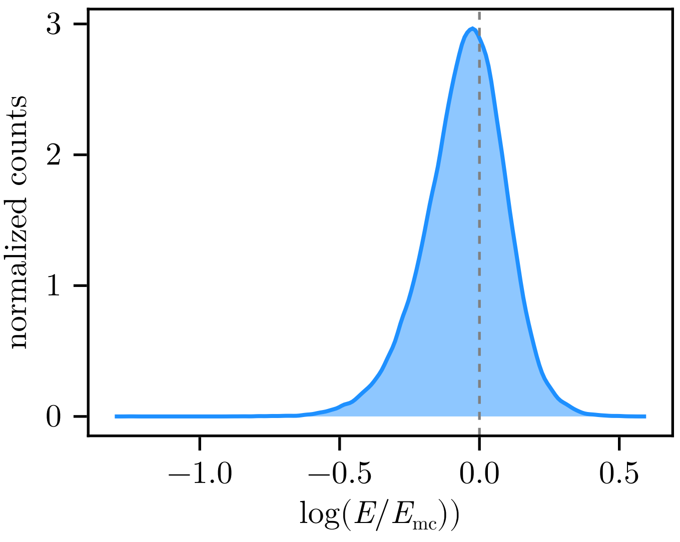

Figure 6.2: Distribution of differences between the predicted logarithmic energy values and Monte Carlo values, for energies between 1 TeV and ≈3.16 TeV. The counts are normalized such that the area under the curve is one.

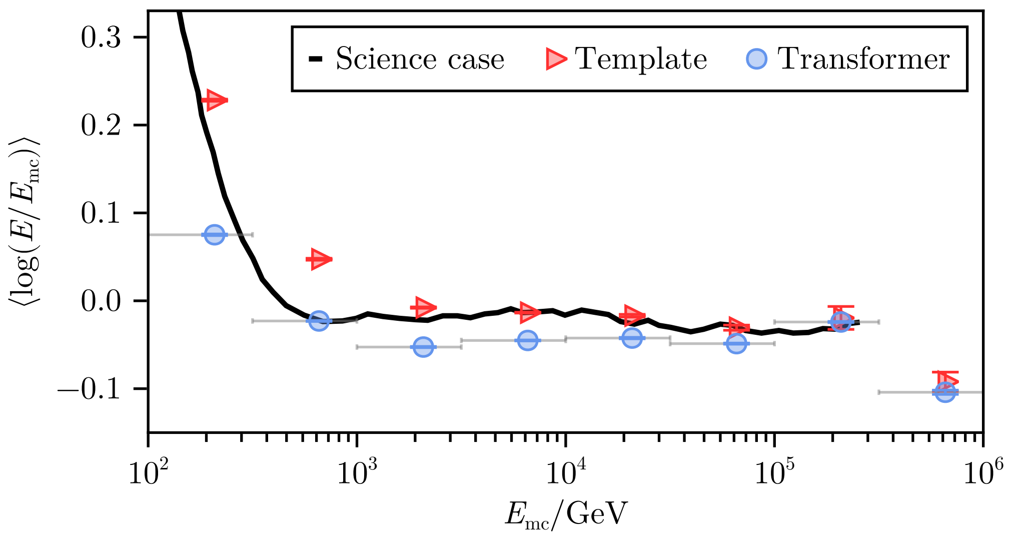

Figure 6.3: Energy bias for different reconstruction methods. A template based method is depicted in red and the Transformer based method in blue. The black line shows the expected energy bias from the SWGO science case (adopted from [8]). The horizontal gray error bars indicate the selected energy bins.

In Figure 6.2 we can see that the mean is slightly shifted towards left, which means the prediction of the Transformer slightly underestimates the energy within this energy range.

In Figure 6.3 and Figure 6.4 the energy bias and the energy resolution are shown for all energy bins. Besides the results of the Transformer method, expected performance goals from the SWGO science case are shown and the results of the template based reconstruction.
In Figure 6.3 we can see that at low energies the bias for both methods is large and energies are overestimated. However, the overestimation for the Transformer is lower than for the templates. The bias at low energies is an expected behavior, as close to the energy threshold of the detector, we can only measure upwards fluctuations from the mean, as lower energetic events will have a limited chance of being detected. In the energy range from 1 TeV to 100 TeV the Transformer slightly underestimates the energy.

Figure 6.4 shows that the energy resolution of the Transformer is better than the one for the templates over the whole energy range. At low energies the Transformer achieves an energy resolution of about 66 %[2]. Above 10 TeV, the energy resolution is less than 15 %. With our method we also surpass the expected energy resolution of the SWGO science-case [8].

[2]Calculated without logarithms as $\sigma(E-E_{mc}/E_{mc})$.

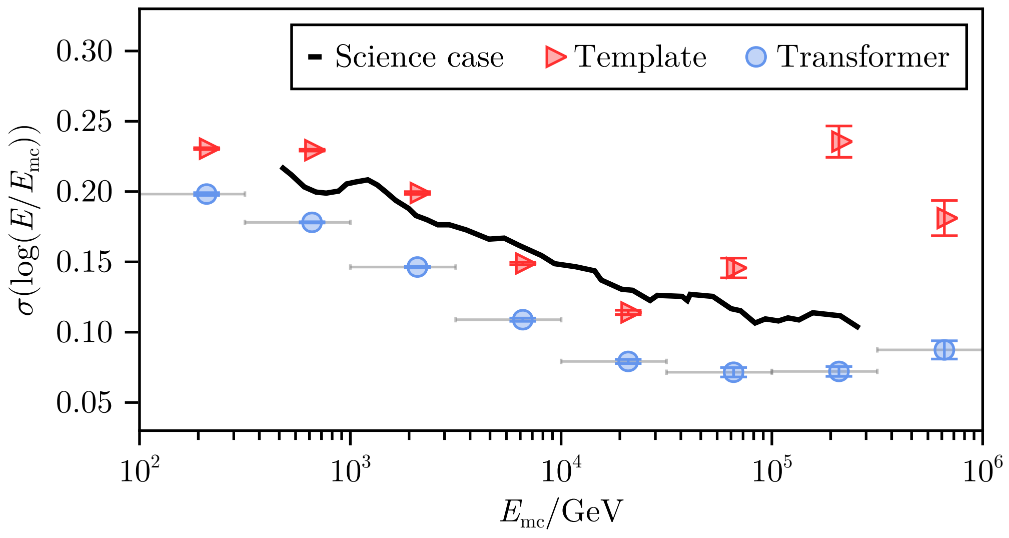

Figure 6.4: Energy resolution for different reconstruction methods. A template based method is depicted in red, and the Transformer based method in blue. The black line shows the expected energy resolution from the SWGO science case (adopted from [8]). The horizontal gray error bars indicate the selected energy bins.

6.2 γ/hadron separation

For the γ/hadron separation a final head was used, which maps the cls-token to a two dimensional vector, like this[3]

$$Y = \mathrm{softmax}(\mathrm{Layernorm}(Z_0')U_{\mathrm{ghs}}) \tag{6.3}$$

with $U_{\mathrm{ghs}} \in \mathbb{R}^{C,2}$. The softmax function is used to map the output of the linear layer to probabilities.

As a loss function, a cross entropy loss was used

$$L = -\frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{2} Y_{n,i}^{\mathrm{mc}} \log(Y_{n,i}). \tag{6.4}$$

The first sum runs over the number of examples N inside the batch and the second runs over columns in Y, where $Y_{n,1}$ is the probability that the event was a γ-ray and $Y_{n,2}$ the probability that the event was a proton. In Figure 6.5 the training and validation losses during the training are shown. The γ/hadron separation was evaluated in the same energy range that was used for the energy

[3]In the software implementation the softmax function is applied directly within the loss function, as this is numerically more stable.

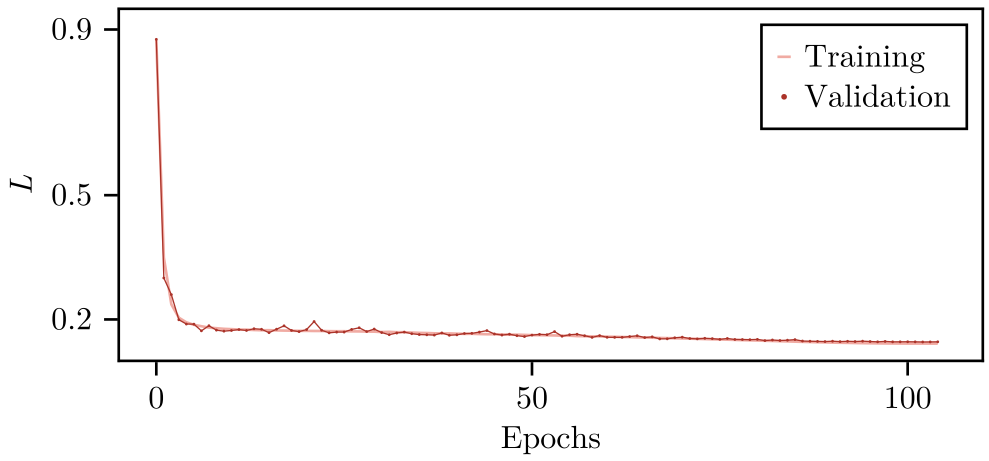

Figure 6.5: Training (solid line) and validation (dotted line) losses of the Transformer trained for γ/hadron separation.

reconstruction. Here not Monte Carlo energy bins were used but energy bins from the reconstructed energy (template).

In Figure 6.6 a histogram is shown, which is filled up with the probabilities $p_\gamma = Y_{i1}$ for all events in the energy range from $E_{\text{reco}} = 1\,\text{TeV}$ to $E_{\text{reco}} \approx 3.16\,\text{TeV}$. The gray line shows the value, which determines for which probability p_γ, we classify an event as a γ-ray event. This cut value is set in such a way, that the gamma efficiency is $\epsilon_\gamma = 0.8$. This means events that lie to the left of the black line are classified as background and all events to the right as γ-rays. The background efficiency ϵ_{bkg} (also called background rejection) is defined as the number of correctly classified background events divided by the number of all background events. 1 - ϵ_{bkg} is called the background contamination.

In the Figure 6.7 the background contamination for the Transformer and for the Graph Neural Network (GNN), is shown.[4] In the upper plot we can see that both methods have a similar performance in γ/hadron separation. At low energies, we have a background rejection of $> 95\,\%$. For energies between 1 TeV and 30 TeV a background rejection of $\geq 98\,\%$. Above 30 TeV greater than 99.9 %. For energies $> 100\,\text{TeV}$ no background is left.

In the lower plot the ratio between the proton background of the Transformer and the GNN is shown. For low energies, we can see that the GNN retains $\approx 10\,\%$ more background. For higher energies, both methods perform equally well, however the error bars are quite large.

[4]In that comparison it was assured that during training the same events were used. The test dataset was also the same for the Transformer and GNN.

