IEKP-KA/2016-11

Study of B Meson Flavor Tagging with Deep Neural Networks at Belle and Belle II

Jochen Frank Gemmler

A Thesis submitted for the Degree of

Master of Science

Institut für Experimentelle Kernphysik,

Department of Physics,

Karlsruhe Institute of Technology (KIT)

August 3rd, 2016

Advisor: Prof. Dr. M. Feindt
Coadvisor: Prof. Dr. G. Quast

Contents

1. Introduction

Particle physics experiments are advancing to increasing energies, testing our understanding of the universe at smaller and smaller scales. The Standard Model of particle physics (SM) describes observed phenomena with remarkable accuracy. It contains all known fundamental particles and three of the four known fundamental interactions. Nearly all tests in high energy physics are in agreement with the SM.

With the discovery of the Higgs boson at the Large Hadron Collider (LHC), the last piece of the SM is now in place [1, 2]. However, there are observations which indicate the limitations of the SM and point towards new physics beyond the SM. Astrophysical and cosmological measurements, e.g. galactic rotation curves or gravity lensing effects imply the existence of dark matter. The SM does not provide a viable dark matter candidate. It is also important to understand the mechanism which generates the matter–antimatter asymmetry in the universe.

According to one of the Sakharov conditions [3], the charge-parity (CP) symmetry has to be violated to a certain degree in the early universe to achieve the observed matter domination. The mechanism to describe CP violation in the SM, which was introduced by Makoto Kobayashi and Toshihide Maskawa [4], does not provide the required order of magnitude. In measurements of neutral B meson decays by the BarBar experiment [5] and the Belle experiment [6], time dependent CP violation in agreement with the SM was observed.

The Belle experiment is located in Tsukuba, Japan, and is dedicated to the study of B meson decays in order to improve the knowledge about CP violation and other aspects of the flavor sector in particle physics. Due to the high variety of successful measurements on an amount of in total 1 ab^{-1} of accumulated data [7], a successor, the Belle II experiment was approved. Measurements with unmatched precision on the angles of the unitary triangle, CP asymmetry, rare decays and studies of the $\Upsilon(5S)$ resonance are planned.

Besides improvements of the experimental setup, the Belle II experiment uses a modernized software framework which includes new techniques to make measurements even more efficient and sensitive. Various improvements in event reconstruction algorithms and a higher degree of automation was achieved. Continuous development incorporates the demand for utilization and evaluation of new analysis techniques in the field of multivariate classification and machine learning.

In the past few years, a revolution in machine learning techniques has changed the way classification processes can be carried out with huge datasets. The so-called deep learning technique [8] utilizes computational models with multiple layers of feature representations. A key aspect is that these representations are not constructed manually but learned by the algorithm. The progress does not only depend on improvements in understanding of the behavior of neural networks, innovative network architectures or training algorithms. It is also driven by the progress in the utilized hardware, designed for parallelized computations.

This thesis covers a successful application of a deep neural network for the determination of the flavor of neutral B mesons, the so-called *flavor tagging*.
At the Belle and Belle II experiments, neutral B mesons occur in entangled pairs, defined as a signal-side B meson (B_{sig}) and tag-side B meson (B_{tag}). They are decay products of the $\Upsilon(4S)$ resonance, which is created in collisions of accelerated electrons and positrons. When analyzing the decay of the signal B meson into an CP eigenstate which is accessible for both B meson flavors, e.g. $B^0 \rightarrow J/\Psi K_S^0$, the original flavor cannot be determined by its decay products. Due to the entanglement, the flavor of the tag-side B meson determines the flavor of the signal-side B meson at the time of the decay of the tag-side B meson.
A full reconstruction, including all intermediate states of the tag-side B meson only is possible in a small amount of cases. Therefore the flavor has to be determined inclusively, via its decay products.

In this thesis, a deep neural network is utilized for the flavor tagging, without creating high level input features or assigning single tracks to certain categories, as it is done in the current flavor tagging approach. Instead, several attributes like particle identification, momentum and spacial variables of reconstructed tracks are used on event basis. The convergence and performance trainings algorithm is heavily influence by adaptable parameters, so-called hyperparameters. They are interdependent, and may have to be adjusted to the data representation and the network architecture. Therefore, their influence on the problem of interest is studied. By using a deep neural network as multivariate classifier, the effective tagging efficiency could be increased by approximately 22 % relative to the current approach on Belle II Monte Carlo (MC).
As part of this thesis, this approach was also integrated into the software framework of the Belle II experiment.

In Chapter 2, an overview of the experimental setup is provided. Understanding the embedding of the multivariate classifier in the global environment helps in identifying the relevant variables, their uncertainties, and recognizing the symmetries of the problem. The principle of flavor tagging and its importance in B meson mixing analyses are discussed in Chapter 3. The basics about artificial neural networks, needed for the construction of the flavor tagger, are reviewed in Chapter 4. In Chapter 5, the utilization of a deep neural network as a flavor tagger is illustrated. The influence of the hyperparameters on the classifier used for flavor tagging is studied and a comparison to the established flavor tagging approach is provided. The results of this thesis are summarized in Chapter 6 and further prospects are discussed.

