Estimating the Mass of Di-Tau Systems in the ATLAS Experiment Using Neural Network Regression

Martin Werres

Masterarbeit in Physik
angefertigt im Physikalischen Institut

vorgelegt der
Mathematisch-Naturwissenschaftlichen Fakultät
der
Rheinischen Friedrich-Wilhelms-Universität
Bonn

April 2019

I hereby declare that this thesis was formulated by myself and that no sources or tools other than those cited were used.

Bonn, ...
Date Signature

1. Gutachter: Prof. Dr. Klaus Desch
2. Gutachter: Priv.-Doz. Dr. Markus Cristinziani

Acknowledgments

I would like to thank Prof. Dr. Klaus Desch for the opportunity to write my master thesis in his research group and for the support throughout the time. Further, I like to thank Priv.-Doz. Dr. Markus Cristinziani for evaluating my thesis. I would also like to thank Philip Bechtle for his great support and the discussions, Peter Wagner and Christian Grefe for their expertise on tau physics and helpful ideas which they were willing to share. I would like to thank Michael Hübner for a lot of help in a broad spectrum of questions and his technical support. I really enjoyed the working atmosphere in the whole research group and would like to thank everyone, especially my office mates, for all the discussion about and beyond physics.

For proofreading my thesis and rehearsing my talk I would like to thank A. Gottwald, M. Hübner, H. Junkerkalefeld, T. Klingl, D. Köhler, P. König, L. Schildgen and M. Wiesner. I do consider to have gained a lot of friends during my time in the research group, but I would also like to thank my friends outside of physics for putting my mind away from my work from time to time. Finally, my thanks go to my family for their support over the past years.

Contents

CHAPTER 1

Introduction

In modern times of particle physics the experiments try to reach ever higher luminosity and higher energies in order to probe the Standard Model of particle physics (SM) to ultra high precision and to possibly find new physics behind the Standard Model (BSM). Not only the machines are upgraded with time but also the analysis tools are refined and sometimes reinvented in order to achieve better accuracy.

The Large Hadron Collider (LHC) is a great example for mankind's curiosity to understand nature. It is one of the largest machines ever built and collides protons with a center-of-mass energy of $\sqrt{s} = 13\,\text{TeV}$. The ATLAS experiment is built around one interaction point to detect the produced particles in the collision. In the year 2012, this setup discovered a new resonance [1], presumably the SM Higgs boson, which was predicted in the 1960s[2–4]. An upgrade is planned to reach higher luminosity [5] and for which also the ATLAS detector is upgraded [6]. With this high luminosity, a lot of data will be produced which demand for a faster analysis.

The Higgs boson plays a major role in the SM because it arises from the electroweak symmetry breaking, the Higgs fields non-zero vacuum expectation value explains the masses of the gauge bosons and also explains the masses of the fermions via the Yukawa coupling. The discovered Higgs boson is not yet tested to full extend and deviations to the predicted properties would open the window for new physics. Promising analyses that directly probe the properties of the found resonance are the Yukawa coupling of the Higgs boson to tau-leptons which can give access to the $\mathcal{CP}$ quantum numbers of the Higgs boson [7–9] and the Higgs boson self interaction which reveals details of the shape of the Higgs potential [10]. Studies on the Higgs self coupling involve two or three Higgs bosons in the final state. It is likely that one of these Higgs boson decays into tau-leptons, because the $H \rightarrow \tau\tau$ decay has a non-negligible branching ratio of $\mathcal{B} = 6.37\%$ [11]. While many promising analyses involve Higgs bosons decaying into tau-lepton pairs, these events have to be accurately distinguished from similar looking background. The main discriminant to separate Higgs bosons from predominantly Z boson background is the invariant mass of the di-tau system [12].

One analysis tool which is getting ever higher popularity is the use of artificial neural networks. While theoretically established since 1943 [13], the recent growth of computational power makes the training of complex artificial neural networks on large data samples possible. Machine learning techniques are already present in modern particle physics analyses, e.g. in the usage of boosted decision trees for the discrimination of signal and background [14]. The trend is to also apply deep learning techniques to high energy physics problems which are harder to interpret for humans but can lead to better results.

The current estimator for the invariant mass of a di-tau system is the Missing Mass Calculator [15]. The MMC is a likelihood-based method which tries to estimate the neutrinos of the tau-lepton decay. For this, individual likelihoods are calculated and multiplied. Possibly, a deep neural network can learn additionally correlation between these variables. A second improvement is that a deep neural network is easy extendable to include additional event related information, which can further improve the estimated mass resolution. While the MMC performs a phase space scan for each individual event in order to predict the mass of a di-tau system, a neural network has to be trained only once and then a map is defined which can calculate the mass of di-tau system by simple matrix multiplication. Thus, a speedup can very likely be gained.

This thesis introduces a setup for training a neural network for the task of estimating the mass of a di-tau system in the ATLAS experiment, where both tau-leptons decay hadronically. The neural network is trained in supervised learning on an unphysical $\gamma^* \rightarrow \tau\tau$ sample, which only includes virtual photons that couple electromagnetically to the tau-leptons and does not take the interference with the weak coupling Z boson into account. A loss function is tested which tries to take the limited target value range of the training sample into account. The performance of the neural network is compared to the MMC performance not only on this unphysical sample but also tested in the use case of separating Higgs and Z bosons.

The thesis is structured to first give an overview of related work in Chapter 2. The theoretical background is given in chapter 3. Chapter 4 describes the ATLAS experiment at the LHC and also sketches how particle reconstruction with the ATLAS experiment is done. It further describes the MMC which is the reference method for this work. An introduction of the technicalities of deep neural networks is presented in chapter 5. The used data samples are described in chapter 6. It also describes the input variables for the deep neural network and which architectures are tested. Chapter 7 is dedicated to the description of how the performance of a deep neural network can be evaluated. It also shows the advantages of using a loss function which tries to take the limited target value range of the training sample into account in the example of a deep neural network trained with Monte Carlo generated information. Finally, the results of the deep neural network performance, which is trained on reconstruction level data, are presented in chapter 8. Impact of selected hyperparameters on the performance are studied and for one specific architecture the performance in the use case of separating Higgs and Z bosons is presented. The thesis is concluded in chapter 9.

CHAPTER 2

Related Work

This work has to be seen in the context of the state of similar problems in the field of machine learning and its applications to high energy physics. For the specific task of the mass estimation of a di-tau system, the Missing Mass Calculator (MMC) tool (see Chapter 4.4) is used as the reference method. There is still ongoing work in the development of the MMC and it is planned to include more variables and the correlation between the probability density functions, which could possibly improve its performance [16].

Another approach to provide an estimator for the di-tau system mass used a boosted regression tree (BRT), trained on Monte Carlo generated $H \rightarrow \tau\tau$ samples which varied the H mass in 5 GeV steps [17, 18]. The BRT was then trained on truth information and therefore could not exploit possible effects of the detector response. It achieved a resolution slightly worse than the MMC [17]. An important observed effect was that the BRT had a non-linear response in the edge regions of the training sample. This seems to be a problem in regression tasks and there is ongoing work in the compensation of edge effects. A recent work tried to give a solution for neural network regression tasks in designing an alternative loss function (see Chapter 5.1.2) [19]. It was developed to reconstruct the energy in a calorimeter using convolutional neural networks, but the concept should be applicable to any neural network regression problem.

Convolutional neural networks are heavily and successfully used in high energy physics, e.g. for top-tagging [20, 21] or for energy reconstruction in NOvA [22]. Other applications of deep learning in high energy physics at the Large Hadron Collider are summarized in [23]. While the spectrum of possible applications of deep learning is very broad, deep feed-forward neural networks are suggested for classification or regression problems. The usage of deep feed-forward neural networks for regression problems is topic of recent studies, e.g. [24], and may find a growing acceptance and popularity.

A study of deep feed-forward neural networks in the search for new exotic particles found that deep neural networks performed better with low level input features in contrast to boosted decision trees or shallow neural networks which performed better with higher level input features [25]. This can be interpreted in a sense, that from physical insights a machine learning approach can extract many useful information, but also that some information can get lost.

A big disadvantage from deep neural networks is that they are very hard to interpret. Many studies try to address that problem and try to illustrate what the neural network does when going from one layer to the next one [26, 27].

CHAPTER 3

The Standard Model of Particle Physics

The Standard Model of particle physics (SM) is a model which aims to describe the nature of elementary particles and the fundamental interactions between them. It is a non-abelian gauge quantum field theory with the internal symmetries of the unitary product group $SU(3)_\mathrm{C} \times SU(2)_\mathrm{L} \times U(1)_\mathrm{Y}$. There are twelve arising gauge bosons which act as the mediators of the fundamental forces, i.e. the strong (8), the weak (3) and the electromagnetic force (1). The SM contains fermions which exist in three generations. The mass of the gauge bosons are explained by the Higgs mechanism with a resulting Higgs boson, while the fermion masses are explained via the Yukawa coupling of the Higgs boson to the fermions.

The SM is renormalizable [28] and mathematically self-consistent, however there are also shortcomings. The fundamental force gravitation as described by general relativity cannot be explained by the SM. Additionally, the SM does not explain the observed baryon asymmetry [29], does not include dark energy [30] which is a possible explanation for the accelerated expansion of the universe [31] and provides no dark matter candidate with the properties deduced from cosmological observations [32]. Despite the shortcomings, the SM is a very successful model as it predicts many properties of particle physics in great accuracy.

After the overview of the particles in the SM, theoretical considerations are explained briefly. It should not be understood as a complete explanation of the SM and at some points it requires understanding of quantum field theory but should lead to an understanding of the physics of the tau-lepton which is relevant for this work. For more in-depth explanations standard textbooks like [33] or [34] are recommended.

3.1 Overview of the Standard Model Particles

Early philosophical thoughts already formulated that matter should consist of some undividable, smallest building blocks. The term *átomos* was systematically introduced by Democritus in the fifth century before Christ. What is called atom today is known to consist of even smaller particles, electrons and protons and neutrons in the nucleus. Protons and neutrons are known to consist of quarks which are fundamental particles in today's understanding, as is the. In the SM these fundamental particles are featured.

The matter building particles are spin $1/2$ fermions. Fermions are subdivided in the group of leptons, which interact in the electroweak interaction, and quarks, which additionally interact in the

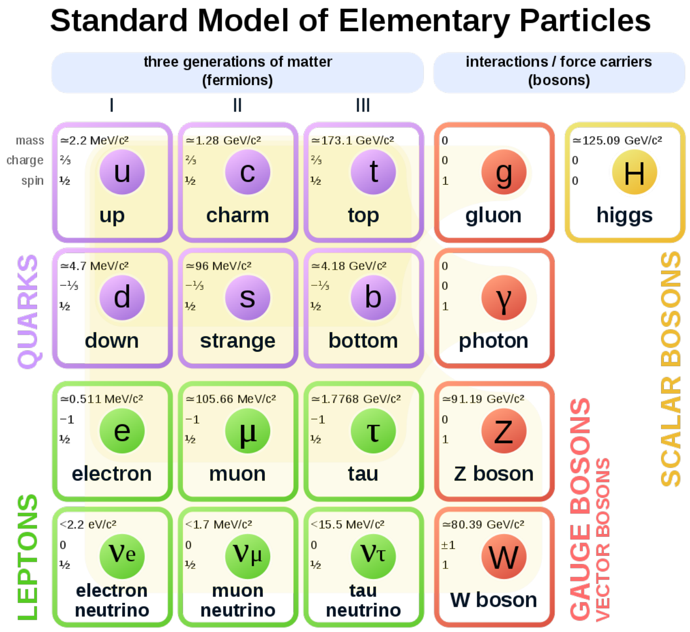


Figure 3.1: Particles of the Standard Model with selected quantum numbers [36].

strong interaction. The electroweak interaction is a theoretical concept which is introduced to unify electromagnetic and weak interactions.

Both types of fermions occur in three generations. Similar particles of different generations have the same quantum numbers and do only differ in the rest mass. The SM gives no explanation why there are exactly three generations. However, until today there is no clear evidence that more generations exist.

The mediators of the three described fundamental forces are the spin 1 bosons. In the SM there is also a spin 0 boson, the Higgs boson. The Higgs boson is the most recently discovered particle [1, 35] and completes the SM in the sense that it was the last puzzle piece for electroweak theory.

All SM particles and some selected quantum numbers are depicted in Fig. 3.1.

3.1.1 Leptons

Leptons are the electrically neutral neutrinos ν_l and the charged leptons l with charge $Q(l) = -1e$. The charged leptons are called electron, muon and tau(-lepton) and abbreviated as e^-, μ^- and τ^-. The three neutrino generations ν_e, ν_μ and ν_τ are the eigenstates of the weak interaction. As the neutrinos have no electric or color charge, they only interact via the weak interaction. Each fermion has an antiparticle. The antiparticles of the charged leptons have opposite charge, while the antiparticles of the neutrinos are produced in weak charged currents with a negatively charged lepton or pair-produced in a weak neutral current. This preserves the lepton number. In the later presented formulation of the SM only the weak eigenstates of the neutrinos occur as massless particles. By the observation of neutrino oscillations (e.g. [37–39]) it can be concluded that the weak eigenstates are mixtures of mass eigenstates ν_1, ν_2 and ν_3 and thus not massless.

3.1.2 Quarks

There are two types of quarks. Up-type quarks have electric charge $Q(u) = 2/3$, down-type quarks have electric charge $Q(d) = -1/3$. Quarks carry color charge which makes them participate in the strong interaction. Antiquarks have opposite signed charge and carry strong anticolor. Due to the running coupling constant α_s of the strong interaction, free quarks can only exist in the asymptotic limit of high energies. This is very important for high energy collider experiments. When quarks are produced the large color potential can create new quark-anti-quark pairs from the vacuum. This continues until all quarks are bound in colorless final states. This process is called hadronization. Hadrons consist of either three (baryons) or two (mesons) valence quarks. The three up-type quarks are called up u, charm c and top t. The down-type quarks are called down d, strange s and bottom b. The top-quark is in a sense special, because it decays before it hadronizes.

3.1.3 Bosons

The electrically neutral and massless photon γ is the mediator of the electromagnetic force. Every electrically charged particle couples to the photon. For this work it is important that a virtual photon can create a $\tau^+\tau^-$ pair in the so called Drell-Yan process [40].

The W bosons are the mediators of weak charged current interactions. There is a positively charged W^+ with electric charge $Q(W^+) = +1e$ and a negatively charged W^- with electric charge $Q(W^-) = -1e$. The W bosons have a rest mass of $(80.379 \pm 0.012)\,\mathrm{GeV}$.

The Z^0 boson is the mediator of weak neutral currents. It is electrically neutral. As it couples to $\tau^+\tau^-$ pairs, it is important for this work. Off-shell Z^0 bosons are experimentally indistinguishable from virtual photons and are also part of the Drell-Yan process. The Z^0 boson has a rest mass of $(91.1876 \pm 0.0021)\,\mathrm{GeV}$.

The gluons are the mediators of the strong force. They are massless, electrically neutral and carry color and anticolor charge of different type. Gluons couple to quarks and can self-interact, i.e. a three and a four-boson vertex exist.

The Higgs boson (H) is different from the other bosons as it is no gauge boson. It arises in the process of spontaneous symmetry breaking in the electroweak theory. The Higgs boson couples to the mass of the fermions via Yukawa coupling. Because of this coupling to the fermion masses and considering that the tau-lepton is the heaviest lepton, the branching ratio of $H \to \tau^+\tau^-$ is 6.37% [11]. The Higgs boson has a rest mass of $(125.18 \pm 0.16)\,\mathrm{GeV}$.

3.2 The Electroweak Sector of the SM

A great success of the SM is the unification of the weak and the electromagnetic force. Historically, the Lagrange density, also called Lagrangian, which describes the electroweak sector of the SM is constructed to reproduce the observations of weak interactions, for example that the parity is maximally violated in charged current interactions [41–43]. However, the resulting mathematical formulation did predict the existence of weak neutral currents and the Z boson [44]. The observed vector bosons, the $W^\pm$ and the Z^0 boson, are massive particles. Without the Higgs mechanism [2–4], which explains the mass of the bosons, as well as masses of the fermions, the electroweak model would not be complete, because it would not be gauge invariant if the particle masses would be explained by Dirac mass terms.

The beauty of the theory is that from the Lagrangian, Feynman rules can be concluded giving a pictorial view on the interactions and the involved couplings. The Lagrangian will be presented with a few comments on phenomenological impacts and consequences for the production and decay of tau-leptons.

The electroweak symmetry group is the direct product of $SU(2)_\mathrm{L}$ and $U(1)_\mathrm{Y}$. After spontaneous symmetry breaking this breaks down to $U(1)_\mathrm{Q}$ which describes the electromagnetic interactions. The subscript L indicates that the $SU(2)$ gauge fields $\vec{W}$ only couple to left-chiral fermions. The left-chiral leptons $(\nu_l, l^-)_\mathrm{L}$ form a doublet with weak isospin $T = 1/2$ while the right-chiral charged leptons l_R^- are $SU(2)_\mathrm{L}$ singlets, only carry hypercharge Y and do only couple to the $U(1)_\mathrm{Y}$ gauge field B. The relation between the weak isospin, the hypercharge and the electric charge is $Q = T_3 + Y$. Right-chiral neutrinos are not included in the SM, as they do not couple to either of the fields. As they are not directly observed so far, it is not known whether they exist. From the observations of neutrino oscillations it is known that there are differences in the neutrino masses of each generation but the underlying mass acquiring concept is not understood.

The fundamental representation of the Higgs field is:

$$\phi = \begin{pmatrix} \phi^+ \\ \phi^0 \end{pmatrix} . \tag{3.1}$$

The Lagrangian for the Higgs field reads:

$$\mathcal{L}_\mathrm{Higgs} = \left| \left(i\partial_\mu - g\frac{\vec{\tau}}{2} \cdot \vec{W}_\mu - g' Y_\phi B_\mu \right) \phi \right|^2 - V(\phi) , \tag{3.2}$$

where $\vec{\tau}$ are the generators of $SU(2)_\mathrm{L}$, g is the coupling constant of $SU(2)_\mathrm{L}$, g' is the coupling constant of $U(1)_\mathrm{Y}$ and $Y_\phi = 1/2$ is the hypercharge of both components of ϕ. The Higgs potential $V(\phi)$ is then:

$$V(\phi) = \mu^2 \phi^\dagger \phi + \lambda (\phi^\dagger \phi)^2 . \tag{3.3}$$

While λ has to be positive in order to bound the potential from below, μ^2 can be negative, which is essential in the SM. In the unitary gauge ϕ^+ is set to 0 and ϕ^0 is real. The global minimum is located at the vacuum expectation value

$$\langle \phi \rangle = \sqrt{\frac{-\mu^2}{2\lambda}} =: \frac{1}{\sqrt{2}} v . \tag{3.4}$$

Fluctuations around the vacuum expectation value can be parametrized by a real scalar field h which is the physical Higgs field:

$$\phi = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ v + h \end{pmatrix} . \tag{3.5}$$

Plugging this in for the field ϕ in (3.2) explains the masses of the vector bosons. It is important to note that the observable vector boson fields are mixtures of the gauge boson fields, which will be explained below.

The electroweak Lagrangian for leptons only reads:

$$
\begin{aligned}
\mathcal{L}_{\mathrm{ew}} = \sum_{l=e,\mu,\tau} &\left[(\overline{\nu_{l\mathrm{L}}}, \overline{l_{\mathrm{L}}}) \gamma^\mu \left(i\partial_\mu - g\frac{\vec{\tau}}{2} \cdot \vec{W}_\mu - g' Y_{\mathrm{L}} B_\mu \right) \begin{pmatrix} \nu_{l\mathrm{L}} \\ l_{\mathrm{L}} \end{pmatrix} + \bar{l}_{\mathrm{R}} \gamma^\mu \left(i\partial_\mu - g' Y_{\mathrm{R}} B_\mu \right) l_{\mathrm{R}} \right] \\
&- \frac{1}{4} B_{\mu\nu} B^{\mu\nu} - \frac{1}{4} \vec{W}_{\mu\nu} \vec{W}^{\mu\nu} + \mathcal{L}_{\mathrm{Higgs}} + \mathcal{L}_{\mathrm{Yukawa}}
\end{aligned}
\tag{3.6}
$$

with

$$
B^{\mu\nu} = \partial^\mu B^\nu - \partial^\nu B^\mu \qquad \text{and} \qquad W_i^{\mu\nu} = \partial^\mu W_i^\nu - \partial^\nu W_i^\mu - g\epsilon_{ijk} W_j^\mu W_k^\nu \; . \tag{3.7}
$$

The physical vector boson fields are mixtures of the fields $\vec{W}$ and B such that

$$
W_\mu^\pm = \frac{1}{\sqrt{2}} \left(W_\mu^1 \mp i W_\mu^2 \right) \tag{3.8}
$$

are the field components of the $W^\pm$ fields and

$$
\begin{pmatrix} A_\mu \\ Z_\mu \end{pmatrix} = \begin{pmatrix} \cos\theta_{\mathrm{W}} & \sin\theta_{\mathrm{W}} \\ -\sin\theta_{\mathrm{W}} & \cos\theta_{\mathrm{W}} \end{pmatrix} \begin{pmatrix} B_\mu \\ W_\mu^3 \end{pmatrix} \tag{3.9}
$$

are the components of the photon and the Z^0 fields. The weak mixing angle is determined through the relation $g'/g = \tan\theta_{\mathrm{W}}$. It relates the couplings to the electric charge via

$$
g' \cos\theta_{\mathrm{W}} = g \sin\theta_{\mathrm{W}} = e \; . \tag{3.10}
$$

The added Lagrangian for the Yukawa coupling $\mathcal{L}_{\mathrm{Yukawa}}$ explains the lepton masses:

$$
\begin{aligned}
\mathcal{L}_{\mathrm{Yukawa}} &= \sum_{l=e,\mu,\tau} -f_l \cdot \overline{l_{\mathrm{R}}^-} \phi^\dagger \begin{pmatrix} \nu_{l\mathrm{L}} \\ l_{\mathrm{L}}^- \end{pmatrix} + \text{h.c.} \\
&\overset{\text{unitary gauge}}{=} \sum_{l=e,\mu,\tau} \left[\underbrace{-f_l \cdot \frac{v}{\sqrt{2}} \overline{l^-} l^-}_{\text{lepton mass term}} \underbrace{-f_l \cdot \frac{h}{\sqrt{2}} \overline{l^-} l^-}_{\text{coupling to } h} \right] ,
\end{aligned}
\tag{3.11}
$$

identifying $f_l \cdot v/\sqrt{2}$ as the lepton mass.

With the Lagrangian $\mathcal{L}_{\mathrm{ew}}$ all interactions of the tau-lepton are defined, which are the most important for this work.

For completeness, the electroweak Lagrangian can be easily extended to include quark interactions, analogously. Left-chiral quarks are introduced as $SU(2)_{\mathrm{L}}$ doublets, right-chiral quarks as singlets:

$$
\begin{pmatrix} u_{\mathrm{L}} \\ d'_{\mathrm{L}} \end{pmatrix}, u_{\mathrm{R}}, d'_{\mathrm{R}}, \begin{pmatrix} c_{\mathrm{L}} \\ s'_{\mathrm{L}} \end{pmatrix}, c_{\mathrm{R}}, s'_{\mathrm{R}}, \begin{pmatrix} t_{\mathrm{L}} \\ b'_{\mathrm{L}} \end{pmatrix}, t_{\mathrm{R}}, b'_{\mathrm{R}}, \tag{3.12}
$$

where d', s' and b' denote the weak eigenstates of the down-type quarks. The up-type quarks are defined to be identical to the mass-eigenstates. The mass eigenstates of the down-type quarks d, s and

b are related to the weak eigenstates by the CKM-Matrix [45, 46]:

$$\begin{pmatrix} d' \\ s' \\ b' \end{pmatrix} = \begin{pmatrix} V_{ud} & V_{us} & V_{ub} \\ V_{cd} & V_{cs} & V_{cb} \\ V_{td} & V_{ts} & V_{tb} \end{pmatrix} \begin{pmatrix} d \\ s \\ b \end{pmatrix} . \tag{3.13}$$

This allows for interactions between different quark generations in weak charged current interactions.

3.3 Quantum Chromodynamics

Quantum Chromodynamics (QCD) describes the strong interaction between quarks and gluons. The underlying gauge symmetry is $SU(3)_\mathrm{C}$, where the subscript C indicates that only particles with color charge participate in strong interactions. This means that all other SM particles, except quarks and gluons are $SU(3)_\mathrm{C}$ singlets. The theoretical foundation was build in the Young-Mills theory [47], which extended the concept of gauge theory to non-abelian $SU(N)$ groups.

The concept of color charge [48–50] demands three quark color fields q_1, q_2 and q_3. Requiring the Lagrangian to be invariant under local phase transformations, the covariant derivatives takes the form:

$$D_\mu = \partial_\mu + igT_a G_\mu^a \, , \tag{3.14}$$

where T_a ($a = 1, ..., 8$) are the generators of $SU(3)_\mathrm{C}$, g is the coupling strength and G_μ^a are the gluon fields.

The Lagrangian describing QCD reads:

$$\mathcal{L}_\mathrm{QCD} = \sum_{q=u,d,c,s,t,b} \left[\sum_{j=1,2,3} \bar{q}_j (i\gamma^\mu \partial_\mu - m) q_j - g(\bar{q}_j \gamma^\mu T_a q_j) G_\mu^a \right] - \frac{1}{4} G_{\mu\nu}^a G_a^{\mu\nu} \, , \tag{3.15}$$

with

$$G_{\mu\nu}^a = \partial_\mu G_\nu^a - \partial_\nu G_\mu^a - g f_{bc}^a G_\mu^b G_\nu^c \, , \tag{3.16}$$

and $f_{bc}^a = f_{abc}$ being the structure constants of the group. The Lagrangian (3.15) describes QCD as a stand-alone theory. But as quarks do also have $SU(2)_\mathrm{L}$ and $U(1)_\mathrm{Y}$ quantum numbers, the Dirac mass term has to be taken out, as it is not gauge invariant in the electroweak theory. The quarks masses are also explained by the Higgs Yukawa coupling.

The two main properties of QCD are color confinement, i.e. that color charged particles cannot be isolated at normal conditions, and asymptotic freedom [51, 52], i.e. that the interaction strength decreases with increasing energy scale so that free color charged particles can exist at the asymptotic limit of high energies.

3.4 The Tau-Lepton

The tau-lepton was discovered in a series of experiments at the Stanford Linear Accelerator Center from 1974 to 1977 [53]. It is the heaviest lepton with a mass of $m_\tau \approx 1.777\,\mathrm{GeV}$ and a lifetime of

category	decay products	branching ratio $\mathcal{B}/\%$
leptonic	$e^{-}\bar{\nu}_{e}\nu_{\tau}$	17.82
	$\mu^{-}\bar{\nu}_{\mu}\nu_{\tau}$	17.39
hadronic - "1-prong"	$h^{-}(\geq 0 \text{ neutrals})\ \nu_{\tau}$	50.03
	$\pi^{-}\nu_{\tau}$	10.82
	$\pi^{-}\pi^{0}\nu_{\tau}$	25.49
	$\pi^{-}2\pi^{0}\nu_{\tau}$	9.26
hadronic - "3-prong"	$h^{-}h^{-}h^{+}(\geq 0 \text{ neutrals})\ \nu_{\tau}$	15.21
	$\pi^{-}\pi^{-}\pi^{+}\nu_{\tau}$	9.31
	$\pi^{-}\pi^{-}\pi^{+}\pi^{0}\nu_{\tau}$	4.62

Table 3.1: Decay channels and respective branching ratios of a τ^{-}-lepton [54]. $h^{\pm}$ stands for charged hadron, neutral references neutral hadrons. The hadronic decay channels are divided into 1 and 3 prong, i.e. 1 or 3 charged particles. Details are given on the most dominant sub-channels.

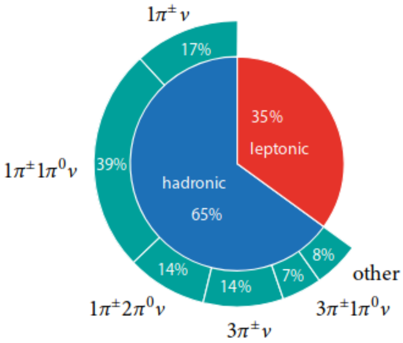


Figure 3.2: Pie chart of the tau-lepton decay modes. The hadronic decay modes are split into the ones which are dominant [55].

$\tau_{\tau} \approx 290\,\text{ps}$ [54]. This results in a mean flight length L of

$$L \approx 49\,\mu\text{m}\frac{E}{\text{GeV}}\,, \tag{3.17}$$

depending on the energy E of the tau-lepton.

Due to the high rest mass, it is the only lepton which can decay hadronically and leptonically. The most frequent decay modes are listed in Tab. 3.1 and visualized in Fig. 3.2.

The interactions between tau-leptons and the bosons in the SM are described by (3.6). The Feynman rules for the interaction vertices are depicted in Figs. 3.3 - 3.6. The different kind of couplings affect the polarization of the tau-leptons. The photon and the Higgs boson couple equally strong to left- and right-chiral tau-leptons. The W^{-} only couples to left-chiral tau-leptons, as the Feynman rule for the vertex contains the left-chiral projection operator $P_{\text{L}} = (1 - \gamma_5)/2$. The Z^{0} couples to left- and right-chiral tau-leptons with strengths $g(0.5 - \sin^2\theta_{\text{W}})/\cos\theta_{\text{W}}$ and $g\sin^2\theta_{\text{W}}/\cos\theta_{\text{W}}$, respectively.

For massless particles (or in the ultra-relativistic limit), the eigenstates of the chirality operator and the helicity operator are identical. This implies that at high energies, the spin of left-chiral tau-leptons

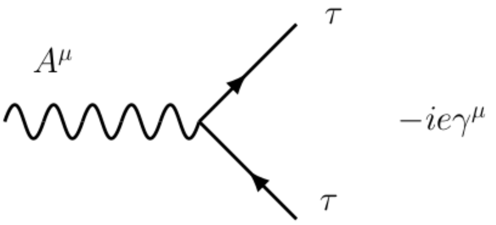


Figure 3.3: Feynman rule for the interaction vertex of tau-leptons with a photon.

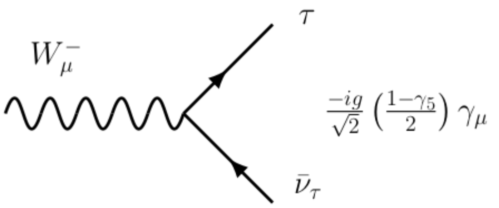

W_μ^-
τ
$\bar{\nu}_\tau$
$\frac{-ig}{\sqrt{2}}\left(\frac{1-\gamma_5}{2}\right)\gamma_\mu$

Figure 3.4: Feynman rule for the interaction vertex of a tau-lepton with a W^- boson.

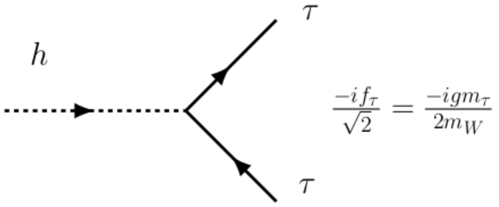


Figure 3.5: Feynman rule for the interaction vertex of tau-leptons with a Higgs boson.

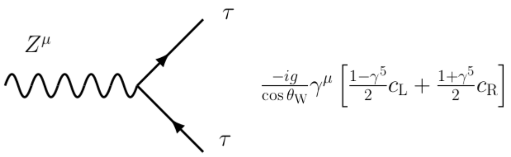

Z^μ
τ
τ
$\frac{-ig}{\cos\theta_{\mathrm{W}}}\gamma^\mu\left[\frac{1-\gamma^5}{2}c_{\mathrm{L}} + \frac{1+\gamma^5}{2}c_{\mathrm{R}}\right]$

Figure 3.6: Feynman rule for the interaction vertex of tau-leptons with a Z^0 boson.

is predominantly anti-parallel to the momentum and for right-chiral tau-leptons parallel.

CHAPTER 4

Experimental Setup

The studies of this work use Monte Carlo simulated data of the ATLAS detector at the Large Hadron Collider (LHC). This chapter aims to describe the LHC and the ATLAS detector. A short explanation of how objects are reconstructed is given, especially how tau-leptons are reconstructed. The current estimator for the mass of a di-tau system is presented, the Missing Mass Calculator tool (MMC) [15]. The MMC will hold for comparison against the neural network approach that will be presented in this work.

4.1 The Large Hadron Collider

The Large Hadron Collider is a circular hadron collider situated near Geneva. With a circumference of approximately 27 km, it is currently the world's largest hadron collider. It is built about 170 m underneath the ground and operated by the Conseil Européen pour la Recherche Nucléaire, CERN. In two oppositely running beam pipes protons can be accelerated to energies up to 6.5 TeV (design energy: 7 TeV). Before the protons are injected into the LHC storage ring, they traverse several small accelerators. At first, the protons are accelerated to 50 MeV in a linear accelerator, the LINAC 2. Afterwards, they are accelerated in three synchrotrons, the Proton Synchrotron Booster (Booster), the Proton Synchrotron (PS) and the Super Proton Synchrotron (SPS) up to an injection energy of 450 GeV. In the LHC, there are four interaction points where the particle beams collide [56]. A particle detector is built at each of the four interaction points: ALICE[1], ATLAS, CMS[2] and LHCb[3]. Additionally, there are two detectors measuring events close to the beam line, namely LHCf[4] and TOTEM[5] [57]. For the studies of this work only proton collisions are simulated. In principle, the LHC could also accelerate ions. A scheme of the LHC setup is depicted in figure 4.1.

The different detectors are specialized for different purposes. The ALICE detector is specialized for analyzing lead ion collisions and investigate possible quark-gluon plasma, a state of matter which is thought to consist of asymptotically free quarks and gluons. The LHCb experiment is specialized to detect small asymmetries between matter and antimatter in the decay of *b* mesons. On the other hand,

[1] **A** **L**arge **I**on **C**ollider **E**xperiment

[2] **C**ompact **M**uon **S**olenoid

[3] **LHC** **b**eauty, reference for the examination of hadrons with *b*-quarks

[4] **LHC** **f**orward

[5] **TOT**al **E**lastic and diffractive cross section **M**easurement

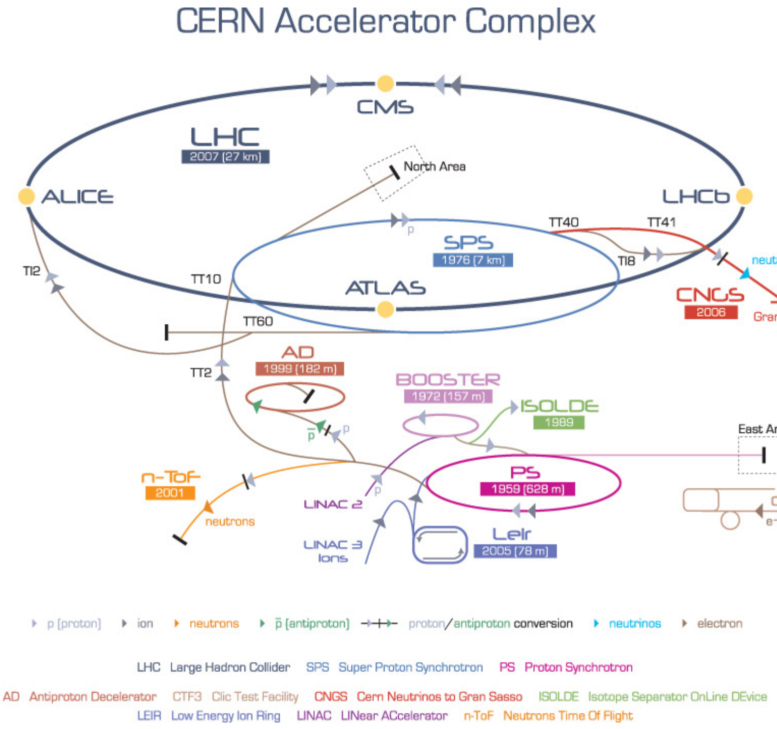


Figure 4.1: Overview of the accelerator system at CERN [58].

the ATLAS and CMS experiments are multipurpose detectors to study a broad variety of processes. The two experiments are set up differently which allows to discover new findings independently and they can confirm each other's discoveries.