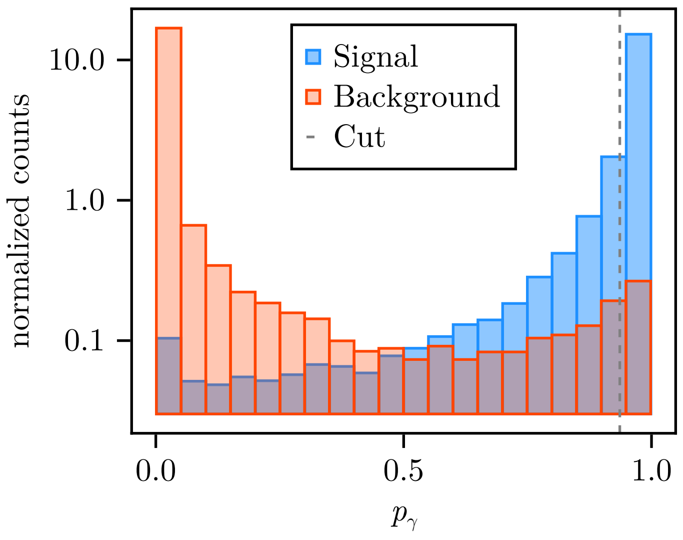

Figure 6.6: Distribution of the predicted γ-ray probabilities from the Transformer. Protons (background events) are highlighted in red and γ-rays (signal events) in blue. The gray line shows the cut value, which determines at which probability an event is classified as signal. Counts are normalized such that the area of the background counts and the area of the signal counts are one.

The error bars in Figure 6.7 were calculated from a Bayesian approach, as discussed in [105]. If a background was left the error bars were calculated as the standard deviation of the probability distribution as given in equation 12 in [105]. The upper limits are calculated at a 95 % confidence level.

6.3 Direction reconstruction

In the direction reconstruction the output of the Transformer should be a unit vector pointing in direction of the shower axis. Thus, the following final head was defined for this last task

$$Y = \text{normalize}\left(\text{Layernorm}(Z_0')U_{\text{dir}} + \langle Y_{mc}\rangle\right), \tag{6.5}$$

with $U_{\text{dir}} \in \mathbb{R}^{C,3}$ and $\langle Y_{mc}\rangle$ the mean direction for the whole training set.

As for the energy reconstruction a mean squared error was used

$$L = \frac{1}{N}\sum_{n}^{N} ||Y_n - Y_n^{\text{mc}}||^2. \tag{6.6}$$

The sums runs over all events within a batch. Y_n is the prediction of the Transformer and Y_n^{mc} the Monte Carlo truth. In the earlier tasks a simple training

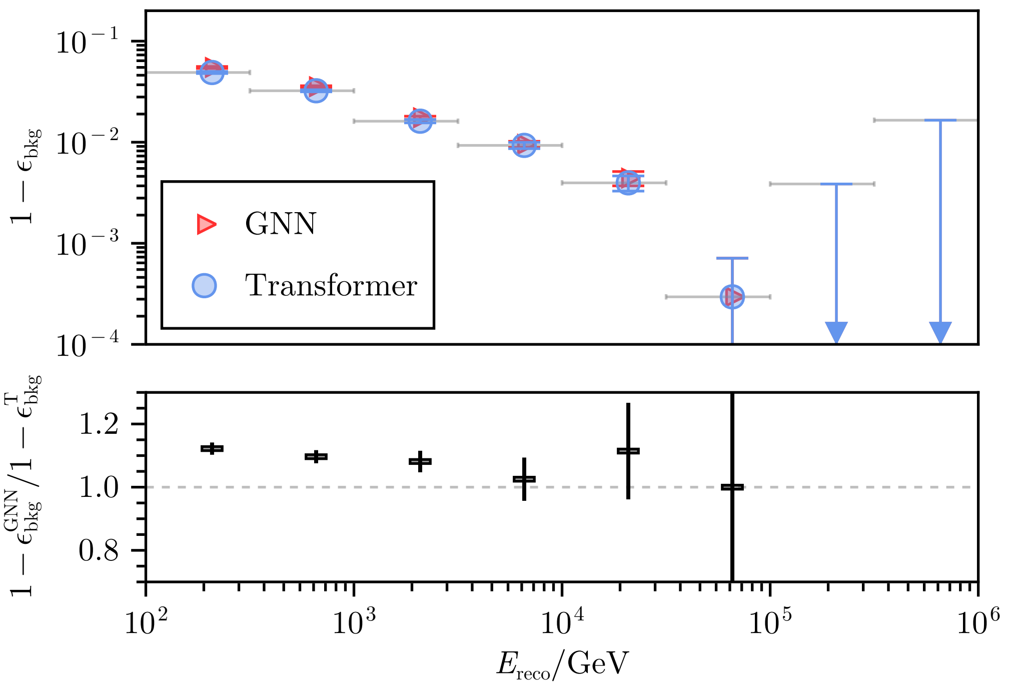

Figure 6.7: γ/hadron separation performance for the Transformer (in blue) and GNN (in red). The upper plot shows the background contamination for both methods. In the last two energy bins no background is left, therefore an upper limit at a 95 % confidence level is shown. The horizontal gray error bars indicate the selected energy bins. The lower plot highlights the ratio of the background contamination of both methods.

procedure was used. Training was done for 100 epochs with a learning rate decay and all events with more than 25 hits were used. The same procedure did not achieve desirable results for the direction reconstruction (these results are shown in Appendix C).

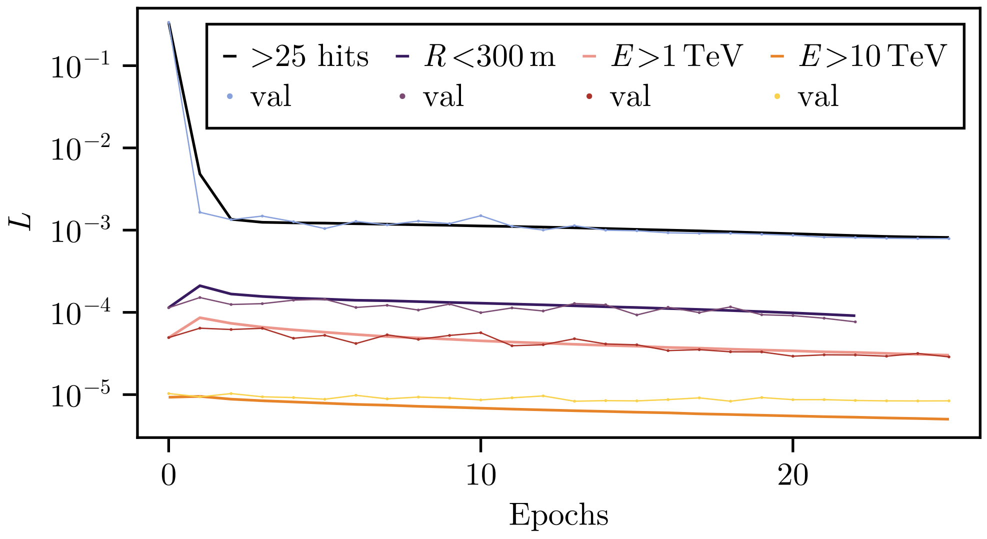

Figure 6.8: Multiple training (solid line) and validation (dotted line) losses of a Transformer trained for direction reconstruction[6].

Instead a so-called **fine-tuning** was done, where a pretrained model is used as a starting point, and this model is trained again on some specific dataset. In our case the following was done: first, a Transformer was trained on events with more than 25 hits. Then this pretrained network was fine-tuned on multiple subsets of events. First on events where the (MC) core landed on the detector ($R < 300$m) and then with events where the (MC) core landed on the detector with energies larger than 1 TeV. In the last fine-tuning step, events where the (MC) core landed on the detector and with (MC) energies larger than 10 TeV were used.

In Figure 6.8 all loss curves for this process are shown. The figure illustrates why it did not work as well without fine-tuning. There are orders of magnitude differences between the losses for the different subsets. That means by training on all events (>25 hits), gradients are dominated by low energy events, where the core landed outside of the detector.

To evaluate the event reconstruction the angles between the Monte Carlo di-

[6]The first fine-tuning training ($R < 300$m) was interrupted 3 epochs to early due to a time limit.

rection vector and the predicted vector are calculated like this

$$\Delta = \arccos{(\hat{Y} \cdot \hat{Y}^{\mathrm{mc}})}. \tag{6.7}$$

In Figure 6.9 the distribution of Δ is shown for events with energies between $E_{\mathrm{mc}} = 1\,\mathrm{TeV}$ to $E_{\mathrm{mc}} \approx 3.16\,\mathrm{TeV}$.

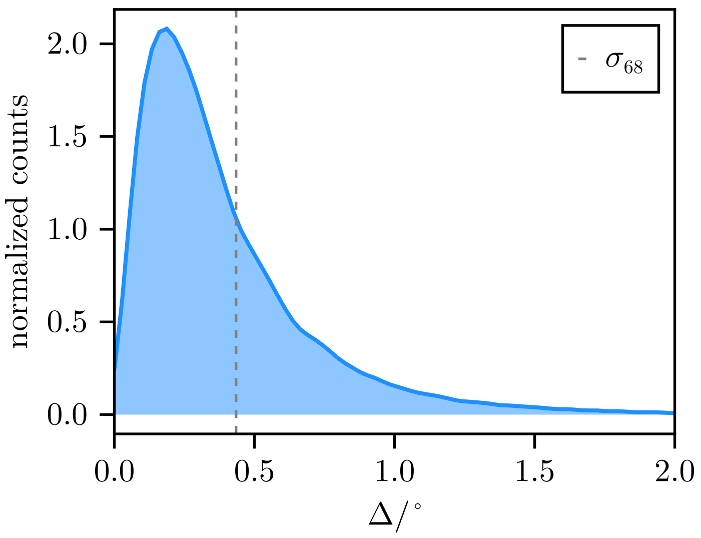

Figure 6.9: Distribution of the angles between predicted directions and the true Monte Carlo directions. The gray dashed line marks the 68 % quantile of the distribution.

We can see that in this figure, the maximum of the distribution is close to zero. The angular resolution, which we take as a measure of the precision of the direction reconstruction, is defined as the 68 % quantile of this distribution (marked by the black line in Figure 6.9).