2. The Belle and Belle II Experiment

The Belle II experiment is designed to investigate electron–positron-collisions and is located at the High Energy Accelerator Research Organization (KEK) in Tsukuba, Japan. The detector operates at the SuperKEKB, where an electron beam with an energy of 7 GeV and a positron beam with the energy of 4 GeV are colliding. The accelerator was upgraded from KEKB, which was used for particle production at the Belle experiment, the predecessor of Belle II.
While both experiments build up on a common foundation, the Belle II experiment benefits from the experience of the preceding experiment. The Belle II Collaboration is responsible for designing, operating and analyzing the data of the Belle II experiment. Approximately 600 members from more than 20 nations are participating in it. The subsequent part mainly focuses on the presumable setup and concepts for Belle II which are essential for the understanding of the development of a B meson flavor tagger, the main subject of this thesis. By pointing out the main differences, the Belle experiment will also be described briefly.
Many details of the Belle II spectrometer can be found in the technical design report [9]. It is used as a main source of information for this chapter, and is followed, if not pointed out otherwise. The Belle Detector is described in [10].

2.1. SuperKEKB

SuperKEKB is an upgrade of the KEKB accelerator [9]. It is an e^+e^-- collider with asymmetric energies at 4 GeV in the low-energy ring (LER) and at 7 GeV in the high-energy ring(HER). This results in a non-zero boost $\beta\gamma = 0.28$ of the $\Upsilon(4S)$ center-of-mass system (CMS) with respect to the laboratory system and allows measurements of time dependent CP violation. The collider uses the so-called nano beam scheme, which reduces the beam aperture to the nano scale and makes it possible to increase the luminosity to $8 \cdot 10^{35}\mathrm{cm}^{-2}\mathrm{s}^{-1}$ [9, p.20] .

SuperKEKB is designed to run mainly at the energy 10.58 GeV of the Υ(4S) resonance [13], but is also planed to take data at the Υ(5S) energy and at off-resonance energies (Fig. 2.1). For the integrated luminosity an accumulation of 50 ab^{-1} data was planned to be reached by 2022 [14] and rescheduled to 2024. It is about 50 times the data amount taken by Belle. This enhances sensitivity in several areas, where possible “New Physics” processes could be detectable. A detailed review of some of these processes and expectations is provided in [15].

One of the advantages of an electron–positron collider over an hadron collider like the LHC are fewer amount of tracks in a collision event . On average, an event of an $\Upsilon(4S)$ resonance decay contains around 11 tracks of interest.

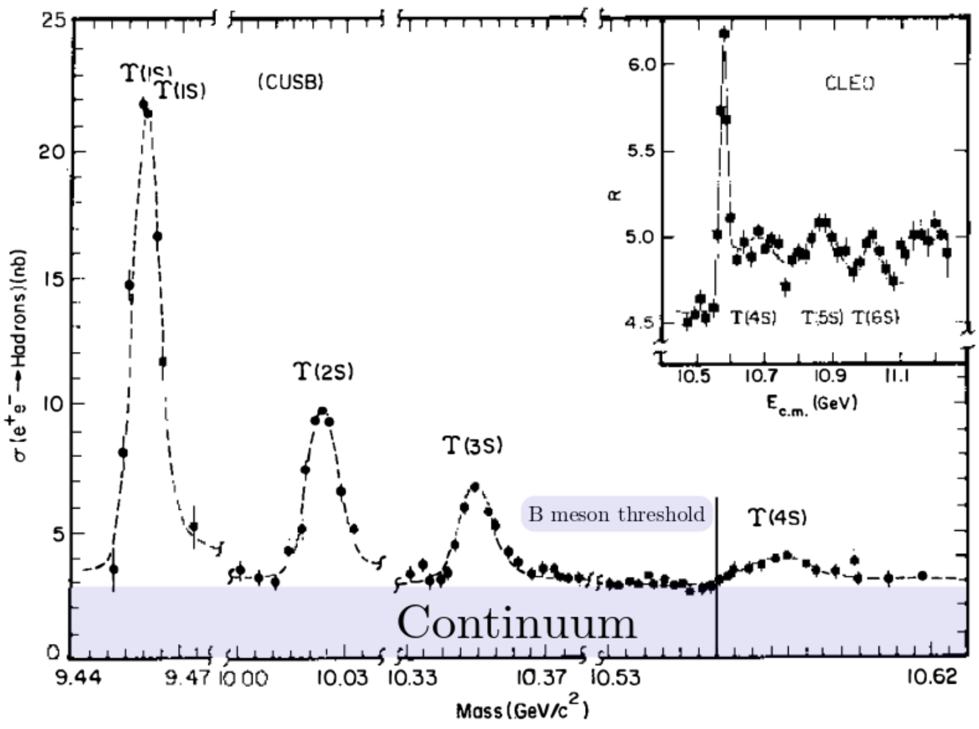


Figure 2.1.: Cross section for hadron production in electron–position collisions with respect to the center-of-mass energy. The energy axis is discontinuous. $B\bar{B}$ pair production occurs above the marked threshold. Taken from [11] which is based on [12].

For each event, beam background and background from other sources have to be discriminated from the particle candidates of interest. The impact of different background sources depends on the detector parts and distances from the interaction point. For example the KLM is affected by radiative Bhabha scattering which induces a neutron background [16]. Another high background source are two photon processes, a main component of the so-called QED background. Additional sources are the synchrotron radiation (mainly from HER), beam-gas scattering and Touschek (intra-bunch) scattering.This has a significant influence on the detector design, as can be seen in the following section.