During operation in 2018, the ATLAS Detector measured a peak instantaneous luminosity of $\mathcal{L} = 21\,\mathrm{nb/s}$ [59]. For the production of particles, the integrated luminosity $L = \int \mathrm{d}t\mathcal{L}$ is important, as the total number of produced particles or events N depends on the integrated luminosity and the cross section σ via the relation $N = L\sigma$. An overview of selected partial cross section for proton-proton collisions depending on the center of mass energy $\sqrt{s}$ is depicted in Fig. 4.2.

As shown in Fig. 4.2, the cross section for Higgs-boson production is roughly three orders of magnitude lower than the cross section for the Z^0, in the LHC energy regime. In the search for Higgs bosons decaying into tau-lepton pairs, the $Z^0 \rightarrow \tau\tau$ decay is the main background, because their rest masses are in the same order of magnitude and their event topology is similar. The main tool to discriminate between H and Z in the di-tau channel is the invariant mass of the di-tau system, which is currently estimated by the MMC. Therefore, a good resolution on the estimated mass is desirable.

4.2 The ATLAS Experiment

The ATLAS Experiment, **A T**oroidal **L**HC **A**pparatu**S** Experiment, is a multipurpose detector. It has a length of approximately 44 m and a diameter of about 25 m (cf. Fig. 4.3). The demands on the ATLAS experiment are to have an acceptance in a large region of pseudorapidity, to cover the whole azimuthal angle and achieve a good energy and momentum resolution of the constituent detectors. To

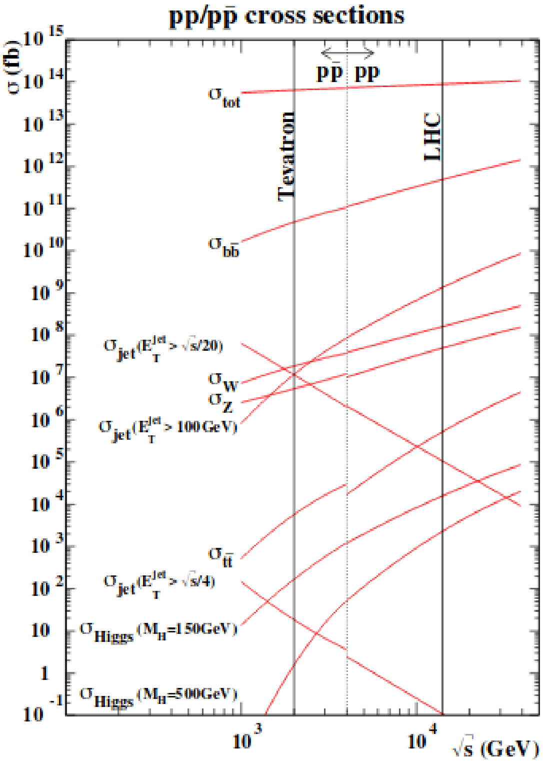


Figure 4.2: Overview of selected cross sections against the center of mass energy $\sqrt{s}$ for proton-anti-proton (below $\sqrt{s} = 4\,\text{TeV}$) and proton-proton collisions (above $\sqrt{s} = 4\,\text{TeV}$) [60].

achieve this, the sub-detectors are built cylindrically around the beam axis in an onion-like structure, symmetrical in forward and backward direction. The three major detector systems are the Inner Detector, the calorimeters and the muon spectrometer. In addition with the magnet system, the Inner Detector can measure the trajectory of charged particles and determine its momentum and charge by the bending in the magnetic field. The calorimeters can measure the energy of electrons, photons and hadrons. And finally the muon spectrometer can identify muons and measure their momentum. Information is from [61].

4.2.1 The ATLAS Coordinate System

The ATLAS coordinate system is a right-handed coordinate system with the origin lying in the nominal interaction point of the proton beams. The z-axis is defined in the direction along the beam pipes, the x-axis points to the center of the LHC storage ring and the y-axis points upwards towards the Earth surface. The azimuthal angle ϕ is measured around the beam pipe ($\tan\phi = {}^{p_y}/_{p_x}$) and the polar angle is the angle relative to the beam pipe ($\tan\theta = {}^{p_y}/_{p_z}$). Instead of the polar angle θ, the pseudorapidity

$$\eta = -\ln\left(\tan\left(\frac{\theta}{2}\right)\right) \tag{4.1}$$

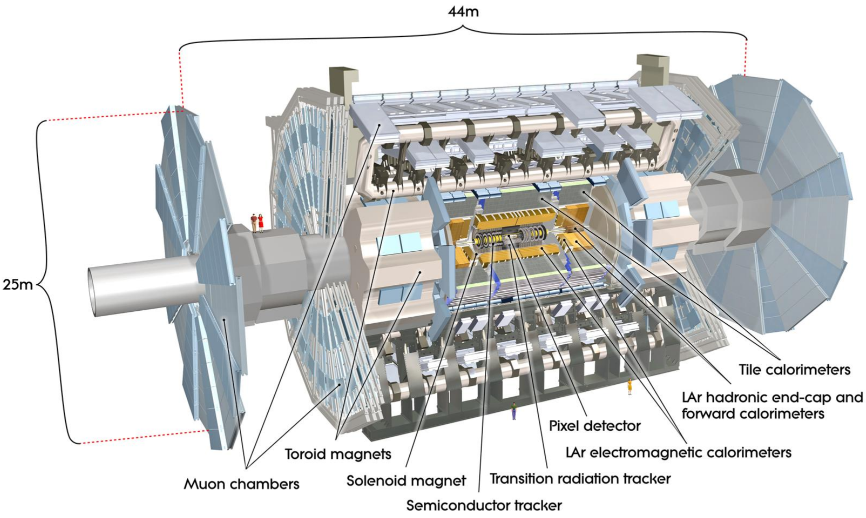

44m
25m
Tile calorimeters
LAr hadronic end-cap and forward calorimeters
Pixel detector
LAr electromagnetic calorimeters
Toroid magnets
Muon chambers
Solenoid magnet
Transition radiation tracker
Semiconductor tracker

Figure 4.3: Labeled sketch of the ATLAS detector [62].

is used to measure the angle relative to the beam pipe. The pseudorapidity has the advantage that differences in pseudorapidity are invariant under Lorentz boosts along the z-axis. Angular differences are measured by $\Delta R = \sqrt{\Delta\phi^2 + \Delta\eta^2}$.

The momentum of a reconstructed particle with the ATLAS detector is represented by the magnitude of the transverse momentum $p_\mathrm{T} = \sqrt{(p_x^2 + p_y^2)}$, ϕ and η. With the Inner Detector only the transverse momentum can be measured but there is also the advantage that, as the initial momentum is only along the z-axis, all transverse momentum should add up to zero. This holds up to the fact, that the particle beams collide with a half-crossing angle of about 120 − 150 µrad.

4.2.2 The Inner Detector

The Inner Detector is approximately 6.2 m long and has a diameter of about 2.1 m. It is situated right around the beam pipe. With a magnetic field of 2 T in parallel to the beam axis, charged particles are bent in the x-y-plane. Tracks associated to particles can be determined by the interaction of the particle with the detector material. Through the curvature of the tracks, caused by the Lorentz force, the transverse momentum could be measured with an accuracy of

$$\frac{\sigma(p_\mathrm{T})}{p_\mathrm{T}} = 0,036\,\text{‰} \cdot \frac{p_\mathrm{T}}{\mathrm{GeV}} \oplus 1,3\,\text{‰}\,, \tag{4.2}$$

where ⊕ represents the addition in quadrature and taking the square root.

The Inner Detector consist of three sub-detectors, the Pixel Detector, the Silicon Tracker and the Transition Radiation Tracker and is depicted in Fig. 4.4.

Pixel Detector

The Pixel Detector is built such that a particle traverses three silicon sensors independent of its pseudorapidity. This is realized by a barrel and an end-cap region. In the barrel region three

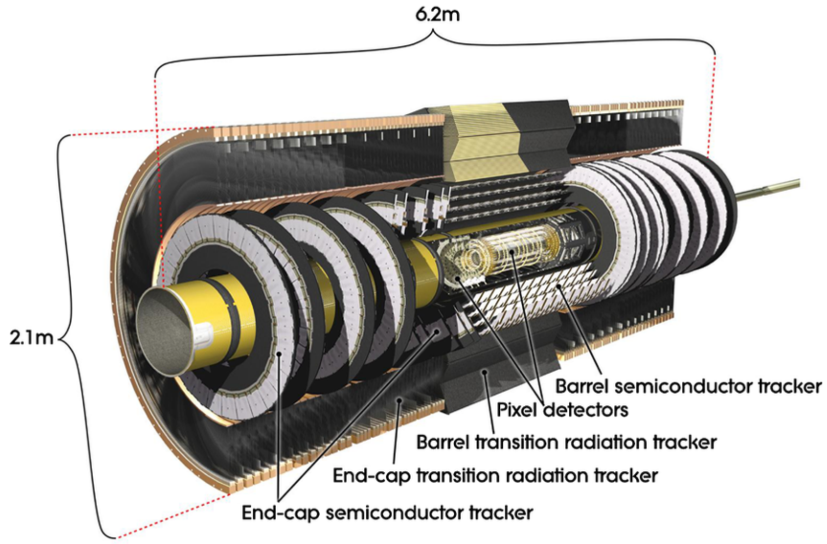

6.2m
2.1m
Barrel semiconductor tracker
Pixel detectors
Barrel transition radiation tracker
End-cap transition radiation tracker
End-cap semiconductor tracker

Figure 4.4: Labeled sketch of the Inner Detector [63].

layers are built cylindrically around the beam axis while in the end-cap regions the detector modules are three discs perpendicular to the beam axis. The innermost layer, the Insertable B-Layer (IBL), was inserted during a shutdown from 2013 to 2015 [64]. The elements of the detector are silicon pixels with a size of a minimum of $400\,\mu\text{m} \times 50\,\mu\text{m}$. It reaches a resolution of about $10\,\mu\text{m}$ in the r-ϕ-plane and covers the range $|\eta| < 2.5$.

Silicon Tracker

The Silicon Tracker embraces the Pixel Detector. It consists of eight layers in the barrel region and nine discs in the end-cap region. It is made of silicon strips allowing a resolution of $17\,\mu\text{m}$ in the r-ϕ-plane and $580\,\mu\text{m}$ in the z-direction. The detector geometry covers the region $|\eta| < 2.5$.

Transition Radiation Tracker

The Transition Radiation Tracker is the outermost part of the Inner Detector. It is a gaseous detector made of straw tubes which contain a 70% Xe, 27% CO_2 and 3% O_2 gas mixture. Each straw tube has a diameter of about 4 mm and works like a drift chamber. The transition radiation, which is emitted when a charged particle passes the boundary of two media of different refraction index, can be used for electron identification. This is because the probability of transition radiation depends on the Lorentz $\gamma = E/m$ which is higher for electrons, due to their small rest mass m at the same energies, compared to other particles. The Transition Radiation Tracker achieves a resolution of $130\,\mu\text{m}$ in the r-ϕ-plane and covers the region $|\eta| < 2.5$.

4.2.3 The Calorimeters

The calorimeter system is divided in an electromagnetic, a hadronic calorimeter and a calorimeter which measures the energy of particles close to the beam line. The system covers the range of $|\eta| < 4.9$. The task of the calorimeters is to measure the energy of the particles and the subdivision results from the different interactions of different particles with matter.

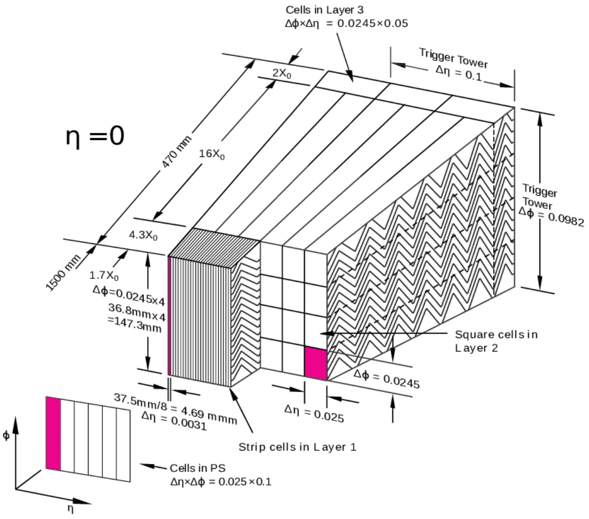


Figure 4.5: Sketch of the lateral and longitudinal segmentation of the ATLAS electromagnetic calorimeter around $\eta = 0$. [66].

The Electromagnetic Calorimeter

The electromagnetic calorimeter (ECAL) is a sampling calorimeter consisting of alternating layers of lead and liquid argon. The lead acts like an absorber because of its high atomic number and density and the liquid argon as scintillating material. As the name suggests, the main task of the ECAL is to measure the energy of photons and electrons which interact via electromagnetic interaction. Incoming electrons radiate off Bremsstrahlphotons, which then produce e^+e^--pairs via pair production. This develops an electromagnetic shower. Likewise, incoming photons produce electromagnetic showers starting with pair production. The relevant length scale is the radiation length X_0, by which the mean energy of the particles in the shower is of the fraction $1/e$ of the initial energy.

The ECAL is divided into a barrel ($|\eta| < 1.475$) and an end-cap region ($1.475 < |\eta| < 3.2$). It consists of a presampling layer (PS) in the range $|\eta| < 1.8$, which measures the energy loss before the ECAL, and three sub-parts of electromagnetic Calorimeters Sampling1-3. The first, Sampling1 has a high granularity of about $\Delta\eta \times \Delta\phi = 0,003 \times 0,01$ in order to make high precision measurements of photons. In the Sampling2, the main energy of the electromagnetic showers are deposited. The Sampling3 measures the rest energy of very high energetic electromagnetic showers. The ECAL in the direction around $\eta = 0$ is depicted in Fig. 4.5. In hadronic tau-lepton decays, neutral pions π^0 occur as decay products. They decay to two photons with a branching ratio of approximately 98.8‰ [54]. These two photons are very close together but usually can be separated due to the high granularity of Sampling1. As the photons of the π^0 decay deposits almost no energy in Sampling3, the Sampling3 can be used as a hadronic calorimeter in tau-lepton decays [65].

The ECAL can measure the energy with a resolution of

$$\frac{\sigma(E)}{E} = 10\text{‰} \cdot \sqrt{\frac{\text{GeV}}{E}} \oplus 0,7\text{‰} . \tag{4.3}$$

The Hadronic Calorimeter

The hadronic calorimeter (HCAL) embraces the ECAL and measures the energy of hadrons. It is divided into a barrel ($|\eta| < 1.7$) and an end-cap region ($1.5 < |\eta| < 3.2$) which uses different detector setups to measure the energy of the particles. In the barrel region, a sampling calorimeter with steel as passive and plastic scintillators as active material is used. In the end-cap region, copper plates are used as passive material and liquid argon as the active material. The hadronic showers are more complex than electromagnetic showers. The hadrons interact strongly with the detector material and can produce other hadrons, e.g. pions, or excite nuclei. The other pions can then decay or the excited nuclei deexcite. As there are neutrinos involved or energy can go into binding energy, the energy of a hadronic shower is more difficult to measure. Hadrons deposit also a fraction of their energy in the ECAL, but as their shower is characterized by the nuclear absorption length λ_a which is much larger than X_0, the shower extends into the HCAL. To ultimately stop the shower, a lot of detector material is needed.

The HCAL can measure the energy with a resolution of

$$\frac{\sigma(E)}{E} = 50\% \cdot \sqrt{\frac{\text{GeV}}{E}} \oplus 3\% \ . \tag{4.4}$$

The Forward Calorimeter

The forward calorimeter (FCAL) measures particles in the very forward direction up to $|\eta| < 4.9$. It is a sampling calorimeter consisting of three layers, using a copper layer and two tungsten layers as passive material and liquid argon as active material. The copper layer is sensible to electromagnetic interactions and the tungsten layers are to stop hadrons. The FCAL can measure the energy with a resolution of

$$\frac{\sigma(E)}{E} = 100\% \cdot \sqrt{\frac{\text{GeV}}{E}} \oplus 10\% \ . \tag{4.5}$$

4.2.4 The Muon System

As muons have a higher rest mass than electrons ($m_\mu \approx 200 m_\mathrm{e}$), they only deposit a small fraction of their energy in the calorimeter system. In order to measure muons there is an additional muon chamber as the outermost part of the ATLAS detector. It covers the range $|\eta| < 2.7$. Basically, the muon system is a tracking detector which measures the bending of muons in the magnetic field of approximately 0.5 T, which is perpendicular to the beam pipe. The detector can measure the momentum of a muon with 1 GeV with a resolution of approximately 10%.

4.3 Object Reconstruction with the ATLAS Detector

Like hinted before, different particles leave different signatures in the detector. The signatures will be explained briefly and some selected signatures are visualized in Fig. 4.6. Only the reconstruction of hadronically decaying tau-leptons will be described in detail, as they are most important for this work.

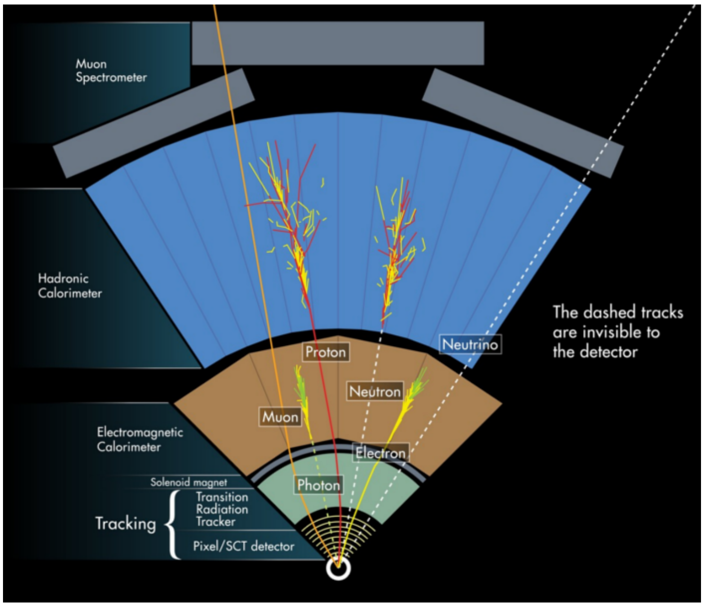


Figure 4.6: Sketch of signatures selected particles leave in the ATLAS detector [67].

4.3.1 Electrons and Photons

Electrons and photons both deposit their energy in the ECAL. While electrons are charged, their signature is a track leading to a energy deposit in the ECAL. Positrons can be distinguished from electrons via the direction of bending in the magnetic field, i.e. their charge sign. Photons only leave a signature in the Inner Detector when they pair produce e^+e^--pair, which is called photon conversion. Algorithms decide, whether a cluster in the ECAL belongs to an electron, an unconverted photon or a converted photon. For details, please see [68] and [69].

4.3.2 Muons

Muons traverse all detector segments and leave a track in the Inner Detector, a small energy deposit in the calorimeters and a track in the muon system. The tracks can then be extrapolated and combined into a muon object. Depending on which information is available, different reconstructed muon types can be defined. Combined muons are tracks reconstructed in the muon system that are extrapolated and matched with a track in the Inner Detector. Segment-tagged muons are tracks in the ID that, once extrapolated, match a local track segment in the muon system, i.e. if only one layer is hit caused by its low p_T or by falling into a low acceptance region of the detector. Calorimeter-tagged muons are tracks that are reconstructed in the Inner Detector and match an energy deposit of a minimal ionizing particle. They have a low purity but can cover regions where the muon system is only partially installed. Lastly, extrapolated muons are tracks in the muon system which are, when extrapolated, loosely originating from the interaction point and can be used to extend the acceptance in the range $2.5 < |\eta| < 2.7$. The muon objects were presented in descending priority, if muon types overlap, the muon type with the highest priority is chosen [70].

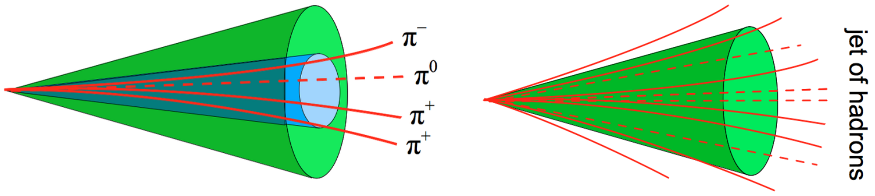


Figure 4.7: Sketch of a typical jet caused by a hadronic tau-lepton decay (left) in comparison to a typical QCD induced jet (right) [74].

4.3.3 Jets

As quarks and gluons are confined in colorless states, they hadronize when produced. As a result, they lead to a varying number of more or less collimated hadrons. These objects are then grouped as a jet. Charged hadrons leave a track in the Inner Detector, but all hadrons deposit their energy in the calorimeters. Therefore, jets are reconstructed from calorimeter cells using the anti-k_t algorithm with $R = 0.4$ [71]. The anti-k_t algorithm is a sequential clustering algorithm that is constructed to be safe against the irradiation of infrared or collinear gluons [72].

4.3.4 Tau-Leptons

As tau-leptons have a very short lifetime, they typically decay before they interact with the detector. Tau-leptons can decay into leptonic or hadronic final states which occur approximately 35% and 65% of the time, respectively. When the tau-lepton decays into light leptons, the leptons look like prompt leptons, as the neutrinos involved escape undetected. Thus, the leptons can then be identified and the neutrinos can lead to missing transverse energy in the detector system. When the tau-leptons decay hadronically, the visible particles are predominantly one or three charged pions with possible additional neutral pions. Thus, the hadronically decaying tau-leptons are very similar to jets with one or three charged tracks and an absolute charge of 1, which is why their reconstruction is also seeded by the anti-k_t algorithm. To differentiate between QCD-jets and tau-jets, it can be utilized, that the hadronic tau-lepton induced jets are on average more collimated than QCD-jets. This is depicted in Fig. 4.7. A boosted decision tree decides whether the jet is more likely to be a QCD- or a hadronic-tau-jet. The input variables used for the boosted decision tree (BDT) describe the jet properties and are given in [73]. The energy of the tau-jet is calibrated with a boosted regression tree, which centers the mean energy of the tau-jet to the actual simulated energy [73]. As the tau-lepton always decays into a tau-neutrino, the tau-jet four-momentum is only the visible part of the tau-lepton four-momentum. In a leptonic decay, there is an additional neutrino.

For tau-jets, the individual constituents and their momentum can be reconstructed using information from the Inner Detector, the ECAL and the HCAL [65]. The idea is to exploit the information of the track for the charged hadrons and estimate their energy deposit in the ECAL as the difference in energy of the track and the energy deposit in the HCAL:

$$E_{h^\pm}^{\mathrm{ECAL}} = E_{h^\pm}^{\mathrm{track}} - E_{h^\pm}^{\mathrm{HCAL}} . \tag{4.6}$$

This energy is then subtracted from the nearest π^0 candidate. The neutral pions almost always decay into two photons which deposit all their energy in the ECAL. Thus the subtraction corrects the

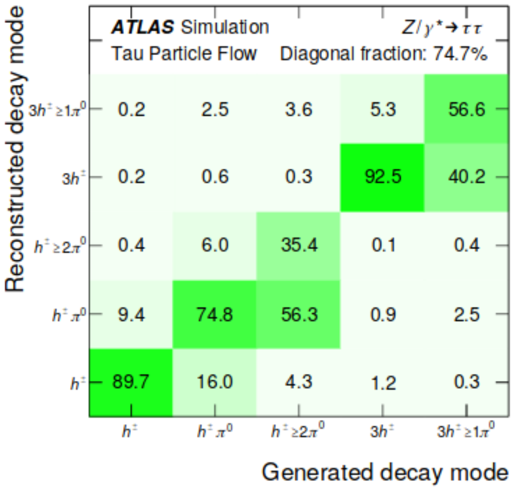


Figure 4.8: Decay mode classification efficiency matrix for simulated $Z \rightarrow \tau\tau$ events with requirements as defined in [65]. The probability of reconstructing a particular decay mode for a given generated decay mode is shown [65].

energy of the neutral cluster. A BDT is used to evaluate the shower shapes in the ECAL and identify neutral pions. This is important, as remnants of charged hadrons could be misclassified as a π^0. The two photons of the π^0 decay lead to two close-by calorimeter hits in the Sampling1 due to its high granularity, but not in all cases. For higher momentum of the initial tau-lepton it gets harder to separate individual neutral pions as the objects are more collimated. After identifying each object of the tau-jet, a decay mode classification is performed. The decay modes which can be identified are $h^{\pm}$, $h^{\pm}\pi^0$, $h^{\pm} \geq 2\pi^0$, $3h^{\pm}$ and $3h^{\pm} \geq 1\pi^0$, or `other` if no decay mode can be associated. A BDT then compares the decay mode classification hypotheses and gives a final decay mode. The performance of the decay mode classification is depicted in Fig. 4.8.

Besides the achievement of this reconstruction technology to discriminate between tau-lepton decay modes, the resolution on the four momentum of the visible tau-jet is crucial, too. The performance on the spacial resolution is depicted in Fig. 4.9 and the energy resolution in Fig. 4.10.

4.3.5 Missing Transverse Energy

As explained in Section 4.2.1, all momenta in the x-y-plane should add up to zero. The missing momentum is then the negative of the vector sum of all measured energy deposits (again without z-component). The missing transverse energy ($\vec{E}_{\mathrm{T}}^{\mathrm{miss}}$) is an estimator for the neutrino system of all neutrinos involved. The resolution on $\vec{E}_{\mathrm{T}}^{\mathrm{miss}}$ depends on the resolution of each individual objects. In order to correct for possible effects of the energy resolution, the objects are calibrated and $\vec{E}_{\mathrm{T}}^{\mathrm{miss}}$ is defined as the negative sum of the vectors of every calibrated object [75]:

$$E_{x,y}^{\mathrm{miss}} = -\left(\sum_{e} E_{x,y}^{e} + \sum_{\gamma} E_{x,y}^{\gamma} + \sum_{\tau} E_{x,y}^{\tau} + \sum_{\mathrm{jets}} E_{x,y}^{\mathrm{jets}} + \sum_{\mu} E_{x,y}^{\mu} + \sum_{\mathrm{soft}} E_{x,y}^{\mathrm{soft}}\right), \qquad (4.7)$$

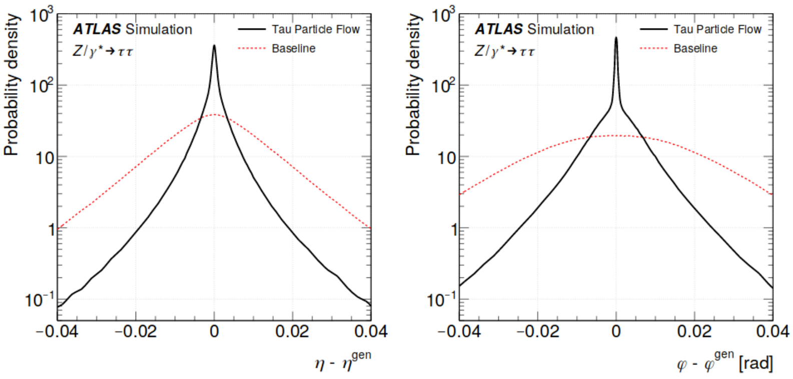


Figure 4.9: Depicted is the probability density of the η (left) and ϕ (right) deviation of the reconstructed direction and the Monte Carlo generated direction. The reconstruction method described in Section 4.3.4 is referred to as Tau Particle Flow [65].

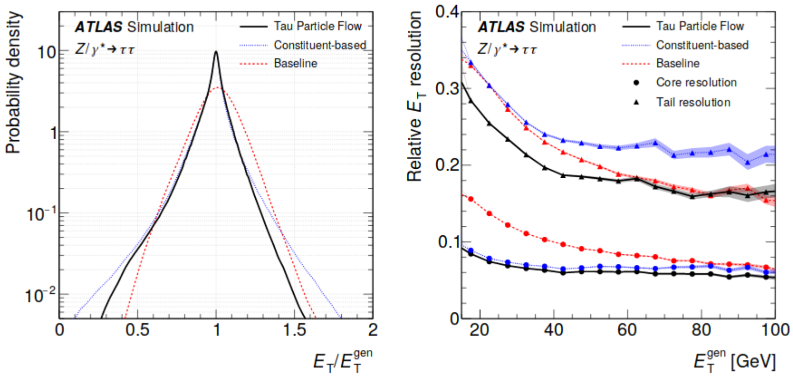


Figure 4.10: Depicted is the probability density of the relative transverse energy deviation (left) and the relative transverse energy resolution as a function of the Monte Carlo generated transverse energy (right). The reconstruction method described in Section 4.3.4 is referred to as Tau Particle Flow [65].

where the labels say to which physical objects the energy belongs. The energies are estimated using the calorimeter information. The soft term is reconstructed from transverse momentum deposited in the detector which is not associated to any physical object. It could be reconstructed either using calorimeter clusters or tracks that are not associated.

The magnitude and ϕ-direction are then calculated by:

$$\begin{aligned} E_{\mathrm{T}}^{\mathrm{miss}} &= \sqrt{\left(E_x^{\mathrm{miss}}\right)^2 + \left(E_y^{\mathrm{miss}}\right)^2} \text{ and} \\ \phi^{\mathrm{miss}} &= \arctan\left(\frac{E_y^{\mathrm{miss}}}{E_x^{\mathrm{miss}}}\right) . \end{aligned} \tag{4.8}$$

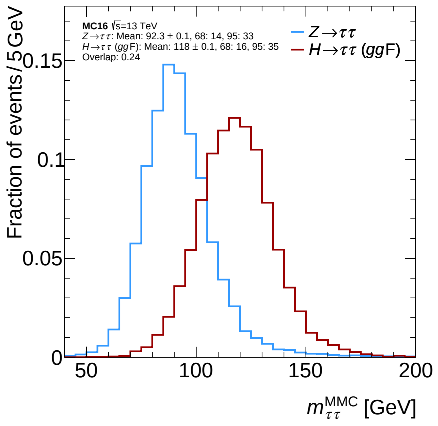


Figure 4.11: Distribution of the predicted di-tau system mass by the MMC for Z/γ^* Monte Carlo generated data (blue) and H Monte Carlo generated data (red). Both distributions are normalized to 1. The overlap describes the area under the intersection of both distributions and measures the discriminating power. The used sample is described in Section 6.1, the sample cuts described in Section 6.1.1.

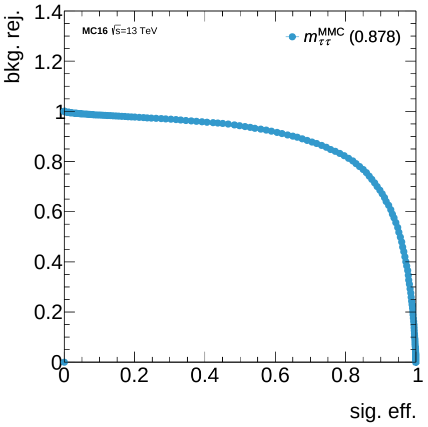


Figure 4.12: Receiver operating characteristics curve where the signal efficiency (sig. eff.) is the true positive rate of correctly classifying a H and background rejection (back. rej.) 1 - the false positive rate of classifying a Z as a H. The used sample is described in Section 6.1, the sample cuts described in Section 6.1.1.

4.4 Di-Tau Mass Estimation

The previous section only considered the detector response of a specific object which can be identified. To characterize and study the event, all object information are combined. In particular resonances like Z or H are of interest which further decay into a pair of leptons among other final states. While some decay channels give a very clean signal and the resonance mass can be calculated accurately, the di-tau channel is experimentally challenging due to the neutrinos involved. It has, however, the feature of being able to give access to measuring polarization and $\mathcal{CP}$ quantum numbers of the resonances [76, 77]. The estimation of the mass of the di-tau system is important to discriminate the resonance of interest, referred to as signal, compared to the background, eg. between H and Z depicted in Figs. 4.11 and 4.12. The here presented Missing Mass Calculator tool (MMC) is used in recent ATLAS analyses, e.g. in the cross section measurement of the Higgs boson decaying into a pair of tau-leptons [12] and is the main discriminator to extract the signal, depicted in Fig. 4.13.

If a resonance, e.g. the H, decays into two tau-leptons, at least two neutrinos occur as decay products. Hence, there are at least 6 unknown variables (3 for each neutrino momentum). If the tau-lepton decays leptonically, there is an additional neutrino. As we cannot estimate the neutrinos individually, the two-neutrino system is estimated with an additional degree of freedom, the mass of the two-neutrino system. Thus, there are 6-8 unknown variables, depending on the decay channel of the tau-leptons. With the estimated neutrino system of each tau-lepton $p_{\mathrm{mis},i}$ and the visible decay products $p_{\mathrm{vis},i}$ the mass of the resonance can be calculated as:

$$m_H = \sqrt{\left(p_{\mathrm{vis},1} + p_{\mathrm{mis},1} + p_{\mathrm{vis},2} + p_{\mathrm{mis},2}\right)^2} \,. \tag{4.9}$$

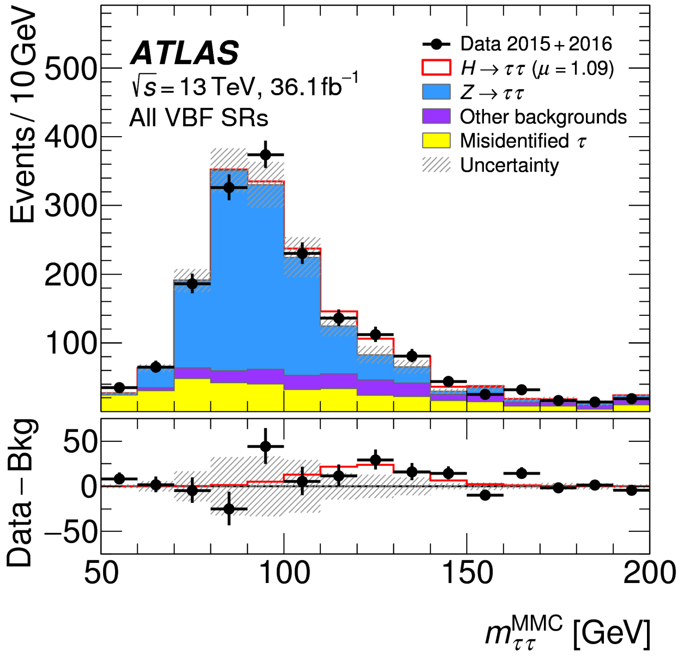


Figure 4.13: Predicted di-tau mass distribution by the MMC. It was used as the main discriminant in a likelihood-fit to extract the signal strength of $H \longrightarrow \tau\tau$. For details, please see [12].