In Figure 6.10 the angular resolution as a function of the (MC) energy of the different methods is compared. For each method, two lines are shown, because the angular resolution is evaluated separately for events where the (MC) core landed on the inner part of the detector and where the (MC) core landed on the outer part of the array. Black lines indicate the expectations of the SWGO science case. We compare our method in blue, to the planefit in red. For events on the inner array, the Transformer and the planefit have a similar performance and both methods are within the expectations of the science case. For these events the Transformer achieves an angular resolution of less than 0.1° for energies between 10 TeV and 100 TeV. For outer events, the Transformer has a better angular resolution than the planefit over the whole energy range. The angular resolution of the Transformer even exceeds the expectations of the science case. Even for these outer events where the shower sampling is worse due to smaller

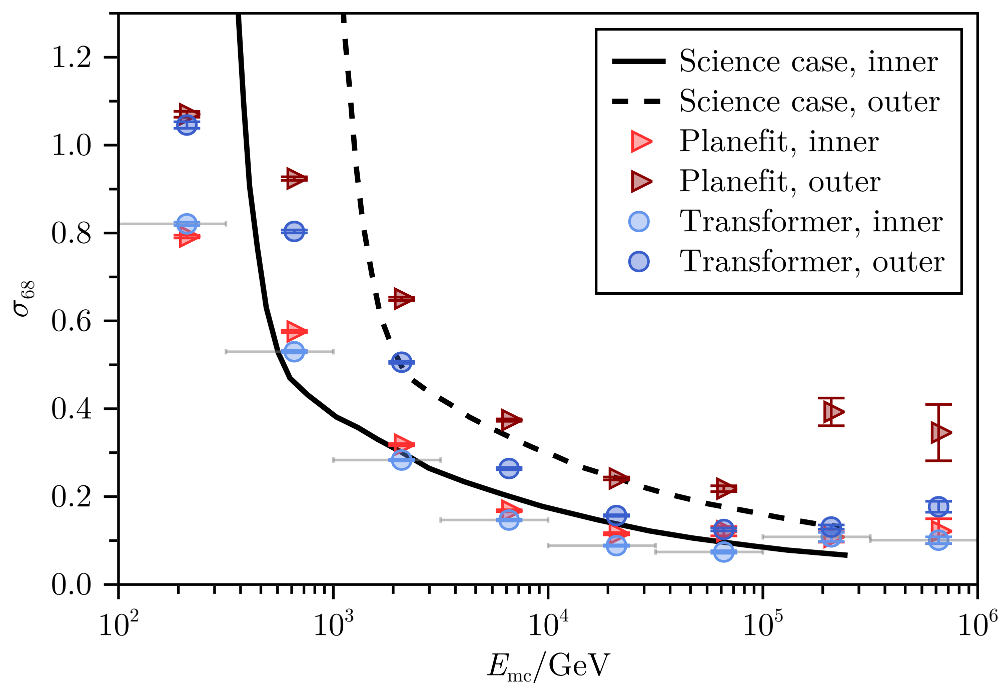

Figure 6.10: Plot of angular resolutions for multiple methods. In black we draw the targeted resolution in the SWGO science case (taken form [8]). In red the results from the plane fit are shown and in blue our results from the Transformer. The angular resolution for each method is separately shown for events where the core landed on the inner part and for events where the core landed on the sparse outer part of the array. The horizontal gray error bars indicate the selected energy bins.

density of tanks, an angular resolution close to 0.1° at energies above 100 TeV was possible.

The increase in angular resolution at the highest energies is probably due to the lack of simulations that were available for the training.

In a paper from 2020, limits on the angular resolution were discussed [49]. The angular resolution was estimated for a perfect ground-based particle detector array, which can be used as a benchmark. At an energy of 100 GeV the authors estimated a best possible angular resolution of $\approx 0.6°$. At 10 TeV $\approx 0.02°$ and at 100 TeV $\approx 0.004°$ (values taken from Fig.5 in [49]). Comparing these values to the angular resolution of our method (for inner events), we see that, at high energies, our achieved resolution falls significantly short of the maximum possible one. However at 10 TeV and lower energies, the angular resolution is within a factor of two or three of the ideal angular resolution.

6.4 Robustness study: Detector aging effects

Over a longer operating time of the detector, aging effects of individual detection units can occur. Some units may get turned off for a while or are defect. In this section we like to study the behavior of the Transformer based event reconstruction, in situations where a fraction of the tanks are turned off or are not working.

As a first test, the trained networks, which we have evaluated in the earlier section, are evaluated again, but this time with a detector where a fraction of tanks were turned off. We varied the fraction of tanks from 1 promille, which correspond roughly to 60 defect tanks, up to a 20 % defect tanks. For each fraction tanks were randomly sampled which are then turned off. Examples of this selection process are shown in the appendix D

It is expected that the performance of the event reconstruction will decrease for two reasons. First, the networks are not trained specifically to be evaluated on a different detector, where tanks are removed. Second, when tanks are defect, the shower is sampled worse, thus less information is available for the reconstruction.

Robustness study: energy reconstruction
In Figure 6.11 we show the result of the inference of the Transformer network on detectors with defect tanks. Each row corresponds to a detector with a different amount of defect tanks. Starting at the top in blue with the least amount of defect tanks and in yellow the most amount. Each row contains the same black lines, which are the bias and resolution for a detector without defect tanks. Barely seeable around the black lines is a band in gray, indicating the error bars.

If we first look at the bias, we can see that as the number of defect tanks is increased the bias is shifted towards larger negative values, which means the energy of events is underestimated. This is an expected behavior, as the total amount of charge induced in all PMTs is a rough estimate for the energy. If we take out tanks, we will have a worse sampling of the footprint and less overall induced charge.

The energy resolution is mostly uneffected. A closer look reveals that for less than 5 % defect tanks, the energy resolution is worsened by less than 1 %. At 20 % defect tanks the energy resolution is worsened by less than 5 %.

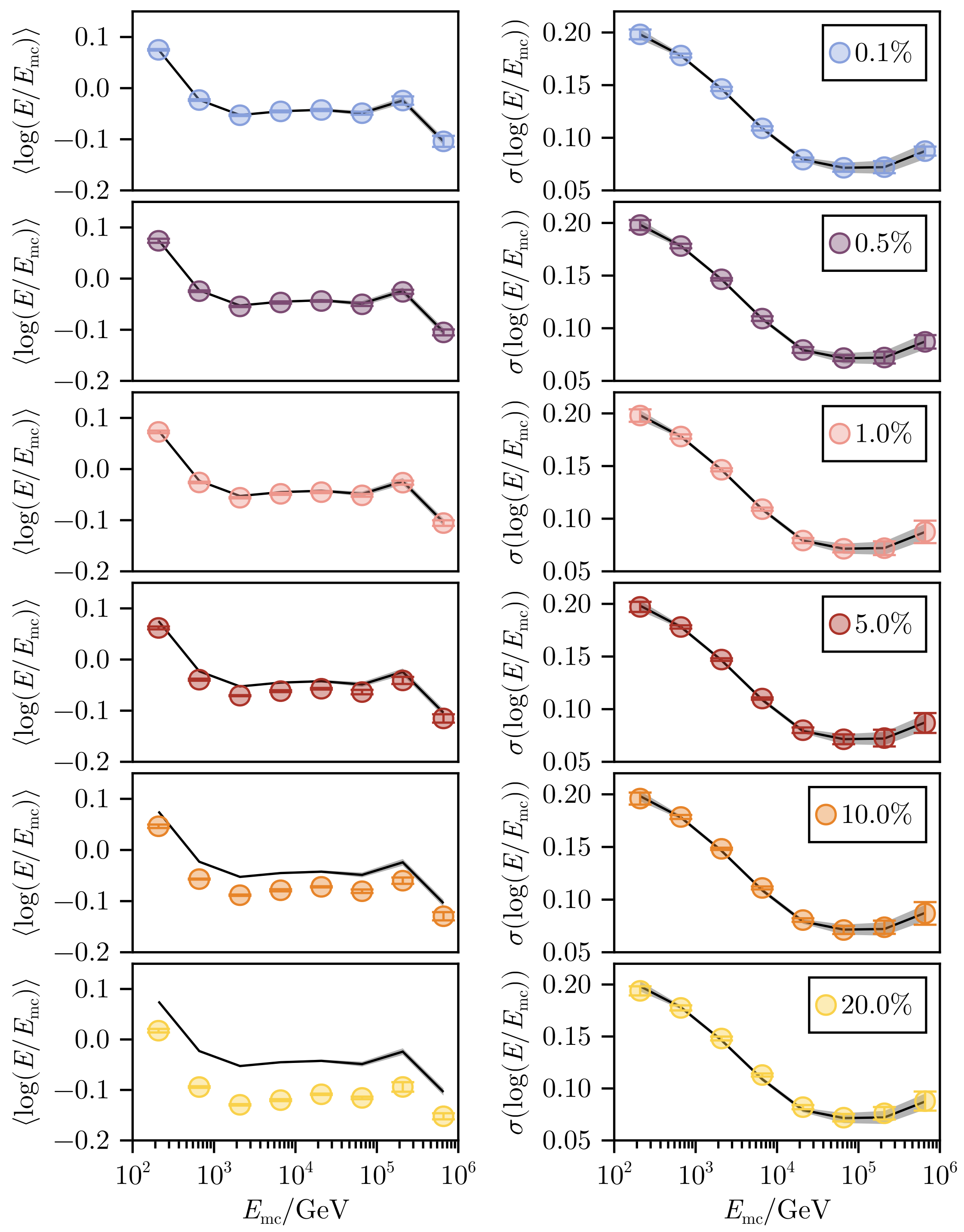

Figure 6.11: Performance in energy reconstruction on detectors where a certain percentage of tanks is turned off. In each row a different percentage of tanks is turned off. The left side shows the bias and the right side the energy resolution. The black line marks the angular resolution for an array without defect tanks.

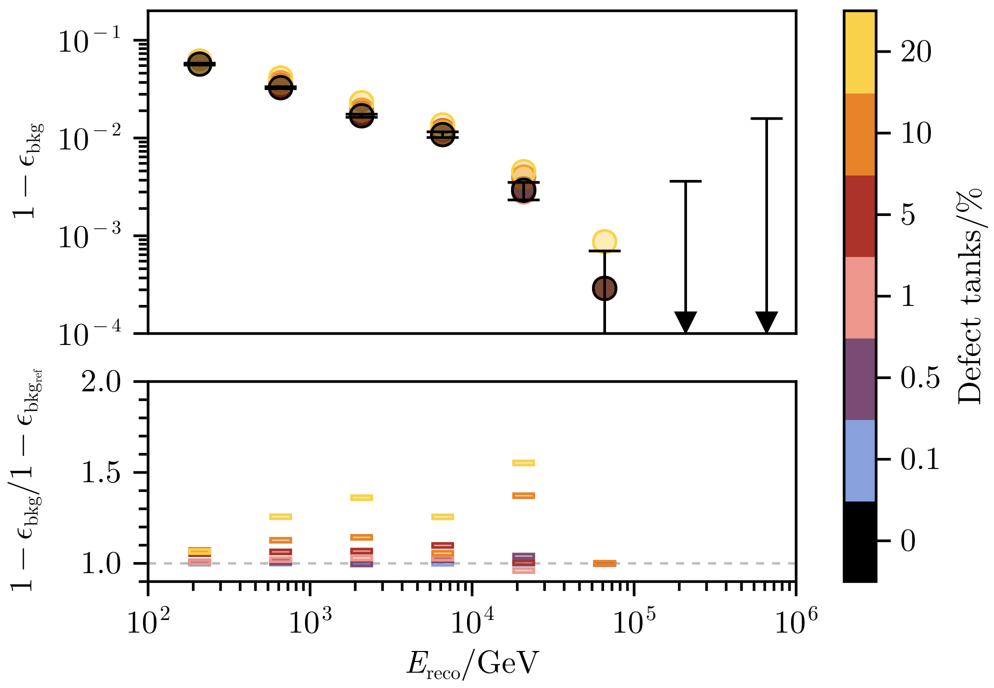

Figure 6.12: Performance of the γ/hadron separation on arrays where a certain percentage of tanks is turned off. The upper plot shows the background contamination, in black for the array without defect tanks and in colors for defect detectors. In the lower plot the ratio of backgrounds is shown.[8]

Robustness study: γ/hadron separation
We have repeated same procedure for the γ/hadron separation. The result is shown in Figure 6.12. In the upper plot the background contamination is shown. In the lower plot, the background contamination for an detector with defect tanks is divided by the background contamination of the ideal detector.