2.2. The Belle and Belle II Detector

The design of the Belle and the Belle II spectrometer is based on similar principles. Some modifications of the Belle II detector compared to its predecessor are due to new concepts for improving the quality of measurement. Others are necessary to cope with the higher radiation intensity and occupancies, which are caused by the increased beam energies.

Both detectors consist of several sub-detectors and will be described more in detail. A short overview of the components from the inside to the outside is presented below, a schematic is provided in Fig. 2.2.

The main parts of Belle II detector are the Silicon Pixel Detector (PXD), the Silicon Vertex Detector (SVD), the Central Drift Chamber (CDC), the Aerogel Ring-Imaging Cherenkov detector (ARICH), the Electromagnetic Calorimeter (ECL) and the K_L and Muon detector (KLM).
In the Belle detector, similar detector components are used, although it does not contain a PXD. For particle identification, not an ARICH but an Aerogel Cherenkov Counter (ACC), Time-of-Flight (TOF) system and Extreme Forward Calorimeter (EFC) is used.

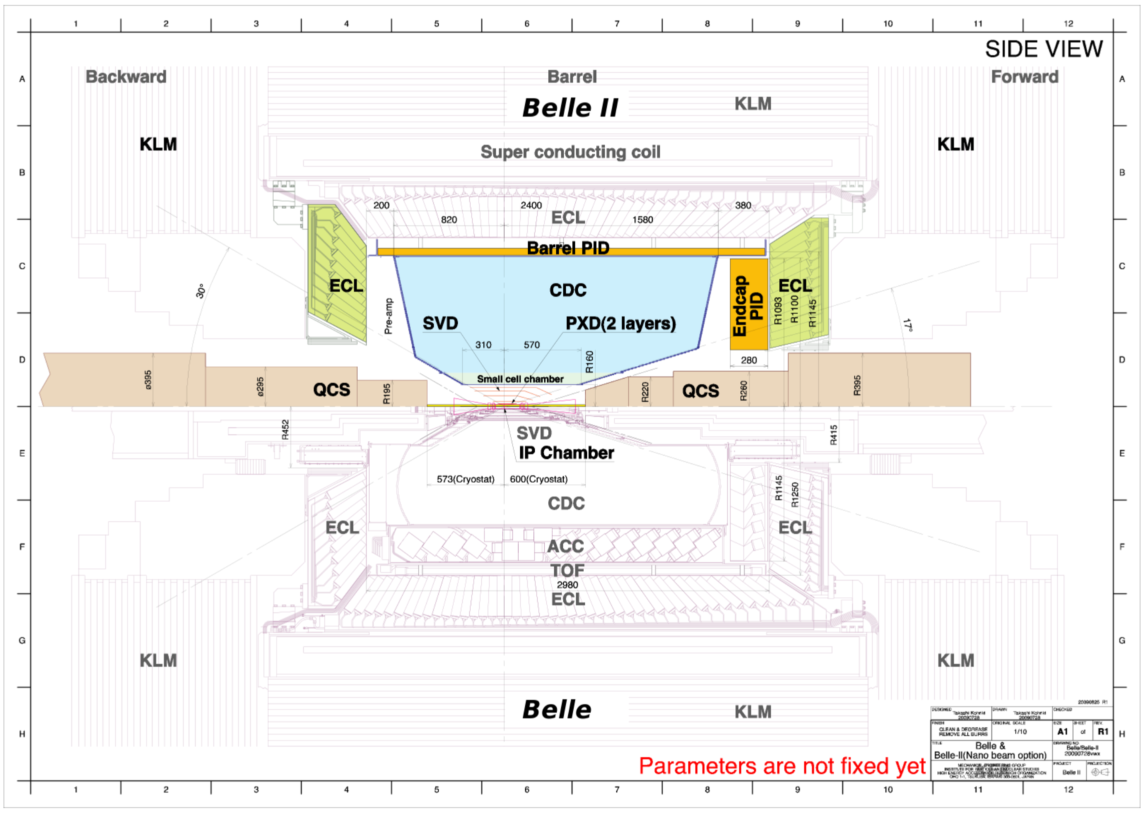


Figure 2.2.: Schematic side view of the Belle II detector (upper half) and the Belle detector (lower half). Taken from [9].

As a result of the enhanced luminosity the described QED backgrounds are expected to be increased up to a factor of 40 [9], which has an influence on the detector design. Furthermore, a better detector hermeticity compared to the Belle detector is targeted [14].

2.2.1. Beam Pipe

The beam pipe has a beryllium coating and is cooled with liquid paraffin. It has a not vanishing crossing angle of $2\phi = 41.5$ mrad, which was increased due to the nano beam scheme compared to the Belle experiment.

2.2.2. Magnetic Field

A magnetic field with the strength of approximately 1.5 T is used to determine the momentum and charge of charged particles. The homogeneity of the magnetic field is achieved with an iron yoke.