However, there are only 4 equations which constrain the neutrino system originating from the measured missing transverse energy and the tau-mass [15]:

$$\begin{aligned} E_x^{\text{miss}} &= |\vec{p}|_{\text{mis},1} \sin\theta_{\text{mis},1} \cos\phi_{\text{mis},1} + |\vec{p}|_{\text{mis},2} \sin\theta_{\text{mis},2} \cos\phi_{\text{mis},2} \\ E_y^{\text{miss}} &= |\vec{p}|_{\text{mis},1} \sin\theta_{\text{mis},1} \sin\phi_{\text{mis},1} + |\vec{p}|_{\text{mis},2} \sin\theta_{\text{mis},2} \sin\phi_{\text{mis},2} \\ m_\tau^2 &= m_{\text{vis},1}^2 + m_{\text{mis},1}^2 + 2\left(\sqrt{|\vec{p}|_{\text{vis},1}^2 + m_{\text{vis},1}^2}\sqrt{|\vec{p}|_{\text{mis},1}^2 + m_{\text{mis},1}^2} - |\vec{p}|_{\text{vis},1}|\vec{p}|_{\text{mis},1}\cos\Delta_{\text{vm},1}\right) \\ m_\tau^2 &= m_{\text{vis},2}^2 + m_{\text{mis},2}^2 + 2\left(\sqrt{|\vec{p}|_{\text{vis},2}^2 + m_{\text{vis},2}^2}\sqrt{|\vec{p}|_{\text{mis},2}^2 + m_{\text{mis},2}^2} - |\vec{p}|_{\text{vis},2}|\vec{p}|_{\text{mis},2}\cos\Delta_{\text{vm},2}\right), \end{aligned} \tag{4.10}$$

where $\Delta_{\text{vm},i}$ is the angle between the visible tau momentum and the neutrino system momentum. Thus, the equation system is under determined.

In order to estimate the neutrino system, a likelihood is calculated which is a product of three individual likelihoods that are extracted from $Z \longrightarrow \tau\tau$ Monte Carlo generated events. The likelihoods considered are the angle between the visible decay products and the neutrino system ΔR_{vm}, the ratio between the absolute of the momentum of the visible decay products and the neutrino system $|\vec{p}|_{\text{vis}}/|\vec{p}|_{\text{mis}}$ and the resolution of the missing transverse energy $\Delta E_{\text{T}}^{\text{miss}}$ [16, 78]:

$$\mathcal{L}(p_{\text{mis},i}) = \mathcal{P}(\Delta R_{\text{vm}}, p_{\text{mis},i}) \times \mathcal{P}(|\vec{p}|_{\text{vis}}/|\vec{p}|_{\text{mis}}, p_{\text{mis},i}) \times \mathcal{P}(\Delta E_{\text{T}}^{\text{miss}}, p_{\text{mis},i}), \tag{4.11}$$

where $\mathcal{P}(x, p)$ describes the likelihood of x given p. The likelihoods are parameterized as functions of different object and event specific variables. All likelihoods differentiate between events where both tau-leptons decay leptonically, both hadronically or one leptonically and the other hadronically. The likelihood $\mathcal{P}(\Delta R_{\mathrm{vm}}, p_{\mathrm{mis},i})$ is parameterized as a function of the visible tau-lepton transverse momentum $p_{\mathrm{T}}^{\mathrm{vis},i}$ and as a function of the number of charged tracks. The likelihood $\mathcal{P}(|\vec{p}|_{\mathrm{vis}}/|\vec{p}|_{\mathrm{mis}}, p_{\mathrm{mis},i})$ is parameterized as a function of the number of charged tracks and differentiates between leading- and sub-leading-tau, i.e. which one has higher transverse momentum. The likelihood $\mathcal{P}(\Delta E_{\mathrm{T}}^{\mathrm{miss}}, p_{\mathrm{mis},i})$ is parameterized as a function of the square root of the sum of all transverse energy $\sqrt{\sum E_{\mathrm{T}}}$ and the angle between the two visible tau decay products $\Delta\phi_{1,2}^{\mathrm{vis}}$. The phase space of all neutrino system configurations is scanned by a Markov chain. The parameter space is again constrained and only physical solutions are tested. The mass of the di-tau system can now be estimated using the neutrino systems with the highest likelihood [16].

CHAPTER 5

Introduction to Deep Neural Networks

The field of machine learning studies algorithms and models so that computer systems can perform on a given task without being explicitly programmed for that task [79]. This work exploits the methods of deep learning, a sub-field of machine learning, which aims to learn features from an input in many different levels of abstraction [80]. Deep learning has reached excellent performance on benchmark tasks, such as speak recognition [81]. Sometimes the performance even equals or surpass human experts, e.g. in the visual detection of skin cancer [82] or the ImageNet classification task [83], respectively. The recent success makes deep learning also very interesting for various tasks in LHC physics [23]. This work will exploit machine learning and deep learning techniques and tries to provide a setup for the task of the di-tau invariant mass reconstruction with the ATLAS detector at the LHC, presenting not only the final setup but also aims to explain the process of model building in the context of the latest research in machine learning.

In machine learning projects, the utilized techniques depend on the desired goal and the available data. As this work uses Monte Carlo generated data which contains accessible information on the simulated detector response, as well as the generated event information, the machine learning task can be expressed as supervised learning. In supervised learning the input $\vec{x}$ pairs with the output $\vec{y}$ so that the algorithm can infer a rule to map the input to a predicted output, which can then be applied to new examples. For a static dimension of the input, deep feed-forward neural networks can be used. In this chapter, deep learning methods are introduced and intuitions how to optimize them are given. In the following section some abbreviations are introduced (in cursive) and used throughout the work, they can be looked up in Appendix A. A lot of intuitions are inspired from [84].

5.1 Feed-Forward Neural Networks

Feed-forward neural networks (FNNs) are *artificial neural networks* (ANNs) and are inspired by the human brain. In the human central nervous system, neurons are excitable cells which receive, process and transmit information via synapses [85]. This structure is emulated in ANNs, however it is not fully understood how the brain learns, whereas in machine learning optimizing algorithms are used [86]. The artificial neurons are the building blocks of an ANN. In general, an artificial neuron receives an input $\vec{x}$ and has a bias b, where the bias represents the constant offset of the artificial neuron. Each component x_i is multiplied with neuron specific weights w_i, so that different neurons can learn different features, given the same input. An artificial neuron is activated with the value z

determined by an activation function $a : \mathbb{R} \rightarrow \mathbb{R}$:

$$z = a\left(\sum_{i=1}^{N}(x_i \cdot w_i) + b\right) . \tag{5.1}$$

In the analogy of the human brain, the activation function a gives a rule if, and which information should be passed on, as the value z can then be used as an input for other artificial neurons. Which activation functions are used for artificial neurons is described in Section 5.1.4. Note that the bias b is often re-expressed as an additional input $x_0 = 1$ and a corresponding weight w_0 so that $b = x_0 \cdot w_0$. The term feed-forward states that in a FNN the information flow has a distinct direction in which the signals are processed. The artificial neurons are grouped into layers, such that the neuron activation $\vec{z}_{l+1}$ of layer $l + 1$ can be calculated from the the activation of layer l before. Models with intermediate layer size L are considered deep if $L > 1$ and have deepness L, because in principle L levels of abstraction can be learned. An ANN, which is considered deep, is called *deep neural network* (DNN). A general sketch of an FNN is presented in Fig. 5.1. For simplicity, only densely connected layers, in which the output of an artificial neuron in layer l is transmitted to every artificial neuron in layer $l + 1$, are considered. FNNs are often used as models because of the universal approximation theorem [87, 88]. This states that FNNs with a single intermediate layer and a finite number of artificial neurons can approximate any continuous function on compact subsets of $\mathbb{R}^n$ for bounded and monotonically-increasing continuous, non-linear activation functions, and was extended to unbounded activation functions, as well [89]. The challenges in machine learning problems are firstly to find the right number of artificial neurons and secondly to train the weights for optimal approximation.

5.1.1 The Forward and Backward Propagation Algorithm

With the definition of how to calculate the activation of an artificial neuron (5.1), the feed-forward algorithm can be expressed. The weight to calculate the transmission of information from the i^{th} artificial neuron $n_{i,l}$ with activation $z_{i,l}$ in layer l to the j^{th} artificial neuron $n_{j,(l+1)}$ in layer $(l + 1)$ will be called $w^l_{i,j}$. The input $\vec{x}$ will be treated as $\vec{z}_0$ and the biases as $z_{0,l} = 1$ with the corresponding weights $w^l_{0,j}$. Each neuron $n_{i,l}$ in layer l has an activation function $a^l(x)$. The intermediate, or hidden, layer size will be denoted as L, the number of artificial neurons in layer l by N^l. The output $\vec{o}$ can be expressed as $\vec{z}_{L+1}$:

Algorithm 1 Forward propagation

```
procedure FORWARD PROPAGATION($\vec{x} := \vec{z}_0$)
    for ($l$ in (0,$L$)) do
        for ($j$ in (1,$N^l$)) do
            $z_{j,l+1} = a^{(l+1)}(\sum_{i=0}^{N^l}(z_{i,l} \cdot w^l_{i,j}))$
        end for
    end for
    return $\vec{z}_{(L+1)} =: \vec{o}$
end procedure
```

With the forward propagation algorithm the output for a given input and a given set of weights can

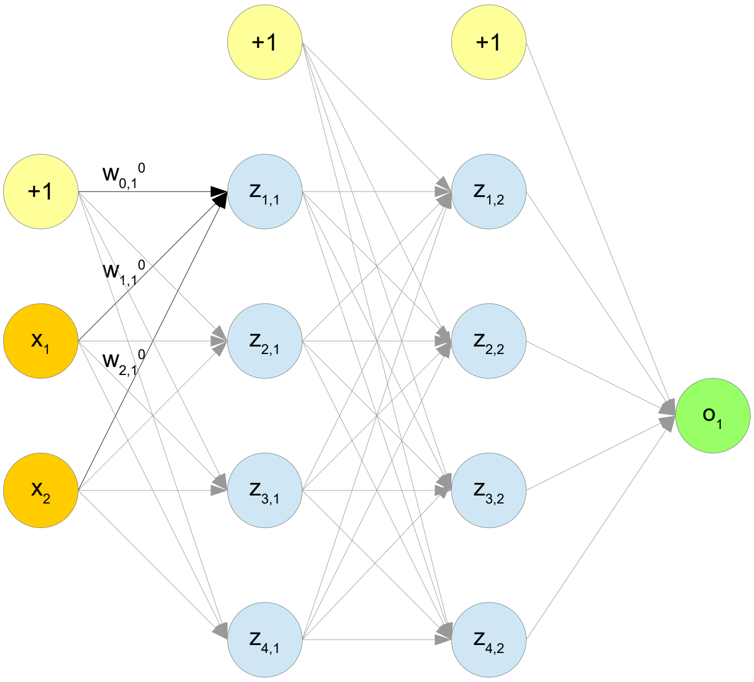


Figure 5.1: Scheme of a FNN with intermediate layer size $L = 2$. For lucidity, only highlighted weights are named. The naming scheme is according to Section 5.1.1.

be calculated. In order to approximate an analytical function or to optimize the mapping, the weights Θ, where Θ contains all weights $w_{i,j}^{l}$, have to be adjusted. As the ANN should learn these weights, a rule how to update them has to be found. Firstly, a loss function $\mathcal{J}(\Theta)$ is defined that should be minimized during the training process. For m training examples, the loss function is defined as:

$$\mathcal{J}(\Theta) = \frac{1}{m} \sum_{k=1}^{m} \mathcal{L}_{\Theta}(\vec{o}, \vec{y}) \,, \tag{5.2}$$

where $\mathcal{L}_{\Theta}(\vec{o}, \vec{y})$ is the individual loss of a training example. The subscript Θ denotes that the output $\vec{o}$ depends on Θ. The individual loss function $\mathcal{L}$ has to be chosen related to the desired task of the ANN. The loss function is a measure of the distance between the predicted output and the true output and can be for example the squared Euclidean distance, see Section 5.1.2. The minimum of the loss function $\min \mathcal{J}(\Theta)$ is typically found by some optimization algorithm. The choice of the optimization algorithm is important for the training performance and various different algorithms are described in Section 5.1.2. However, they all need to compute the impact of a weight $w_{i,j}^{l}$ to the loss function $\mathcal{J}(\Theta)$. The calculation of the partial derivative

$$\frac{\partial \mathcal{J}(\Theta)}{\partial w_{i,j}^{l}} := \Delta_{i,j}^{l} \tag{5.3}$$

can be done by the backward propagation algorithm [90]. The backward propagation algorithm exploits the chain rule, let

$$z_{j,l} = a^l \underbrace{\left(\sum_{i=0}^{N^{(l-1)}} (z_{i,(l-1)} \cdot w_{i,j}^{(l-1)}) \right)}_{:=b_j^{(l-1)}} = a^l \left(b_j^{(l-1)} \right) , \tag{5.4}$$

where $b_j^{(l-1)}$ is the input for the artificial neuron $n_{j,l}$.

Then

$$\Delta^l_{i,j} = \frac{1}{m} \sum_{k=1}^{m} \frac{\partial \mathcal{L}_\Theta(\vec{o},\vec{y})}{\partial z^{(k)}_{j,(l+1)}} \cdot \frac{\partial z^{(k)}_{j,(l+1)}}{\partial b^{l,(k)}_j} \cdot \frac{\partial b^{l,(k)}_j}{\partial w^l_{i,j}} = \frac{1}{m} \sum_{k=1}^{m} \frac{\partial \mathcal{L}_\Theta(\vec{o},\vec{y})}{\partial z^{(k)}_{j,(l+1)}} \cdot a^{'(l+1)}(b^{l,(k)}_j) \cdot z^{(k)}_{i,l} , \tag{5.5}$$

where $a^{'l}$ denotes the derivative of the activation function a^l. The later two terms of the product can be calculated easily for every i, j, where the first term can only be calculated for the output layer $L + 1$ straightforwardly. Let

$$\frac{\partial \mathcal{L}_\Theta(\vec{o},\vec{y})}{\partial z^{(k)}_{j,(L+1)}} \cdot a^{'(l+1)}(b^{l,(k)}_j) := \delta^{(L+1),(k)}_j , \tag{5.6}$$

then

$$\frac{\partial \mathcal{L}_\Theta(\vec{o},\vec{y})}{\partial z^{(k)}_{i,L}} = \sum_{j=1}^{N^{(L+1)}} \frac{\partial \mathcal{L}_\Theta(\vec{o},\vec{y})}{\partial z^{(k)}_{j,(L+1)}} \cdot \frac{\partial z^{(k)}_{j,(L+1)}}{\partial z^{(k)}_{i,L}} = \sum_{j=1}^{N^{(L+1)}} \delta^{(L+1),(k)}_j \cdot w^L_{i,j} . \tag{5.7}$$

Combining all the findings, this gives the backward propagation algorithm which calculates $\Delta\Theta = (\mathbf{\Delta}^0,...,\mathbf{\Delta}^L)$ [1].

Algorithm 2 Backward propagation

```
procedure BACKWARD PROPAGATION($\mathbf{Z} = (\vec{z}_0, ..., \vec{z}_{(L+1)})$,$\Theta$)
    $\delta^{(L+1),(k)}_j = \frac{\partial \mathcal{L}_\Theta(\vec{o},\vec{y})}{\partial z^{(k)}_{j,(L+1)}} \cdot a^{'(L+1)}(b^{L,(k)}_j)$
    $\Delta^{(L)}_{i,j} = \frac{1}{m} \sum_{k=1}^{m} \delta^{(L+1),(k)}_j \cdot z^{(k)}_{i,L}$
    for ($l$ in (0,$L-1$)) do
        $\delta^{(L-l),(k)}_j = \sum_{q=1}^{N^{(L-l+1)}} \delta^{(L-l+1),(k)}_q \cdot w^{(L-l)}_{i,j}$
        $\Delta^{(L-l-1)}_{i,j} = \frac{1}{m} \sum_{k=1}^{m} \delta^{(L-l),(k)}_j \cdot a^{'(L-l)}(b^{(L-l-1),(k)}_j) \cdot z^{(k)}_{i,(L-l-1)}$
    end for
    return $\Delta\Theta = (\mathbf{\Delta}^0,...,\mathbf{\Delta}^L)$
end procedure
```

In the forward and backward propagation algorithm only matrix and vector multiplication are computed. This allows computers to efficiently perform these algorithms if a vectorized implementation is used.

[1] $\mathbf{\Delta}^l$ is the matrix containing $\Delta^l_{i,j}$ for every i, j.

5.1.2 Loss Functions and Optimization Algorithms

As explained, the model learns the weights Θ during an optimization process, minimizing the loss formally introduced in Eq. 5.2. The choice of the individual loss function $\mathcal{L}_\Theta(\vec{o}, \vec{y})$ is important and depends on the task of the FNN on which it should perform. The problem of this work is a regression problem. There exist several loss functions which are specialized for classification problems but are not applicable here. The two main loss functions which can be used are:

$$\begin{aligned} \mathcal{L}_\Theta(\vec{o}, \vec{y}) &= (\vec{o} - \vec{y})^2 \quad \text{squared error} \\ \mathcal{L}_\Theta(\vec{o}, \vec{y}) &= \|\vec{o} - \vec{y}\| \quad \text{absolute error} \end{aligned} \tag{5.8}$$

Both are means of measure for the distance of the predicted output to the given reference values. The absolute error is the Euclidean distance and the squared error is the square of the Euclidean distance. As stated before, the partial derivative of the individual loss function with respect to the output can be calculated easily, which is exploited in the backward propagation algorithm.

Recent work on neural network regression task developed a new loss function, which should correct for edge effects [19]. First of all, edge effects can occur when the target y is limited in some range (a, b). It is costly for the neural network estimator to predict values below a or above b because they were not seen in the training sample. If there is only one input feature x_1 and some linear correspondence $y \propto x_1$ with a Gaussian smearing σ, then the distribution y against x_1 has a hard cut in the range of $y < a$ and $y > b$. This results to the fact that the neural network thinks the probability distribution for x_1 to map to y is a cut off Gaussian. This is sketched in Fig. 5.2. It is proposed that the neural network should maximize the likelihood product which is equivalent to minimizing the sum of the negative log-likelihood, where for the individual likelihood a normal distribution with mean $\mu = o$ is assumed. The boundary correction comes by weighting the likelihood value with the inverse of the overlap of the assumed normal distribution with the sample region (a, b):

$$\begin{aligned} &\min - \sum_i \ln\left(\frac{\mathcal{N}(y^{(i)}|\mu = o^{(i)}, \sigma)}{\int_a^b \mathrm{d}y^{(i)} \mathcal{N}(y^{(i)}|\mu = o^{(i)}, \sigma)}\right) \\ \Leftrightarrow &\min \sum_i \underbrace{\frac{\left(o^{(i)} - y^{(i)}\right)^2}{2\sigma^2} + \ln\left(\operatorname{erf}\left(\frac{o^{(i)} - a}{\sqrt{2}\sigma}\right) - \operatorname{erf}\left(\frac{o^{(i)} - b}{\sqrt{2}\sigma}\right)\right) + \ln\left(\sqrt{2\pi}\sigma\right)}_{\mathcal{L}_\Theta(o,y)} \; . \end{aligned} \tag{5.9}$$

However, this cost function comes with the caveat that the standard deviation σ has to be estimated. The standard deviation itself can also depend on the true value y. While for calorimetry it is a reasonable assumption that $\sigma \propto \sqrt{E}$, for other problems the dependency of σ and y can differ. This can make the DNN training very difficult with this particular loss function.

The simplest method to minimize the loss function $\mathcal{J}(\Theta)$ is to use gradient descent [91]. There the weights are updated proportional to the negative of the derivative of the loss function to the current point:

$$\Theta \leftarrow \Theta - \alpha \frac{\partial \mathcal{J}(\Theta)}{\partial \Theta} \, , \tag{5.10}$$

where α is the learning rate and the operator $a \leftarrow b$ is defined as the variable a gets updated with the

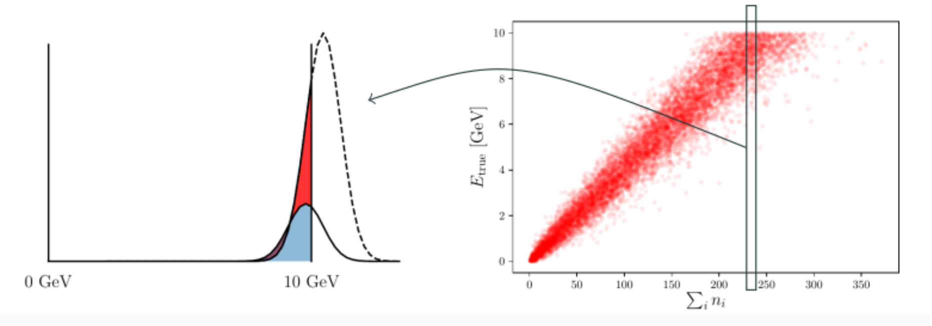


Figure 5.2: Scheme of how a limited target range effects the probability functions for input features. Here in the task of energy reconstruction in a calorimeter with the number of calorimeter hits as input. On the right hand side, the generated particle energy is shown as a function of the calorimeter hits. On the left hand side, the distribution of events is shown as a function of the generated energy for a fixed number of calorimeter hits in a region where boundary effects play a role [19].

value b. The gradient can be calculated by the backward propagation algorithm and all weights are updated simultaneously. This is repeated until the loss function converges to a local minimum. Two small variations to gradient descent are stochastic gradient descent (SGD) and mini-batch gradient descent which try to approximate the minimization of $\mathcal{J}(\Theta)$. In stochastic gradient descent, the training examples $((\vec{x}^{(1)}, \vec{y}^{(1)}),...,(\vec{x}^{(m)}, \vec{y}^{(m)}))$ are shuffled randomly and only a single example is used to update the weights [92, 93]. This is repeated until every training example is seen and is called an epoch:

$$\Theta \leftarrow \Theta - \alpha \frac{\partial \mathcal{L}_\Theta(\vec{o}^{(i)}, \vec{y}^{(i)})}{\partial \Theta} , \tag{5.11}$$

where the partial derivative of $\mathcal{L}_\Theta$ with respect to Θ can be calculated similarly to the backward propagation algorithm.

Mini-batch gradient descent works similar as SGD with the difference that s examples are used for each update, where s is the mini-batch size. Again, the updates are repeatedly performed until every training example was seen:

$$\Theta \leftarrow \Theta - \alpha \frac{1}{s} \sum_{i=1}^{s} \frac{\partial \mathcal{L}_\Theta(\vec{o}^{(i)}, \vec{y}^{(i)})}{\partial \Theta} . \tag{5.12}$$

Note that for $s = 1$ mini-batch gradient descent equals SGD, for $s = m$ it equals gradient descent. Mini-batch gradient descent has a few advantages compared to SGD and gradient descent. While SGD can be very noisy, i.e. the direction of the weight adjustment can be totally different for different training examples, gradient descent can be slow in the training process. This could be due to plateaus in the multi-dimensional loss function, where the derivatives are small. The noise of mini-batch gradient descent can pull the weight vector randomly away from plateaus or even away from local optima. Note that points of zero gradient in the loss function in multi-dimensional space are more likely to be saddle points than local optima [94]. However, finding the global optimum can not be guaranteed.

The fact that the loss function depends on many parameters makes it very unintuitive. However, if the loss function is imagined to depend only on two parameters $\mathcal{J}(\theta_1, \theta_2)$, then some intuitions can be gained. A possible contour plot could look for example like as depicted in Fig. 5.3. There, the

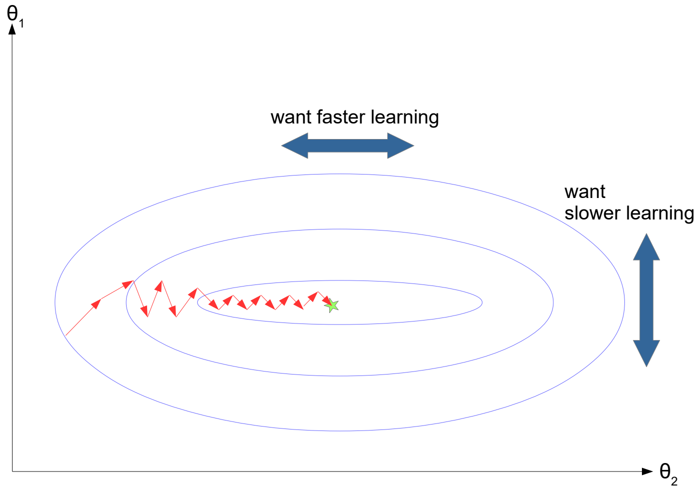


Figure 5.3: Scheme of a contour plot of $\mathcal{J}(\theta_1, \theta_2)$. Lines connect space-points of equal loss. The point of optimal loss is represented by the star. The weight updates are visualized by the red arrows.

parameter updates can be viewed as a two dimensional vector ($\delta\theta_1$,$\delta\theta_2$). The component $\delta\theta_1$ gets updated alternating positively and negatively, whereas $\delta\theta_2$ gets updated only positively. It can be concluded that the updates can be stabilized, when they are averaged. This can also speed up training. In order to achieve this, a momentum term is introduced [95]. The presented algorithms can be easily extended to include the momentum term. The weight updates are then defined recursively:

$$u_t = \underbrace{-\alpha \frac{1}{s} \sum_{i=1}^{s} \frac{\partial \mathcal{L}_\Theta(\vec{o}^{(i)}, \vec{y}^{(i)})}{\partial \Theta}}_{\mathrm{d}\Theta} + \beta u_{t-1} = \mathrm{d}\Theta + \beta u_{t-1} \, , \tag{5.13}$$

where $\mathrm{d}\Theta$ is the update on Θ as in the (mini-batch) gradient descent algorithm, $\beta < 1$ is the so-called momentum and u_t is the t^{th} actual update on Θ. Notice that it is similar to the calculation of an exponentially weighted moving average without bias correction, which would look like

$$u_t = (1 - \beta)\mathrm{d}\Theta + \beta u_{t-1} \, . \tag{5.14}$$

The parameters are then updated according to

$$\Theta_t \leftarrow \Theta_{t-1} + u_t \, . \tag{5.15}$$

There exist a variety of advanced optimization algorithms which all aim to achieve better convergence rates to the minimum. To give a small overview, the established advanced optimization algorithms are:

- Nesterov accelerated momentum (NAM) [96]
- RMSprop [97]
- Adagrad [98]

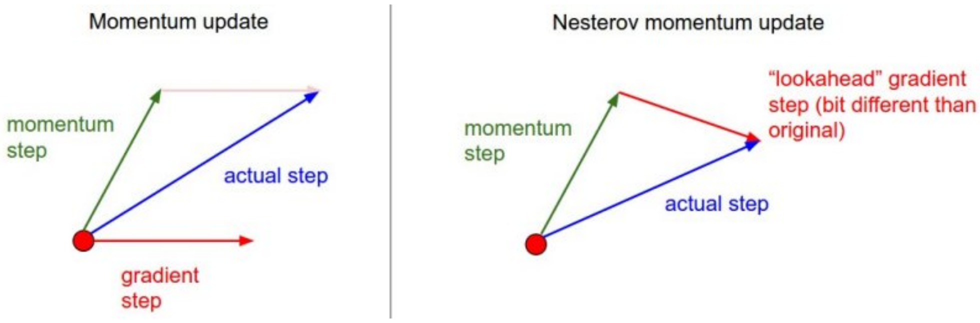


Figure 5.4: Scheme of how Nesterov accelerated momentum (right) differs from SGD with standard momentum (left). The green arrow represents the momentum step βu_{t-1} in Eq. 5.13 and Eq. 5.16, the red arrow represents the gradient step $\mathrm{d}\Theta$ and the blue arrow the actual step $\mathrm{d}\Theta + \beta u_{t-1}$ [102].

- Adadelta [99]
- Adam [100]
- Nadam [101] .

Nesterov accelerated momentum

Nesterov accelerated momentum is similar to SGD with a momentum term, except that the partial derivative is calculated with respect to an estimated parameter update $\Theta_{t-1+\beta u_{t-1}}$. This guarantees a stronger theoretical performance and in practice leads to slightly better results. The weight updates are then defined as:

$$u_t = \underbrace{-\alpha \frac{1}{s} \sum_{i=1}^{s} \frac{\partial \mathcal{L}_{(\Theta_{t-1}+\beta u_{t-1})}(\vec{o}^{(i)}, \vec{y}^{(i)})}{\partial(\Theta_{t-1} + \beta u_{t-1})}}_{\mathrm{d}\Theta} + \beta u_{t-1} \, . \tag{5.16}$$

Graphically, the algorithm is depicted in Fig. 5.4.

RMSprop

RMSprop tries to tackle the same problem as the momentum term, namely to dampen out oscillations in parameter updates. It is an adaptive learning optimization algorithm, as the learning rate gets adapted for each parameter θ in Θ individually. The idea is to divide the parameter updates by the square root of the exponentially weighted moving average of the square of the parameter updates:

$$\begin{aligned} s_{(\partial\theta)^2}(t) &= \beta s_{(\partial\theta)^2}(t-1) + (1-\beta)\,(\partial\theta(t))^2 \\ \theta(t) &= \theta(t-1) - \frac{\alpha}{\sqrt{s_{(\partial\theta)^2}(t) + \epsilon}} \partial\theta(t) \, , \end{aligned} \tag{5.17}$$

where $\epsilon > 0$ to ensure not to divide by 0 and

$$\partial\theta(t) = \sum_{i=1}^{s} \frac{\partial \mathcal{L}_\Theta(\vec{o}^{(i)}, \vec{y}^{(i)})}{\partial\theta} \, . \tag{5.18}$$

By doing so, the weights get updated more slowly if the previous updates were big and the other way around.

Adagrad and Adadelta

Adagrad and Adadelta are like RMSprop adaptive learning optimization algorithms. They are not tested in this work and for completeness described in Appendix B.

Adam

Adam is essentially a combination of RMSprop and momentum. The only difference is that Adam applies a bias correction to the averages:

$$\begin{aligned}
u(t) &= (1-\beta_1)\partial\theta + \beta_1 u(t-1) \\
\hat{u}(t) &= \frac{u(t)}{1-(\beta_1)^t} \\
s_{(\partial\theta)^2}(t) &= \beta_2 s_{(\partial\theta)^2}(t-1) + (1-\beta_2)\left(\partial\theta(t)\right)^2 \\
\hat{s}_{(\partial\theta)^2}(t) &= \frac{s_{(\partial\theta)^2}(t)}{1-(\beta_2)^t} \\
\theta(t) &= \theta(t-1) - \frac{\alpha}{\sqrt{\hat{s}_{(\partial\theta)^2}(t) + \epsilon}}\hat{u}(t) \, .
\end{aligned} \tag{5.19}$$

Nadam

Nadam works identically to Adam simply replacing the momentum term $u(t)$ in Eq- 5.19 by the introduced Nesterov accelerated momentum in Eq. 5.16.

5.1.3 Standardization of the Input

Another consequence of ellipse shaped contour plots, like in Fig. 5.3, is that the input data should be preprocessed before the training procedure. Standardization of the input features, i.e. all input features standard normal distributed, is a common requirement to make ANNs better trainable. Often, the shape of the input features is neglected but they are transformed to have zero mean and unit variance [103]. The basic intuition is that the loss function is on average more symmetric for standardized inputs. To give an example, one could think of a DNN which task it is to classify jets as QCD-jets or tau-leptons. This could take e.g. the transverse momentum $p_\mathrm{T}/\mathrm{GeV}$, the number of tracks n_tracks or the isolation cone $\Delta R_\mathrm{isolation}$ as input features. The problem is that $p_\mathrm{T}/\mathrm{GeV}$ and $\Delta R_\mathrm{isolation}$ are in different number ranges. If the initial values of the weights Θ are small, then the gradient on the weights which pass on the information of $p_\mathrm{T}/\mathrm{GeV}$ are likely to be much bigger than the other gradients and training can become unstable. If all input variables are in the same number region, this problem could potentially be avoided.

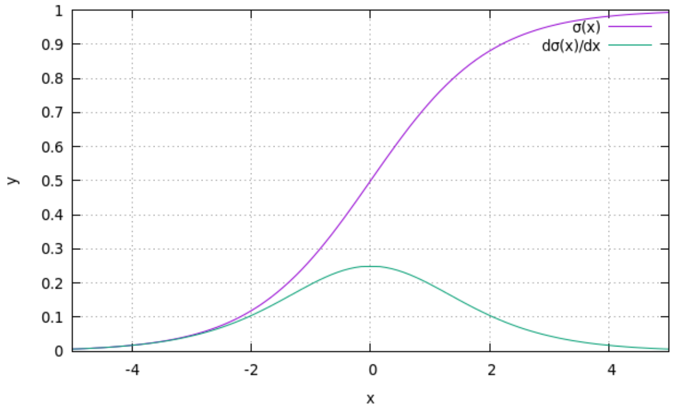


Figure 5.5: Illustration of the logistic activation function $\sigma(x)$ and its derivative to show the vanishing gradient problem.

5.1.4 Activation Functions and Weight Initialization

Activation functions are crucial for the ANN to learn complex non-linear behaviors. If every artificial neuron is activated by the linear function $a(x) = x$, then the output can only be a linear combination of the input plus a bias. The universal approximation theorem was first proven for the logistic activation function

$$a(x) = \sigma(x) = \frac{1}{1 + \exp(-x)}, \tag{5.20}$$

which is a sigmoid function. The characteristic S-shaped curve is shown in Fig. 5.5. The problem of sigmoid functions is that the absolute of the gradient is smaller one. When training a FNN with a gradient based optimizer, the gradients $\partial\mathcal{J}/\partial w^l_{i,j}$ can be very small. Because of the details of the backward propagation, the later layers affect the gradient on the previous layers, resulting in even smaller gradients in the first layers. This problem is known as the vanishing gradient problem [104]. It slows down the learning in FNN with many intermediate layers and makes it very difficult to train. On the other hand, if the gradients in the later layers are large, i.e. > 1, this can lead to very big gradients in the previous layers and is known as the exploding gradient problem [105]. Both problems can make learning very unstable.

The vanishing gradient problem can be addressed with using other activation functions. A very popular activation function is the rectified linear unit (ReLU)

$$a(x) = \begin{cases} 0, & \text{if } x \leq 0 \\ x, & \text{otherwise} \end{cases}. \tag{5.21}$$

As the ReLU is not differentiable, which is a necessary condition to use the backward propagation algorithm, the derivative is simply defined as

$$a'(x) = \begin{cases} 0, & \text{if } x \leq 0 \\ 1, & \text{otherwise} \end{cases}. \tag{5.22}$$

Another tool which tries to address a stable learning is weight initialization. Often an ANN is trained from scratch, meaning that there is no initial guess for the weight parameters Θ. If all weights were initialized by the same value, then the derivatives $\partial \mathcal{J} / \partial w_{i,j}^{l}$ would be equal for every i, j, meaning that the layers can effectively learn only one feature. Therefore, the weights need to be initialized differently, to assure a symmetry breaking. Often this is done by initializing the weights randomly. Empirically, it was found that poorly initialized DNNs are very difficult to train [106]. Several works on how to initialize the weights exist. All methods are similar in a way that they propose a uniform distribution ($\mathcal{U}(a, b)$, where a and b are the upper and lower bound) or normal distribution ($\mathcal{N}(\mu, \sigma^2)$, where μ is the mean and σ^2 the variance) with mean zero, differing in the bounds and standard deviation but all taking the number of input nodes, sometimes also of output nodes, into account. Denoting the number of input nodes by n_{in} and the number of output nodes by n_{out}, the different initialization methods are:

- LeCun normal [107]: $\mathcal{N}\left(0, \frac{1}{n_{\text{in}}}\right)$
- LeCun uniform [107]: $\mathcal{U}\left(-\sqrt{\frac{3}{n_{\text{in}}}}, \sqrt{\frac{3}{n_{\text{in}}}}\right)$
- Glorot normal [108]: $\mathcal{N}\left(0, \frac{2}{n_{\text{in}}+n_{\text{out}}}\right)$
- Glorot uniform [108]: $\mathcal{U}\left(-\sqrt{\frac{6}{n_{\text{in}}+n_{\text{out}}}}, \sqrt{\frac{6}{n_{\text{in}}+n_{\text{out}}}}\right)$
- He normal [83]: $\mathcal{N}\left(0, \frac{2}{n_{\text{in}}}\right)$
- He uniform [83]: $\mathcal{U}\left(-\sqrt{\frac{6}{n_{\text{in}}}}, \sqrt{\frac{6}{n_{\text{in}}}}\right)$.