Already in the upper plot we can clearly see that for 10 % and 20 % the background is increased. In the lower plot we can see that the background for detectors with less than 5 % defect tanks, the background is at most increased by 10 %.

Robustness study: direction reconstruction.
Figure 6.13 is structured similar to Figure 6.11. In the left plots the angular resolution for inner events is shown. The angular resolution for outer events is shown on the right. The angular resolution is barely affected for arrays with defect tanks. For less than 1 % defect tanks the performance is only decreased by less than 1 %. At 20 % defect tanks, the angular resolution is worse by roughly 10 %.

[8]In the sixth energy bin the ratio for 20 % ($\approx$3) is not shown.

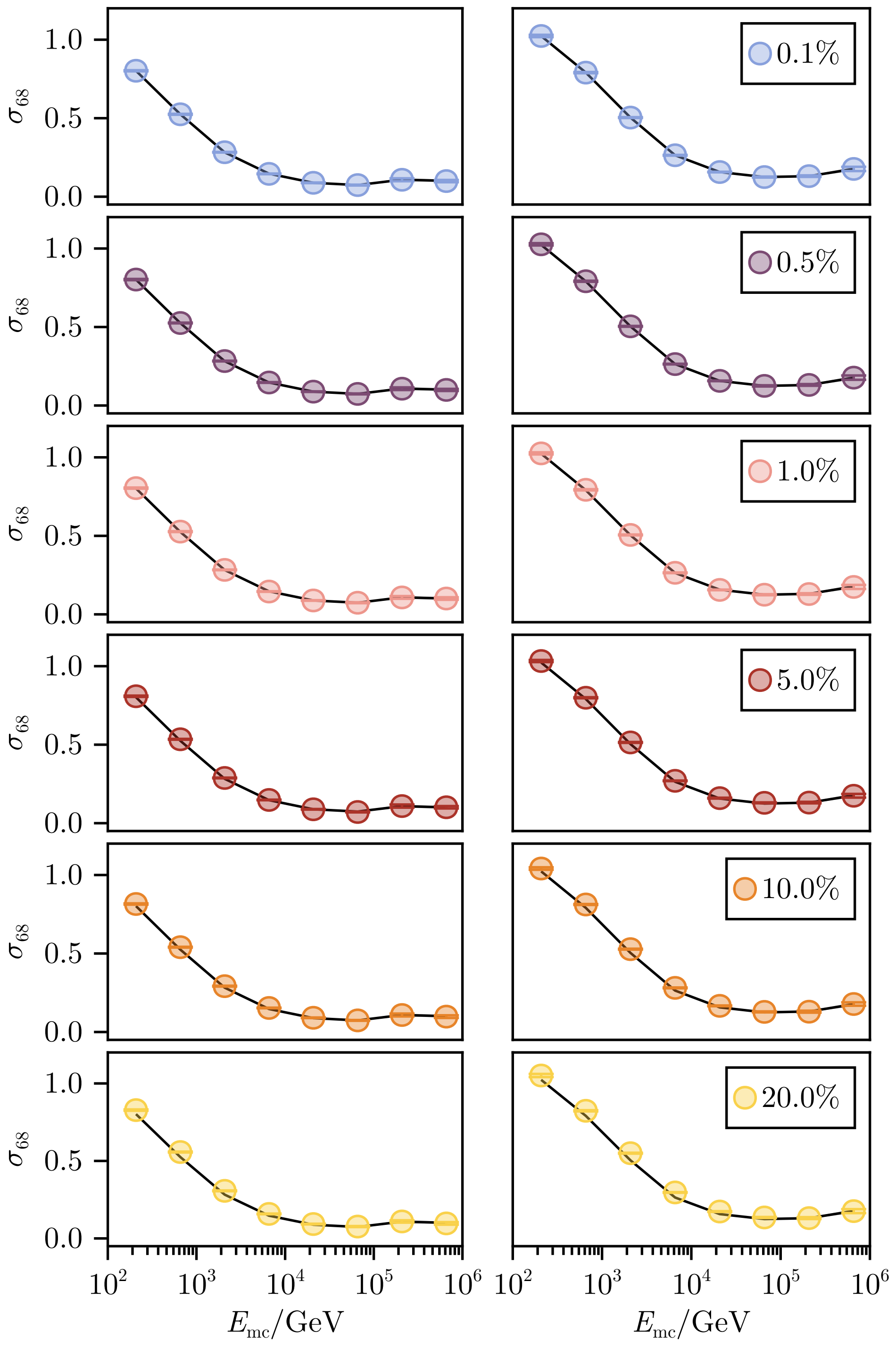

Figure 6.13: Performance in direction reconstruction on detectors where a certain percentage of tanks is turned off. In each row a different percentage of tanks is turned off. The left side shows the angular resolution for inner events and the right side for outer events. The black line marks the angular resolution for an array without defect tanks.

Chapter 7

Summary and outlook

In this work, a Transformer based method for event reconstruction was applied for SWGO, a future ground-based particle detector, which will be located in the Southern Hemisphere. In this array, individual detection units utilize the water-Cherenkov technique, observing directly the secondary particles within an air shower. If multiple PMTs within different water tanks register secondary particles, it is possible to infer the particle type that initiated the shower, and to reconstruct the direction and energy of the primary particle. The event reconstruction for such observatories is a challenging task, as only information of one horizontal slice of the shower is available.

With the Transformer it is possible to achieve state of the art performance in energy reconstruction, γ/hadron separation and direction reconstruction.
The Transformer based energy reconstruction, reaches an energy resolution of $\approx 66\,\%$ at low energies, $\approx 30\,\%$ at a few TeV, and at about 100 TeV, $\approx 15\,\%$. Thus the energy resolution is better than the standard template based method of SWGO.
The background rejection is improved to $\approx 95\,\%$ at low energies, and even outperformes the GNN in that regime. For energies above a few TeV a background rejection of greater than 99 % is possible and even better than 99.9 % at the highest energies, which is comparable to the GNN approach.
In the direction reconstruction an angular resolution of less than 0.1° is reached for energies between 10 TeV and 100 TeV (for events where the core lands on the inner array). Even for events where the core landed on the sparse part of the detector, it is possible to reach an angular resolution close to 0.1°, at energies above 100 TeV. With these results the Transformer was comparable to the standard method of SWGO, and even slightly better for outer events.

By implementing a bucketing strategy for the training of the Transformer networks, all this is possible, with only one GPU that was used during the training process, which shows that also for large sequence lengths a Transformer can be trained efficiently and thus, with a low energy footprint.

In addition, the robustness of the developed algorithm was evaluated, by turning a significance fraction of the detector off. Even without being explicitly trained for such a situation, the decrease in performance is acceptable. Angular resolution and energy resolution are barely affected. However, an increase in background is observed, especially if the fraction of defect tanks is large ($\geq 5\,\%$).

This work demonstrates that Transformers are well suited for event reconstruction for ground-based particle detectors. However, more work is needed to investigate potential differences between simulations and real data that could effect the performance. In the near future, one idea is to evaluate the potential of a combined neural network: a GNN and a Transformer. These together could open the window for an even improved reconstruction, by simplifying the extraction of local and global features. Also a revised robustness study would be interesting, where during the training process, defect tanks and additional noise are considered.

Acronyms

γ-ray gamma ray. 57

ADAM adaptive moment estimation. 32, 40, 55

CMB cosmic microwave background. 9

CORSIKA COsmic Ray SImulations for KAscade. 21

CTA Cherenkov Telescope Array. 9, 19

EAS Extensive Air Showers. 10, 15, 24, 25

GELU Gaussian Error Linear Unit. 37, 53

GNN Graph Neural Network. 66, 68, 77

GP Ground Parameter. 24

GPU Graphical Processing Unit. 62

H.E.S.S High Energy Stereoscopic System. 15

HAWC High-Altitude Water Cherenkov. 2, 9, 19, 24, 25, 26, 50

IACT Imaging Atmospheric Cherenkov Telescope. 1, 9, 14, 15, 16, 19, 26

LDF lateral distribution function. 25

LHAASO Large High Altitude Air Shower Observatory. 2, 9

LLM Large Language Model. 2, 32, 45, 47, 54, 56, 62

MC Monte Carlo. 25

ML machine mearning. 29

MLP Multi Layer Perceptron. 24, 35, 36, 38, 46, 49, 53, 55, 59

NLP Natural Language Processing. 2, 47

PINC Parameter for IdeNtifying Cosmic rays. 24, 25

PMT Photo Multiplier Tube. 15, 16, 19, 20, 21, 22, 23, 24, 25, 50, 51, 56, 61, 72, 76, 82

PWN Pulsar Wind Nebula. 8

RELU Rectified Linear Unit. 37, 46

SGD Stoachastic Gradient Descent. 32, 40

SNR Supernova Remnant. 8, 9, 26

SOTA state of the art. 2, 56

SWGO Southern Wide-Field Gamma-Ray Observatory. 2, 9, 19, 21, 24, 25, 26, 27, 31, 43, 49, 50, 52, 56, 58, 61, 64, 65, 70, 76

ViT Vision Transformer. 51, 53

Appendix A

Distributions for protons simulations

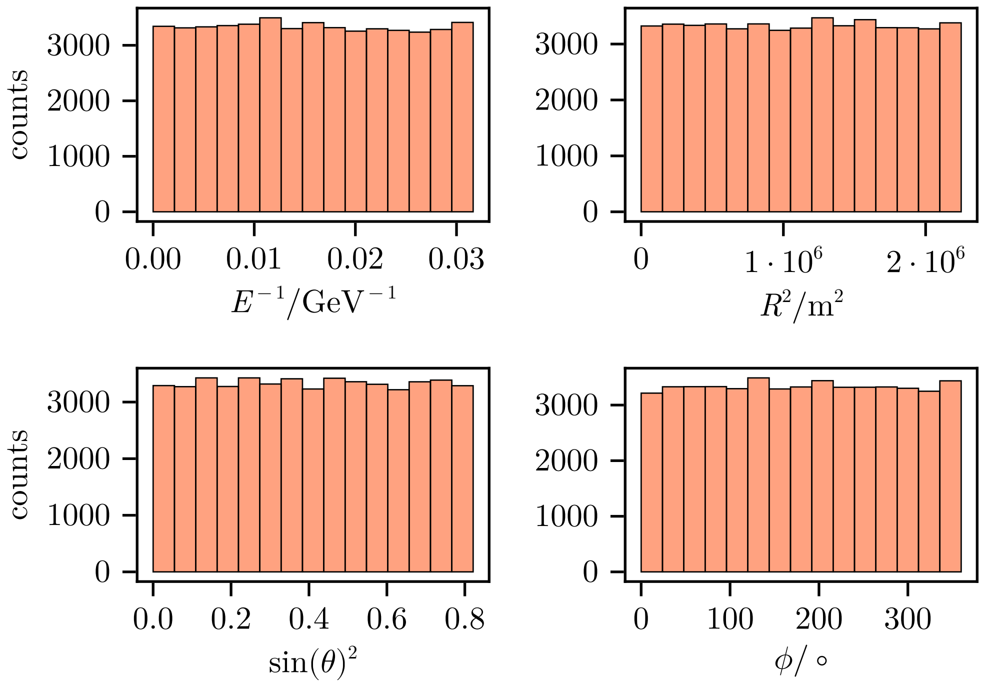

Figure A.1: Distributions of input parameters of simulated proton air showers.