2.2.3. Silicon Pixel Detector

In contrast to the Belle experiment, a two-layer silicon pixel detector (PXD) is introduced which is based on the DEPFET (DEPleted Field Effect Transistor) technology [9]. Both layers are combinations of singular modules which are arranged around the beam axis. In the first layer 8 modules, in the second layer 12 modules are used. Due to the upgrade, high rates on beam-induced and QED background are expected at the innermost detector layers. As a result, a silicon strip detectors cannot be deployed here. During a readout interval of 20 μs, an occupancy in the order of 1% is expected. Comparing this to the number of hits related to the particles of interest, shows that this is actually a high amount.

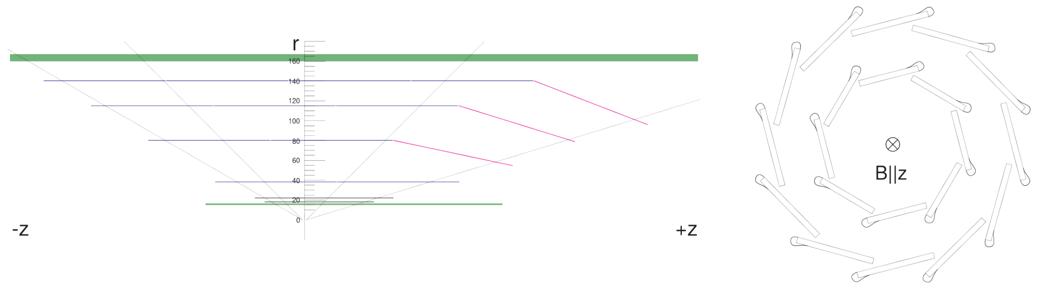


Figure 2.3.: Schematic side view of the PXD and SVD (left) and of a cross section of the readout concept for the SVD (right). Taken from [9].

2.2.4. Silicon Vertex Detector

The Silicon Vertex Detector (SVD) consists of a four layer structure located around the PXD. Each layer is built of silicon strip sensors with a double sided readout. The readout on the n-side is orientated along the $r - \phi$-plane (referred to as short strips) while on the p-side along the z-plane (referred to as long strips). This leads to a different resolution for each direction respectively. The sensors are arranged in a "Windmill" like structure, so that the readout chips on the sensors are covered from the direction of the beam axis by another sensor (Fig. 2.3).

The sensors in forward region are slanted to reduce the material, which is traversed by charged particles originating from the interaction point. This decreases the negative effects on resolution for the outer detector parts. The SVD covers (and surpasses) the full Belle II angular acceptance 17-150° [9, p.139, p.142].

The SVD of Belle was upgraded from 3 to 4 layers during the data taking of the Belle experiment. At Belle II the positioning of the super layers is altered with respect to setup of the predecessor experiment, to allow the positioning of the PXD.

The read-out of the SVD can be used to find tracks down to a very low momentum (less than 100 MeV/c). It allows to gain more detailed vertex information in track reconstruction and for particle identification (PID) information. Together with the PXD it is named as Vertex Detector (VXD).

2.2.5. Central Drift Chamber

The Central Drift Chamber (CDC) is a cell-wire structure, filled with a Helium-Methane mixture. The main differences to its predecessor at Belle are the size, wire structure and improved electronic readout system. The wires are arranged in two different shapes, parallel to the beam axis (axial layer) or slightly tilted (stereo layer). The torsion can be performed in two directions (U, V). In total 56 layers are grouped into 9 superlayers. The CDC of Belle II can cope with an increased beam background of a factor 20 compared to Belle, although less is expected. The CDC is asymmetric and covers the default experiments angular acceptance [17-150°]. Information from the CDC is used for track reconstruction, particle identification (PID) and trigger decisions for charged particles.

2.2.6. Electromagnetic Calorimeter

The Electromagnetic Calorimeter (ECL) is used for the detection of photons, determination of their energy energy deposition, angular information and for the electron identification. It also provides trigger information and is used for luminosity measurements. The sensitive material consists of 6624 CsI(TI) scintillation crystals in the barrel-region and 2112 CsI crystals in

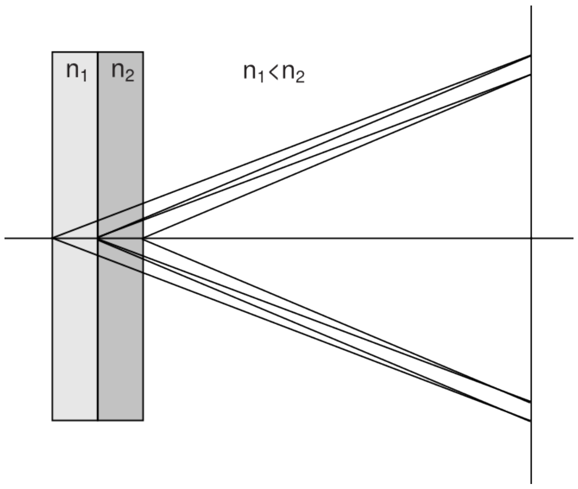


Figure 2.4.: Schematic view of the proximity focusing Ring-Imaging Cherenkov detector. The refractive indices n_1 and n_2 of the radiator materials are different. Taken from [9].

the end-cap-region which have a reduced scintillation decay time but a smaller light output. Readout is performed with two photodiodes per crystal on the rear surface. Its angular coverage of 12.4 - 155.1° is in agreement with the Belle II detector hermeticity demands. Parts of the Belle ECL are reused in Belle II.