The initialization methods are sometimes designed for specific activation functions, e.g. the He normal/uniform initialization method is specialized on rectified activation functions. Notice that the bias weights $w_{0,j}^{l}$ are often initialized differently than the other weights, e.g. set to zero. The intuition of the weight initialization is that the ANN has to be trained from scratch and the weights should be small to not have artificial neurons with large activation values in order to avoid the exploding gradient problem.

5.1.5 Self-Normalizing Neural Networks

With the information above, a FNN $\mathcal{F}$ could be set up, defining a mapping $\mathcal{F}(\vec{x}) = \vec{o}$. However, many decisions have to be made, such as choosing a number of intermediate layers, the number of artificial neurons for each layer, an activation function for each artificial neuron, a loss function, an optimization algorithm (and its parameters) and an initialization scheme for all the weights. Having chosen all these things, it is not guaranteed that the training will work efficiently. How to analyze the training procedure will be discussed later. However, there exist also a few more tricks to stabilize the training procedure. Having said that standardization of the input is important, it was seen that it can also be useful to normalize the input of each layer. The Batch Normalizing Transform algorithm uses the statistics in a mini-batch to transform the input of each artificial nodes [109]. The algorithm is depicted in Algorithm 3 in Appendix B. The whole procedure is done to reduce the so called internal shift. To illustrate what this is, it is useful to cover the first few layers of a DNN and just look at the last few layers and the output. This describes a new ANN which task it is to find a mapping of a given input to a given output. However, during training the input varies, as the network is trained. If the input varies very rapidly, then the last layers cannot focus on learning distinct features but maybe tend to find different features with every training step. If the input values are confined in some sense, or change only slowly, this can help to make the training procedure more stable. In FNN batch normalization is not as successful as for different architectures, i.e. convolutional neural networks or recurrent neural networks. There exist different techniques to normalize FNNs, and for this work the setup of *self-normalizing neural networks* (SNNs) is chosen [110]. SNNs do not explicitly perform a transformation on the inputs to each artificial neuron but ensure that the neurons $z_{i,l}$ have a fixed point with zero mean and unit variance, to which they always tend. To achieve this a special activation function is used together with the presented LeCun normal weight initialization method. The activation function is the scaled exponential linear unit (SELU) and is given by

$$a(x) = \mathrm{selu}(x) = \lambda \begin{cases} \alpha \exp(x) - \alpha, & \text{if } x \leq 0 \\ x, & \text{otherwise} \end{cases}, \tag{5.23}$$

with $\lambda = 1.0507$ and $\alpha = 1.6733$. Note that the function can reach negative and positive values, has a slope greater one for positive values and a saturation for negative values. All of this is needed for the self-normalizing properties. The LeCun normal weight initialization is important for the mapping of one layer to the next. If a layer has activation $z_{i,l}$ which are independent but share mean $\mu = 0$ and variance $\sigma = 1$, then the input to $n_{j,(l+1)}$ is a weighted sum of these independent variables. For that, the central limit theorem (CLT) holds. If the weights have mean $\mu(\vec{w}_i^l) = 0$ and variance $\sigma(\vec{w}_i^l) = 1/n_{\text{in}}$, then the input distribution of neuron $n_{j,(l+1)}$ is approximately a normal distribution $\mathcal{N}(\mu\nu, \sigma\tau)$, where $\nu = \sum_i \mu(\vec{w}_i^l)$ and $\tau = \sum_i \sigma(\vec{w}_i^l)$, hence $\sim \mathcal{N}(0, 1)$. In the CLT, the distribution is closer to a normal distribution, the larger the input size of a given artificial neuron is, i.e. layers with many artificial neurons are required for SNNs. This is ensured by the fact that the layers of DNNs are typically very broad. The advantages of SNNs are that they do not suffer from the exploding or vanishing gradient problem, because the activation of the artificial neurons are bounded and the variance close to one. Also strong regularization schemes, which will be explained later, can be applied to SNNs. For detailed proofs of the self-normalizing properties, see [110].

5.2 Optimizing Neural Network Architectures

In the previous section tools to build and train a FNN were introduced. By restricting to SNNs, the activation function of the artificial neurons and the initialization scheme is fixed. There are still many parameters to choose and twist. This section tries to give guidelines on how to choose and optimize the so-called hyperparameters of the FNN in order to optimize its performance. First of all, possibilities how to check the performance and the predictive power of a FNN are introduced. Of course, performance and predictive power are closely related, i.e. a FNN with low performance is expected to have a low predictive power on unseen new examples but for the development of a FNN it is useful to think of both things separately and find (nearly) orthogonal adjusting mechanisms.

In order to estimate the predictive power of a FNN, we need to hold back a piece of the data set which is large enough to be representative. This data set is orthogonal to the rest which will be called training data. However, if the parameters of the FNN are adjusted to perform good on the held back data, then the estimation of the predictive power is biased. In order to avoid this problem, the data set is split into three orthogonal pieces which will be called the training, cross-validation and test sample. The training sample contains the input output pairs which the FNN will get for training, the cross-validation, or development sample will be used for choosing which configuration has the best predictive power and the test sample is only used for getting an unbiased estimate for the predictive power of the FNN. The splitting of the samples has to be done carefully and depending on the task in a strategic way. Sometimes it is fine to just randomly split the whole sample but this could lead into problems. Imagine the task of classifying mice, cats and dogs in a data set of labeled pictures, if the development set would not contain any cat pictures, it could be that when analyzing the outcome of the test set all cat pictures were misclassified as dog pictures. The relative size of the three individual data sets depends on the absolute size of the whole data set. There exist no strict rule on how to split the data set. As explained later, it is desirable to have a very large training data set. Therefore, it can be fine, to split the data set into $\sim$ 98% training sample, $\sim$ 1% cross-validation and $\sim$ 1% test data sample, for very large data sets. For small data sets, it can be useful to split the data set only into two orthogonal data sets, a training and a test data set, and then use k-fold cross validation on the training set [111]. In the process, the FNN has to be trained k times, which can be counterproductive for large data sets, as the training then takes very long time.

After having split the data set into training, cross-validation and test set, some metrics have to be defined which measure the performance and the predictive power. The predictive power is the performance of the FNN on unseen new data. When comparing two different FNNs it is desirable to have a single, real-valued evaluation metric which tells which of the two FNNs performs better. For regression tasks, this can be the loss on the cross-validation set. In practice, this is not always achievable, because the single metric can not take all the effects into account that can occur. When realizing a strange effect or behavior of the FNN performance for some training examples, it can be useful to formulate necessary conditions. For regression tasks, this is often more difficult than for classification tasks because there exist more, and more advanced metrics, such as the F1-score [111], the accuracy or the area under the ROC-curve. When realizable, it is always good to look into examples that are not well predicted by the FNN and see if there are any patterns. Also, it is always good to have something to compare the FNN against. For image classification this can be a human (expert), as humans are very good on natural perception tasks. The comparison tells, how good a FNN can at least be, and therefore can tell if the performance can be increased. Having set a decision rule to rank the performance of different FNNs, the hyperparameters can be varied and the best FNN can be chosen.

5.2.1 Bias and Variance

Bias and Variance are two concepts which explain how well a ANN generalizes features. The theoretical foundations are that the expected squared error of a model $\hat{f}(x)$ on unseen new data (x, y) following the underlying model $f(x)$ can be decomposed into[112]

$$\mathrm{E}\left[(y - \hat{f}(x))^2\right] = \left(\mathrm{Bias}\left[\hat{f}(x)\right]^2 + \mathrm{Var}\left[\hat{f}(x)\right] + \mathrm{Var}\left[f(x)\right]\right) . \tag{5.24}$$

In practice this is not usable, as the underlying model is unknown and the error can only be measured as a whole and not separated into terms resulting from bias or variance or the irreducible error Var $[f(x)]$. Therefore, it is looked at regions of either high bias or high variance. The concepts are feasible when

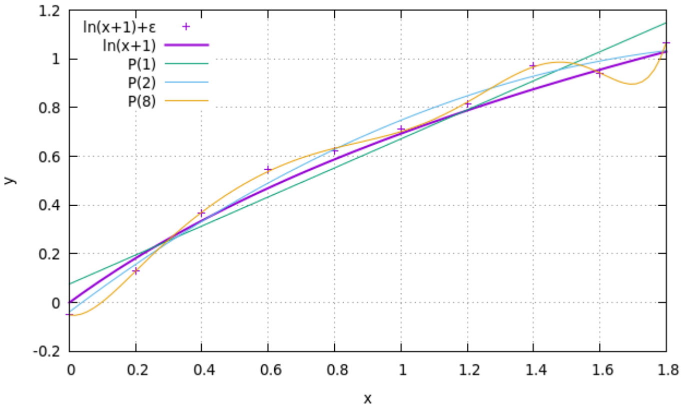


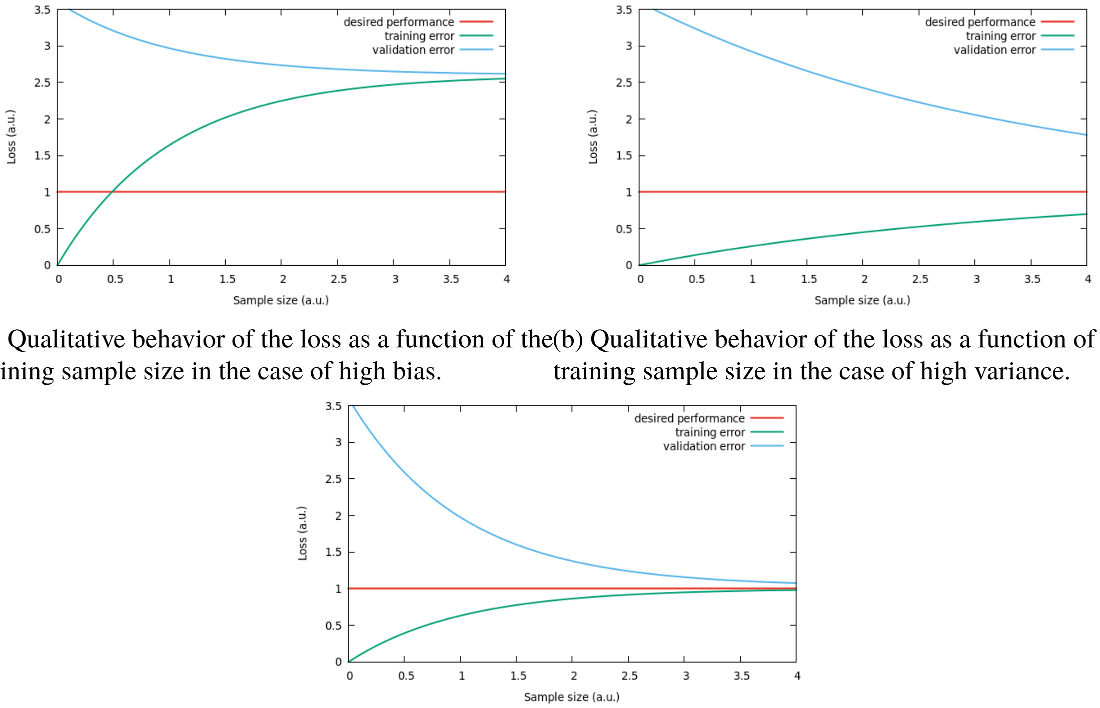
Figure 5.6: Random generated data according to $f(x) = \ln(x+1) + \epsilon$ in comparison to the function $\ln(x+1)$ and three fitted polynomials of order 1, 2 and 8.

looking at a function with some random noise. Let

$$f(x) = \ln(x+1) + \epsilon \qquad \text{for } x \in [0, 2]\,, \tag{5.25}$$

where ϵ is some random noise distributed like $\mathcal{N}(0, 0.05)$. In order to approximate the underlying model, three polynomials are fitted to random generated data points. Denoting a polynomial of order n by $P(n)$. The results are depicted in Fig. 5.6. The first order polynomial underfits the data in the sense that is misses the flattening of the underlying function for larger values. It suffers from high bias, i.e. when predicting new values it would fail due to a too simple model of the data. The second order polynomial has a steeper rise for smaller values and flattens out for larger. This more or less represents the underlying function and would be considered a good model. The polynomial of order eight fits every training example very well but can only limitedly generalize the underlying function. It suffer from high variance, i.e. when predicting new values it would fail, because the model predicts a complex behavior in between the training examples. For complex DNN tasks the low order polynomial represents a very shallow FNN with only a few artificial neurons, the high order polynomial a complex FNN with many artificial neurons.

To diagnose whether an ANN suffers from high bias or high variance or both, it is good to look at learning curves. There are two possible ways to track the learning. The first method is to plot the training and the cross-validation error of a fully trained model for different training sample sizes. If the training error is fast above the desired performance error and the cross-validation error close to the training error, the model suffers from high bias. If there is a large gap between the training and the cross-validation error, this hints that the model suffers from high variance. The qualitative shape of these learning curves are depicted in Fig.5.7. A major advantage of this method is that in case of high variance, it can be estimated how much more training data would be needed, as in the high variance scenario the loss would converge towards the optimal performance value for ever larger training samples. The drawback of this method is that a model has to be trained with many different sample sizes and thus it takes a long time to obtain the learning curve. A second method to obtain a learning curve is to plot the training error and cross-validation error against the number of epochs. A high training error, a high cross-validation error and low discrepancy between the two errors hints at



(a) Qualitative behavior of the loss as a function of the training sample size in the case of high bias.

(b) Qualitative behavior of the loss as a function of the training sample size in the case of high variance.

(c) Qualitative behavior of the loss as a function of the training sample size in the case of optimal performance.

Figure 5.7: Behavior of the loss as a function of the training sample size in different scenarios.

high bias while a small training error, a high cross-validation error and large discrepancy between the two hints at high variance. In the case of high variance, the cross-validation sometimes rises at some point because then the model weights are adjusted to learn the specific features of the training example which is bad for the generalization quality of the model. The advantage of the second method is that the learning curve can be obtained during training and the model is only trained once with the fully available training sample.

In order to say whether a training error is high or low, it is good to have a reference value, e.g. performance of existing models or human level performance. When this is not available, it can be good to go into regimes of high bias, i.e. a model with low complexity, and regimes of high variance, i.e. very complex models, in order to get boundaries for the optimal training error.

After diagnosing high bias or high variance or sometimes a combination of both, there are a few mechanisms which can control those. If the ANN suffers from high bias, it is useful to increase the model complexity by using more artificial neurons and/or more intermediate layers. Before making the model too complex it should be checked, whether the model training was successful, i.e. did not stop because of slow training in a plateau region. If the ANN suffer from high variance, the model complexity can be reduced. However, this is not always desirable, e.g. if the performance on the training set is still below, the desired performance. Then it could help to collect more data, i.e. this can limit overfitting. Collecting more data can be (time) costly. Two other things which can reduce high variance are using regularization or dropout.

5.2.2 Regularization

Regularization and dropout (described in Appendix B) are two mechanisms which address high variance. As explained, high variance occurs when the model is too complex. The complexity is described by the number of intermediate layers and the number of artificial neurons. The number of intermediate layers does also give the level of abstraction of the learned features, the number of artificial neurons the number of features which can be learned. Therefore, it is good to have ways to cope with high variance other than reducing the model complexity. Regularization introduces an additional term to the loss function

$$\mathcal{J}(\Theta) = \frac{1}{m}\sum_{i=1}^{m} \mathcal{L}_\Theta(\vec{o}, \vec{y}) + \frac{1}{m}\sum_{l=0}^{L}\sum_{i,j}\left[\lambda_1 |(w^l_{i,j})| + \lambda_2 (w^l_{i,j})^2\right] , \tag{5.26}$$

where λ_1 is the regularization parameter for the so-called L1 regularization, and λ_2 the regularization parameter for the so-called L2 regularization. The two different regularization schemes have different effects on the weights in the ANN. L1 regularization favors sparse weights, i.e. a lot of weights are zero, while with L2 regularization the weights get very small and it is often called weight decay. When $\alpha \mathrm{d}\Theta$ describes the weight updates calculated with the back propagation algorithm without any regularization, then the weight updates with L2 regularization reads

$$\Theta \leftarrow (\Theta - \frac{2\lambda_2\alpha}{m}\Theta) - \alpha \mathrm{d}\Theta . \tag{5.27}$$

Because the weights are kept either sparse or small, the model complexity is reduced and the model gets less prone to overfitting.

5.2.3 Hyperparameter Tuning

As introduced in this section, there are ways to judge the quality of a DNN and ways to improve its performance. Additional regularization parameters were introduced and added to the set of hyperparameters which should be optimized. As the training of an individual DNN setup usually takes very long time, a grid hyperparameter search is computationally very expensive. There are some things which could be done in order to avoid grid searches. As it is hard to tell in advance, which hyperparameters are the most important and where their optimal region lies, a grid search can be very inefficient. Because some hyperparameters are more important than others and/or more sensitive to small deviations, a random sampling of the hyperparameters exploits a larger variety of values for all parameters and is therefore recommended. Parameters which are more sensitive to small deviations should also be sampled according to logarithmic scales, rather than linear scales. In order to get a flavor of the ranking of the hyperparameters it can be good to start with a given set of hyperparameters and vary some of them by hand.

Often the learning rate α is the most important hyperparameter and is therefore recommended to be tested first. This can then limit the range of α where good results are expected. If the training error does not further decrease, the learning rate can be lowered, which allows to go closer to the local minimum and can cause jumps in the learning curve. This should not be done too early, as it can instantly decrease the training loss but could decelerate the convergence progress.

The more advanced optimization algorithms have additional hyperparameters but they do also come with default values and they are considered to have a smaller impact on the DNN performance than the learning rate. However, they do have advantages over the standard mini-batch SGD and should be used or at least tested. The choice of which optimization algorithm to use can be viewed as a hyperparameter itself.

A very important hyperparameter is the mini-batch size s, which is usually chosen as a power of 2, e.g. 32, 64, ..., 512. Recent research hinted that larger mini-batch sizes can lead to the optimizer going into sharp minima which lead to poorer generalization and worse predictive power [113]. Sharp minima mean that a small variation of the input vector leads to a large deviation on the output and a much higher loss value. Therefore, only $s = 32$ and $s = 64$ are considered.

The first focus should lie on optimizing the training error, and therefore choose a number of intermediate layers and of artificial neurons in each layer. As very complex DNNs take a very long time to train, it should definitely be started with less complex neural networks and decided whether the training performance is close to the desired performance. By ruling out regions in which high bias occurs, the possible range of a random sampled hyperparameter search can be decreased. After optimizing the training error, the validation error should give a diagnosis on whether the network suffers from high variance. If that is the case, then dropout should be used and tested with keeping probabilities of $p = 0.95$ and $p = 0.90$. This is simply to the fact that it is much more difficult to find good regularization parameters. In tasks where there is no reference value for a desired performance, it can be very hard to find an optimal interplay between optimizing the training error and avoid variance at the same time.

CHAPTER 6

Data Sample and DNN Setup

This chapter aims to describe how to set up an environment where a DNN can be trained for the specific task of the mass estimation of a di-tau system. While in the previous chapter all tools were described how a DNN can be set up, this chapter also takes into account which information should be used as input for a DNN and which data sample is suitable for the training. As stated before, this work uses Monte Carlo generated data where event generated information is accessible for supervised learning. First, a short overview about the data sample is given and why it was chosen. Secondly, the variables are introduced which will be fed into the DNN. Lastly, the architectures and hyperparameters which will be tested for the DNN are introduced.

6.1 Data Sample

The data sample and the available statistics are the most important part in the development of the neural network di-tau mass estimator. As high statistics are necessary in order to successfully train a DNN, there is also a strong interplay of the data used for training and what it can predict. Similar to a Taylor expansion of a function $f(x)$ around a point x_0 which is not expected to predict well for x with $|x - x_0| >> 1$, the DNN is not expected to predict masses well outside the training mass range. A suitable data sample is needed so that the DNN can fulfill the requirements of exploiting the decay kinematics, covering a broad mass range, especially the mass range of 90 – 125 GeV, and of delivering an unbiased result. The mass range of 90 − 125 GeV is of particular interest because there the Z and H resonances lie. As explained the estimated di-tau system mass is the main discriminant for $H \longrightarrow \tau\tau$ analyses with $Z/\gamma^* \longrightarrow \tau\tau$ background and the neural network estimator should perform well in this region. Due to the estimators resolution, the mass range of the training sample should have a safety margin to both sides. It cannot be guaranteed that the DNN is able to exploit the decay kinematics but with high statistics in a broad mass range the fundamentals can be given. In order to prevent the DNN from being biased, i.e. having a preferred mass, the mass spectrum should be flat over the whole mass range. The necessity of having a broad mass range can be explained by the mass resolution again. If the mass window would be narrower than the theoretical achievable mass resolution, the best prediction would always be close to the mean mass.

For the realization of the data sample's demands, an unphysical $\gamma^* \longrightarrow \tau^{\text{had}}\tau^{\text{had}}$ data sample was used, where τ^{had} indicates that the tau-lepton decays hadronically. The unphysical sample has virtual photons as intermediate states and re-weighted matrix elements in order to modify the mass distribution.

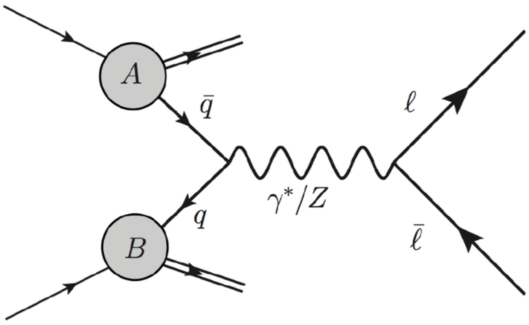


Figure 6.1: Feynman diagram of the Drell-Yan process [114].

As only hadronic tau decay modes where simulated, the estimator is limited to the prediction of di-tau systems, where both tau-leptons decay hadronically. Other decay channels would have additional neutrinos which the DNN cannot know about. Note, that the unphysical $\gamma^* \to \tau^{\text{had}}\tau^{\text{had}}$ process has a different coupling to left- and right-chiral tau-leptons than the physical Drell-Yan process, where an interference between $\gamma^* \to \tau\tau$ and $Z \to \tau\tau$ occurs. A Feynman diagram of a Drell-Yan process is shown in Fig. 6.1. The sample is unphysical because in experiments the process $\gamma^* \to \tau\tau$ cannot be observed but only with the interference of γ^* and Z. In order to compare the DNN di-tau mass estimator to more realistic scenarios, the neural network estimator can be tested on $Z/\gamma^* \to \tau\tau$ and $H \to \tau\tau$ simulated data. These samples can also be used to judge on the discriminant power of the neural network estimator.

The simulation of the $\gamma^* \to \tau^{\text{had}}\tau^{\text{had}}$ data sample is based on the Pythia8 [115] event generator. The $H \to \tau\tau$ data is simulated with the POWHEG BOX [116, 117] framework interfaced with the Pythia8 event generator. The $Z/\gamma^* \to \tau\tau$ data sample is simulated with the SHERPA [118] framework. Many different statistically independent data samples are combined into one data sample in order to have more statistics. An overview of the data samples is given in Table 6.1. The mass distribution of the $\gamma^* \to \tau^{\text{had}}\tau^{\text{had}}$ is depicted in Fig. 6.2. In order to correct the shape of the mass distribution, the events receive weights that flatten the mass distribution in bins of 0.1 GeV. The weighted mass distribution is shown in Fig 6.3. The mass range is limited from 60 − 220 GeV because the simulation started at 60 GeV and higher masses than 220 GeV are considered to be not relevant for the mass range of interest, namely 90 – 125 GeV. Note, that the energy resolution of the detectors depends on the energy of the particles and with higher invariant masses the average energies of the individual particles will be higher. Also the $E_{\text{T}}^{\text{miss}}$ resolution depends on the sum of all transverse energies. Thus, the resolution on possible event characterizing variables depends on the initial simulated invariant mass of the di-tau system. A broader mass range means more statistics and gives the hope that the DNN is able to learn more general features like Lorentz boosts, even when not understandable for a human interpreter.

6.1.1 Event Selection

A fully simulated data sample goes through two steps of cuts applied on event level before the data will be interpreted. The first step is the derivation step, where the data sample is reduced to the events interesting for the final state. And the second step further employs selection and is mainly used to define signal regions. For the DNN training, there is no signal region in that sense but the cuts are

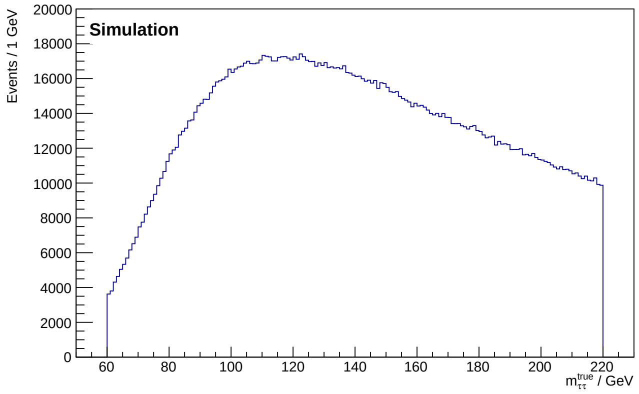


Figure 6.2: Histogram of the generated di-tau mass distribution $m_{\tau\tau}^{\text{true}}$ of the $\gamma^* \longrightarrow \tau^{\text{had}}\tau^{\text{had}}$ sample.

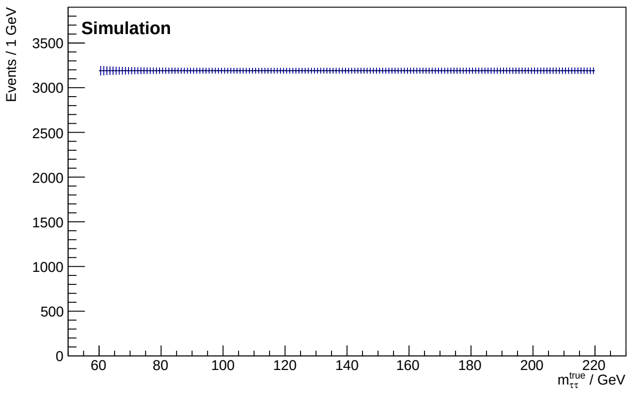


Figure 6.3: Histogram of the generated di-tau mass distribution $m_{\tau\tau}^{\text{true}}$ of the $\gamma^* \longrightarrow \tau^{\text{had}}\tau^{\text{had}}$ sample with event weights.

process	MC sample	N_{events}
$\gamma^* \longrightarrow \tau^{\text{had}}\tau^{\text{had}}$	`mc16_13TeV.425200.Pythia8EvtGen_A14NNPDF23LO_Gammatautau_MassWeight.deriv.DAOD_HIGG4D3.e5468_s3126_r10201_r10210_p3703`	2 142 022
$Z/\gamma^* \longrightarrow \tau\tau$	combined Sherpa samples, full list in Appendix C	259 654
$H \longrightarrow \tau\tau$	`mc16_13TeV.345123.PowhegPy8EG_NNLOPS_nnlo_30_ggH125_tautauh30h20.deriv.DAOD_HIGG4D3.e5814_s3126_r10201_p3627`	14 066

Table 6.1: Overview of the used Monte Carlo data samples. The number of events is after event cuts described in Section 6.1.1.

motivated to give the DNN a safer training environment.

Derivation Cuts

The derivation cuts are applied to all data samples. On this level the applied cuts aim for two hadronically decaying tau-leptons. Every jet is a (hadronically-decaying-) tau candidate if it fulfills the requirement:

- `(nTracks + nWideTracks ≥ 1) and (nTracks + nWideTracks ≤ 8)`,

where `nTracks` is the number of tracks in the core region $\Delta R \leq 0.2$ and `nWideTracks` is the number of tracks in the isolation region $0.2 < \Delta R \leq 0.4$.

It is further required that there are at least two tau candidates in the event with $p_{\mathrm{T}}^{\tau_1} \geq 33\,\mathrm{GeV}$ and $p_{\mathrm{T}}^{\tau_2} \geq 23\,\mathrm{GeV}$, respectively. One of the two tau candidates also needs to pass the loose working point in the BDT-based tau identification algorithm (cf. [73]).

Event Selection for DNN Training

These cuts are applied to the $\gamma^* \longrightarrow \tau^{\mathrm{had}}\tau^{\mathrm{had}}$ sample and are chosen to have a reasonable setup for the DNN training. The cuts are selected so that little events fail the criteria in order to have high training statistics. Thus, there is no specific trigger requirement, just any of the triggers need to fire.

A tau candidate needs to have

- `(nTracks = 1) or (nTracks = 3)`,

in order to exploit that a hadronically decaying tau-lepton has one or three charged pions/kaons as decay products. The sum of the absolute charge of the tracks has to be one: `AbsCharge = 1`.

All candidates have to fall into the η region $0 \leq |\eta| \leq 1.37$ or $1.52 \leq |\eta| \leq 2.5$. The forbidden region is in the transition region between the barrel and end-cap region.

There can be more than two tau candidates but only the two with the highest transverse momentum are considered. The last requirement on reconstruction level is that the two tau candidates have to originate from the same vertex.

Additional cuts on the truth/generated level are applied to define a safe working region. Both tau candidates need to have a truth-matched tau-lepton which decays hadronically. This was chosen to ensure that the DNN does not try to learn features from misidentified jets. The mass of the system of the two truth tau-leptons has to lie in the region $60\,\mathrm{GeV} \leq m_{\tau\tau}^{\mathrm{truth}} \leq 220\,\mathrm{GeV}$.

Event Selection for the Z and H Discrimination

The cuts on the $Z/\gamma^* \longrightarrow \tau\tau$ sample and $H \longrightarrow \tau\tau$ sample are chosen to be close to the selection in the $H \longrightarrow \tau\tau$ cross-section analysis (given in brackets if different) [12], only slightly less strict. As here, the prediction of the mass of the di-tau system is important for signal and background discrimination, the DNN estimator can be tested under conditions relevant for analyses.

The selection demands for

- exactly two tau candidates with
  - `(nTracks = 1) or (nTracks = 3)`

  - $0 \leq |\eta| \leq 1.37$ or $1.52 \leq |\eta| \leq 2.5$

- at least medium (tight) tau identification BDT working point
- $p_{\mathrm{T}}^{\tau_1} \geq 35\,\mathrm{GeV}(40\,\mathrm{GeV})$ and $p_{\mathrm{T}}^{\tau_2} \geq 25\,\mathrm{GeV}(30\,\mathrm{GeV})$
- the charge product of the tau candidates has to be -1
- both tau candidates must have a valid decay mode, i.e. not `other` (see Section 4.3.4)
- no light lepton
- $E_{\mathrm{T}}^{\mathrm{miss}} \geq 20\,\mathrm{GeV}$
- $\Delta\phi(\vec{E}_{\mathrm{T}}^{\mathrm{miss}}, (\vec{p}^{\tau_1} + \vec{p}^{\tau_2}))$ (no requirement)
- $0.8 < \Delta R(\vec{p}^{\tau_1}, \vec{p}^{\tau_2}) < 2.4(2.5)$
- $|\Delta\eta(\vec{p}^{\tau_1}, \vec{p}^{\tau_2})| < 1.5$

6.2 DNN Input Variables

The input variables for the DNN are inspired by the MMC input. The DNN is firstly trained with basic kinematic features in order to find stable training regions. Later, also additional information is taken into account and the impact on the DNN training is studied. The kinematic input directly follows from Eq. 4.10, namely the three momentum of the visible tau decay products and the two components of the missing transverse energy. The MMC further uses the number of tracks of each tau decay. This gives in total ten input parameters:

- $p_{\mathrm{T}}^{\tau_1}, \eta^{\tau_1}, \phi^{\tau_1}$, `nTracks`($\tau_1$)
- $p_{\mathrm{T}}^{\tau_2}, \eta^{\tau_2}, \phi^{\tau_2}$, `nTracks`($\tau_2$)
- $E_{\mathrm{T}}^{\mathrm{miss}}, \phi^{\mathrm{miss}}$.

Many variables are interesting for auxiliary information. The additional features that are tested in this work are:

- tau decay channel
- `MET significance`
- number of vertices
- number of jets (the MMC differentiates between 0 and > 0 jets [16]) .

The tau decay channel can give additional information because there the intermediate resonance is encoded. The intermediate resonances, e.g. π, ρ or a_1, have different masses, and therefore the angle between the visible decay products and the neutrino should be different. What could also have an effect is the different tau momentum resolution in the different decay channels [119].

The `MET significance` is the estimated error on the missing transverse energy divided by the magnitude of the missing transverse energy. This could be a useful input feature because the error on the missing transverse energy gives indicates of how precisely the neutrino momenta can be estimated.

The number of vertices and the number of jets can both have an effect on the resolution of the missing transverse energy.

It is important to notice that the input of a DNN should be uncorrelated. As stated in Section 2 higher level input features are not expected to lead to better DNN performance and are not studied.

6.2.1 Additional Input Variables for Future Work

There are more event and object related features which are interesting to study. As the main focus of this work lies in finding a safe and stable setup for the DNN training, not every detail is studied. In principle, feature selection and their representation can have a major impact on improving the DNN performance. This is typically desirable when it comes to gaining the last few percent (or permille) in accuracy. Additional information could be:

- impact parameters d_0 and z_0
- jet kinematics
- missing transverse energy with respect to primary vertex

The impact parameters d_0 and z_0 measure the distance of the secondary vertex to the primary vertex. This could be helpful in tau decays with three tracks because then the exact position of the tau decay would give further constraints. However, the resolution on the secondary vertex is a very limiting factor.

The jet kinematics can be more helpful than just the number of jets since a mismeasurement of the missing transverse energy could be correlated with the direction of the jets and their momentum.

As the missing transverse energy is measured event-wise it may be helpful to try to give an estimate of the missing transverse energy with respect to the primary vertex. However, this cannot take final state radiation into account.

Beside additional input features, the representation of the input feature can also have an impact on the DNN performance. In [120, 121] measurements of angles were made in the rest-frame of the visible tau decay. Inspired by this, an idea could be to give the momenta in the rest-frame of the visible resonance. As this work does not aim to fully optimize the setup, impacts on different input representations is suggested for further studies.

6.2.2 Standardization of Input Variables

The input variables were standardized, i.e. subtracted by their mean and divided by their standard deviation. The distribution of the different input features are depicted in Appendix D. Note that the standardization transforms the features to have zero mean and unit variance but does not change the shape of the distributions. For the standardization the SCIKIT-LEARN package was used [103].

The decay channel is not a numerical number, and therefore an one-hot encoding is suggested to represent them. This maps each decay mode to a unit vector of standard base. This is a simple way to find an orthogonal representation. The decay channels are:

- $h^{\pm} \rightarrow (1, 0, 0, 0, 0, 0)^{\mathrm{T}}$
- $h^{\pm}\pi^{0} \rightarrow (0, 1, 0, 0, 0, 0)^{\mathrm{T}}$
- $h^{\pm} \geq 2\pi^{0} \rightarrow (0, 0, 1, 0, 0, 0)^{\mathrm{T}}$
- $3h^{\pm} \rightarrow (0, 0, 0, 1, 0, 0)^{\mathrm{T}}$
- $3h^{\pm} \geq 1\pi^{0} \rightarrow (0, 0, 0, 0, 1, 0)^{\mathrm{T}}$
- `other` $\rightarrow (0, 0, 0, 0, 0, 1)^{\mathrm{T}}$

6.2.3 Suggestion for Future Work

The shapes of the variable distribution are not Gaussian (see Appendix D). It may be worthwhile to study the impact of outliers on the DNN training. The robustness of DNNs is a topic of recent study, i.e. [122]. A simple way to test the robustness could be a check whether input with outliers produces particularly inaccurate results or whether the performance improves when data with outliers is not used for training.