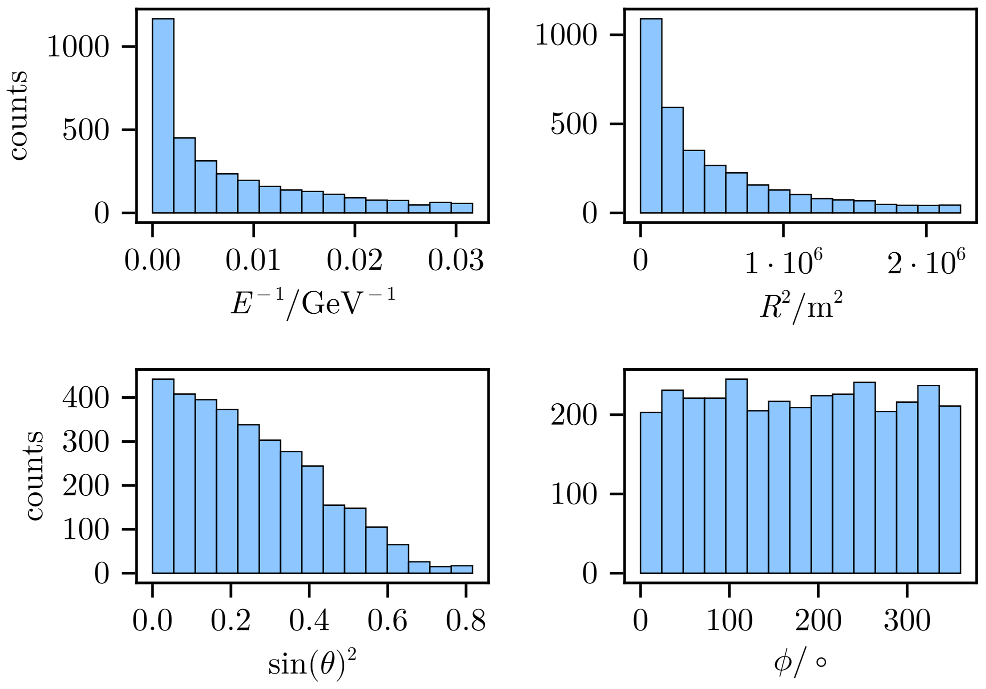

Figure A.2: Distributions of input parameters of simulated proton air showers, which were captured by at least one PMT.

Appendix B

Distributions for all energy bins

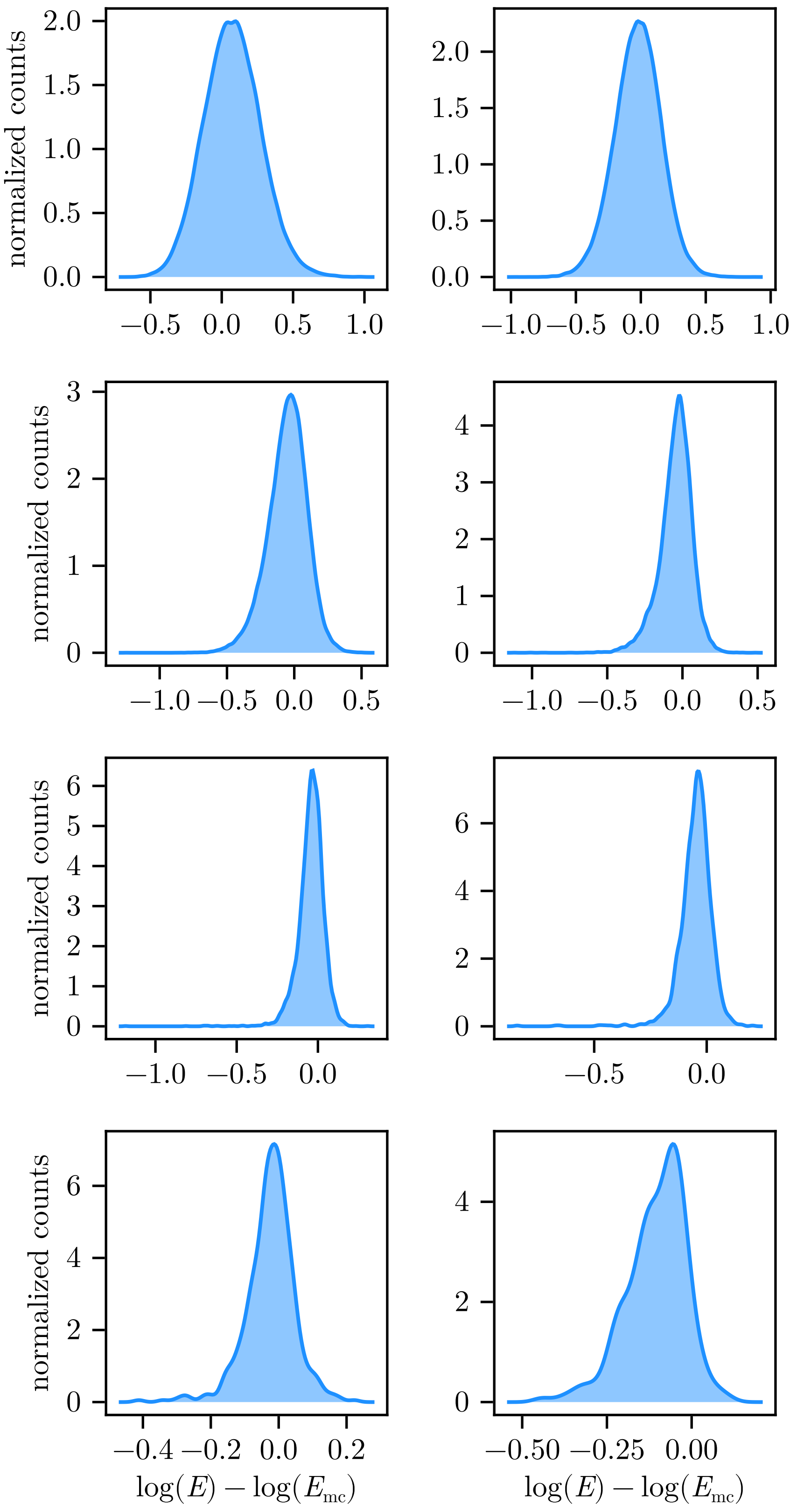

Figure B.1: Distributions, like 6.2, but for all energy bins as used in Figure 6.3. Top left is the lowest energy bin, top right the second lowest, and so on.

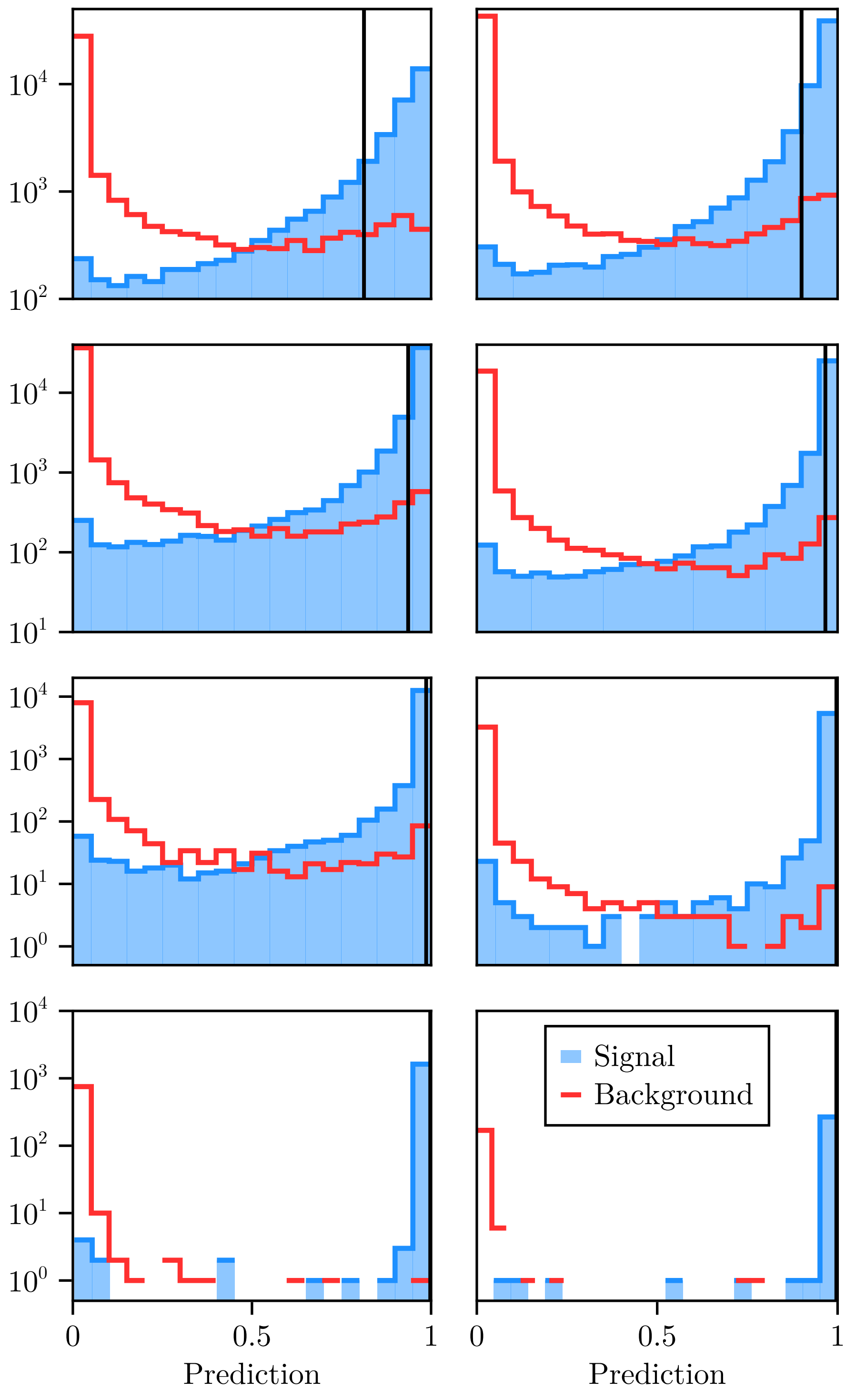

Figure B.2: Histograms, like Figure 6.6, but for all energy bins. Top left is the lowest energy bin, top right the second lowest, and so on.