2.2.7. Particle Identification

In the barrel and end-cap regions, systems for particle identification are installed. Their main purpose is to provide information for the discrimination between kaons and pions. The **Time-Of-Propagation counter** (TOP) is located in the barrel region. 16 arrays of quartz bars are arranged circularly at the outer wall of the CDC. Charged particles which move faster than speed of light of the medium cause a cone of Cherenkov light inside the material. It is guided via total reflection to photomultiplier tubes (PMT) located at the end of the bars. The two dimensional detector information and the timing information is used for reconstructing the Cherenkov image and particle discrimination, which is correlating with the particle velocity. The proximity-focusing **Aerogel Ring-Imaging Cherenkov detector** (ARICH) is located in the forward end-cap region. It is composed of a radiator, an expansion volume and an array of position sensitive photon detectors. To reduce the emission point uncertainty and therefore reduce the spread of the Cherenkov rings a two layer aerogel radiator with different refraction indices is used. A schematic is shown in Fig. 2.4.

For the Belle experiment an **aerogel Cherenkov counter** (ACC) and a **time-of-flight** detector (TOF) was employed for particle identification, both located in the barrel region. The former used an aeorgel as active Cherenkov material. The latter provides time measurements for slower particles.

2.2.8. $\mathbf{K_L}$ and Muon Detector

The **$\mathbf{K_L}$ and muon** detector (KLM) is built of alternating sensor layers and iron plates. K_L interact with the iron plates and induce showers, which can be detected in glass-electrode resistive plate chambers (RPC). Between two glas-electrodes, separated by noryl and epoxid, a high voltage is distributed. Charged particles cause an ionization of the gas mixture, which can be measured. As the penetration depth and shower shape of kaons and muons are very characteristic, they permit a good discrimination between both particles. For the muon reconstruction, tracks determined from other detector parts and extrapolated to the KLM. Similar to the ECL, parts of the Belle KLM are reused in Belle II.

2.3. Tracking at Belle II

Tracking quality has a huge influence on the discriminating variables used for the study in this thesis. Tracking can be separated into two parts, the track finding and the track fitting. For further information about tracking at Belle II, see [17, 18]. Track finding attempts to assign detector responses, so-called hits, to their provoking particles. At the Belle II experiment, different approaches are used, based on a global track recognition and local hit tracing. The basic assumption of the global track finder is a helical trajectory of a track which is caused by a charged particle in a constant magnetic field. The helix can be described by five independent parameters. A convention for the reference point is the point of closest approach (POCA) to the interaction point. The final track reconstruction combines information of multiple detectors.
Track fitting is defined as the process of fitting a track model to the assigned hits of a track, in order to determine the helix parameters. To consider momentum loss of a particle, interacting with matter inside the detector, the assumption of constant helix parameters along the track is being dropped. At Belle II the fitting procedure includes a Kalman filter, which is implemented in the `GenFit` toolkit [19]. A higher number of correctly assigned hits can increase the quality of the fit.

2.4. The Belle II Analysis Software Framework

The Belle II Analysis Software Framework (BASF2) is used for online (during the process of data taking) as well as for offline analysis [20]. Although it benefits from the experience acquired with the Belle software framework (BASF), the Belle II software framework was completely rewritten and provides different interfaces and more automated approaches than its predecessor. While most of the calculation intensive parts are written in C++, C and Fortran, a python interface is implemented via the boost C++ libraries. This enables the user to perform major parts of his analysis in a python script referred to as *steering file* [20, p.4] with access to common tools needed for analysis.
The framework has a modular structure, where the modules can be arranged in a linear container called path. The data store is designed to grant simultaneously access for different modules. It is based on ROOT tables [21], python modules can access it via the PyROOT interface. The data, which will be available on analysis level, is stored in a format called mini Data Summary Tables (mDST) and contains not raw detector information but already processed information like the helix parameters, at the perigee of the track. In addition combined likelihood ratios for particle identification from different detector parts are also available in the mDST format.
BASF2 provides training of multivariate classifiers within and without the steering file level. The most common way is an interface to the TMVA package [22], a rich toolkit for multivariate data analysis. It also provides an automated approach for the hierarchical reconstruction of B mesons [23]. For physics analysis and testing purpose Monte Carlo can be generated with the integrated EvtGen [24] interface.
Although the framework is still under heavy development, studies on converted Belle data are in preparation.

2.5. Processing Belle Legacy Data

In order to benefit from improvements of the Belle II software or to apply the flavor tagging approach introduced in this thesis on data taken by the previous experiment, the conversion of Belle legacy data has to be possible. BASF2 provides the conversion package `b2bii`, with which Belle objects are partially converted into the BASF2 format. More details can be found in [25, p.119ff]. Particle identification is partially converted to log-likelihood objects of

BASF2, although the ACC information is mapped to the ARICH and TOF to TOP since those detectors are replaced in the Belle II experiment. As part of this thesis, the conversion of SVD hit patterns, together with the number of hits in the CDC and the SVD was added. On Monte Carlo, the Belle II electron ID and kaon ID show a much better background rejection versus signal efficiency power in classification trainings than their Belle correspondents [25, p.122].

Furthermore the conversion of Belle beam parameters, which are loaded from an online database, is integrated. During the conversion of Monte Carlo, several filters (cuts) are applied to coincide with filters which were applied to the measured data.