6.3 Data Sample Splitting

In order to evaluate the training process, the $\gamma^{*} \rightarrow \tau^{\mathrm{had}}\tau^{\mathrm{had}}$ data sample is split into a training (train), a cross-validation (xval) and a test sample. The cross-validation sample is used to evaluate the training during the development process and the test sample is used to estimate the performance after a model is selected. The splitting of the data sample is orientated on the mass distribution of the $\gamma^{*} \rightarrow \tau^{\mathrm{had}}\tau^{\mathrm{had}}$ sample. As explained it is desirable to have a data sample which mass distribution is flat in mass.

The data sample was first sorted into bins of 0.1 GeV. Then, from every bin the same amount of events were used in order to define a flat region (319/GeV, i.e. the lowest occupancy of all bins). From this flat region 10% (32) of the events per bin were used for the training and the cross-validation sample (in total 51 200 events each). The rest of the events were then used to define a flat training region (train flat) with events that are used for training but with a flat mass distribution (408 000 events). The events of the region flat training and the formerly not used were put together, defining the training sample. This sample is then weighted to have a flat mass distribution. The training sample is used for the DNN training while the flat training subsample is used for the evaluation plots (see Chapter 7). An overview of the sample splitting is given in Tab. 6.2.

A simple data sample splitting is used because k-fold cross-validation is computationally more expensive and the loss in the training sample size is manageable while the cross-validation sample and test sample are big enough to give a good estimate of the DNN performance. The standardization transformation, i.e. the mean and the standard deviation of the input variables, is calculated on the training sample and then performed on all samples.

6.4 DNN Architecture

The DNN training is performed using the Keras framework [123] with TensorFlow [124] as backend. Different setups are used in order to make consistency checks and to study the impact of some selected

sample	subsample	events
train		2 039 622
	train flat	408 000
xval		51 200
test		51 200

Table 6.2: Overview of the splitting of the $\gamma^* \to \tau^{\mathrm{had}}\tau^{\mathrm{had}}$ sample.

hyperparameter	test range
# hidden layers	1,2,3,4,5
# nodes / layer	1,2,4,8,16,32
optimizer	NAM, Adam, Nadam

Table 6.3: Overview of the hyperparameters and how they are varied in order to study their impact on the DNN training.

hyperparameters. The tested architectures are depicted in Tab. 6.3. The number of hidden layers and the number of nodes per hidden layer is varied. For all setups no regularization method is used, and the squared error is used as the loss function. The activation function and the weight initialization scheme is chosen according to the SNN setup (see Section 5.1.5). The training of the different setups will be performed with three different optimization algorithms with mini-batch size $s = 32$, the Nesterov accelerated momentum algorithm (NAM), the Adam and the Nadam algorithm. This is repeated for two different random initializations, i.e. different random seeds so that the results are reproducible. All these studies aim to check how stable the DNN training is and if good results can be achieved.

The hyperparameters of the optimization algorithms are chosen according to the standard values of [123]:

- NAM: $\alpha = 0.01, \beta = 0.9$
- Adam: $\alpha = 0.001, \beta_1 = 0.9, \beta_2 = 0.999$
- Nadam: $\alpha = 0.002, \beta_1 = 0.9, \beta_2 = 0.999$

The training time is fixed to 200 epochs. If the training loss does not decrease within 15 epochs, the value α is decreased by a factor of 2. Further, the gradient is clipped to 2, i.e. if the norm of the calculated gradient is larger than 2, it is scaled to 2. This is used for numerical stability in regions with high gradients [125]. The value 2 is chosen randomly and it is another hyperparameter which has to be optimized or tested for further optimization.

CHAPTER 7

DNN Evaluation Metrics

This chapter aims to introduce and motivate the evaluation metric plots which are used to measure the DNN performance. They are exemplarily shown for the MMC predictions for the $\gamma^* \to \tau^{\text{had}}\tau^{\text{had}}$ sample which will be used for comparison with the DNN predictions. As stated before, for the development of a DNN it is important to know how precise the DNN estimation can become. Unfortunately, there is no recipe for estimating the best possible performance. In order to give a very upper limit, a DNN will be trained with truth information as input and the evaluation metric plots will be used for this particular example. This is of course not a valid estimation for the best possible, achievable precision of a DNN trained with detector information, as the limiting factor should arise from the DNN architecture itself and not from the data. However this validates if a DNN can learn something in the safe environment of no detector resolution effects.

7.1 Evaluation Metrics

The DNN performance at training time is measured by the loss function. It should be decreasing with the number of epochs and for a non-high-variance case, the loss function value evaluated on the cross-validation data should be near the training data value. While the learning curve is good for comparing different DNN setups and getting a flavor of the variance, it has also a few down sides. For example it can not be concluded if the predictions are equally good in every mass interval. In order to compare the DNN with the MMC, a few plots are shown which can evaluate the performance and also check the consistency.

Firstly, the distribution of the predicted di-tau system mass is considered. This contains a lot of information. If the estimator has a strong bias, the distribution could peak at some mass value. Moreover, the distribution should look continuous. If this is not the case this would need further investigation. The distribution does also give insights on the resolution dependency on the truth mass. If the resolution would be considered Gaussian with a constant standard deviation, then the distribution of the predicted masses would be given by the convolution of the Gaussian with the uniform distribution (depicted in Fig. 7.1). A dependency of the standard deviation of the truth mass would lead to other distributions. Exemplary for a $\sigma \propto \sqrt{m_{\tau\tau}^{\text{truth}}}$ and a $\sigma \propto m_{\tau\tau}^{\text{truth}}$ dependency the distributions are simulated and depicted in Figs. 7.2 and 7.3, respectively. The di-tau system masses predicted by the MMC are depicted in Fig. 7.4. Comparing the distribution in Fig. 7.4 with the

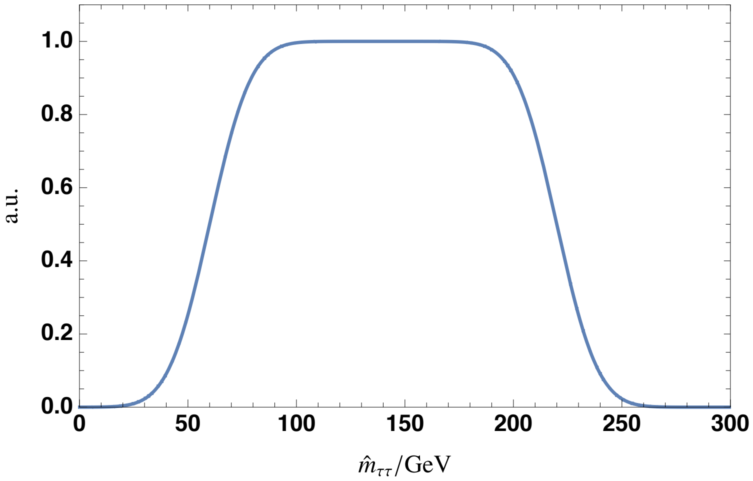


Figure 7.1: Convolution of a Gaussian (with $\sigma = 15\,\text{GeV}$) with a uniform distribution $\mathcal{U}(60\,\text{GeV}, 220\,\text{GeV})$. This gives the distribution of the expected predicted mass $\hat{m}_{\tau\tau}$ if a Gaussian-shaped resolution of $\sigma = 15\,\text{GeV}$ is assumed.

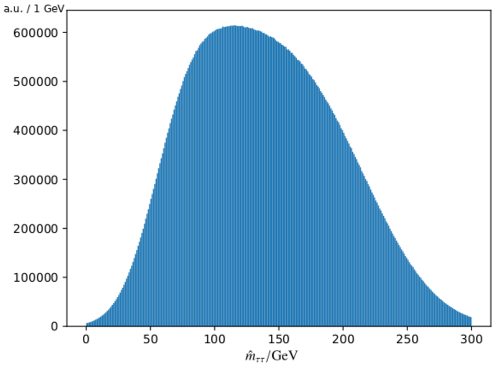


Figure 7.2: Histogram of a simulated distribution of the expected predicted mass $\hat{m}_{\tau\tau}$ if a Gaussian-shaped resolution of $\sigma = 3\,\text{GeV}\sqrt{m_{\tau\tau}^{\text{truth}}/\text{GeV}}$ is assumed.

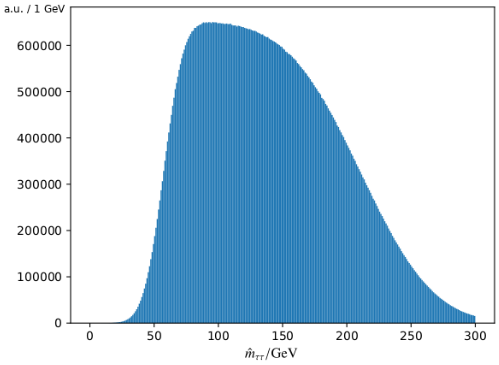


Figure 7.3: Histogram of a simulated distribution of the expected predicted mass $\hat{m}_{\tau\tau}$ if a Gaussian-shaped resolution of $\sigma = 0.2 \cdot m_{\tau\tau}^{\text{truth}}$ is assumed.

theoretically expected distributions Figs. 7.1-7.3, it seems as if the resolution depends approximately linearly on the truth di-tau system mass because of the similarity with Fig. 7.3 but further evaluation plots are needed. However, the Gaussian distribution is very idealized and the tails are typically non Gaussian.

As from the distribution of the predicted mass the deviation from the truth mass can not be concluded, additional information is needed. Therefore the predicted mass is plotted against the truth mass in a two-dimensional scatter plot, depicted in Fig. 7.5. There it can be seen how much the predicted mass correlates with the truth mass, if there are any shifts and also what resolution the mass estimator has. The correlation factor is an additional real value number which can be used to quickly compare different estimators, besides comparing their loss values. In principle, this plot contains all

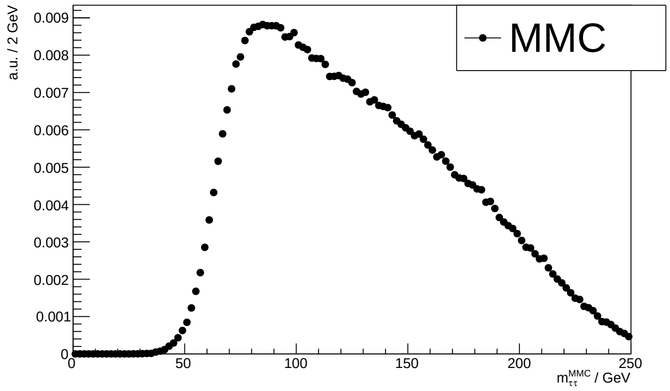


Figure 7.4: Histogram of the predicted di-tau mass distribution of the $\gamma^* \to \tau^{\text{had}}\tau^{\text{had}}$ sample by the MMC, in the flat training subsample.

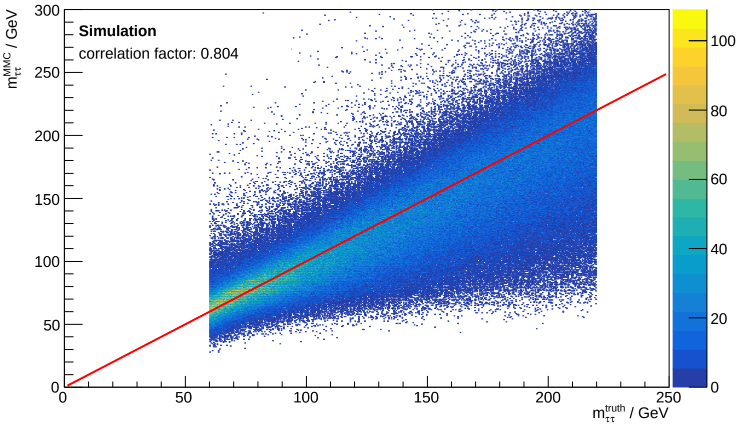


Figure 7.5: Scatter plot of the predicted di-tau mass distribution of the $\gamma^* \to \tau^{\text{had}}\tau^{\text{had}}$ sample by the MMC against the truth mass $m_{\tau\tau}^{\text{truth}}$, in the flat training subsample. The correlation factor is 0.804. The red line describes an ideal linear response $m_{\tau\tau}^{\text{est}} = m_{\tau\tau}^{\text{truth}}$.

the information needed to evaluate the DNN performance. But as the resolution is very hard to read off by eye, an additional auxiliary plot is made which is easier to interpret. As The underlying truth mass distributions are always flat because they cannot be read off by eye an otherwise the plot would be hard to interpret. This is why the flat training subsample is needed. Additionally, a linear response $m_{\tau\tau}^{\text{est}} = m_{\tau\tau}^{\text{truth}}$ is plotted to the scatter plot in red.

A possible way to visualize a possible relative shift from the estimator to the truth mass is plotting the mean value of $(m_{\tau\tau}^{\text{estimated}}/m_{\tau\tau}^{\text{truth}}) - 1$ in bins of the truth mass. This can also show trends or hint at a high bias. The resolution is added to the plot as error bars of the central values. The error bars enclose the range corresponding to one standard deviation in both directions, the $15.865 - 84.135$ percentile range. For the MMC prediction this is depicted in Fig. 7.6.

To further visualize the size of the width, the half of the widths corresponding to 1σ and 2σ

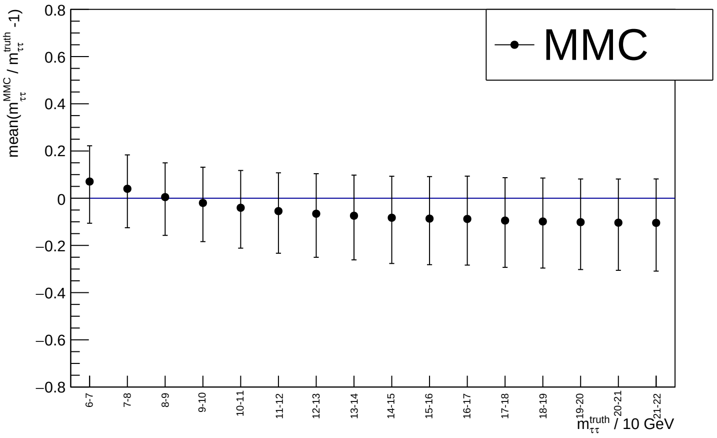


Figure 7.6: Depicted is the mean of the relative deviation $m_{\tau\tau}^{\mathrm{MMC}}/m_{\tau\tau}^{\mathrm{truth}} - 1$ of the $\gamma^* \rightarrow \tau^{\mathrm{had}}\tau^{\mathrm{had}}$ flat training subsample in 10 GeV bins of the truth mass. The error bars enclose the percentile range corresponding to 1σ.

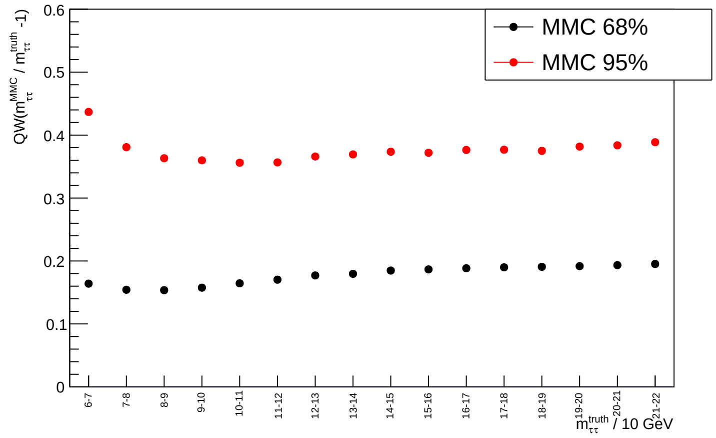


Figure 7.7: Depicted are the half of the quantile widths of the relative deviation $m_{\tau\tau}^{\mathrm{MMC}}/m_{\tau\tau}^{\mathrm{truth}} - 1$ of the $\gamma^* \rightarrow \tau^{\mathrm{had}}\tau^{\mathrm{had}}$ flat training subsample in 10 GeV bins of the truth mass. The black dot corresponds to a 1σ range, the red dot to a 2σ range.

(2.275 – 97.725 percentile range) are also plotted in bins of the truth mass. This is easier to read off. This can also give an estimate how Gaussian a distribution is. For a Gaussian distribution, the range corresponding to 2σ should be exactly twice as wide as the range corresponding to 1σ. For the MMC this is depicted in Fig. 7.7. It can be seen that the relative resolution is rather flat which means that the uncertainty of the estimated mass is roughly linear with respect to the truth mass.

7.2 DNN with Truth Level Information Input

The training of a DNN with truth level information is mainly a check how much a DNN with a given architecture can learn from the low level information. As input variables the basic variables described in Section 6.2 are chosen, without auxiliary information (10 input variables in total). The architecture tested will be:

- SNN setup
- 4 hidden layers with 16 nodes each
- Adam optimizer ($\alpha = 0.001, \beta_1 = 0.9, \beta_2 = 0.999$)
- no regularization
- 200 epochs
- reduce α by a factor 0.5 if loss does not decrease for 15 epochs
- squared error loss
- truth mass as target
- mini-batch size $s = 32$

The results are summarized in Figs. 7.8-7.14. A first evaluation is to look at the learning curve. From the loss function $\mathcal{J}(\Theta)$, the square root $\mathcal{J}'(\Theta)$ is calculated:

$$\mathcal{J}'(\Theta) = \sqrt{\mathcal{J}(\Theta)} = \sqrt{\left(\sum_i \frac{1}{w_i}\left(\frac{m_{\tau\tau}^{\text{pred.}} - m_{\tau\tau}^{\text{truth}}}{\text{GeV}}\right)^2\right)}, \tag{7.1}$$

where w_i are the event weights used to weight the distribution flat in mass. This RMS value is easier to interpret. The value $\mathcal{J}'(\Theta)$ is evaluated at the end of each epoch and plotted against the number of trained epochs in Fig. 7.8. It can be concluded that there is only little variance because the loss value for the training and cross-validation sample differ only by a small amount. It gives a consistent result, i.e. the training point with the lowest loss also gives the lowest cross-validation error. The steps in the training learning curve emerge because of a decrease of α.

The mass distribution, depicted in Fig. 7.9, follows neither of the theoretically motivated distributions from Figs. 7.1, 7.2 or 7.3. It rather looks like the distribution of the predicted mass tries to emulate a flat mass distribution. It is striking, that the predicted masses are very unlikely to be outside the training mass region of 60 GeV – 220 GeV.

The distribution of the estimated masses against the truth masses give more insights. For the flat training subsample and the cross-validation sample they are depicted in Figs. 7.10 and 7.11. Firstly, the correlation factor of the two histograms are very high 0.905 (train flat) and 0.903 (xval) which means that the DNN estimates the mass very accurately. The small deviation further underlines that there is only little variance. But the predicted masses in the edge regions are very asymmetric. The DNN prefers to predict masses inside the known mass region. There is also a shift in the most probable estimated mass depending on the truth mass. This trend can be seen in the mean of the relative deviation, depicted in Fig 7.12. This effect is believed to occur because of the squared error loss function which does not correct for edge effects. In order to compensate the edge effects, the loss function with boundary correction derived in Eq. 5.9 is tested. Unfortunately, this new loss function cannot be tested to full detail. As a best guess, a linear dependency of the standard deviation

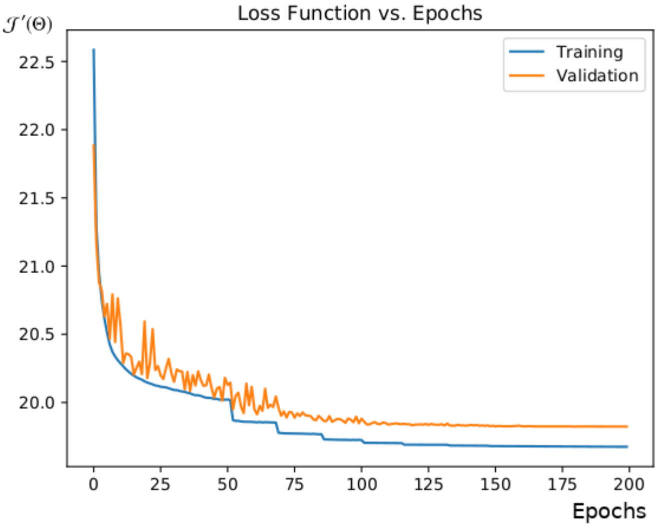


Figure 7.8: Square root of the loss against the number of training epochs.

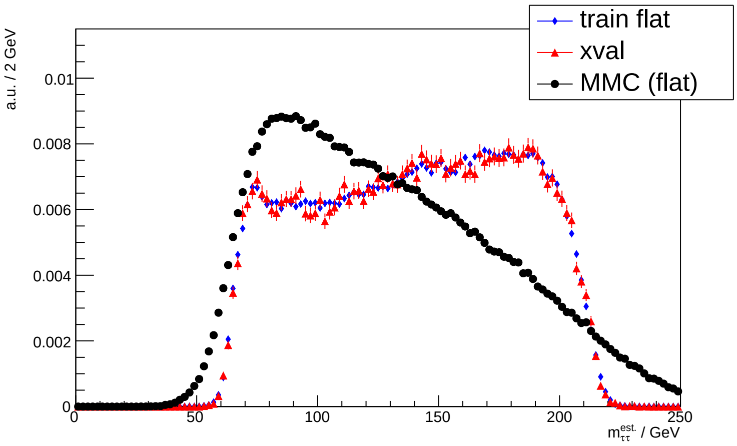


Figure 7.9: Distribution of the estimated masses for the flat training subsample (blue), the cross-validation sample (red) and the MMC predictions for the flat training subsample.

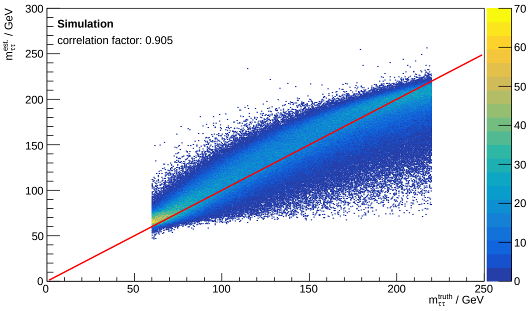


Figure 7.10: Scatter plot of the DNN predicted di-tau mass distribution against the truth mass $m_{\tau\tau}^{\text{truth}}$ in the flat training subsample. The correlation factor is 0.905.

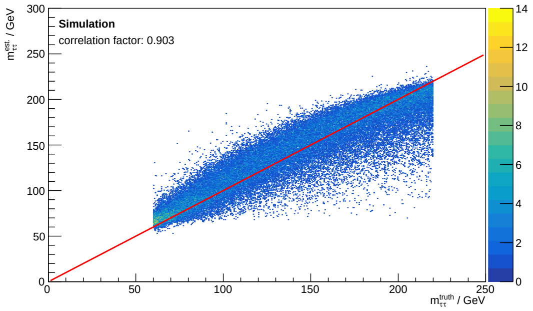


Figure 7.11: Scatter plot of the DNN predicted di-tau mass distribution against the truth mass $m_{\tau\tau}^{\text{truth}}$ in the cross-validation sample. The correlation factor is 0.903.

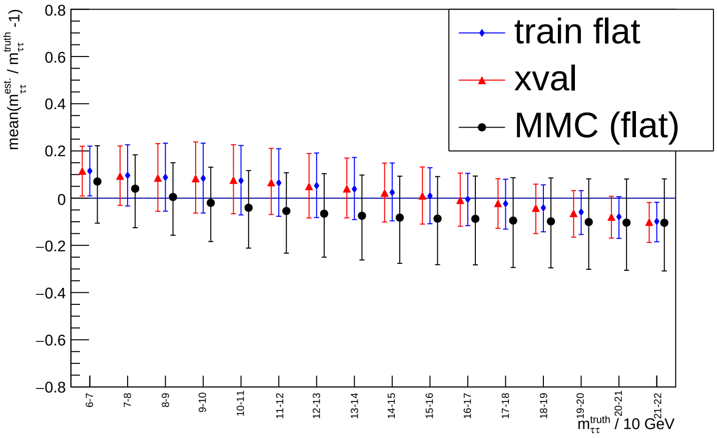


Figure 7.12: Mean of the relative deviation $m_{\tau\tau}^{\text{est}}/m_{\tau\tau}^{\text{truth}} - 1$ compared for the flat training subsample (blue), the cross-validation sample (red) and the MMC predictions for the flat training subsample.

$\sigma = c \cdot m_{\tau\tau}^{\text{truth}}$ is assumed (based on the estimated MMC dependency) and the coefficient $c = 0.12$ is a very rough estimate of the 68% quantile width from Figs. 7.13 and 7.14.

7.2.1 Boundary Corrected Loss Function

The boundary corrected loss function can be a promising tool to counter the effect of learning global features from the training sample, i.e. the training target range. However, it is not commonly used and therefore not fully tested. For the di-tau system mass prediction it has to be tested whether the assumption of a normal distribution with $\sigma = c \cdot m_{\tau\tau}^{\text{truth}}$ is valid or if this runs into problems. For a fair comparison, the same hyperparameters are used like in Section 7.2, exchanging only the loss function. The results are summarized in Figs. 7.15-7.22.

Firstly, the learning, depicted in Figs. 7.15 and 7.16, is not as stable as in the case of the squared error loss. There are small jumps in the learning curve of the training sample and even bigger jumps

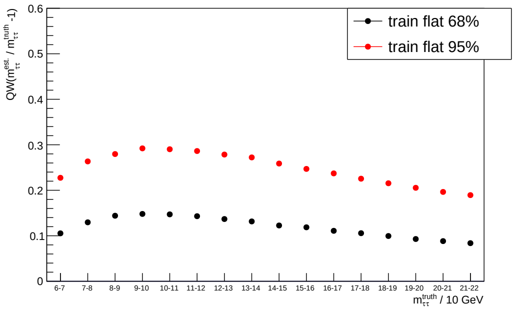


Figure 7.13: Depicted are the half of the quantile widths of the relative deviation $m^{\text{est}}_{\tau\tau}/m^{\text{truth}}_{\tau\tau} - 1$ of the flat training subsample in 10 GeV bins of the truth mass. The black dot corresponds to a 1σ range, the red dot to a 2σ range.

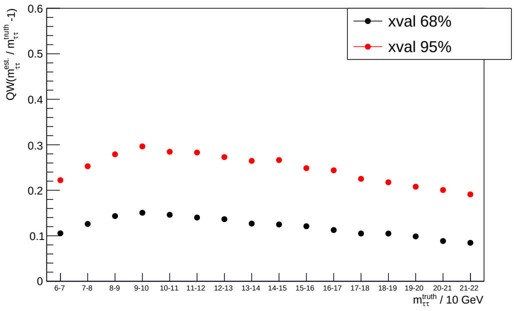


Figure 7.14: Depicted are the half of the quantile widths of the relative deviation $m^{\text{est}}_{\tau\tau}/m^{\text{truth}}_{\tau\tau} - 1$ of the cross-validation sample in 10 GeV bins of the truth mass. The black dot corresponds to a 1σ range, the red dot to a 2σ range.

in the cross-validation sample. Evaluating the RMS

$$\sqrt{\left(\sum_i \frac{1}{w_i}\left(\frac{m^{\text{pred.}}_{\tau\tau} - m^{\text{truth}}_{\tau\tau}}{\text{GeV}}\right)^2\right)} \tag{7.2}$$

after each epoch shows, that the RMS in the cross-validation is very noisy and it is hard to see whether there is any improvement at all. The RMS is not expected to be monotonically decreasing as it is not optimized. However, further investigation is needed and possibly this result of an unstable learning curve has to be confirmed by other experiments. There are a few things that may cause these behaviors. Maybe the initial value of the learning rate is too high or maybe the estimate of σ is too inaccurate for the problem. Possible further investigation could be to test different optimizers and learning rates, train the DNN with pretrained weights (e.g. by a DNN trained with the squared error loss function) or estimate σ differently, maybe on an epoch-wise basis.

Despite the problems with the optimization, the other evaluation plots look better. The DNN now also predicts masses outside the training target range, see Fig. 7.17. This comes with a lower accuracy of the predictions, i.e. a lower correlation factor, see Figs. 7.18 and 7.19. This has to be considered when looking at correlation factors predicted by DNNs trained with the squared error, that these DNNs probably overestimate the correlation due to edge effects. The correlation factors of the flat training subsample (0.880) and the cross-validation sample (0.877) differ only a little suggesting that the DNN does not overtrain. It is striking, that the relative resolution gets worse for higher truth masses. If this is an artifact from the loss function itself or a result of a suboptimal training has to be investigated in further research. However, the relative resolution is roughly constant in the region of interest, see Figs. 7.20 and 7.21, and there is no obvious trend in the shifts of the relative deviation against the truth mass, see Fig. 7.22.

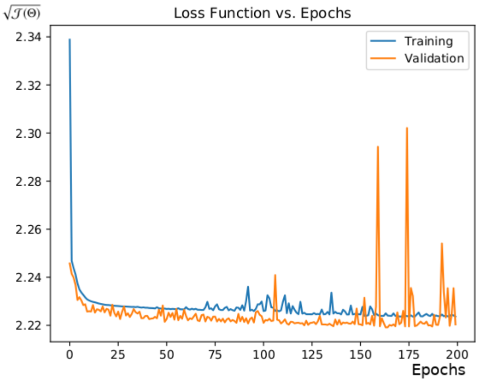


Figure 7.15: Square root of the boundary corrected loss against the number of training epochs.

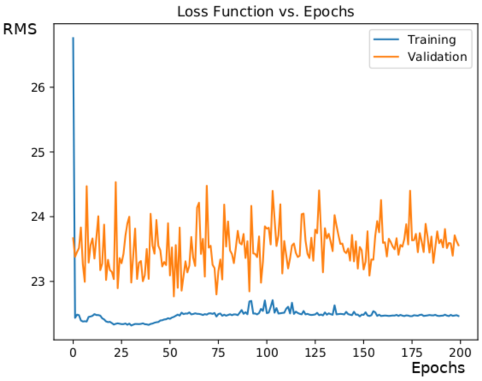


Figure 7.16: RMS of the predicted masses against the number of training epochs.

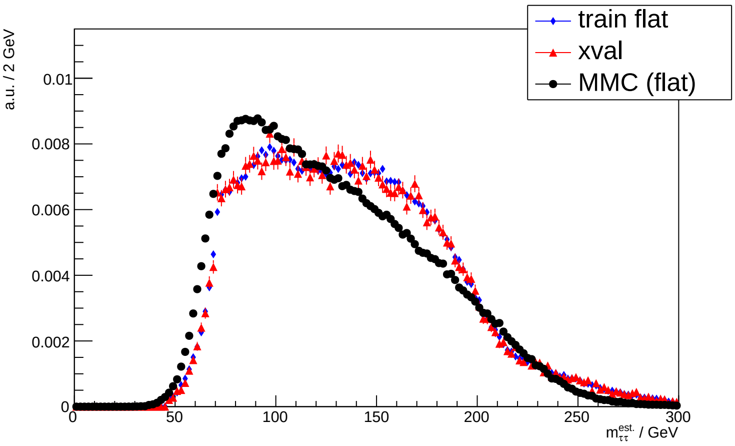


Figure 7.17: Distribution of the estimated masses for the flat training subsample (blue), the cross-validation sample (red) and the MMC predictions for the flat training subsample.

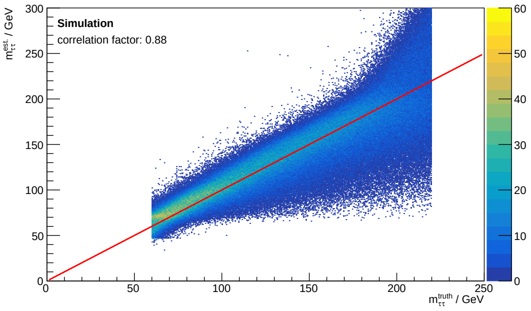


Figure 7.18: Scatter plot of the DNN predicted di-tau mass distribution against the truth mass $m_{\tau\tau}^{\text{truth}}$ in the flat training subsample. The correlation factor is 0.880.

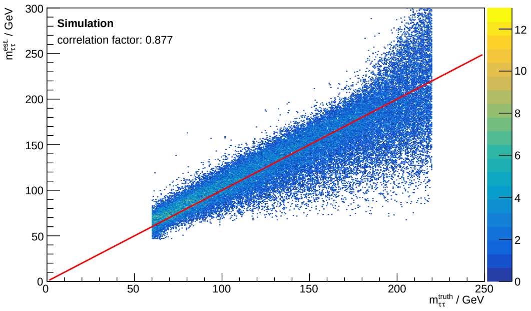


Figure 7.19: Scatter plot of the DNN predicted di-tau mass distribution against the truth mass $m_{\tau\tau}^{\text{truth}}$ in the cross-validation sample. The correlation factor is 0.877.

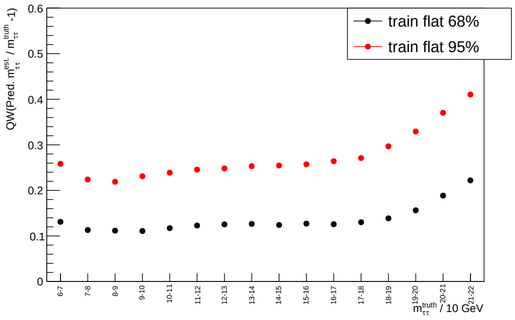


Figure 7.20: Depicted are the half of the quantile widths of the relative deviation $m_{\tau\tau}^{\text{est}}/m_{\tau\tau}^{\text{truth}} - 1$ of the flat training subsample in 10 GeV bins of the truth mass. The black dot corresponds to a 1σ range, the red dot to a 2σ range.

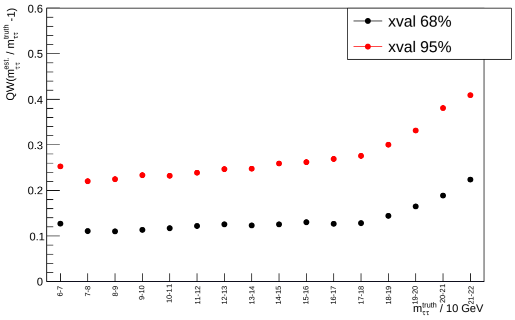


Figure 7.21: Depicted are the half of the quantile widths of the relative deviation $m_{\tau\tau}^{\text{est}}/m_{\tau\tau}^{\text{truth}} - 1$ of the cross-validation sample in 10 GeV bins of the truth mass. The black dot corresponds to a 1σ range, the red dot to a 2σ range.

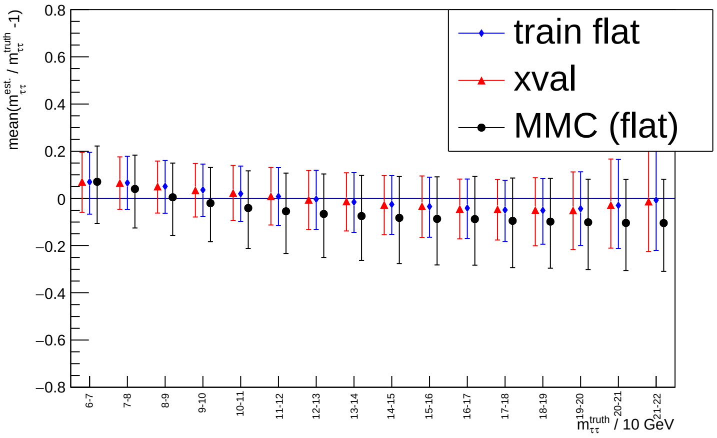


Figure 7.22: Mean of the relative deviation $m_{\tau\tau}^{\text{est}}/m_{\tau\tau}^{\text{truth}} - 1$ compared for the flat training subsample (blue), the cross-validation sample (red) and the MMC predictions for the flat training subsample.