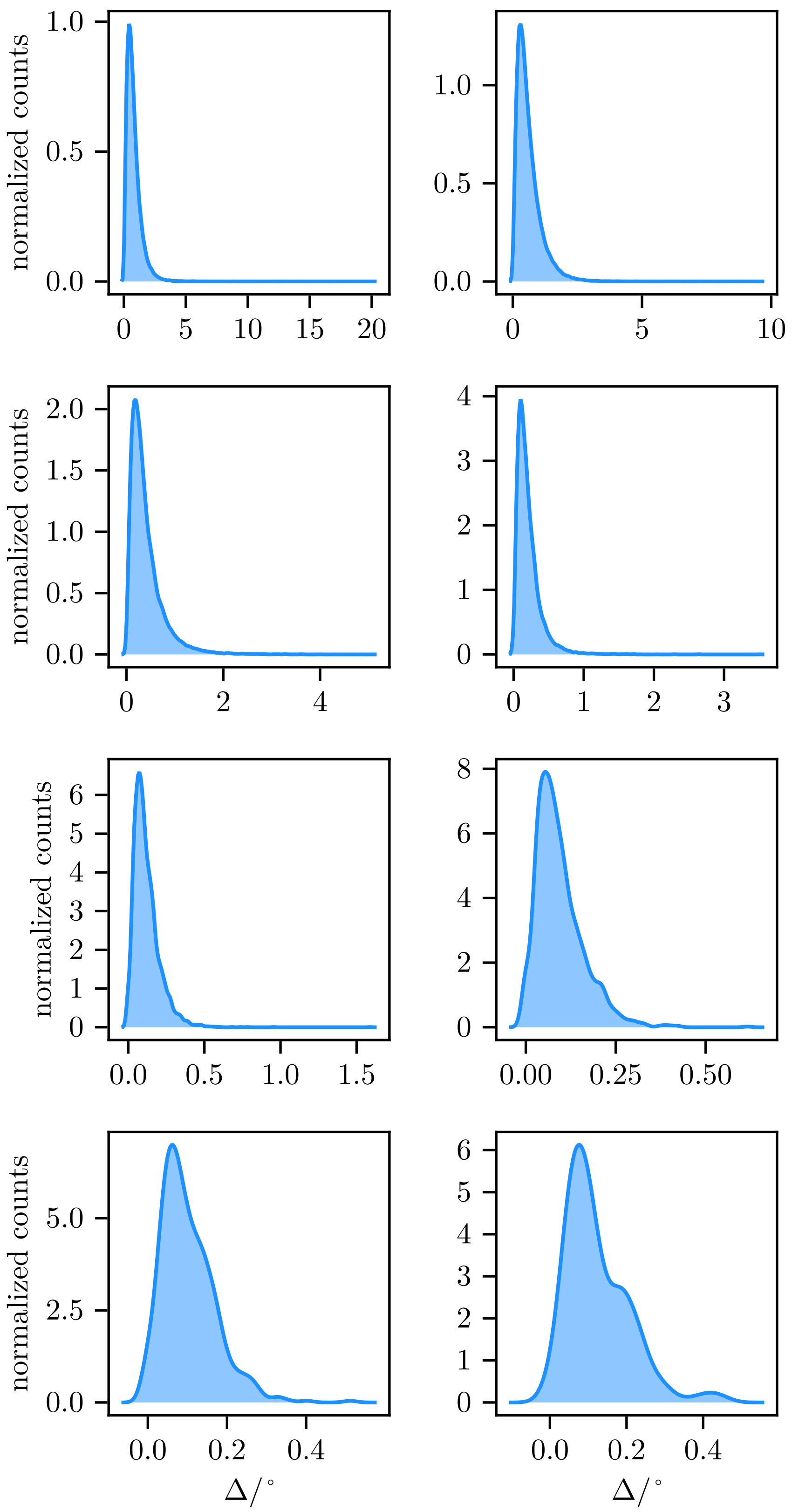

Figure B.3: Distributions, like 6.9 but for all energy bins. Top left is the lowest energy bin, top right the second lowest, and so on.

Appendix C

Direction reconstruction without fine-tuning

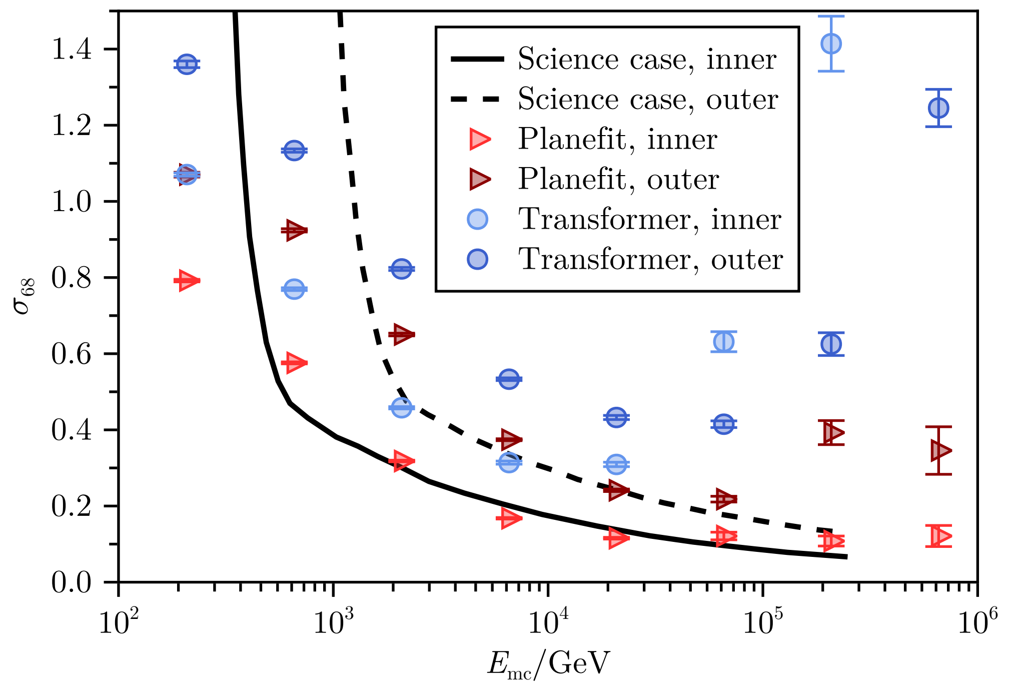

Figure C.1: Plot of angular resolutions for multiple methods. Same plot as Figure 6.10, but without the finetuning during training.

Appendix D

Sampling of defect tanks.

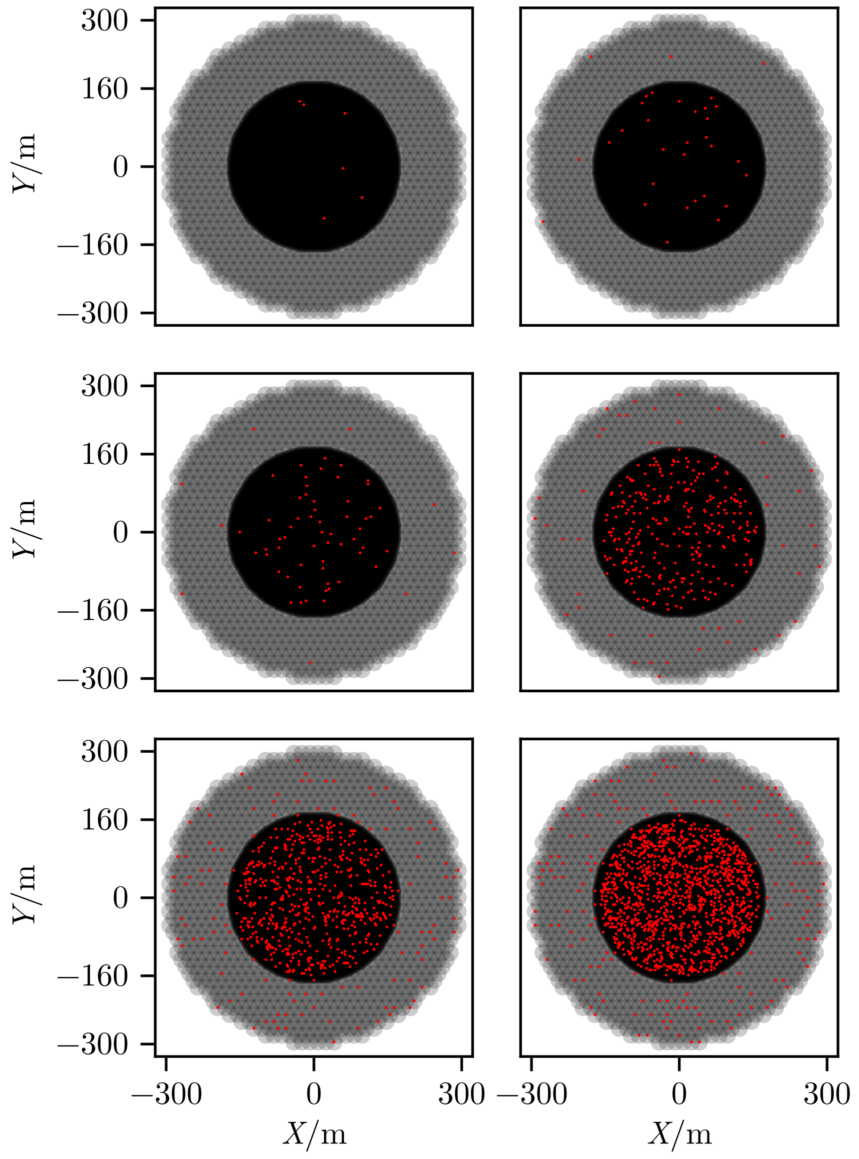

Figure D.1: Defect tanks for robustness study.

Bibliography

[1] A. Comte, *The Positive Philosophy of Auguste Comte.* Blanchard, 1858.

[2] N. deGRASSE TYSON, M. A. Strauss, and J. R. Gott, *Welcome to the Universe: An Astrophysical Tour.* Princeton University Press, 2016.

[3] E. Hubble, "A Relation between Distance and Radial Velocity among Extra-Galactic Nebulae," *Proceedings of the National Academy of Science*, vol. 15, pp. 168–173, Mar. 1929.

[4] A. Wolszczan and D. A. Frail, "A planetary system around the millisecond pulsar PSR1257 + 12," *Nature*, vol. 355, pp. 145–147, Jan. 1992.

[5] D. Bose, V. R. Chitnis, P. Majumdar, and A. Shukla, "Galactic and Extragalactic Sources of Very High Energy Gamma-rays," *Eur. Phys. J. Spec. Top.*, vol. 231, pp. 27–66, Jan. 2022.

[6] F. M. Rieger, E. de Ona-Wilhelmi, and F. A. Aharonian, "Tev astronomy," *Frontiers of Physics*, vol. 8, pp. 714–747, 2013.

[7] T. C. Weekes, M. F. Cawley, D. J. Fegan, K. G. Gibbs, A. M. Hillas, P. W. Kowk, R. C. Lamb, D. A. Lewis, D. Macomb, N. A. Porter, P. T. Reynolds, and G. Vacanti, "Observation of TeV Gamma Rays from the Crab Nebula Using the Atmospheric Cerenkov Imaging Technique," *The Astrophysical Journal*, vol. 342, p. 379, July 1989.

[8] A. Albert, R. Alfaro, H. Ashkar, C. Alvarez, J. Álvarez, J. Arteaga-Velázquez, H. Solares, R. Arceo, J. Bellido, S. BenZvi, *et al.*, "Science case for a wide field-of-view very-high-energy gamma-ray observatory in the southern hemisphere," *arXiv preprint arXiv:1902.08429*, 2019.