3. Flavor Tagging

This chapter briefly introduces the relevant parts of the Standard Model for this thesis. Flavor tagging, which is described in Section 3.4, is an essential ingredient in the measurement of the charge parity violation (CPV) in B meson systems.

3.1. Flavor Physics

The standard model Lagrangian obeys a $SU(3)_C \times SU(2)_L \times U(1)_Y$ gauge symmetry, where the latter part is spontaneously broken as described by the Higgs mechanism resulting in a $SU(3)_C \times U(1)_{EM}$ symmetry. Lepton and quark fields couple to the Higgs field via the Yukawa coupling. A transformation between mass and flavor eigenstate of the lepton fields result in a matrix structure of the effective coupling to a (massive) W bosons. This transformation is described by a 3x3 complex and unitary matrix, the Cabibbo-Kobyashi-Maskawa (CKM) matrix

$$V_{CKM} = \begin{pmatrix} V_{ud} & V_{us} & V_{ub} \\ V_{cd} & V_{cs} & V_{cb} \\ V_{td} & V_{ts} & V_{tb} \end{pmatrix}. \tag{3.1}$$

Unitary triangles, which are used to visualize measurements, follow from unitary conditions of the CKM matrix. The most frequently examined unitary triangle arises from multiplying the first and the third column. One of its major benefits is the clear difference of the angles from zero and the fact that its side lengths are of the same order of magnitude

$$V_{ud}V_{ub}^* + V_{cd}V_{cb}^* + V_{td}V_{tb}^* = 0. \tag{3.2}$$

The CKM matrix can be parametrized by 4 parameters: 3 angles (θ_{12}, θ_{13}, θ_{23}) and one complex phase $e^{i\delta}$, where the complex phase is the relevant part for measuring CPV. The Wolfenstein parametrization [26] is commonly used as a way to parametrize the CKM matrix since it shows a clear hierarchy of quark mixing. Hereby $|V_{us}|$, which also can be expressed as the sine of the cabbibo mixing angle θ_{12}, is substituted with

$$\sin(\theta_{12}) = |V_{us}| = \lambda \tag{3.3}$$

and the CKM is expanded in λ

$$V_{CKM} = \begin{pmatrix} 1-\lambda^2/2 & \lambda & A\lambda^3(\rho - i\eta) \\ -\lambda & 1-\lambda^2/2 & A\lambda^2 \\ A\lambda^3(1-\rho-i\eta) & -A\lambda^2 & 1 \end{pmatrix} + \mathcal{O}(\lambda^4). \tag{3.4}$$

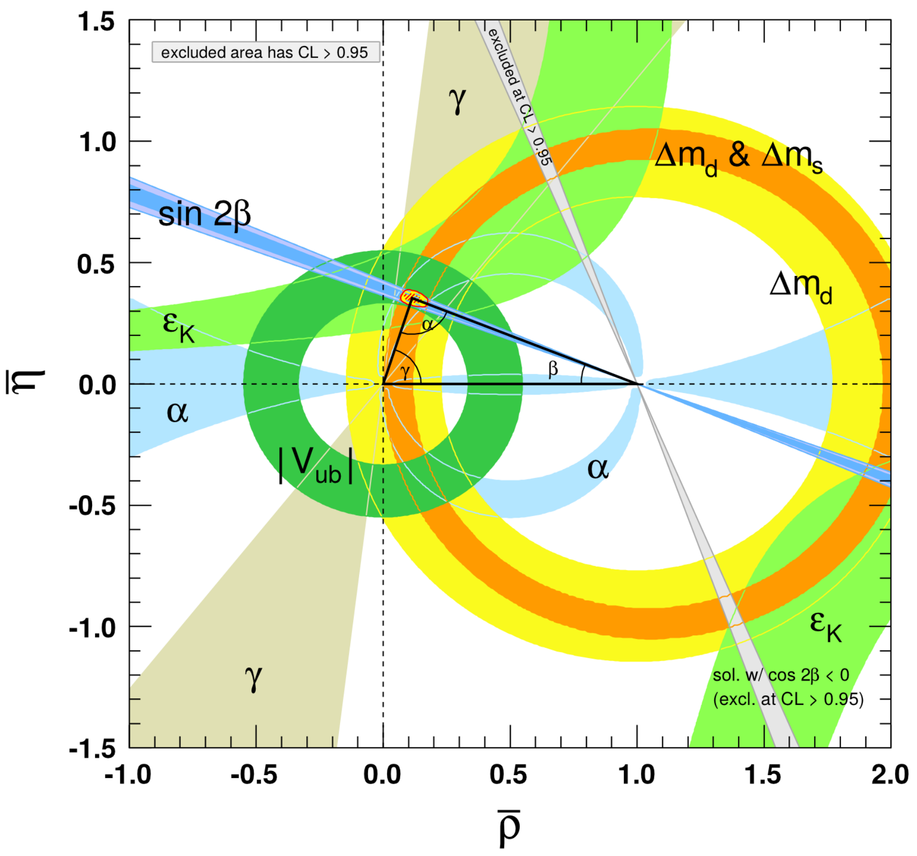


Figure 3.1.: Constraints and global fit results in the $\bar{\rho}$-$\bar{\eta}$-plane. The parameters $\bar{\rho}+i\bar{\eta} = -(V_{ud}V_{ub})/(V_{cd}V_{cb})$ can be expressed in terms of λ, ρ and η: $\bar{\rho} = \rho(1-\lambda^2/2+...)$ and $\bar{\eta} = \eta(1-\lambda^2/2+...)$. Taken from [28].