CHAPTER 8

Results

In Section 7.2 it was shown that a DNN can approximate the mass of a di-tau system with low level input features at truth level. Now, also detector effects are included and the input variables are given at reconstruction level. The DNN performance is compared to the MMC performance. For the comparison it is important to remember that the MMC is trained on $Z/\gamma^* \rightarrow \tau\tau$ data while the DNN on the explained $\gamma^* \rightarrow \tau^{\text{had}}\tau^{\text{had}}$ data sample. Therefore, the comparison is difficult depending on which data sample the performance is tested, since each estimator is expected to perform better on the data sample it is trained on. The idea is to train the DNN on the $\gamma^* \rightarrow \tau^{\text{had}}\tau^{\text{had}}$ data sample and then test it in the task of separating $Z/\gamma^* \rightarrow \tau\tau$ and $H \rightarrow \tau\tau$ events.

8.1 Impact of the DNN Architecture

In order to study the impact of the architecture on the DNN performance, some hyperparameters are varied as described in Section 6.4. For the comparison, not all evaluation metric plots are shown but rather the correlation factors between the predicted mass and the truth mass on the cross-validation samples are compared. This is of course not sufficient for a detailed analysis. A differentiated analysis is exemplarily shown when necessary.

8.1.1 Impact of DNN Depth

For a fixed number of nodes $N = 16$, the number of hidden layers L is varied in order to study the impact of the depth of the DNN on the performance. The DNN is trained with the Adam optimization algorithm. The impact of this on the correlation factor is depicted in Fig. 8.1. It can be seen that the correlation factor flattens out for $L \geq 3$ and that even for $L = 1$ the correlation factor (0.828) is higher than the correlation factor for the MMC (0.804). This would suggest that the DNN performs better on the $\gamma^* \rightarrow \tau^{\text{had}}\tau^{\text{had}}$ sample than the MMC. Unfortunately, a few effects occur which dampen this success. To quickly sketch the problem, the DNN suffers from edge effects but the mass distribution is also not smooth in the intermediate mass range for $L = 5$. The mass distribution for $L = 5$ is shown in Fig. 8.2. While theoretically a DNN should become more precise with more hidden layers, it might be, that the training is too difficult and the weights got stuck in a plateau region. In a plateau region the loss does not decrease or only very slowly and is very hard to distinguish from a (local) optimum. For $L < 5$ the mass distributions are shown in Appendix E.1.

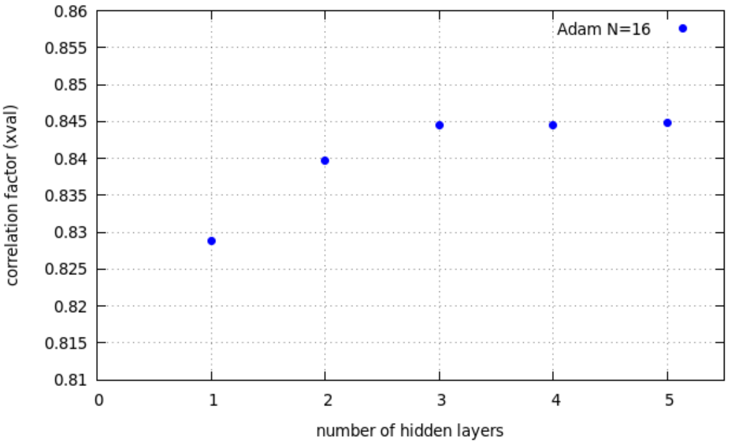


Figure 8.1: Correlation factor of the predicted mass against truth mass scatter plot on the cross-validation sample for different number of hidden layers. All hidden layers have $N = 16$ nodes per layer and the DNNs are trained with the Adam optimizer.

The edge effects are still present, i.e. the DNN only very rarely predicts masses outside the training mass range of 60 GeV – 220 GeV, which also results in a shift in the relative deviation, shown in Fig. 8.3. The relative resolution also depends on the truth mass, see Fig. 8.4. This is caused by the squared error loss, because the squared error loss only takes the distance of the predicted output and truth output into account. Thus, the DNN is optimized for a minimal distance independent of the truth output, and thus the error is held constant. In order to achieve a constant relative resolution the loss function has to be changed to:

$$\mathcal{L} = \frac{\left(o^{(i)} - y^{(i)}\right)^2}{y^{(i)}} \,. \tag{8.1}$$

The good resolution for high truth masses might further be caused by the limited prediction range of the DNN estimator, see Fig. 8.5.

To conclude, there is a general trend that the DNN can achieve a higher correlation between predicted mass and truth mass with more hidden layers. This effect flattens out for more hidden layers ($L \geq 3$) so that at some point the correlation does not increase with more layers anymore. This is expected, however increasing the number of hidden layers may also run into difficulties with the DNN training. Furthermore, the edge effects play a major role in the DNN predictions and the global feature of the training mass range has a higher impact on the DNN trained with reconstruction level information than on the DNN trained with truth level information.

8.1.2 Impact of DNN Broadness

In order to study the effect of the DNN Broadness, i.e. how many nodes are used per hidden layer, the number of hidden layers $L = 4$ is held constant and the number of nodes per layer is varied. The DNNs are trained with the Adam optimization algorithm. The results are depicted in Fig. 8.6. It can be seen, that the correlation factor flattens out, starting at $N = 8$. The edge effects are still present and not presented again. Despite the edge effects, studies like these can help to get a feeling in which range of

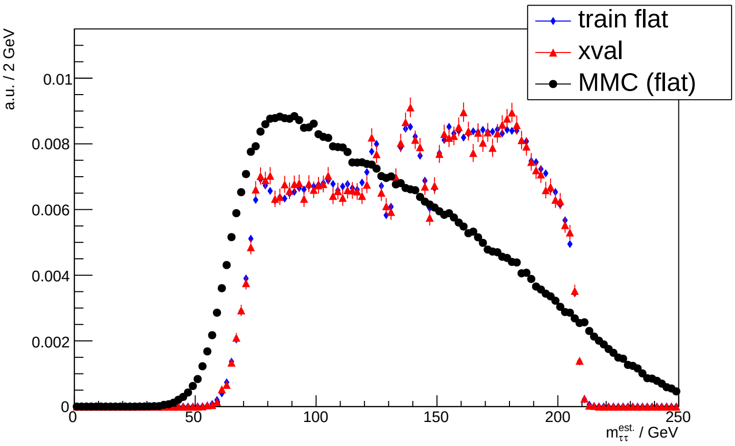


Figure 8.2: Mass distribution of the DNN predicted masses for the flat training subsample (blue), the cross-validation sample (red) and the MMC predicted masses (black). The DNN has $L = 5$ hidden layers with $N = 16$ nodes and is trained with the Adam optimization algorithm.

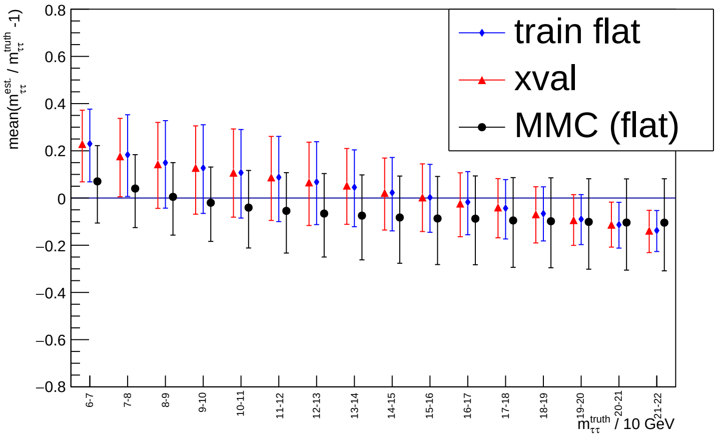


Figure 8.3: Mean of the relative deviation of the DNN predictions for the flat training subsample (blue), the cross-validation sample (red) and the MMC predictions (black). The DNN has $L = 5$ hidden layers with $N = 16$ nodes and is trained with the Adam optimization algorithm.

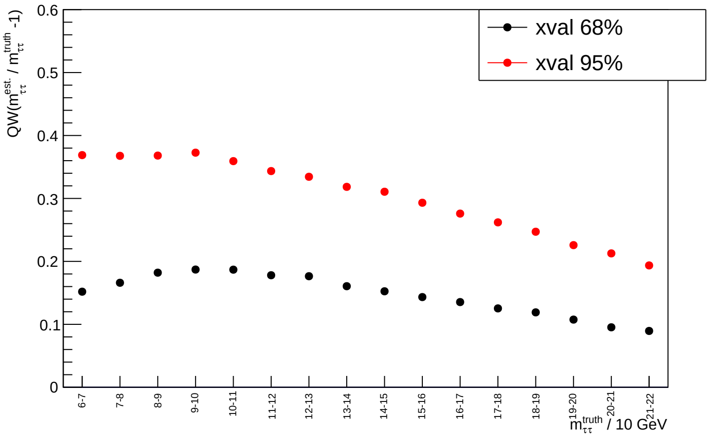


Figure 8.4: Half of the quantile widths corresponding to 1σ (black) and 2σ (red) of the relative deviation $m_{\tau\tau}^{\text{est}}/m_{\tau\tau}^{\text{truth}} - 1$ of the cross-validation sample in 10 GeV bins of the truth mass. The DNN has $L = 5$ hidden layers with $N = 16$ nodes and is trained with the Adam optimization algorithm.

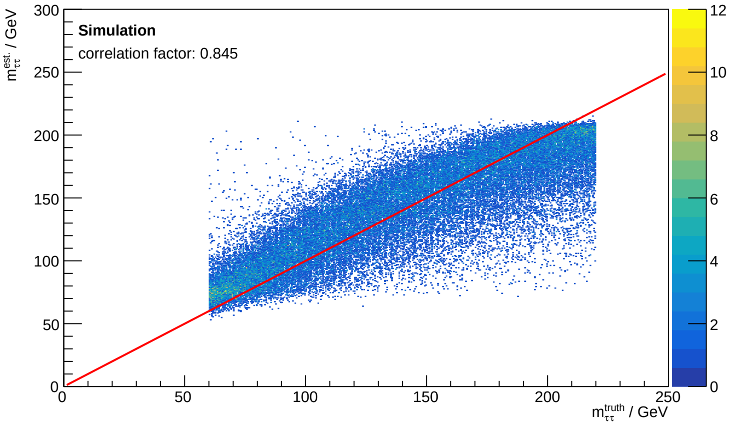


Figure 8.5: Scatter plot of the DNN predicted di-tau mass distribution against the truth mass $m_{\tau\tau}^{\text{truth}}$ in the cross-validation sample. The correlation factor is 0.845. The DNN has $L = 5$ hidden layers with $N = 16$ nodes and is trained with the Adam optimization algorithm.

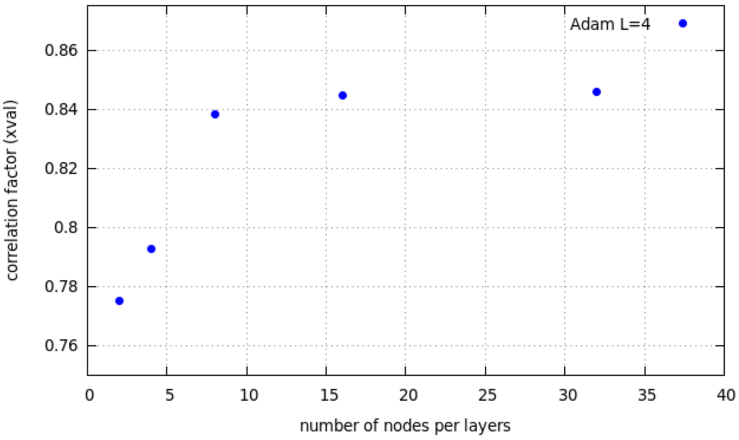


Figure 8.6: Correlation factor of the predicted mass against truth mass scatter plot on the cross-validation sample for different number of nodes per layer. The DNNs have 4 hidden layers and are trained with the Adam optimizer.

the hyperparameters good results can be achieved. This could then lower the cost of a hyperparameter scan, if some regions are already excluded. For the tested architectures the DNN training did not suffer from large overfitting. For higher numbers of nodes per hidden layer regularization techniques should be used. A possible way how to determine the transition point could be determining the point where the correlation factor between the predicted mass and the truth mass on the cross-validation sample starts to drop down with increasing number of nodes per hidden layer. Additionally, the learning curve should be considered to keep track of high bias, i.e. overtraining.

8.1.3 Impact of Different Optimization Algorithms

In order to test the impact of different optimization algorithms on the DNN performance, different DNN configurations are set up and then trained with the NAM, the Adam and the Nadam optimization algorithm. The different tested algorithms have different theoretical convergence behavior and could lead to different results. The tested configurations, denoted by (L,N), are (1,2), (2,4), (3,8), (4,16) and (5,32). There, the weights are initialized with the same random seeds in order to have the same starting point for every optimization algorithm. The results are depicted in Fig. 8.7. It can be seen, that in general the Nadam optimization algorithms leads to the best results. It is slightly better than the Adam optimization algorithm. It is noticeable, that the NAM optimization algorithm leads to the best result for the configuration (1,2) which is even to better than the result of configuration (2,4) trained with the NAM algorithm. This leads to the question, what could cause this observed effect?

A possible explanation could be, that the DNN gets stuck in a local optima or a plateau region for some training cases. The Adam and Nadam optimizer might be similar enough, that there the DNNs end up in the same or a very similar end point, but that the NAM optimization algorithm takes a very different path and ends up in a better (1,2) or worse local optima (2,4). In order to verify this hypothesis, two tests are proposed. Firstly, the Adam and Nadam optimizer are considered. It is studied how similar the predictions of DNNs trained with the two optimization algorithms are. And secondly, the same configurations are trained again with another random initialization, i.e. different

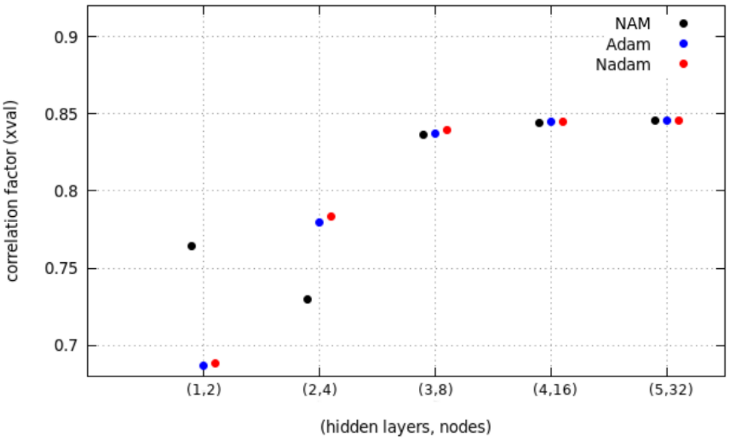


Figure 8.7: Comparison of the correlation factor of the predicted mass against truth mass scatter plot on the cross-validation sample for different DNN configurations and different optimization algorithms.

random seeds, in order to see whether the DNNs end up in the same or similar end points as before.

Comparison of the Adam and Nadam Optimization Algorithm

The Adam and Nadam optimizer achieve very similar correlation factors between the predicted mass and the truth mass. Therefore, it is studied whether they predict similar results for the same examples. For this, the relative deviation of the DNN trained with the Adam optimizer and the DNN trained with the Nadam optimizer is calculated for the (5,32) configuration. There the deviation of the correlation $\Delta r = 1.9 \cdot 10^{-5}$ is particularly small. This is then compared to the relative deviations of both DNNs with the relative deviation of the DNN trained with the NAM optimizer in the same configuration. This is depicted in Fig. 8.8. It can be seen that the deviation of the predicted values of the DNN trained with the Adam algorithm and the DNN trained with the Nadam algorithm is not significantly smaller than the deviation of any other combination. Thus, the hypothesis, that the Adam and Nadam end up in a similar end point which is further different to the end point of the NAM algorithm is not supported.

8.1.4 Impact of Random Initialization

In order to understand why the DNNs trained with different optimization algorithms end up having very different correlation coefficients between the predicted mass and the truth mass, especially in the configurations (1,2) and (2,4), all configurations are trained again with different initial weights Θ of the neural network. This is done by changing the random seeds of the TensorFlow and numpy random generators. The correlation factors are again computed for the correlation between the predicted mass and the truth mass on the cross-validation sample and then compared to the ones obtained with the random initialization used before. For the training with the Adam optimization algorithm, the results are depicted in Fig. 8.9. It can be seen that for the configurations (1,2) and (3,8) the obtained correlation factors differ quite significantly for the different initializations. This suggests that the DNN can really get stuck in different optima or plateau regions. This would also imply that the same configurations should be trained multiple times in order to find the optimal weights. A last cross-check is to have a look at the learning curves for the setup (1,2) or (3,8) trained with the Adam optimization

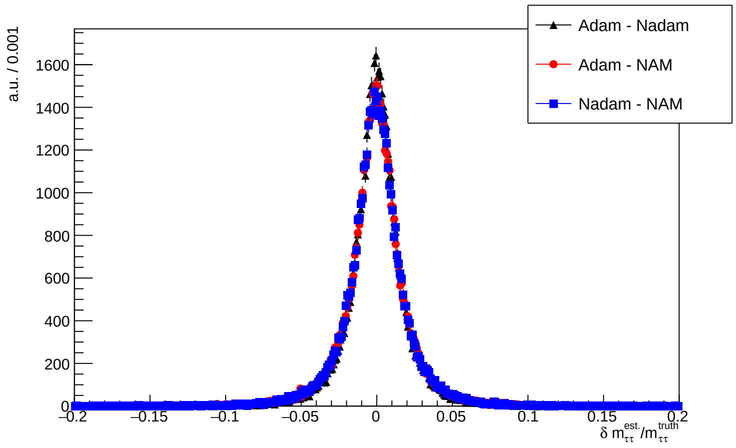


Figure 8.8: Comparison of the relative deviations of the DNN predictions. The DNNs are trained with the different optimization algorithms in the configuration of $L = 5$ hidden layers and $N = 32$ hidden nodes. The predictions were made on the cross-validation sample.

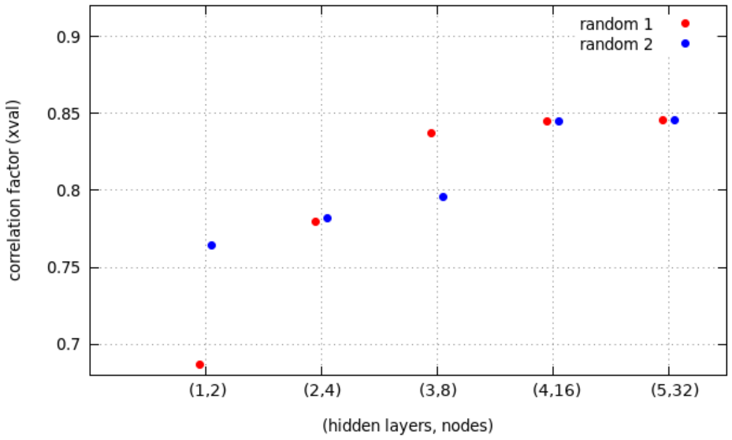


Figure 8.9: Comparison of the correlation factor of the predicted mass against truth mass scatter plot on the cross-validation sample for different DNN configurations trained with the Adam optimization algorithm for different random weight initialization.

algorithm for the different initialization and check whether one training is not finished, in the sense that the loss function is still decreasing. Exemplarily, this is done for the configuration (2,4) and shown in Figs. 8.10 and 8.11. There it can be seen that the square root of the loss starts with a much lower value in Fig. 8.10 compared to Fig. 8.11 but that both learning curves seem to have achieved the end of their training after 200 epochs. A last cross-check would be to check whether deviations for different random initialization still occur if the optimal hyperparameters of the optimization algorithm have been found. Therefore, a hyperparameter scan for the parameter of the optimization algorithm would be needed. A suggestion to address this problem would be to use a more stable optimization algorithm, possibly a Markov chain based procedure, e.g. the Metropolis-Hastings algorithm, if this would be feasible.

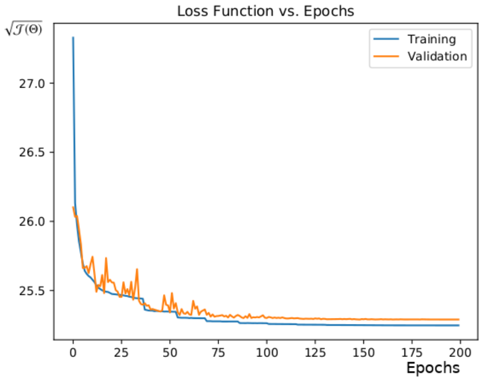


Figure 8.10: Square root of the loss against the number of training epochs for the configuration (3,8) trained with the Adam optimization algorithm in the first random initialization scheme, from Section 8.1.3.

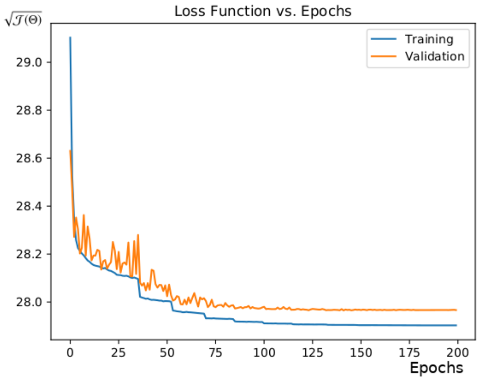


Figure 8.11: Square root of the loss against the number of training epochs for the configuration (3,8) trained with the Adam optimization algorithm in the second random initialization scheme, from Section 8.1.4.

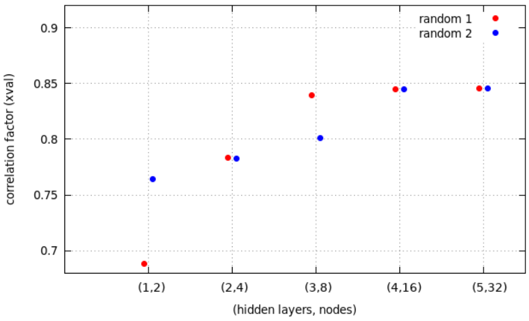


Figure 8.12: Comparison of the correlation factor of the predicted mass against truth mass scatter plot on the cross-validation sample for different DNN configurations trained with the Nadam optimization algorithm for different random weight initialization.

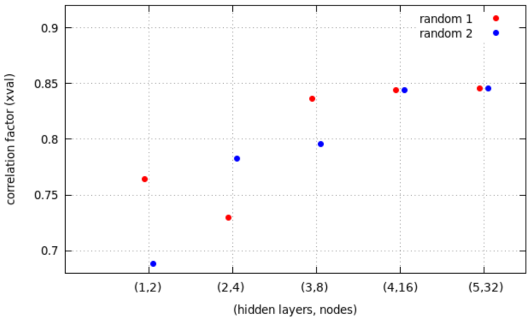


Figure 8.13: Comparison of the correlation factor of the predicted mass against truth mass scatter plot on the cross-validation sample for different DNN configurations trained with the NAM optimization algorithm for different random weight initialization.

For the training with the other two optimization algorithms, the Nadam and NAM algorithm, the results are depicted in Figs. 8.12 and 8.13. They support the local optima or plateau region hypothesis in the sense, that there are also deviations between the different random weight initializations.

Coming back to the hypothesis that the DNNs trained by the Adam and Nadam optimizer end up in very similar end points when trained with the same initial random weights, this hypothesis is not rejected by the results because both the DNNs trained with the Adam and Nadam algorithms show larger deviations in the configurations (1,2) and (3,8) which are in the same direction and very similar in size.

To conclude the study of the different optimization algorithms, there are only small deviations between the Adam and Nadam optimizer and both tend to be better than the NAM optimizer. Getting stuck in local minima or plateau regions is a difficulty in the training process and it could be countered by training the same configuration multiple times with different random initialization or by a hyperparameter scan of the hyperparameters of the optimizer.

8.2 $m_{\text{vis}}/m_{\text{generated}}$ as target

As the target for the DNN training so far the Monte Carlo generated mass of the di-tau system, also referenced as the truth mass, was used. Another possible target could be the ratio of the visible mass divided by the truth mass. The visible mass m_{vis} is the invariant mass of the system defined by the visible decay products of both tau decays. In principle, this is no new information because the visible mass can be calculated from the momentum vectors of the visible decay products which is used as input information for the DNN. A hope could be, that with this target the DNN is easier to train, in a sense, that it leads to better results. In order to study the impact of this new target $m_{\text{vis}}/m_{\text{generated}}$ on the DNN training, the training is compared to a DNN with the truth mass as target for a few selected configurations. Again, the correlation factors are compared. For the DNN trained with $m_{\text{vis}}/m_{\text{generated}}$ as target, the measured visible mass is used for the final prediction, according to:

$$m_{\tau\tau}^{\text{est.}} = m_{\text{vis}}/o\,, \tag{8.2}$$

where $m_{\tau\tau}^{\text{est.}}$ is the final estimate of the mass of the di-tau system and o is the output of the DNN.

In order to get a good understanding of the impact of the new target, for the configurations (1,2), (2,4), (3,8), (4,16) and (5,32) a DNN gets trained with the Adam optimization algorithm and is then compared to the same configurations also trained with the Adam optimization algorithm, but with the truth mass as target. The comparison of the achieved correlation factors on the cross-validation sample is depicted in Fig. 8.14. For the other tested optimization algorithms there were only tiny deviations from the results obtained by training with the Adam optimization algorithm, and therefore they are omitted. It can be seen in Fig. 8.14, that the correlation factor is very high with $r \approx 0.811$ fo the simplest configuration (1,2) but does not improve as much as before with the truth mass as target. The correlation factor for the configuration with the most trainable parameters (5,32), $r \approx 0.844$, is slightly lower than the correlation factor $r \approx 0.846$ obtained by training with the truth mass as target.

As the correlation factor alone is not enough for a detailed analysis, a more differentiated analysis is presented for one specific configuration. In order to have a good comparison to the training with the truth mass as target, again the configuration (5,16) trained with the Adam optimizer is chosen to be presented in more detail.

Firstly, the mass distribution, depicted in Fig. 8.15, looks more smoothly compared to the mass distribution predicted by the DNN trained with the truth mass as target (cf. Fig. 8.2). It is roughly flat, but it still does not or only very rarely predict masses outside the training mass range. The scatter plot of the DNN predicted mass against the truth mass, depicted in Fig. 8.16, looks similar to the one obtained with the truth mass as target (cf. Fig. 8.5). The trend in the mean of the relative deviation, depicted in Fig. 8.17, is still present, however slightly less prominent (cf. Fig. 8.3). The quantile widths are compared to the ones obtained by the DNN prediction with the truth mass as target and are also compared to the MMC prediction obtained quantile widths in Figs. 8.18 and 8.19. There it can be seen that the quantile widths are smaller for the DNN predictions at high and low masses. But essentially there, the DNN predictions are not perfectly valid. This underlines the importance of fixing the edge effect problem.

As it is not clear how to cope with the edge effects for the case of $m_{\text{vis}}/m_{\text{generated}}$ as target and because for the truth mass as target a promising possibility is already tested at truth level, the ratio of

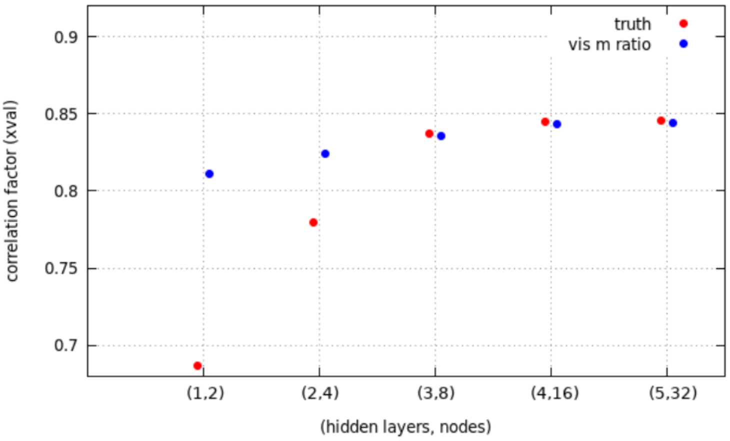


Figure 8.14: Comparison of the correlation factor of the predicted mass against truth mass scatter plot on the cross-validation sample for different DNN configurations of a DNN trained with the truth mass as target (red) and with the ratio of the visible mass and the truth mass as target (blue). Both DNNs are trained with the Adam optimization algorithm.

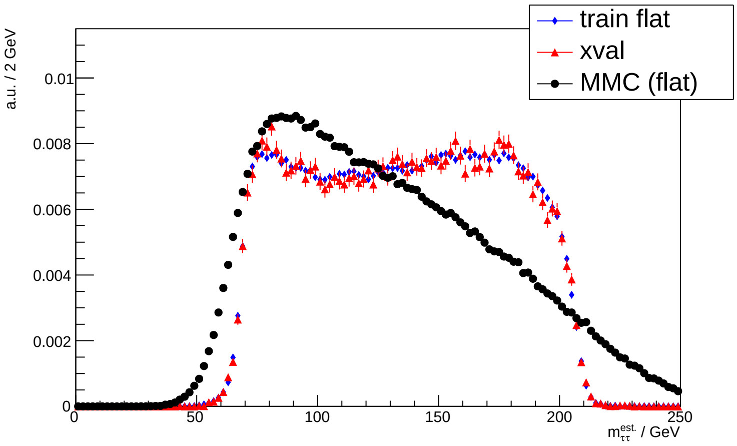


Figure 8.15: Mass distribution of the predicted masses for the flat training subsample (blue), the cross-validation sample (red) and the MMC (black). The DNN has $L = 5$ hidden layers with $N = 16$ nodes and is trained with the Adam optimization algorithm. The training target is the ratio of the visible mass and the truth mass $m_{\text{vis}}/m_{\text{generated}}$.

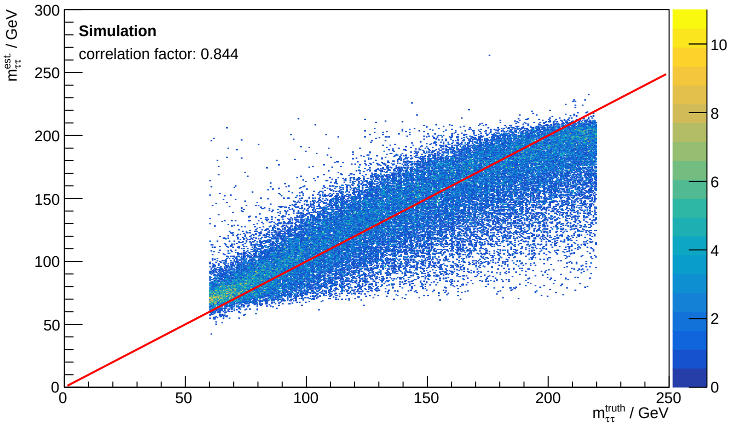


Figure 8.16: Scatter plot of the DNN predicted di-tau mass distribution against the truth mass $m_{\tau\tau}^{\text{truth}}$ in the cross-validation sample. The correlation factor is 0.845. The DNN has $L = 5$ hidden layers with $N = 16$ nodes and is trained with the Adam optimization algorithm.

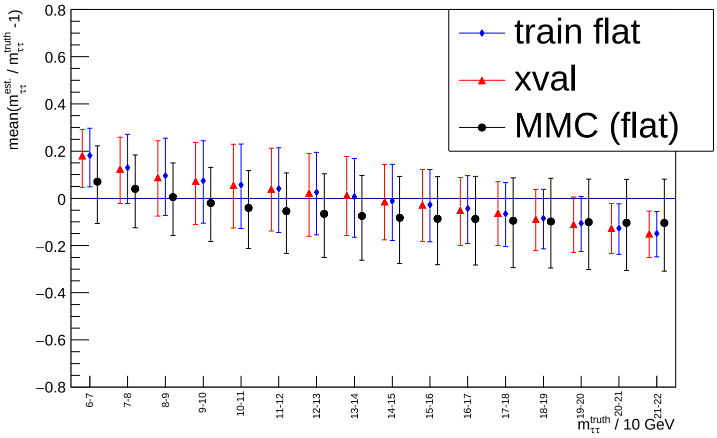


Figure 8.17: Mean of the relative deviation for the flat training subsample (blue), the cross-validation sample (red) and the MMC (black). The DNN has $L = 5$ hidden layers with $N = 16$ nodes and is trained with the Adam optimization algorithm.

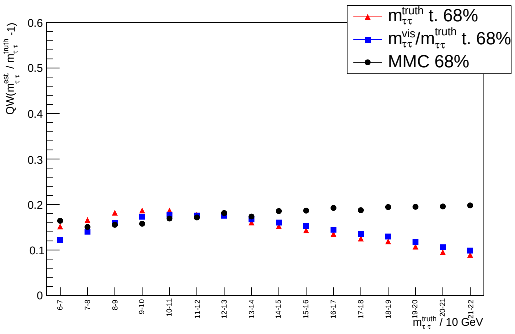


Figure 8.18: Comparison of the half of the quantile widths corresponding to 1σ of the relative deviation $m_{\tau\tau}^{\text{est}}/m_{\tau\tau}^{\text{truth}}-1$ of the cross-validation sample in 10 GeV bins of the truth mass for a DNN trained with the truth mass as target (red), a DNN trained with the ratio of the visible mass and the truth mass as target (blue) and the MMC (black).

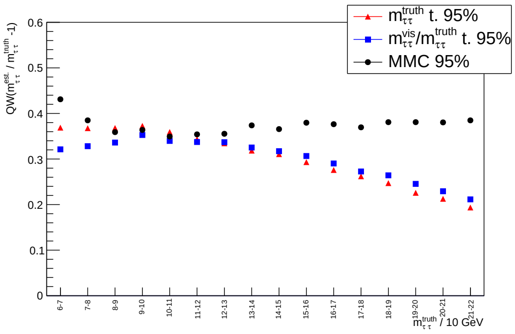


Figure 8.19: Comparison of the half of the quantile widths corresponding to 2σ of the relative deviation $m_{\tau\tau}^{\text{est}}/m_{\tau\tau}^{\text{truth}}-1$ of the cross-validation sample in 10 GeV bins of the truth mass for a DNN trained with the truth mass as target (red), a DNN trained with the ratio of the visible mass and the truth mass as target (blue) and the MMC (black).

the visible mass to the truth mass is not studied further.

8.3 Boundary Effect Correction

In order to deal with the effects that arise from the boundaries, two possibilities could be studied. The first one is calibrating the DNN predicted output and the second method is training the DNN using the boundary corrected loss function, derived in (5.9).

The usage of boundary corrected loss function can potentially solve the boundary effect problem. Therefore, it is studied in more detail. While the boundaries are given by the training sample, i.e. $a = 60\,\text{GeV}$ and $b = 220\,\text{GeV}$, a prior guess of the resolution σ is needed. Following the argumentation of Section 7.2.1 that the relative resolution of the MMC is roughly constant and estimating the resolution as the 68% half-quantile width of the DNN, where it is roughly equal to the MMC resolution at $m_{\tau\tau}^{\text{truth}} \approx 100\,\text{GeV}$ (cf. Fig. 8.18), the prior guess of the resolution is chosen to be $\sigma = 0.18 \cdot m_{\tau\tau}^{\text{truth}}$. Note that the prior guess could heavily effect the achievable DNN performance. Further, note that the prior guess is not a specific characteristic of the boundary corrected loss function but of any loss function. It is often neglected by choosing the square error loss, where the resolution is assumed to be constant.