[9] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, A. Bridgland, C. Meyer, S. A. A. Kohl, A. J. Ballard, A. Cowie, B. Romera-Paredes, S. Nikolov, R. Jain, J. Adler, T. Back, S. Petersen, D. Reiman, E. Clancy, M. Zielinski, M. Steinegger, M. Pacholska, T. Berghammer, S. Bodenstein, D. Silver, O. Vinyals, A. W. Senior, K. Kavukcuoglu, P. Kohli, and D. Hassabis, "Highly accurate protein structure prediction with AlphaFold," *Nature*, vol. 596, pp. 583–589, Aug. 2021.

[10] D. Guest, K. Cranmer, and D. Whiteson, "Deep Learning and its Application to LHC Physics," *Annu. Rev. Nucl. Part. Sci.*, vol. 68, pp. 161–181, Oct. 2018.

[11] ICECUBE COLLABORATION, "Observation of high-energy neutrinos from the Galactic plane," *Science*, vol. 380, pp. 1338–1343, June 2023.

[12] A. Vaswani *et al.*, "Attention is all you need," *Advances in Neural Information Processing Systems*, 2017.

[13] T. B. Brown *et al.*, "Language models are few-shot learners," *arXiv preprint arXiv:2005.14165*, 2020.

[14] E. Lorenz and R. Wagner, "Very-high energy gamma-ray astronomy: A 23-year success story in high-energy astroparticle physics," *EPJ H*, vol. 37, pp. 459–513, Aug. 2012.

[15] V. F. Hess, "Concerning observations of penetrating radiation on seven free balloon flights," in *A Source Book in Astronomy and Astrophysics, 1900–1975*, pp. 13–20, Harvard University Press, 1979.

[16] W. Hanlon. `https://web.physics.utah.edu/{~}whanlon/spectrum.html`.

[17] V. Schönfelder, *The Universe in Gamma Rays*. Springer Science & Business Media, Mar. 2013.

[18] W. L. Kraushaar and G. W. Clark, "Search for Primary Cosmic Gamma Rays with the Satellite Explorer XI," *Phys. Rev. Lett.*, vol. 8, pp. 106–109, Feb. 1962.

[19] W. B. Atwood *et al.*, "THE LARGE AREA telESCOPE ON THE FERMI GAMMA-RAY SPACE telESCOPE MISSION," *ApJ*, vol. 697, p. 1071, May 2009.

[20] "Fermi website." `https://fermi.gsfc.nasa.gov/ssc/observations/types/allsky/`.

[21] "Wikipedia: Crab nebula." `https://commons.wikimedia.org/wiki/File:Crab_Nebula_in_Multiple_Wavelengths.png`. This work is licensed under the Creative Commons Attribution-Share Alike 3.0 Unported license. (https://creativecommons.org/licenses/by-sa/3.0/deed.en).

[22] L. Kay, S. Palen, and G. Blumenthal, *21st century astronomy*. WW Norton & Company, 2016.

[23] J. J. Hester, "The Crab Nebula: An Astrophysical Chimera," *Annu. Rev. Astron. Astrophys.*, vol. 46, pp. 127–155, Sept. 2008.

[24] R. Bühler and R. Blandford, "The surprising Crab pulsar and its nebula: A review," *Rep. Prog. Phys.*, vol. 77, p. 066901, June 2014.

[25] F. Aharonian, A. Akhperjanian, M. Beilicke, K. Bernlohr, H.-G. Borst, H. Bojahr, O. Bolz, T. Coarasa, J. L. Contreras, J. Cortina, S. Denninghoff, M. V. Fonseca, M. Girma, N. Gotting, G. Heinzelmann, G. Hermann, A. Heusler, W. Hofmann, D. Horns, I. Jung, R. Kankanyan, M. Kestel, A. Kohnle, A. Konopelko, D. Kranich, H. Lampeitl, M. Lopez,

E. Lorenz, F. Lucarelli, O. Mang, D. Mazin, H. Meyer, R. Mirzoyan, A. Moralejo, E. Ona-Wilhelmi, M. Panter, A. Plyasheshnikov, G. Puhlhofer, R. De Los Reyes, W. Rhode, J. Ripken, G. Rowell, V. Sahakian, M. Samorski, M. Schilling, M. Siems, D. Sobzynska, W. Stamm, M. Tluczykont, V. Vitale, H. J. Volk, C. A. Wiedner, and W. Wittek, "The Crab Nebula and Pulsar between 500 GeV and 80 TeV: Observations with the HEGRA Stereoscopic Air Cerenkov Telescopes," *ApJ*, vol. 614, pp. 897–913, Oct. 2004.

[26] A. Giuliani and M. Cardillo, "Supernova remnants in gamma rays," *Universe*, vol. 10, no. 5, p. 203, 2024.

[27] A. Fraknoi, "The OpenStax free astronomy textbook and its open education resources hub," in *Astronomical Society of the Pacific Conference Series*, vol. 533, p. 1, 2022.

[28] K.-H. Kampert and A. A. Watson, "Extensive Air Showers and Ultra High-Energy Cosmic Rays: A Historical Review," *EPJ H*, vol. 37, pp. 359–412, Aug. 2012.

[29] W. Heitler, *The quantum theory of radiation.* Courier Corporation, 1984.

[30] J. Matthews, "A Heitler model of extensive air showers," *Astroparticle Physics*, vol. 22, pp. 387–397, Jan. 2005.

[31] J. D. Jackson, *Classical Electrodynamics.* New York, NY: Wiley, 3rd ed. ed., 1999.

[32] H. Bethe and W. Heitler, "On the stopping of fast particles and on the creation of positive electrons," *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, vol. 146, no. 856, pp. 83–112, 1934.

[33] T. K. Gaisser, R. Engel, and E. Resconi, *Cosmic Rays and Particle Physics: 2nd Edition.* Cambridge University Press, June 2016.

[34] M. Barrantes, JF. Valdés-Galicia, O. Musalem, A. Hurtado, M. Anzorena, R. Garcia, R. Taylor, Y. Muraki, Y. Matsubara, T. Sako, *et al.*, "Atmospheric corrections of the cosmic ray fluxes detected by the solar neutron telescope at the summit of the sierra negra volcano in mexico," *Geofísica internacional*, vol. 57, no. 4, pp. 253–275, 2018.

[35] F. Aharonian, J. Buckley, T. Kifune, and G. Sinnis, "High energy astrophysics with ground-based gamma ray detectors," *Rep. Prog. Phys.*, vol. 71, p. 096901, Aug. 2008.

[36] G. Di Sciascio, "Ground-based Gamma-Ray Astronomy: An Introduction," *J. Phys.: Conf. Ser.*, vol. 1263, p. 012003, June 2019.

[37] I. Tamm, "General characteristics of radiation emitted by systems moving with superlight velocities with some applications to plasma physics," in *Selected Papers*, pp. 55–67, Springer, 1991.

[38] I. Frank and Ig. Tamm, "Coherent visible radiation of fast electrons passing through matter," in *Selected Papers* (B. M. Bolotovskii, V. Y. Frenkel, and R. Peierls, eds.), pp. 29–35, Berlin, Heidelberg: Springer Berlin Heidelberg, 1991.

[39] S. Funk, "Ground- and Space-Based Gamma-Ray Astronomy," *Annual Review of Nuclear and Particle Science*, vol. 65, pp. 245–277, Oct. 2015.

[40] "Hess website." `https://www.mpi-hd.mpg.de/HESS/pages/about/telescopes/`.

[41] W. Hofmann, HESS. Collaboration, *et al.*, "Status of the HESS project," in *Proceedings of the 27th International Cosmic Ray Conference. 07-15 August, 2001. Hamburg, Germany. Under the Auspices of the International Union of Pure and Applied Physics (IUPAP)., p. 2785*, vol. 7, p. 2785, 2001.

[42] A. J. Smith, "The MILAGRO gamma ray observatory," 2005.

[43] "HAWC-Observatory." `https://www.hawc-observatory.org`.

[44] A. U. Abeysekara *et al.*, "The High-Altitude Water Cherenkov (HAWC) observatory in México: The primary detector," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 1052, p. 168253, July 2023.

[45] V. Joshi and A. Jardin-Blicq, "Hawc high energy upgrade with a sparse outrigger array," *arXiv preprint arXiv:1708.04032*, 2017.

[46] P. Abreu, A. Albert, R. Alfaro, C. Alvarez, R. Arceo, P. Assis, F. Barao, J. Bazo, J. Beacom, J. Bellido, *et al.*, "The southern wide-field gamma-ray observatory (swgo): A next-generation ground-based survey instrument for vhe gamma-ray astronomy," *arXiv preprint arXiv:1907.07737*, 2019.

[47] "SWGO SITE SELECTION [SWGO]." `https://www.swgo.org/SWGOWiki/doku.php?id=site_press_release`.

[48] B. S. Acharya *et al.*, "Introducing the CTA concept," *Astroparticle Physics*, vol. 43, pp. 3–18, Mar. 2013.

[49] W. Hofmann, "On angular resolution limits for air shower arrays," *Astroparticle Physics*, vol. 123, p. 102479, Dec. 2020.

[50] A. Smith, R. Conceição, and Schoorlemmer, "Summary of A&S Group Activities(internal swgo document)," Mar. 2023.

[51] S. Kunwar, H. Goksu, J. Hinton, H. Schoorlemmer, A. Smith, W. Hofmann, and F. Werner, "A double-layered Water Cherenkov Detector array for Gamma-ray astronomy," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 1050, p. 168138, May 2023.

[52] A. Smith, R. Conceição, Schoorlemmer, J. Glombitza, Velasco, and L. , "Reference Analysis Chain Description(internal swgo document)," June 2024.

[53] D. Heck, J. Knapp, JN. Capdevielle, G. Schatz, T. Thouw, *et al.*, "CORSIKA: A Monte Carlo code to simulate extensive air showers," *Forschungszentrum Karlsruhe Karlsruhe*, 1998.

[54] J. Pretz, S. BenZvi, and Z. Ren, "Simulation Weighting v4 (internal document)," Aug. 2016.

[55] J. K. Blitzstein and J. Hwang, *Introduction to Probability.* Chapman and Hall/CRC, 2019.

[56] R. Brun, L. Urban, F. Carminati, S. Giani, M. Maire, A. McPherson, F. Bruyant, and G. Patrick, "GEANT: Detector description and simulation tool," tech. rep., CERN, 1993.

[57] A. Albert, R. Alfaro, C. Alvarez, A. Andrés, J. Arteaga-Velázquez, D. A. Rojas, H. A. Solares, R. Babu, E. Belmont-Moreno, A. Bernal, *et al.*, "Performance of the hawc observatory and tev gamma-ray measurements of the crab nebula with improved extensive air shower reconstruction algorithms," *The Astrophysical Journal*, vol. 972, no. 2, p. 144, 2024.

[58] A. U. Abeysekara *et al.*, "Measurement of the Crab Nebula Spectrum Past 100 TeV with HAWC," *ApJ*, vol. 881, p. 134, Aug. 2019.

[59] R. Alfaro *et al.*, "Gamma/Hadron Separation with the HAWC Observatory," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 1039, p. 166984, Sept. 2022.

[60] The SWGO Collaboration, "Status of the SWGO air shower reconstruction using a template-based likelihood method," in *Proceedings of 38th International Cosmic Ray Conference — PoS(ICRC2023)*, (Nagoya, Japan), p. 593, Sissa Medialab, Aug. 2023.