Here A, ρ and η are real parameters [27], introduced to preserve unitarity, and are defined by

$$\sin(\theta_{23}) = A\lambda^2 \qquad \sin(\theta_{13})e^{-i\delta} = A\lambda^3(\rho - i\eta). \tag{3.5}$$

It is notable that this structure is not derived by an overlaying theoretical framework but determined by measurements for some of the 19 free parameters of the standard model. The value of the perturbation parameter is approximately $\lambda \approx 0.22$.

In the SM there are three generation of quarks. Each generation contains an up type quark (up, charm, top) and a down type quark (down, strange, bottom). Transitions between both types are allowed via coupling to a charged W boson. The branching fraction of the transition is governed by the corresponding matrix elements of the CKM matrix. Transitions between generations are allowed as well, but are suppressed.

Due to the color charge of gluons and gluon self interaction, quarks only appear in bound states. This phenomena is called confinement. A pair formed by a quark and an anti-quark is called a meson. Mesons which contain a bottom (b) quark are referred to as B mesons. B mesons with an up quark carry the charge ± 1 while B^0 mesons, which carry a charm or a down quark are neutral. In this thesis the object of interest is the flavor of neutral B_d^0 mesons with down quark and are abbreviated in the following with B^0. Its anti-particle, the $\bar{B}^0$ meson, contains a $\bar{b}$ and a down (d) quark.

3.2. Charge Parity Violation and B^0 Meson Mixing

While fields in the SM are invariant under the combined transformation of charge, parity and time (CPT), the CP symmetry is violated. This is incorporated in the SM as an irreducible complex phase, in the CKM also referred to as the weak phase. CP violation in the B meson system was measured in 2001 by the Belle and the BarBar experiments and led to the Nobel Price of M. Kobayashi and T. Maskawa in 2008. For the measurement, $B \to J/\Psi K_S^0$ decays were investigated. The subsequent introduction into B meson mixing follows the argumentation in [29]. The neutral B meson sector is attractive for measurements of CPV since B^0 meson pairs are in an entangled state with their corresponding antiparticle (B meson mixing) which is defined by CP transformation.
While regarding time evolution into a mixed state, an effective Hamiltonian can be constructed via the Wigner-Weisskopf approximation which describes a two state system. This Hamiltonian does not obey probability conservation since the decay products are ignored (not Hermitian) but can be divided into a real part, which represents the mass matrix, and into an imaginary part, which represents the decay matrix. Diagonalization leads to two different states, a light and a heavy B meson (B_L and B_H) which also have different lifetimes. When observing the decay into an CP eigenstate $f = f_{CP}$, the asymmetry simplifies from

$$A_f(t) = \frac{\Gamma(\bar{B}^0(t) \to f) - \Gamma(B^0(t) \to f)}{\Gamma(\bar{B}^0(t) \to f) + \Gamma(B^0(t) \to f)} \tag{3.6}$$

to

$$A_f(t) = S_f \sin(\Delta m \ t) \tag{3.7}$$

in $J/\Psi K_S^0$ decays, which allows measurements of the parameter β in the CKM unitary triangle, since there are no hadronic terms involved. The decay coefficient S_f therefore only depends on the weak phase.

3.3. Principles of Flavor Tagging

Decays of the $\Upsilon(4S)$ resonance are of main interest at Belle and Belle II. The resonance decays roughly in equal parts into neutral or charged B meson pairs with a total branching fraction of approximately 0.96 [13].
The final state of the decay of the meson can be flavor unspecific, but the knowledge of the flavor of signal B meson (B_{sig}) is mandatory for many analysis [30, p. 100ff]. Fortunately, the neutral B mesons are in an entangled state (Section 3.2). As a consequence, the flavor of a B meson at the time of its decay determines the flavor of the accompanying B meson (B_{tag}). The determination of its flavor via the final state particles of its decay is referred to as *flavor tagging*. A full reconstruction of the complete event is only possible for a small fraction of all events [23]. Therefore an inclusive approach is used for flavor tagging. In a flavor specific final state of the tag side, particle attributes, as described in Section 3.4, provide sufficient information for the flavor determination.

For measurements of the time-dependent CPV with the signal decay $J/\Psi K_s^0$, the signal side is accessible from both B flavors and flavor tagging has to be employed. Measurement of the distance between the decay vertices of both B meson candidates allows the determination of the decay time difference between the B mesons

$$\Delta t = \frac{\Delta z}{\beta_\gamma c}. \tag{3.8}$$

The flavor tagging efficiency has an direct impact on the error of measurements of the CP asymmetry (Section 5.5.2).