The setup of the DNN training is chosen as described in Section 7.2, changing only the loss function to the boundary corrected loss function with $\sigma = 0.18 \cdot m_{\tau\tau}^{\text{truth}}$. The results are summarized in Figs. 8.20 -8.26.

The learning curve, where the square root of the loss is depicted after each epoch, shown in Fig. 8.20, implies that the learning is still unstable. This has a huge impact on the RMS (cf. Eq. 7.2) as a function of the number of epochs, depicted in Fig. 8.21. More effort has to be put in stabilizing the training process.

The distribution of the DNN predicted mass, shown in Fig. 8.22, is very similar to the MMC predicted distribution. It is striking that there is a small irregularity at $m_{\tau\tau}^{\text{est.}} \approx 47\,\text{GeV}$. This could be

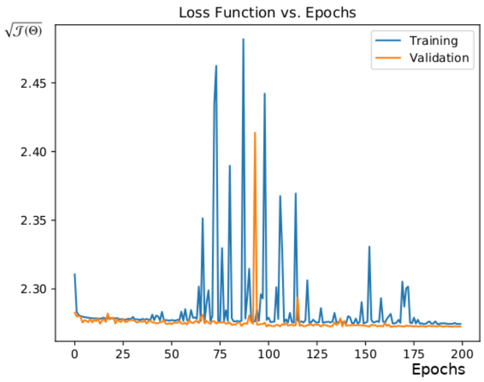


Figure 8.20: Square root of the boundary corrected loss against the number of training epochs.

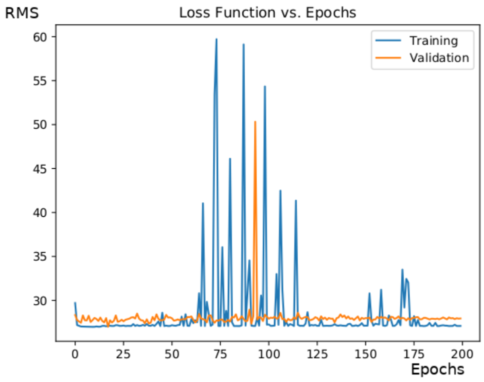


Figure 8.21: RMS of the predicted masses against the number of training epochs.

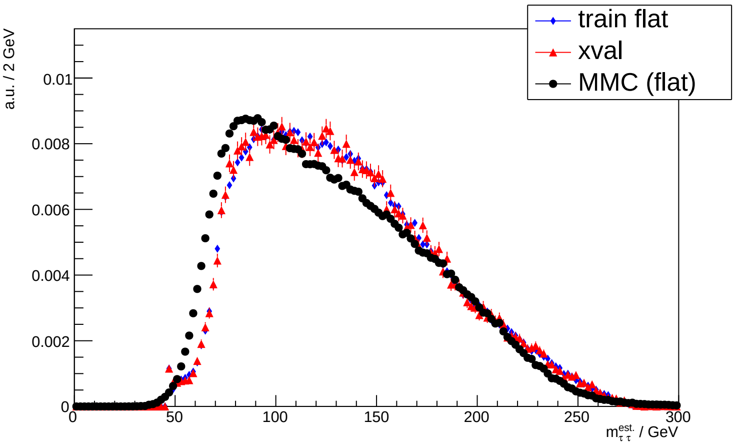


Figure 8.22: Mass distribution of the DNN predicted masses for the flat training subsample (blue), the cross-validation sample (red) and the MMC predicted masses (black).

an effect of the unstable training and has to be further investigated, if this irregularity is still present after a hyperparameter scan.

The scatter plot of the DNN estimated mass against the truth generated mass, depicted in Fig. 8.23, shows that the DNN does indeed predict masses outside the training mass range, but also that there is a sharp edge at $m_{\tau\tau}^{\text{est.}} \approx 47\,\text{GeV}$. The correlation factor of $r = 0.821$ is higher than the correlation factor between the MMC predicted mass and truth mass of $r^{\text{MMC}} = 0.801$, which suggests that the DNN performs better. From Figs. 8.18 and 8.19 it turned out that the DNN predicted resolution without boundary correction is predominantly better in the edge regions. In order to check if this is still the case, the predicted resolution of the DNN, trained with the boundary corrected loss function, is compared to the MMC predicted resolution in Figs. 8.24 and 8.25. It can be seen that the 68%-quantile widths are consistently smaller in the range of $70 - 200\,\text{GeV}$ for the DNN predictions and roughly equally good in the edge regions. The 95%-quantile widths are consistently smaller in the whole tested mass region for the DNN predictions. This suggests that the DNN really performs better on the

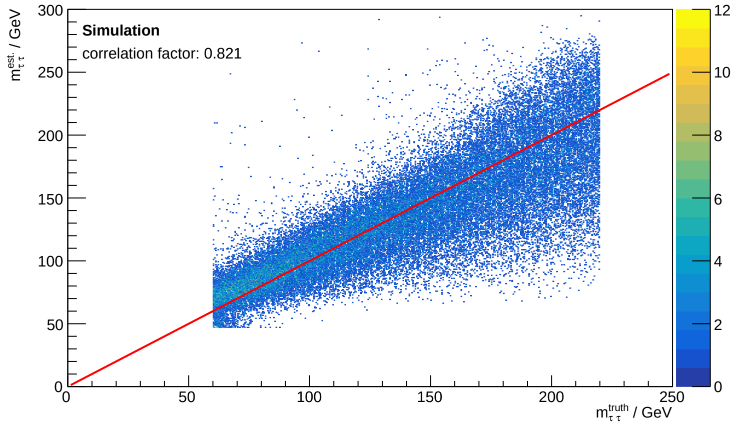


Figure 8.23: Scatter plot of the DNN predicted di-tau mass distribution against the truth mass $m_{\tau\tau}^{\text{truth}}$ in the cross-validation sample. The correlation factor is 0.821.

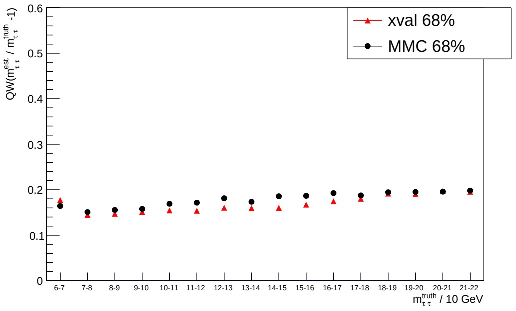


Figure 8.24: Comparison of the half of the quantile widths corresponding to 1σ of the relative deviation $m_{\tau\tau}^{\text{est}}/m_{\tau\tau}^{\text{truth}}-1$ of the cross-validation sample in 10 GeV bins of the truth mass for a DNN trained with the boundary corrected loss function (red) and the MMC (black).

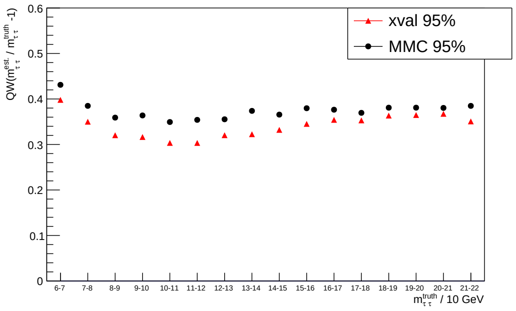


Figure 8.25: Comparison of the half of the quantile widths corresponding to 2σ of the relative deviation $m_{\tau\tau}^{\text{est}}/m_{\tau\tau}^{\text{truth}}-1$ of the cross-validation sample in 10 GeV bins of the truth mass for a DNN trained with the boundary corrected loss function (red) and the MMC (black).

$\gamma^* \to \tau^{\text{had}}\tau^{\text{had}}$ sample.

As a last cross-check the mean of the relative resolution is considered and depicted in Fig. 8.26. There is still a small trend in overestimating small masses and underestimating higher masses. However, this trend is less prominent than in the case of using the squared error loss function (cf. Fig. 8.3). Further notice that the trend is remarkably similar to the trend of the MMC predictions.

With this results it is promising to test the DNN in the test case of separating Z and H events and compare it to the MMC in this particular task.

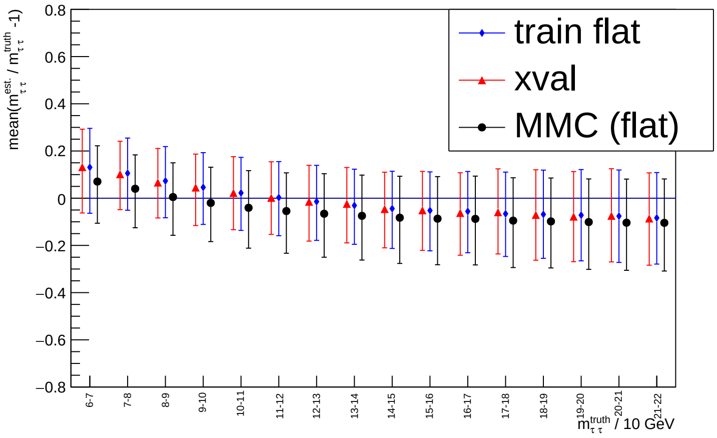


Figure 8.26: Mean of the relative deviation for the DNN predictions on the flat training subsample (blue), the cross-validation sample (red) and the MMC predictions (black).

8.4 Z and H Discrimination

In order to test the DNN estimator, trained with the boundary corrected loss function from Section 8.3, it is applied to the unseen $Z/\gamma^* \to \tau\tau$ and $H \to \tau\tau$ samples. For analyses studying $H \to \tau\tau$ decays, it is important to discriminate between the H boson as signal and the Z boson as background. In order to do so, the predicted mass distributions for both samples and the overlap between them are considered. For optimal separation the overlap should vanish. The mass histograms are normalized to one in order to get an estimate on the probability density function of the mass distributions. For the DNN predicted masses, the histogram of the Z and H samples and their overlap is depicted in Fig. 8.27. The irregularity at $m^{\text{est.}}_{\tau\tau} \approx 47$ GeV is still present in the $Z/\gamma^* \to \tau\tau$ sample and is consistent with the observations made before. Furthermore, it can be seen that the mass of the Z boson is slightly overestimated with 93.9 GeV (91.2 GeV) and the mass of the H boson slightly underestimated with 120 GeV (125 GeV). The resolution and overlap has to be compared to the MMC predicted values, depicted in Fig. 8.28. The MMC has a slightly better 68%-core resolution on the $Z/\gamma^* \to \tau\tau$ sample (14 GeV) compared to the DNN (15 GeV), but the DNN has a slightly better 95%-tail resolution on the $H \to \tau\tau$ sample (33 GeV) than the MMC (35 GeV). The other values, including the overlap, are identical.

For the discriminating power, the Receiver Operator Characteristics (ROC) curve and the area under this curve are good evaluation metrics. The ROC curve is obtained when applying cuts to the variable of interest, i.e. the predicted di-tau mass, and classifying everything above the cut as signal and below as background. For each cut the signal efficiency and the background rejection is calculated and plotted for every possible cut. The ROC curves for the DNN predictions and the MMC predictions are shown in Fig. 8.29. The area under the ROC curve is nearly identical for the MMC predictions (0.878) and the DNN predictions (0.877). Thus, both estimators for the di-tau mass achieve approximately the same discriminating power in the task of separating Z and H.

From the results it can be checked, whether there are transition effects from training the DNN on the $\gamma^* \to \tau^{\text{had}}\tau^{\text{had}}$ sample and predicting the mass of di-tau systems in $Z/\gamma^* \to \tau\tau$ and $H \to \tau\tau$ samples. Estimating the relative resolution of the DNN estimator from the $\gamma^* \to \tau^{\text{had}}\tau^{\text{had}}$ sample

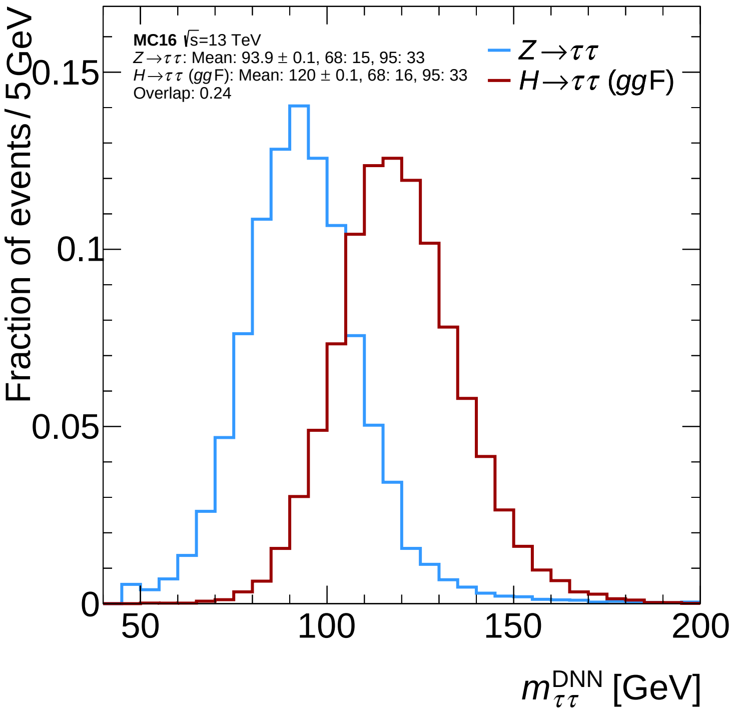


Figure 8.27: Histogram of the normalized mass distribution predicted by the DNN of the $Z/\gamma^* \rightarrow \tau\tau$ sample (blue), the $H \rightarrow \tau\tau$ sample (red) and the overlap between them (0.24).

at the Z and H mass to be roughly $\sigma = 0.16\%$ (cf. Fig. 8.24), resolutions of roughly 15 GeV (Z) and 20 GeV (H) were expected. For the Z sample this resolution estimate is in good agreement with the observed 15 GeV and for the H sample the observed resolution of 16 GeV is actually better than expected. It seems that the transition for the MMC has a bigger impact, because the MMC achieved a worse resolution on the $\gamma^* \rightarrow \tau^{\mathrm{had}}\tau^{\mathrm{had}}$ sample than on the $Z/\gamma^* \rightarrow \tau\tau$ and $H \rightarrow \tau\tau$ samples.

Estimating the relative deviation of the DNN predictions from the $\gamma^* \rightarrow \tau^{\mathrm{had}}\tau^{\mathrm{had}}$ sample at the the Z and H mass, one would expect an absolute deviation of roughly +5 GeV and −6 GeV, respectively (cf Fig. 8.26). Observed are deviations of roughly +3 GeV and −5 GeV, respectively. The expected and observed deviations are in a good agreement. Thus it could be promising to put more effort into getting rid of the relative deviation.

Concluding, the training on the $\gamma^* \rightarrow \tau^{\mathrm{had}}\tau^{\mathrm{had}}$ sample and its application to the $Z/\gamma^* \rightarrow \tau\tau$ sample works very good and gives consistent results. For the application of the $H \rightarrow \tau\tau$ sample, the DNN works better than expected. Further investigations could study if this results from the different spin of the particles, i.e. 1 for γ^* and 0 for H or if this is due to something different. The DNN achieves approximately the same discriminating power in separating Z and H bosons as the MMC, which is the main result of this work.

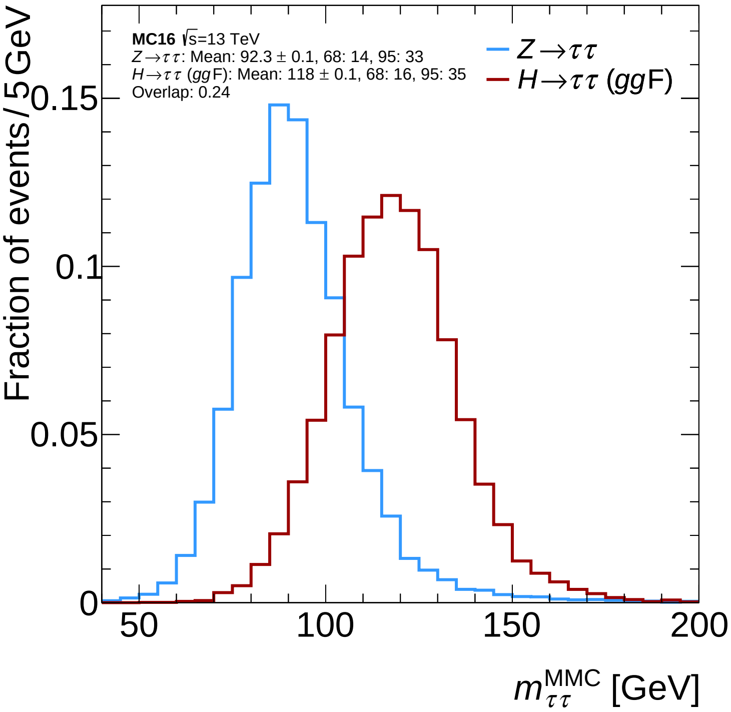


Figure 8.28: Histogram of the normalized mass distribution predicted by the MMC of the $Z/\gamma^* \rightarrow \tau\tau$ sample (blue), the $H \rightarrow \tau\tau$ sample (red) and the overlap between them (0.24).

8.5 Impact of Additional Auxiliary Features

With the main result, that the DNN can perform equally good as the MMC on physical $Z/\gamma^* \rightarrow \tau\tau$ and $H \rightarrow \tau\tau$ samples, a question could be whether it exceeds the MMC performance when trained with more auxiliary information. The additional auxiliary variables are introduced in Section 6.2. In order to study their impact on the DNN training, a DNN is trained with the basic information and one additional input variable at a time. And finally with all additional input variables (combined). The same hyperparameters as in Section 8.3 were chosen in order to have a reference. For every DNN the correlation factor between the DNN predicted masses and the truth masses on the cross-validation sample is extracted. This is then compared to the correlation factor for the DNN without auxiliary information (0.821). The results are depicted in Table 8.1.

Unfortunately, any of the additional auxiliary input variables led to any improvement on the DNN performance and all information combined seem to have lowered the DNN performance. This results have to be treated cautiously, because it was already seen that a DNN has to be trained multiple times and with different random initialization in order to get an actual estimate of its capability.

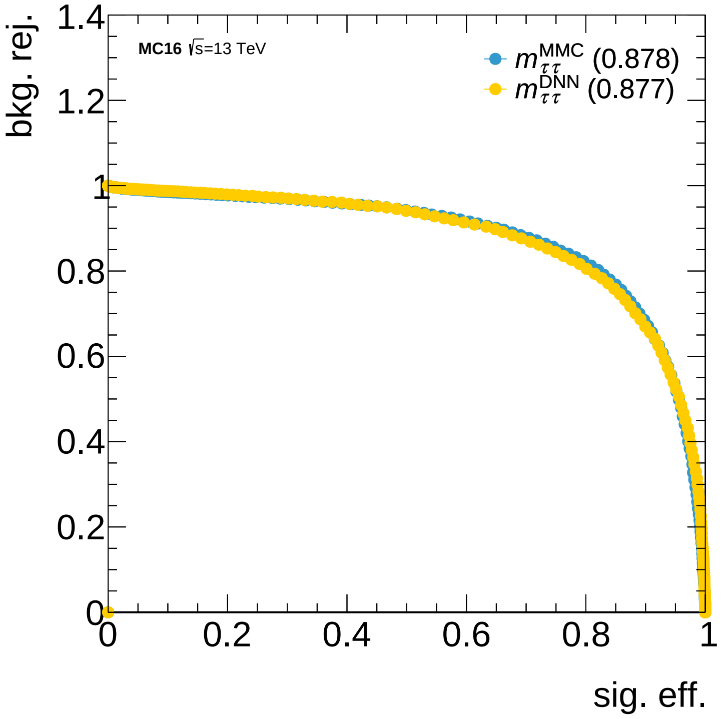


Figure 8.29: ROC curves of the MMC predictions (blue) and the DNN predictions (yellow) for the task of discriminating between the $Z/\gamma^* \to \tau\tau$ and $H \to \tau\tau$ sample.

auxiliary variable	correlation factor
tau decay channel	0.820
`MET significance`	0.820
number of vertices	0.820
number of jets	0.820
combined	0.804

Table 8.1: Overview of the correlation factor between the DNN predicted mass and the truth mass on the cross-validation sample with additional auxiliary information.

CHAPTER 9

Conclusion

9.1 Summary

The accurate reconstruction of the mass of a di-tau system is important for many physics analyses, especially for studies of the Higgs boson decaying into tau-leptons. There, the estimated mass of the di-tau system is the main discriminant for distinguishing between signal and background. A small improvement in the resolution of the mass estimate of a di-tau system can lead to a big improvement in the signal significance because the estimated mass of the Z and H resonance overlap. An improvement in the resolution thus narrows both resonances, lowering the overlap from both sides. The current used estimator for the mass of a di-tau system is the Missing Mass Calculator (MMC).

In this thesis, a setup for a deep neural network training is introduced, in which the deep neural network can learn to estimate the mass of a di-tau system. In this setup, the momentum of the visible tau-decay products and the missing transverse energy serve as input for the deep neural network. A loss function is tested which takes the limited value range into account and tries to counter edge effects that occur when using the squared error loss function for the training. This boundary corrected loss function improves the training of the deep neural network in the sense that the relative deviation of the predicted masses is on average closer to zero and the mass resolution gets better in the middle of the value range, compared to using the squared error loss function. The training of the deep neural network is performed on an unphysical $\gamma^* \to \tau^{\mathrm{had}}\tau^{\mathrm{had}}$ sample, where both tau-leptons decay hadronically. On this data sample the deep neural network estimator outperforms the MMC slightly in the 68%-core resolution and significantly in the 95%-tail resolution.

While the data sample used for training is unphysical, i.e. it only includes the electromagnetic coupling of virtual photons to the tau-leptons and does not take the interference with the weak coupling Z boson into account, the transition to physical samples is studied. The trained deep neural network estimator is applied to $Z/\gamma^* \to \tau\tau$ and $H \to \tau\tau$ samples and the mass resolution is compared to the MMC. The deep neural network estimator has a slightly worse 68%-core resolution than the MMC on the $Z/\gamma^* \to \tau\tau$ data sample with 15 GeV (14 GeV). Both estimators have the same 95%-tail resolution on the $Z/\gamma^* \to \tau\tau$ data sample with 33 GeV. On the $H \to \tau\tau$ data sample, both estimators have a 68%-core resolution of 16 GeV. In the 95%-tail resolution the deep neural network estimator has a better resolution than the MMC with 33 GeV (35 GeV). In the task of separating H as signal and Z as background, the estimators achieve very similar results. Using the area under the receiver operating characteristics curve as the metric, the neural network value (0.877) is only minimally lower than the value achieved by the MMC (0.878). The results achieved on the physical sample are consistent with the results on the unphysical $\gamma^* \rightarrow \tau^{\mathrm{had}}\tau^{\mathrm{had}}$ sample suggesting that further improvements on the deep neural network trained on the unphysical sample would lead to improvements on the physical sample as well.

The achievement of performing equally well in the task of separating Z and H bosons without a fully optimized deep neural network architecture encourages to put more effort in the development of a neural network based estimator of the mass of a di-tau system.

9.2 Future Work

In order to investigate some findings of this thesis, a hyperparameter scan is needed and is suggested for future analyses building on this work. As the usage of the boundary corrected loss function is the biggest improvement on the deep neural network performance, but the training with this loss function is not particularly stable with the set of used hyperparameters, it has to be investigated whether the learning is still unstable when the optimal hyperparameters are found including additional regularization techniques, e.g. dropout.

Another finding which has to be confirmed is the dependency on the initial weights of the deep neural network on its achievable performance. Possibly, this dependency vanishes once the optimal set of hyperparameters is found. If not it could be considered using other optimization techniques, such as the Metropolis-Hastings algorithm.

With a potentially more stable training setup, the impact of additional auxiliary information could be revisited and potentially even further improvement achieved. For new additional auxiliary variables suggestions are made in Section 6.2.1. Also the impact of the representation of the input variables could be studied and the way they are preprocessed, as suggested in Section 6.2.3.

As the presented estimator should only be capable of predicting the mass of di-tau systems where both tau-leptons decay hadronically, a similar analysis could be done for other final states, including one or two leptonically decaying tau-leptons. For the analysis of the case where one tau-lepton decays hadronically and one tau-lepton leptonically a Monte Carlo simulation request was set up for a similar unphysical data sample like the one presented which can be used for a similar analysis. Unfortunately, due to time limitations, this could not be done during this work but is suggested for future work.

Bibliography

[1] T. A. Collaboration, *Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC*, (2012), eprint: `arXiv:1207.7214`.

[2] F. Englert and R. Brout, *Broken Symmetry and the Mass of Gauge Vector Mesons*, Phys. Rev. Lett. **13** (9 1964) 321, URL: `http://link.aps.org/doi/10.1103/PhysRevLett.13.321`.

[3] P. Higgs, *Broken symmetries, massless particles and gauge fields*, Physics Letters **12** (1964) 132, ISSN: 0031-9163, URL: `http://www.sciencedirect.com/science/article/pii/0031916364911369`.

[4] G. S. Guralnik, C. R. Hagen and T. W. B. Kibble, *Global Conservation Laws and Massless Particles*, Phys. Rev. Lett. **13** (20 1964) 585, URL: `https://link.aps.org/doi/10.1103/PhysRevLett.13.585`.

[5] *High Luminosity LHC*, Last viewed: 10.03.2019, URL: `https://home.cern/science/accelerators/high-luminosity-lhc`.

[6] E. Stanecka, *ATLAS Upgrade for High Luminosity LHC*, (2018), URL: `https://cds.cern.ch/record/2647063`.

[7] K. Desch et al., *Probing the CP nature of the Higgs boson at linear colliders with tau spin correlations: The Case of mixed scalar - pseudoscalar couplings*, Phys. Lett. **B579** (2004) 157, arXiv: `hep-ph/0307331 [hep-ph]`.

[8] R. Harnik et al., *Measuring CP violation in $h \rightarrow \tau^+\tau^-$ at colliders*, Physical Review D **88** (2013).

[9] S. Berge and W. Bernreuther, *Determining the CP parity of Higgs bosons at the LHC in the tau to 1-prong decay channels*, Phys. Lett. **B671** (2009) 470, arXiv: `0812.1910 [hep-ph]`.

[10] M. J. Dolan, C. Englert and M. Spannowsky, *Higgs self-coupling measurements at the LHC*, JHEP **10** (2012) 112, arXiv: `1206.5001 [hep-ph]`.

[11] L. H. C. S. W. Group et al., *Handbook of LHC Higgs Cross Sections: 1. Inclusive Observables*, (2011), eprint: `arXiv:1101.0593`.

[12] A. Collaboration, *Cross-section measurements of the Higgs boson decaying into a pair of tau-leptons in proton-proton collisions at $\sqrt{s} = 13$ TeV with the ATLAS detector*, 2018, eprint: `arXiv:1811.08856`.

[13] W. S. McCulloch and W. Pitts, *A logical calculus of the ideas immanent in nervous activity*, The bulletin of mathematical biophysics **5** (1943) 115, ISSN: 1522-9602, URL: `https://doi.org/10.1007/BF02478259`.

[14] Y. Coadou, *Boosted Decision Trees and Applications*, EPJ Web Conf. **55** (2013) 02004.

[15] A. Elagin et al., *A New Mass Reconstruction Technique for Resonances Decaying to di-tau*, (2010), eprint: `arXiv:1012.4686`.

[16] M. Hübner, *Personal communication*, 2019.

[17] S. B. Dugan O'Neil Quentin Buat, *Boosted Regression Trees approach forresonance mass reconstruction*, 2015, URL: `https://indico.cern.ch/event/383016/contributions/1810017/attachments/763878/1047974/BRT_update-S.Bahrasemani-March-01.pdf`.

[18] Q. B. Matthew Quenneville, *Estimating $m_{\tau\tau}$ with Boosted Regression Trees*, 2014, URL: `https://indico.cern.ch/event/358811/contributions/849289/attachments/714626/981116/quentin_buat_brt_introduction.pdf`.

[19] H. S. Simon Schnake and P. Schleper, *Studies of Energy Reconstruction with Deep Learning at the LHC*, DPG Conference Talk, Aachen 2019, 2019.

[20] G. Kasieczka et al., *Deep-learning top taggers or the end of QCD?*, Journal of High Energy Physics **2017** (2017) 6, ISSN: 1029-8479, URL: `https://doi.org/10.1007/JHEP05(2017)006`.

[21] S. Macaluso and D. Shih, *Pulling out all the tops with computer vision and deep learning*, Journal of High Energy Physics **2018** (2018) 121, ISSN: 1029-8479, URL: `https://doi.org/10.1007/JHEP10(2018)121`.

[22] P. Baldi et al., *Improved Energy Reconstruction in NOvA with Regression Convolutional Neural Networks*, 2018.

[23] D. Guest, K. Cranmer and D. Whiteson, *Deep Learning and its Application to LHC Physics*, (2018), eprint: `arXiv:1806.11484`.

[24] P. Bechtle et al., *SCYNet: testing supersymmetric models at the LHC with neural networks*, The European Physical Journal C **77** (2017) 707, ISSN: 1434-6052, URL: `https://doi.org/10.1140/epjc/s10052-017-5224-8`.

[25] P. Baldi, P. Sadowski and D. Whiteson, *Searching for Exotic Particles in High-Energy Physics with Deep Learning*, Nature communications **5** (2014) 4308.

[26] J. J. Thiagarajan et al., *Understanding Deep Neural Networks through Input Uncertainties*, CoRR **abs/1810.13425** (2018).

[27] G. Montavon, W. Samek and K.-R. Müller, *Methods for interpreting and understanding deep neural networks*, Digital Signal Processing **73** (2018) 1, ISSN: 1051-2004, URL: `http://www.sciencedirect.com/science/article/pii/S1051200417302385`.

[28] G. 't Hooft, *Renormalizable Lagrangians for Massive Yang-Mills Fields*, Nucl. Phys. **B35** (1971) 167, [,201(1971)].

[29] A. D. Sakharov, *Violation of CP Invariance, c Asymmetry, and Baryon Asymmetry of the Universe*, Pisma Zh. Eksp. Teor. Fiz. **5** (1967) 32, [Usp. Fiz. Nauk161,61(1991)].

[30] P. J. E. Peebles and B. Ratra, *The Cosmological constant and dark energy*, Rev. Mod. Phys. **75** (2003) 559, [,592(2002)], arXiv: `astro-ph/0207347 [astro-ph]`.

[31] A. G. Riess et al., *Observational evidence from supernovae for an accelerating universe and a cosmological constant*, Astron. J. **116** (1998) 1009, arXiv: `astro-ph/9805201 [astro-ph]`.

[32] P. A. R. Ade et al., *Planck 2015 results. XIII. Cosmological parameters*, Astron. Astrophys. **594** (2016) A13, arXiv: `1502.01589 [astro-ph.CO]`.

[33] F. Halzen and A. D. Martin, *Quarks and Leptons: An Introductory Course in Modern Particle Physics*, Wiley, 1984, ISBN: 9780471887416.

[34] M. Thomson, *Modern Particle Physics*, 3rd ed., Cambridge University Press, 2015.

[35] T. C. Collaboration, *Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC*, (2012), eprint: `arXiv:1207.7235`.

[36] *Standard Model*, Last viewed: 26.01.2019, URL: `https://de.wikipedia.org/wiki/Standard_Model`.

[37] Y. Fukuda et al., *Evidence for Oscillation of Atmospheric Neutrinos*, Phys. Rev. Lett. **81** (8 1998) 1562, URL: `http://link.aps.org/doi/10.1103/PhysRevLett.81.1562`.

[38] Q. R. Ahmad et al., *Direct Evidence for Neutrino Flavor Transformation from Neutral-Current Interactions in the Sudbury Neutrino Observatory*, Phys. Rev. Lett. **89** (1 2002) 011301, URL: `https://link.aps.org/doi/10.1103/PhysRevLett.89.011301`.

[39] T. Araki et al., *Measurement of Neutrino Oscillation with KamLAND: Evidence of Spectral Distortion*, Phys. Rev. Lett. **94** (8 2005) 081801, URL: `https://link.aps.org/doi/10.1103/PhysRevLett.94.081801`.

[40] S. D. Drell and T.-M. Yan, *Massive Lepton Pair Production in Hadron-Hadron Collisions at High-Energies*, Phys. Rev. Lett. **25** (1970) 316, [Erratum: Phys. Rev. Lett.25,902(1970)].

[41] T. D. Lee and C. N. Yang, *Question of Parity Conservation in Weak Interactions*, Phys. Rev. **104** (1 1956) 254, URL: `https://link.aps.org/doi/10.1103/PhysRev.104.254`.

[42] C. S. Wu et al., *Experimental Test of Parity Conservation in Beta Decay*, Phys. Rev. **105** (4 1957) 1413, URL: `http://link.aps.org/doi/10.1103/PhysRev.105.1413`.

[43] M. Goldhaber, L. Grodzins and A. W. Sunyar, *Helicity of Neutrinos*, Phys. Rev. **109** (3 1958) 1015, URL: `http://link.aps.org/doi/10.1103/PhysRev.109.1015`.

[44] S. Glashow, *Partial Symmetries of Weak Interactions*, Nucl. Phys. **22** (1961) 579; A. Salam, “Weak and Electromagnetic Interactions”, *Elementary particle theory. Relativistic groups and analyticity. Proceedings of the Eighth Nobel Symposium*, ed. by N. Svartholm, Stockholm: Almquist & Wiksell, 1968 367; S. Weinberg, *A Model of Leptons*, Phys. Rev. Lett. **19** (1967) 1264.

[45] N. Cabibbo, *Unitary Symmetry and Leptonic Decays*, Phys. Rev. Lett. **10** (12 1963) 531, URL: `https://link.aps.org/doi/10.1103/PhysRevLett.10.531`.

[46] M. Kobayashi and T. Maskawa, *CP-Violation in the Renormalizable Theory of Weak Interaction*, Progress of Theoretical Physics **49** (1973) 652, eprint: `/oup/backfile/content_public/journal/ptp/49/2/10.1143/ptp.49.652/2/49-2-652.pdf`, URL: `http://dx.doi.org/10.1143/PTP.49.652`.

[47] C. N. Yang and R. L. Mills, *Conservation of Isotopic Spin and Isotopic Gauge Invariance*, Phys. Rev. **96** (1 1954) 191, URL: `https://link.aps.org/doi/10.1103/PhysRev.96.191`.

[48] O. W. Greenberg, *Spin and Unitary Spin Independence in a Paraquark Model of Baryons and Mesons*, Phys. Rev. Lett. **13** (1964) 598.

[49] M. Y. Han and Y. Nambu, *Three Triplet Model with Double SU(3) Symmetry*, Phys. Rev. **139** (1965) B1006, [,187(1965)].

[50] H. Fritzsch, M. Gell-Mann and H. Leutwyler, *Advantages of the color octet gluon picture*, Physics Letters B **47** (1973) 365, ISSN: 0370-2693, URL: `http://www.sciencedirect.com/science/article/pii/0370269373906254`.

[51] D. J. Gross and F. Wilczek, *Ultraviolet Behavior of Non-Abelian Gauge Theories*, Phys. Rev. Lett. **30** (26 1973) 1343, URL: `https://link.aps.org/doi/10.1103/PhysRevLett.30.1343`.