[61] V. Joshi, J. Hinton, H. Schoorlemmer, R. López-Coto, and R. Parsons, "A Template-based γ-ray Reconstruction Method for Air Shower Arrays," *J. Cosmol. Astropart. Phys.*, vol. 2019, pp. 012–012, Jan. 2019.

[62] S. Funk, J. Glombitza, F. Leitl, M. Schneider, and C. van Eldik, "Application of graph networks to SWGO," May 2024.

[63] A. Smith, "Basic Direction Fitting with AERIE(internal swgo document)," Aug. 2020.

[64] H. Schoorlemmer and on behalf of the SWGO collaboration, "A next-generation ground-based wide field-of-view gamma-ray observatory in the southern hemisphere," in *Proceedings of 36th International Cosmic Ray Conference — PoS(ICRC2019)*, vol. 358, p. 785, SISSA Medialab, July 2021.

[65] T.-p. Li and Y.-q. Ma, "ANALYSIS METHODS FOR RESULTS IN GAMMA-RAY ASTRONOMY," *ApJ. . .*, vol. 272, no. 1, 1983.

[66] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning.* MIT Press, 2016.

[67] S. M. Stigler, "Gauss and the Invention of Least Squares," *The Annals of Statistics*, vol. 9, pp. 465–474, May 1981.

[68] D. Teets and K. Whitehead, "The Discovery of Ceres: How Gauss Became Famous," *Mathematics Magazine*, vol. 72, no. 2, pp. 83–93, 1999.

[69] A. Karpathy, *Connecting Images and Natural Language.* PhD thesis, Stanford University, 2016.

[70] F. Marquardt, "Machine learning and quantum devices," *SciPost Physics Lecture Notes*, p. 029, May 2021.

[71] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," Jan. 2017.

[72] I. Neutelings, "Tikz: Neural networks." `https://tikz.net/neural_networks/`, 2021/0009. This work is licensed under the Creative Commons Attribution 4.0 International License (https://creativecommons.org/licenses/by-sa/4.0/).

[73] C. M. Bishop and N. M. Nasrabadi, *Pattern Recognition and Machine Learning*, vol. 4. Springer, 2006.

[74] N. L. Carothers, "A short course on approximation theory," *Bowling Green State University, Bowling Green, OH*, vol. 38, 1998.

[75] B. Bauer and M. Kohler, "On deep learning as a remedy for the curse of dimensionality in nonparametric regression," *The Annals of Statistics*, vol. 47, pp. 2261–2285, Aug. 2019.

[76] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, vol. 65, no. 6, pp. 386–408, 1958.

[77] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, "Activation Functions: Comparison of trends in Practice and Research for Deep Learning," Nov. 2018.

[78] D. Hendrycks and K. Gimpel, "Gaussian Error Linear Units (GELUs)," June 2023.

[79] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 249–256, JMLR Workshop and Conference Proceedings, Mar. 2010.

[80] M. Erdmann, J. Glombitza, G. Kasieczka, and U. Klemradt, *Deep Learning for Physics Research.* World Scientific, 2021.

[81] S. Hanson and L. Pratt, “Comparing Biases for Minimal Network Construction with Back-Propagation,” in *Advances in Neural Information Processing Systems*, vol. 1, Morgan-Kaufmann, 1988.

[82] S. G. a. J. Howard, “Fast.ai - AdamW and Super-convergence is now the fastest way to train neural nets.” https://www.fast.ai/posts/2018-07-02-adam-weight-decay.html, July 2018.

[83] I. Loshchilov and F. Hutter, “Decoupled Weight Decay Regularization,” Jan. 2019.

[84] J. Riebesell, “Tikz: Dropout.” `https://tikz.net/dropout/`. This work is published under a MIT License.

[85] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A Simple Way to Prevent Neural Networks from Overfitting,”

[86] J. Riebesell, “Tikz: Skip connection.” `https://tikz.net/skip-connection/`. This work is published under a MIT License.

[87] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” Dec. 2015.

[88] J. L. Ba, J. R. Kiros, and G. E. Hinton, “Layer Normalization,” July 2016.

[89] “Tiktokenizer.” `https://tiktokenizer.vercel.app/?model=meta-llama%2FMeta-Llama-3-8B`.

[90] R. Xiong, Y. Yang, D. He, K. Zheng, S. Zheng, C. Xing, H. Zhang, Y. Lan, L. Wang, and T.-Y. Liu, “On Layer Normalization in the Transformer Architecture,” June 2020.

[91] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, “An image is worth 16x16 words: Transformers for image recognition at scale,” *arXiv preprint arXiv:2010.11929*, 2020.

[92] J. Shlomi, P. Battaglia, and J.-R. Vlimant, “Graph neural networks in particle physics,” *Mach. Learn.: Sci. Technol.*, vol. 2, p. 021001, Dec. 2020.

[93] R. Lam, A. Sanchez-Gonzalez, M. Willson, P. Wirnsberger, M. Fortunato, F. Alet, S. Ravuri, T. Ewalds, Z. Eaton-Rosen, W. Hu, *et al.*, “Graphcast: Learning skillful medium-range global weather forecasting,” *arXiv preprint arXiv:2212.12794*, 2022.

[94] N. Choma, F. Monti, L. Gerhardt, T. Palczewski, Z. Ronaghi, P. Prabhat, W. Bhimji, M. M. Bronstein, S. R. Klein, and J. Bruna, “Graph neural networks for icecube signal classification,” in *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 386–391, IEEE, 2018.

[95] J. Glombitza, V. Joshi, B. Bruno, and S. Funk, “Application of graph networks to background rejection in Imaging Air Cherenkov Telescopes,” *J. Cosmol. Astropart. Phys.*, vol. 2023, p. 008, Nov. 2023.

[96] I. Watson, A. Albert, R. J. Alfaro, C. Alvarez, A. Andres, J. C. Arteaga Velazquez, D. O. Avila Rojas, H. A. Ayala Solares, R. Babu, E. Belmont-Moreno, T. Capistrán Rojas, S. Yun, A. Carramiñana, F. Carreon-Gonzalez, U. Cotti, J. Cotzomi, S. Coutiño de León, E. de la Fuente, D. Depaoli, C. L. de León, R. Diaz Hernandez, J. C. Díaz Vélez, B. Dingus, M. Durocher, M. DuVernois, K. Engel, M. C. Espinoza Hernández, J. Fan, K. Fang, N. I. Fraija, J. A. Garcia-Gonzalez, F. Garfias, H. Goksu, M. M. González, J. A. Goodman, S. J. Groetsch, J. P. Harding, S. Hernández Cadena, I. Herzog, J. Hinton, B. Hona, D. Huang, F. Hueyotl-Zahuantitla, P. Hüntemeyer, A. Iriarte, V. Joshi, S. Kaufmann, D. Kieda, A. Lara, J. LEE, W. H. Lee, H. Leon Vargas, J. Linnemann, A. L. Longinotti, G. Luis-Raya, K. Malone, J. Martínez-Castro, J. Matthews, P. Miranda-Romagnoli, J. A. Montes, J. A. Morales Soto, M. Mostafa, L. Nellen, M. U. Nisa, R. Noriega-Papaqui, L. Olivera-Nieto, N. Omodei, Y. Pérez Araujo, E. G. Pérez Pérez, A. Pratts, C. D. Rho, D. Rosa-Gonzalez, E. Ruiz-Velasco, H. I. Salazar, D. Salazar-Gallegos, A. Sandoval, M. Schneider, G. Schwefer, J. Serna-Franco, A. J. Smith, Y. Son, W. R. Springer, O. Tibolla, K. Tollefson, I. Torres, R. Torres Escobedo, R. M. Turner, F. Ureña-Mena, E. Varela, L. Villaseñor, X. Wang, I. J. Watson, F. Werner, K. Whitaker, E. J. Willox, H. Hongyi Wu, H. Zhou, and K. S. Caballero Mora, “Deep learning for the HAWC gamma-ray observatory,” *PoS*, vol. ICRC2023, p. 927, 2023.

[97] T. Dao, D. Y. Fu, S. Ermon, A. Rudra, and C. Ré, “FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness,” June 2022.

[98] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. C. Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M.-A. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. E. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, and T. Scialom, “Llama 2: Open Foundation and Fine-Tuned Chat Models,” July 2023.

[99] V. Khomenko, O. Shyshkov, O. Radyvonenko, and K. Bokhan, “Accelerating recurrent neural network training using sequence bucketing and multi-GPU data parallelization,” in *2016 IEEE First International Conference on Data Stream Mining & Processing (DSMP)*, pp. 100–103, Aug. 2016.

[100] H. Bukhari, D. Chakraborty, P. Eller, T. Ito, M. V. Shugaev, and R. Ørsøe, "IceCube – Neutrinos in Deep Ice The Top 3 Solutions from the Public Kaggle Competition," Oct. 2023.

[101] W. Zhang, W. Wei, W. Wang, L. Jin, and Z. Cao, "Reducing BERT computation by padding removal and curriculum learning," in *2021 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, pp. 90–92, 2021.

[102] L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and J. Han, "On the Variance of the Adaptive Learning Rate and Beyond," Oct. 2021.

[103] X. Zhai, A. Kolesnikov, N. Houlsby, and L. Beyer, "Scaling Vision Transformers," June 2022.

[104] "Huggingface." `https://huggingface.co/meta-llama/Meta-Llama-3-70B`.

[105] T. Ullrich and Z. Xu, "Treatment of Errors in Efficiency Calculations," Jan. 2007.

Danksagung

- Ich danke Janika für die beste Unterstützung die man sich nur wünschen kann.
- Jonas, Martin und Franzi für die hilfreichen Diskussionen, Ideen und Ratschläge.
- Johannes, dafür dass er immer mit einem Lächeln und guter Laune für Studenten bei jeglichen Fragen zu Hilfe steht.
- Tom, für alles was du für mich in den letzten 4 Jahren getan hast.

Eigenständigkeitserklärung

Hiermit versichere ich, Markus Pirke (22577513), die vorgelegte Arbeit selbstständig und ohne unzulässige Hilfe Dritter sowie ohne die Hinzuziehung nicht offengelegter und insbesondere nicht zugelassener Hilfsmittel angefertigt zu haben. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen und wurde auch von keiner anderen Prüfungsbehörde bereits als Teil einer Prüfung angenommen.

Die Stellen der Arbeit, die anderen Quellen im Wortlaut oder dem Sinn nach entnommen wurden, sind durch Angaben der Herkunft kenntlich gemacht. Dies gilt auch für Zeichnungen, Skizzen, bildliche Darstellungen sowie für Quellen aus dem Internet.

Mir ist insbesondere bewusst, dass die Nutzung künstlicher Intelligenz verboten ist, sofern diese nicht ausdrücklich als Hilfsmittel von dem Prüfungsleiter bzw. der Prüfungsleiterin zugelassen wurde. Dies gilt insbesondere für Chatbots (insbesondere ChatGPT) bzw. allgemein solche Programme, die anstelle meiner Person die Aufgabenstellung der Prüfung bzw. Teile derselben bearbeiten könnten.

Ort, Datum	Unterschrift