[52] H. D. Politzer, *Reliable Perturbative Results for Strong Interactions?*, Phys. Rev. Lett. **30** (26 1973) 1346, URL: `https://link.aps.org/doi/10.1103/PhysRevLett.30.1346`.

[53] M. L. Perl et al., *Evidence for Anomalous Lepton Production in $e^+ - e^-$ Annihilation*, Phys. Rev. Lett. **35** (22 1975) 1489, URL: `http://link.aps.org/doi/10.1103/PhysRevLett.35.1489`.

[54] M. Tanabashi et al., *Review of Particle Physics*, Phys. Rev. D **98** (3 2018) 030001, URL: `https://link.aps.org/doi/10.1103/PhysRevD.98.030001`.

[55] M. Schultens, *Search for Supersymmetry with Hadronically and Leptonically Decaying Tau Leptons at the ATLAS Experiment*, dissertation: University Bonn, 2016, URL: `https://www.lhc-ilc.physik.uni-bonn.de/ergebnisse/dateien/t00000078.pdf?c=t&id=78`.

[56] *The accelerator complex*, Last viewed: 20.03.2019,
URL: `https://home.cern/science/accelerators/accelerator-complex`.

[57] *CERN Experiments*, Last viewed: 20.03.2019,
URL: `https://home.cern/science/experiments`.

[58] *CERN Komplex*, Last viewed: 10.03.2019, URL: `http://www.lhc-facts.ch/`.

[59] *LuminosityPublicResultsRun2*, Last viewed: 14.03.2019, URL: `https://twiki.cern.ch/twiki/bin/view/AtlasPublic/LuminosityPublicResultsRun2`.

[60] L. S. Group et al., *Physics Interplay of the LHC and the ILC*, (2004),
eprint: `arXiv:hep-ph/0410364`.

[61] T. A. Collaboration, *The ATLAS Experiment at the CERN Large Hadron Collider*,
Journal of Instrumentation **3** (2008) S08003,
URL: `https://doi.org/10.1088%2F1748-0221%2F3%2F08%2Fs08003`.

[62] T. A. Collaboration, *The ATLAS Experiment at the CERN Large Hadron Collider*,
Journal of Instrumentation **3** (2008) S08003,
URL: `http://stacks.iop.org/1748-0221/3/i=08/a=S08003`.

[63] F. Ahmadov et al., *The ATLAS Inner Detector commissioning and calibration*,
The European Physical Journal C **70** (2010) 787.

[64] M. Capeans et al., *ATLAS Insertable B-Layer Technical Design Report*,
tech. rep. CERN-LHCC-2010-013. ATLAS-TDR-19, 2010,
URL: `https://cds.cern.ch/record/1291633`.

[65] G. Aad et al.,
Reconstruction of hadronic decay products of tau leptons with the ATLAS experiment,
Eur. Phys. J. **C76** (2016) 295, arXiv: `1512.05955 [hep-ex]`.

[66] A. Collaboration, *Measurement of the photon identification efficiencies with the ATLAS detector using LHC Run 2 data collected in 2015 and 2016*, 2018.

[67] *How ATLAS Detects Particles*, Last viewed: 18.03.2019,
URL: `http://collider.physics.ox.ac.uk/detecting.html`.

[68] G. Aad et al., *Electron performance measurements with the ATLAS detector using the 2010 LHC proton-proton collision data*, Eur. Phys. J. **C72** (2012) 1909,
arXiv: `1110.3174 [hep-ex]`.

[69] M. Aaboud et al., *Measurement of the photon identification efficiencies with the ATLAS detector using LHC Run-1 data*, Eur. Phys. J. **C76** (2016) 666,
arXiv: `1606.01813 [hep-ex]`.

[70] Atlas Collaboration, *Muon reconstruction performance of the ATLAS detector in proton–proton collision data at* $\sqrt{s} = 13\,\text{TeV}$,
The European Physical Journal C **76** (2016) 292, ISSN: 1434-6052,
URL: `https://doi.org/10.1140/epjc/s10052-016-4120-y`.

[71] Atlas Collaboration, *Jet Calibration and Systematic Uncertainties for Jets Reconstructed in the ATLAS Detector at* $\sqrt{s} = 13\,\text{TeV}$, ATL-PHYS-PUB-2015-015 **76** (2015),
URL: `http://cds.cern.ch/record/2037613`.

[72] M. Cacciari, G. P. Salam and G. Soyez, *The anti-ktjet clustering algorithm*, Journal of High Energy Physics **2008** (2008) 063, URL: `https://doi.org/10.1088%2F1126-6708%2F2008%2F04%2F063`.

[73] T. A. collaboration, *Measurement of the tau lepton reconstruction and identification performance in the ATLAS experiment using pp collisions at* $\sqrt{s} = 13\,\text{TeV}$, (2017).

[74] Y. Sakurai, "The ATLAS Tau Trigger Performance during LHC Run 1 and Prospects for Run 2", *Proceedings, 2nd Conference on Large Hadron Collider Physics Conference (LHCP 2014): New York, USA, June 2-7, 2014*, 2014, arXiv: `1409.2699 [hep-ex]`, URL: `http://www.slac.stanford.edu/econf/C140602.2/papers/1409.2699v1.pdf`.

[75] B. H. Brunt et al., *Expected performance of missing transverse momentum reconstruction for the ATLAS detector at* $\sqrt{s} = 13$ *TeV*, tech. rep. ATL-COM-PHYS-2015-347, CERN, 2015, URL: `https://cds.cern.ch/record/2013489`.

[76] A. Collaboration, *Measurement of* τ *polarisation in* $Z/\gamma^* \to \tau\tau$ *decays in proton–proton collisions at* $\sqrt{s} = 8\,\text{TeV}$ *with the ATLAS detector*, European Physical Journal C **78** (2018).

[77] *Probing the* $\mathcal{CP}$ *nature of the Higgs boson coupling to* τ *leptons at HL-LHC*, tech. rep. ATL-PHYS-PUB-2019-008, CERN, 2019, URL: `http://cds.cern.ch/record/2665667`.

[78] M. Hübner, *Effects of tau decay product reconstruction in aHiggs CP analysis with the ATLAS experiment*, MA thesis: Universität Bonn, 2016.

[79] J. R. Koza et al., "Automated Design of Both the Topology and Sizing of Analog Electrical Circuits Using Genetic Programming", *Artificial Intelligence in Design '96*, ed. by J. S. Gero and F. Sudweeks, Springer Netherlands, 1996 151, ISBN: 978-94-009-0279-4, URL: `https://doi.org/10.1007/978-94-009-0279-4_9`.

[80] L. Deng and D. Yu, *Deep Learning: Methods and Applications*, now, 2014, ISBN: 9781601988140, URL: `https://ieeexplore.ieee.org/document/8187230`.

[81] A. Graves, A.-r. Mohamed and G. Hinton, *Speech Recognition with Deep Recurrent Neural Networks*, 2013, eprint: `arXiv:1303.5778`.

[82] A. Esteva et al., *Dermatologist-level classification of skin cancer with deep neural networks*, Nature **542** (2017) 115, URL: `http://dx.doi.org/10.1038/nature21056`.

[83] K. He et al., *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*, 2015, eprint: `arXiv:1502.01852`.

[84] A. Ng and Y. B. Mourri, *Improving Deep Neural Networks: Hyperparameter tuning, Regularization and Optimization*, URL: `https://www.coursera.org/learn/deep-neural-network`.

[85] R. F. Schmidt, F. Lang and M. Heckmann, *Physiologie des Menschen*, Springer-Verlag Berlin Heidelberg, 2011.

[86] A. H. Marblestone, G. Wayne and K. P. Kording,
Toward an Integration of Deep Learning and Neuroscience,
Frontiers in Computational Neuroscience **10** (2016) 94, ISSN: 1662-5188,
URL: `https://www.frontiersin.org/article/10.3389/fncom.2016.00094`.

[87] G. Cybenko, *Approximation by superpositions of a sigmoidal function*,
Mathematics of Control, Signals and Systems **2** (1989) 303, ISSN: 1435-568X,
URL: `https://doi.org/10.1007/BF02551274`.

[88] K. Hornik, M. Stinchcombe and H. White,
Multilayer feedforward networks are universal approximators, Neural Networks **2** (1989) 359,
ISSN: 0893-6080,
URL: `http://www.sciencedirect.com/science/article/pii/0893608089900208`.

[89] S. Sonoda and N. Murata,
Neural Network with Unbounded Activation Functions is Universal Approximator, (2015),
eprint: `arXiv:1505.03654`.

[90] S. Dreyfus, *The numerical solution of variational problems*,
Journal of Mathematical Analysis and Applications **5** (1962) 30, ISSN: 0022-247X,
URL: `http://www.sciencedirect.com/science/article/pii/0022247X62900045`.

[91] A. Cauchy, *Méthode générale pour la résolution des systèmes d'équations simultanées*, 1847
536, URL: `https://gallica.bnf.fr/ark:/12148/bpt6k2982c.image.f540.pagination.langEN`.

[92] H. Robbins and S. Monro, *A Stochastic Approximation Method*,
The Annals of Mathematical Statistics **22** (1951) 400, ISSN: 00034851,
URL: `http://www.jstor.org/stable/2236626`.

[93] J. Kiefer and J. Wolfowitz, *Stochastic Estimation of the Maximum of a Regression Function*,
Ann. Math. Statist. **23** (1952) 462, URL: `https://doi.org/10.1214/aoms/1177729392`.

[94] Y. Dauphin et al., *Identifying and attacking the saddle point problem in high-dimensional non-convex optimization*, 2014, eprint: `arXiv:1406.2572`.

[95] D. E. Rumelhart, G. E. Hinton and R. J. Williams,
Learning representations by back-propagating errors, Nature **323** (1986) 533,
ISSN: 1476-4687, URL: `https://doi.org/10.1038/323533a0`.

[96] Y. E. NESTEROV,
A method for solving the convex programming problem with convergence rate $O(1/k^2)$,
Dokl. Akad. Nauk SSSR **269** (1983) 543,
URL: `http://www.cis.pku.edu.cn/faculty/vision/zlin/1983-A%20Method%20of%20Solving%20a%20Convex%20Programming%20Problem%20with%20Convergence%20Rate%20O(k%5E(-2))_Nesterov.pdf`.

[97] G. Hinton, N. Srivastava and K. Swersky,
rmsprop: Divide the gradient by a running average of its recent magnitude,
Last viewed: 12.02.2019, URL: `http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf`.

[98] J. Duchi, E. Hazan and Y. Singer, *Adaptive Subgradient Methods for Online Learning and Stochastic Optimization*, J. Mach. Learn. Res. **12** (2011) 2121, ISSN: 1532-4435, URL: `http://dl.acm.org/citation.cfm?id=1953048.2021068`.

[99] M. D. Zeiler, *ADADELTA: An Adaptive Learning Rate Method*, 2012, eprint: `arXiv:1212.5701`.

[100] D. P. Kingma and J. Ba, *Adam: A Method for Stochastic Optimization*, 2014, eprint: `arXiv:1412.6980`.

[101] T. Dozat, "Incorporating Nesterov Momentum into", 2015.

[102] *Convolutional Neural Networks for Visual Recognition*, Last viewed: 11.03.2019, URL: `http://cs231n.github.io/neural-networks-3/`.

[103] F. Pedregosa et al., *Scikit-learn: Machine Learning in Python*, Journal of Machine Learning Research **12** (2011) 2825.

[104] S. Hochreiter, *Untersuchungen zu dynamischen neuronalen Netzen*, (1991).

[105] Y. Bengio, P. Simard and P. Frasconi, *Learning long-term dependencies with gradient descent is difficult*, IEEE Transactions on Neural Networks **5** (1994) 157, ISSN: 1045-9227.

[106] I. Sutskever et al., "On the importance of initialization and momentum in deep learning", *Proceedings of the 30th International Conference on Machine Learning*, ed. by S. Dasgupta and D. McAllester, vol. 28, Proceedings of Machine Learning Research 3, PMLR, 2013 1139, URL: `http://proceedings.mlr.press/v28/sutskever13.html`.

[107] Y. LeCun et al., "Efficient BackProp", *Neural Networks: Tricks of the Trade, This Book is an Outgrowth of a 1996 NIPS Workshop*, Springer-Verlag, 1998 9, ISBN: 3-540-65311-2, URL: `http://dl.acm.org/citation.cfm?id=645754.668382`.

[108] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks", *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS'10). Society for Artificial Intelligence and Statistics*, 2010.

[109] S. Ioffe and C. Szegedy, *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*, 2015, eprint: `arXiv:1502.03167`.

[110] G. Klambauer et al., "Self-Normalizing Neural Networks", *Advances in Neural Information Processing Systems 30*, ed. by I. Guyon et al., Curran Associates, Inc., 2017 971, URL: `http://papers.nips.cc/paper/6698-self-normalizing-neural-networks.pdf`.

[111] F. Mosteller and J. W. Tukey, "Data analysis, including statistics", *Handbook of Social Psychology, Vol. 2*, ed. by G. Lindzey and E. Aronson, Addison-Wesley, 1968.

[112] G. James et al., *An Introduction to Statistical Learning – with Applications in R*, vol. 103, Springer Texts in Statistics, Springer, 2013, ISBN: 978-1-4614-7137-0.

[113] N. S. Keskar et al.,
On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima, 2016,
eprint: `arXiv:1609.04836`.

[114] S. Farid, *Search for New Physics Beyond the Standard Model*, PhD thesis, 2016.

[115] T. Sjöstrand et al., *An Introduction to PYTHIA 8.2*, Comput. Phys. Commun. **191** (2015) 159,
arXiv: `1410.3012 [hep-ph]`.

[116] S. Alioli et al., *A general framework for implementing NLO calculations in shower Monte Carlo programs: the POWHEG BOX*, JHEP **06** (2010) 043, arXiv: `1002.2581 [hep-ph]`.

[117] S. Frixione, P. Nason and C. Oleari,
Matching NLO QCD computations with parton shower simulations: the POWHEG method,
Journal of High Energy Physics **2007** (2007).

[118] T. Gleisberg et al., *Event generation with SHERPA 1.1*, (2008), eprint: `arXiv:0811.4622`.

[119] *Substructure Plotbook*, pp. 17-19, 2018,
URL: `https://cernbox.cern.ch/index.php/s/tiI9KheBwHUOAaU`.

[120] R. Józefowicz, E. Richter-Was and Z. Was, *Potential for optimizing the Higgs boson CP measurement in $H \to \tau\tau$ decays at the LHC including machine learning techniques*,
Phys. Rev. **D94** (2016) 093001, arXiv: `1608.02609 [hep-ph]`.

[121] K. Lasocha et al.,
Machine learning classification: case of Higgs boson CP state in $H \to \tau\tau$ decay at LHC,
2018, eprint: `arXiv:1812.08140`.

[122] F. Pernkopf et al., *Efficient and Robust Machine Learning for Real-World Systems*, 2018,
eprint: `arXiv:1812.02240`.

[123] F. Chollet et al., *Keras*, `https://keras.io`, 2015.

[124] Martin Abadi et al., *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*,
Software available from tensorflow.org, 2015, URL: `https://www.tensorflow.org/`.

[125] R. Pascanu, T. Mikolov and Y. Bengio,
On the difficulty of training Recurrent Neural Networks,
30th International Conference on Machine Learning, ICML 2013 (2012).

[126] N. Srivastava et al., *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*,
Journal of Machine Learning Research **15** (2014) 1929,
URL: `http://jmlr.org/papers/v15/srivastava14a.html`.

APPENDIX A

Acronymes

ANN Artificial neural network - is a general term for a computational system of artificial neurons inspired by the human brain. There exist many different types of ANNs.

FNN Feed-forward neural network - is an artificial neural network in which the information flow is in a distinct *forward* direction. A FNN consist of an input layer, at least one intermediate layer and an output layer. The number of intermediate, or hidden, layers L describes the depth of a FNN.

DNN Deep neural network - is an artificial neural network with many intermediate layers. A DNN does not have to be a feed-forward neural network, but these are the only ones which are considered in this work. Thus, they are used synonymously.

SNN Self-normalizing neural network - a feed-forward neural network where the inputs of the intermediate layers $l > 1$ are normalized without explicit transformation, only by the choice of the activation function a of the artificial neurons and the weight initialization scheme.

APPENDIX B

Algorithms

Algorithm 3 Batch Normalizing Transform

```
procedure BATCH NORMALIZING TRANSFORM($\vec{z}_{i,l} = (z_{i,l}^{(0)}, ..., z_{i,l}^{(m)})$)    ▷ $m$ training examples
    $\mu_{\mathcal{B}} \leftarrow \frac{1}{s} \sum_{k=1}^{s} z_{i,l}^{(k)}$    ▷ compute mean on mini-batch
    $\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{s} \sum_{k=1}^{s} \left( z_{i,l}^{(k)} - \mu_{\mathcal{B}} \right)^2$    ▷ compute variance on mini-batch
    for ($j$ in (1,$m$)) do
        $\hat{z}_{i,l}^{(j)} \leftarrow \frac{z_{i,l}^{(j)} - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}}$    ▷ normalize every training example
        $z_{i,l}^{(j)'} \leftarrow \gamma \hat{z}_{i,l}^{(j)} + \beta := \mathrm{BN}_{\gamma\beta}(z_{i,l}^{(j)})$    ▷ scale and shift, $\gamma$ and $\beta$ are learnable parameters
    end for
    return $\vec{z}'_{i,l}$
end procedure
```

Adagrad

Adagrad is like RMSprop an adaptive learning optimization algorithm. The divisor differs from RMSprop as it takes all previous updates into account and not only the exponentially moving average:

$$\begin{aligned} s_{(\partial\theta)^2}(t) &= s_{(\partial\theta)^2}(t-1) + (\partial\theta(t))^2 \\ \theta(t) &= \theta(t-1) - \frac{\alpha}{\sqrt{s_{(\partial\theta)^2}(t) + \epsilon}} \partial\theta(t) \, , \end{aligned} \tag{B.1}$$

where again, ϵ ensures not to divide by 0. The problem with Adagrad is, that the learning rate $\alpha' = \alpha / \sqrt{s_{(\partial\theta)^2}(t) + \epsilon}$ gets smaller and smaller during training such that for some weights it will eventually be too small to learn something new.

Adadelta

Adadelta is an extension to Adagrad and very similar to RMSprop. In Eq. 5.17 the parameter updates of θ would have a different unit than θ itself, when θ is not dimensionless. Adadelta

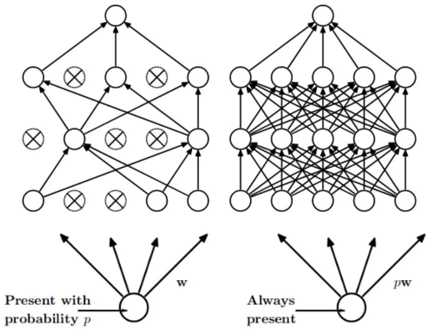


Figure B.1: Illustration of an ANN with dropout during training time (left) and testing time (right), edited after [126].

therefore defines a second exponentially weighted moving average which calculates the average of the actual weight updates $\Delta\theta(t)$ and updates the weights according to:

$$\begin{aligned} s_{(\partial\theta)^2}(t) &= \beta s_{(\partial\theta)^2}(t-1) + (1-\beta)\left(\partial\theta(t)\right)^2 \\ \theta(t) &= \theta(t-1) - \underbrace{\frac{\sqrt{s_{(\Delta\theta)^2}(t-1)+\epsilon}}{\sqrt{s_{(\partial\theta)^2}(t)+\epsilon}}\partial\theta(t)}_{\Delta\theta(t)} \\ s_{(\Delta\theta)^2}(t) &= \beta s_{(\Delta\theta)^2}(t-1) + (1-\beta)\left(\Delta\theta(t)\right)^2 \ . \end{aligned} \tag{B.2}$$

Thus, there is no need to set an initital learning rate α.

Dropout

Dropout is another way to reduce variance. The idea of dropout is to randomly drop artificial neurons during at training time. By doing this, a set of different possible ANNs with shared weights is trained and averaged over. To achieve this, an artificial neuron is dropped with probability $1 - p$ during training, i.e. set to zero, while the weights are multiplied with a factor p at test time [126]. A sketch of an ANN with dropout during training is depicted in Fig. B.1. This forces the artificial neurons to be more robust, i.e. to not rely on many input features, motivated by the way genes are combined during reproduction. For the setup of SNNs, the dropout technique is adjusted, such that the artificial neurons are not set to zero, but to the value of a not-activated neuron, i.e. $\lim_{x\to-\infty} a(x) = -\lambda\alpha = \alpha'$, and called alpha-dropout. In order to preserve mean and variance and to keep the self-normalizing property, the input to an artificial neuron in the next layer, after performing alpha-dropout to the previous layer, is transformed according to

$$b_j^{(l-1)} \leftarrow c \cdot b_j^{(l-1)} + d \tag{B.3}$$

with

$$c = \sqrt{p \cdot (1 + (1-p)(\alpha')^2)} \qquad \text{and} \qquad d = -c \cdot \alpha'(1-p) \ . \tag{B.4}$$

It was found that keeping probabilities p between $90 - 95\%$ lead to good results [110].

APPENDIX C

Combined Sherpa Samples

For the $Z/\gamma^* \rightarrow \tau\tau$ sample a combination of statistically independent SHERPA simulated data samples is used. The samples are:

- `mc16_13TeV.308094.Sherpa_221_NNPDF30NNLO_Ztautau2jets_Min_N_TChannel.deriv.DAOD_HIGG4D3.e5767_s3126_r10201_p3749`
- `mc16_13TeV.344775.Sherpa_221_NNPDF30NNLO_Ztautau_MAXHTPTV0_70_h30h20.deriv.DAOD_HIGG4D3.e5585_s3126_r10201_p3749`
- `mc16_13TeV.344779.Sherpa_221_NNPDF30NNLO_Ztautau_MAXHTPTV70_140_h30h20.deriv.DAOD_HIGG4D3.e5585_s3126_r10201_p3749`
- `mc16_13TeV.344782.Sherpa_221_NNPDF30NNLO_Ztautau_MAXHTPTV140_280_h30h20.deriv.DAOD_HIGG4D3.e5585_s3126_r10201_p3749`
- `mc16_13TeV.364212.Sherpa_221_NN30NNLO_Ztt_Mll10_40_MAXHTPTV70_280_BVeto.deriv.DAOD_HIGG4D3.e5421_s3126_r10201_p3759`
- `mc16_13TeV.364213.Sherpa_221_NN30NNLO_Ztt_Mll10_40_MAXHTPTV70_280_BFilter.deriv.DAOD_HIGG4D3.e5421_s3126_r10201_p3759`
- `mc16_13TeV.364214.Sherpa_221_NN30NNLO_Ztt_Mll10_40_MAXHTPTV280_E_CMS_BVeto.deriv.DAOD_HIGG4D3.e5421_s3126_r10201_p3759`
- `mc16_13TeV.364215.Sherpa_221_NN30NNLO_Ztt_Mll10_40_MAXHTPTV280_E_CMS_BFilter.deriv.DAOD_HIGG4D3.e5421_s3126_r10201_p3759`

APPENDIX D

Input Variable Distributions

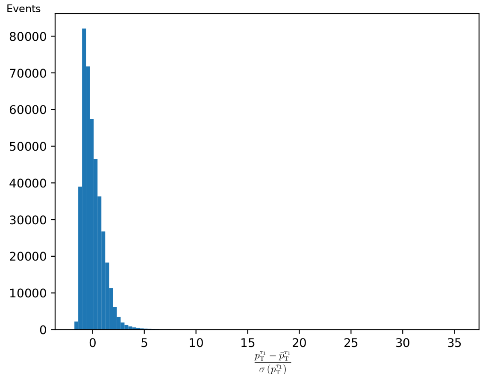


Figure D.1: Distribution of the scaled visible transverse momentum of the higher energetic tau-lepton.

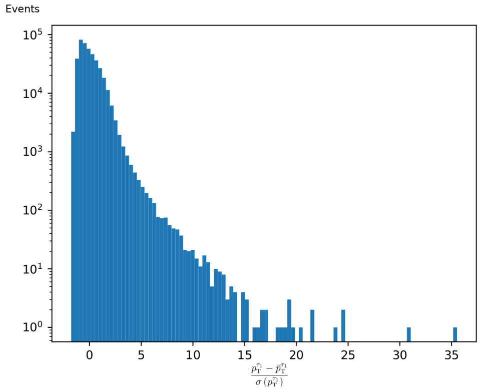


Figure D.2: Distribution of the scaled visible transverse momentum of the higher energetic tau-lepton with a logarithmic *y*-axis.

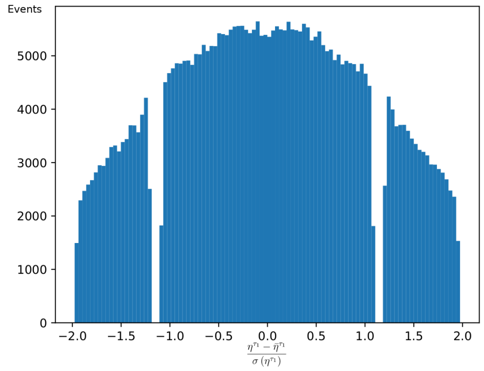


Figure D.3: Distribution of the scaled pseudorapidity of the visible momentum of the higher energetic tau-lepton.

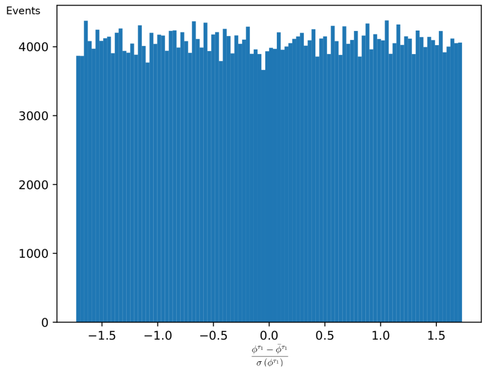


Figure D.4: Distribution of the scaled azimuthal angle of the visible momentum of the higher energetic tau-lepton.

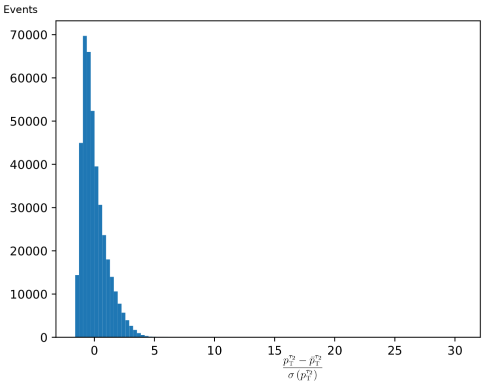


Figure D.5: Distribution of the scaled visible transverse momentum of the lower energetic tau-lepton.

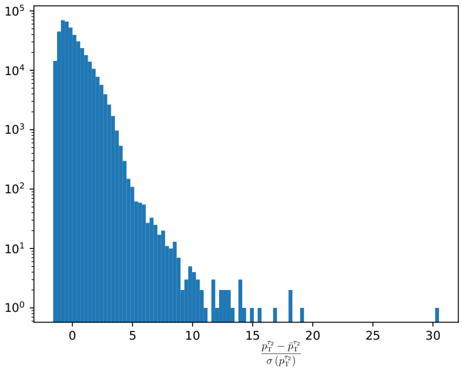


Figure D.6: Distribution of the scaled visible transverse momentum of the lower energetic tau-lepton with a logarithmic *y*-axis.

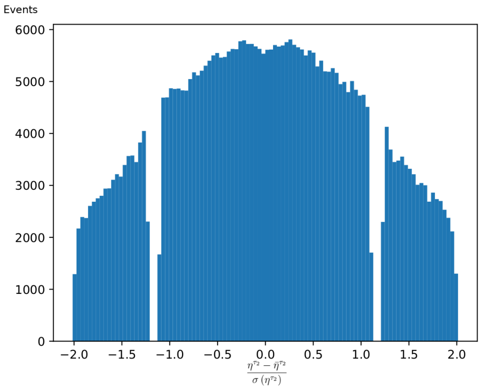


Figure D.7: Distribution of the scaled pseudorapidity of the visible momentum of the lower energetic tau-lepton.

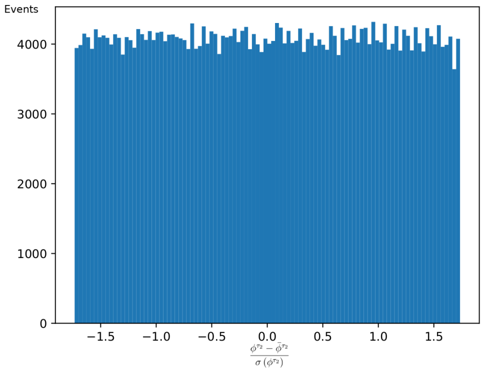


Figure D.8: Distribution of the scaled azimuthal angle of the visible momentum of the lower energetic tau-lepton.

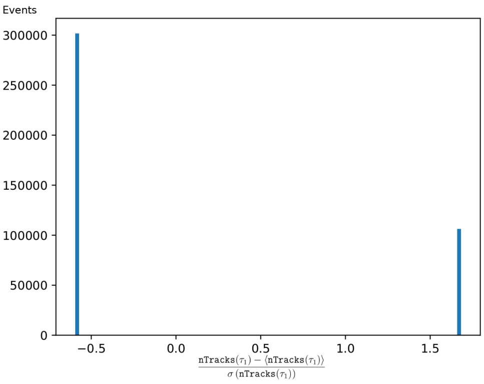


Figure D.9: Distribution of the scaled charged tracks of the higher energetic tau-lepton.

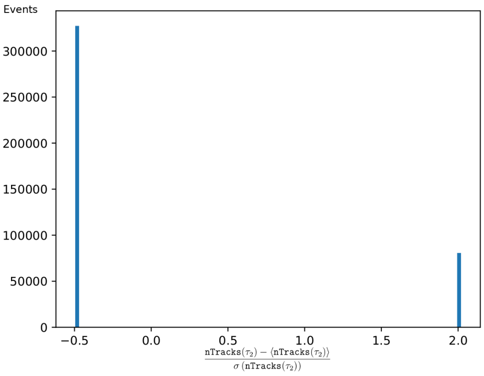


Figure D.10: Distribution of the scaled charged tracks of the lower energetic tau-lepton.

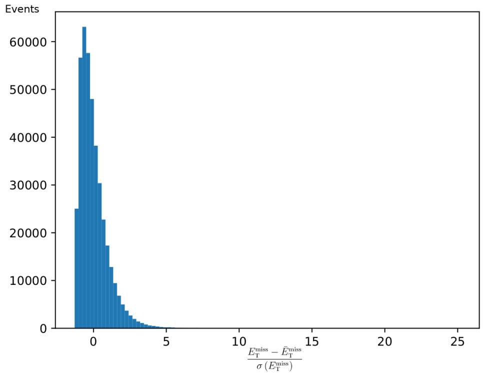


Figure D.11: Distribution of the scaled magnitude of the missing transverse energy.

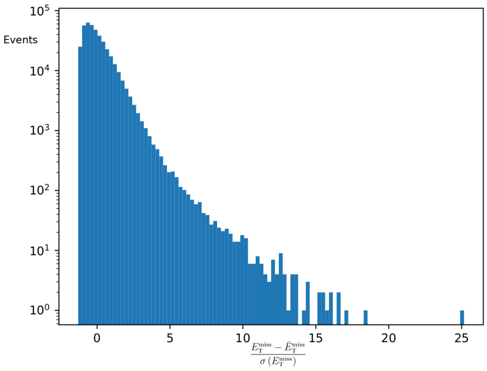


Figure D.12: Distribution of the scaled magnitude of the missing transverse energy with a logarithmic *y*-axis.

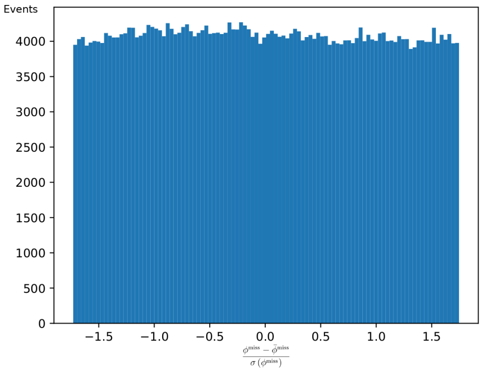


Figure D.13: Distribution of the scaled azimuthal angle of the missing transverse energy.

APPENDIX E

Results - Additional Plots

E.1 Impact of DNN Depth - Additional Plots

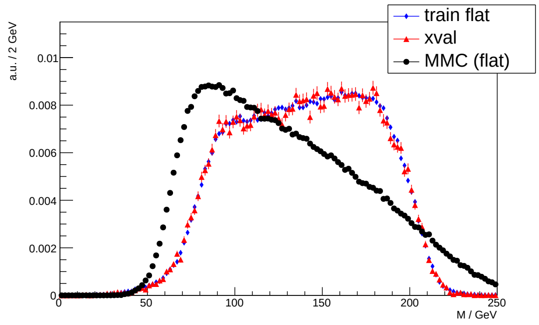


Figure E.1: Mass distribution of the predicted masses for the flat training subsample (blue), the cross-validation sample (red) and the MMC (black). The DNN has $L = 1$ hidden layers with $N = 16$ nodes and is trained with the Adam optimization algorithm.

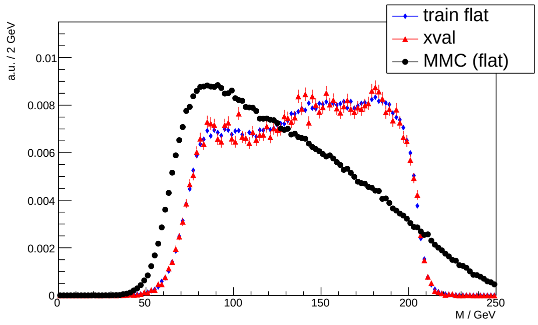


Figure E.2: Mass distribution of the predicted masses for the flat training subsample (blue), the cross-validation sample (red) and the MMC (black). The DNN has $L = 1$ hidden layers with $N = 16$ nodes and is trained with the Adam optimization algorithm.

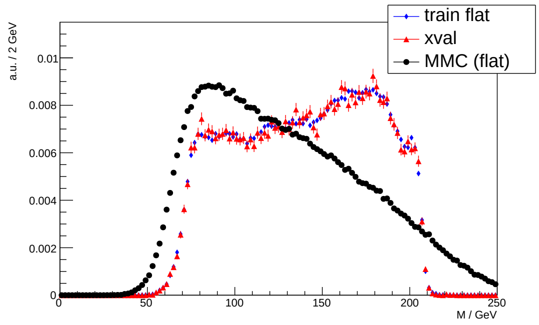


Figure E.3: Mass distribution of the predicted masses for the flat training subsample (blue), the cross-validation sample (red) and the MMC (black). The DNN has $L = 1$ hidden layers with $N = 16$ nodes and is trained with the Adam optimization algorithm.

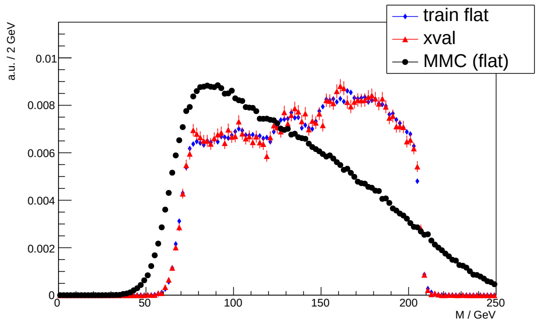


Figure E.4: Mass distribution of the predicted masses for the flat training subsample (blue), the cross-validation sample (red) and the MMC (black). The DNN has $L = 1$ hidden layers with $N = 16$ nodes and is trained with the Adam optimization algorithm.

List of Figures

List of Tables