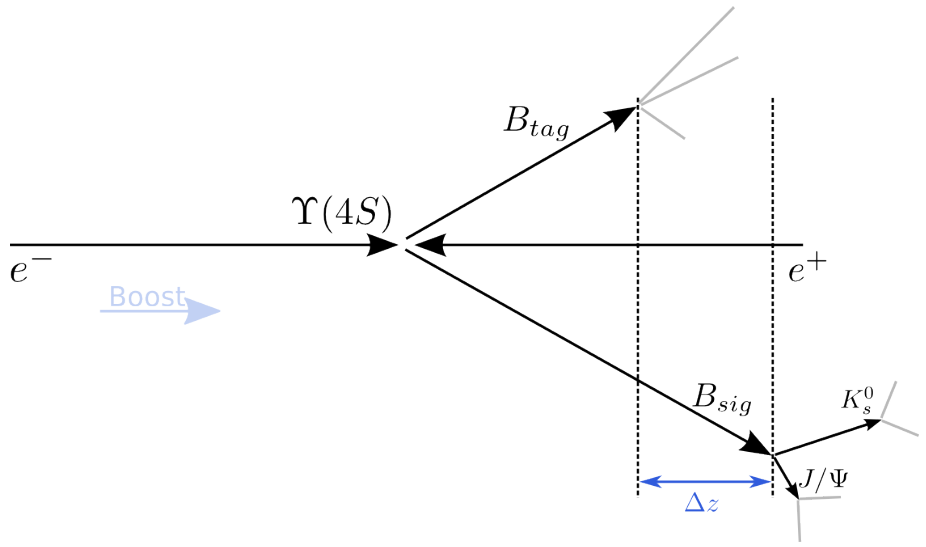


Figure 3.2.: Scheme of an typical $\Upsilon(4S)$ event at an asymmetric lepton collider.

3.4. Flavor Tagging at Belle II

The aim of this thesis is to investigate a new approach to flavor tagging, utilizing deep neural networks. Before describing the deep neural network tagger, the current, category-based method [31, 32] will be reviewed, as understanding the creation of categories is also useful for variable selection and network architecture selections of the new approach.

The current approach divides the tagging process into different sub-classifications, which are combined by a final classifier. A scheme of the approach is shown in Fig. 3.3. Furthermore, a special treatment for hadronic B decays in including the Full Event Interpretation [23] was tested, but this is only applicable for a small subset of possible B decays.

At *track level* several classifiers are used to assign each track to a specific category.
In the **lepton category**, leptons directly originating from the decay of a B meson (*primary leptons*) are distinguished from secondary leptons which originate from intermediate decay products. Since primary leptons are produced by a decaying W boson, the charge provides immediate information of the decaying B meson, see Eq. (3.9). Here X denotes a hadronic particle which contains a c quark (and may also decay)

$$B^0 \to X^- l^+ \nu_l. \tag{3.9}$$

A positive charge of a primary lepton implies a B^0.
The charge of a *secondary lepton*, in contrast, implies directly the opposite flavor of the B meson

$$b \to W^- c \qquad c \to s l^+ \nu_l. \tag{3.10}$$

Secondary leptons can be separated by momentum and angular distribution of the decay products (angle between missing momentum and lepton).
In analogy of the lepton category a discrimination between slow and fast **pions** is performed, which also indicates a direct or intermediate decay. Slow pions, that are produced in D^* to D meson decays are produced almost at rest in the center-of-mass system (CMS) of its mother particle. Pions are the most frequent final state particles in B meson decays.

The most reliable source is probably the **kaon category** [30, p.101]. There are two kaon sub-classifiers, one focusing on the $b \to c \to s$ transitions, the other on $s\bar{s}$ quark pair popping and $b \to c\bar{c}(d, s)$ decays. The flavor of the former decay is directly determined by the flavor of the charged kaon. A likely decay product for the latter decay is a K_s^0 meson, containing a strange quark from these decays.

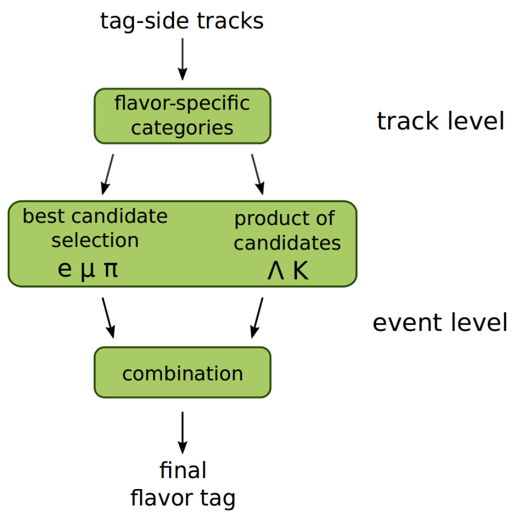


Figure 3.3.: Mechanism of the flavor tagging algorithm as implemented in BASF2. Taken from [31].

Another category using $b \to c \to s$ decays, is the classification of **lambda** baryons. Even thought this decay has only a small branching fraction, it is an essential part of the tagging algorithm. Lambda baryons have to be reconstructed from protons and pions, so a candidate selection has to be performed to fulfill a defined level of reconstruction quality.

At the event level, the different categorized tracks are assigned via multivariate classification to different event categories, which consider a combination of multiple tracks.

Correlation between those subcategories are exploited for combining them to a final, single discriminator to distinguish between neutral B^0 and $\bar{B}^0$ mesons.

4. Artificial Neural Networks

Artificial neural networks (Section 4.2) are one of the core subjects of this thesis. They are used for multivariate classification (Section 4.1). In this chapter the most relevant aspects for this thesis are pointed out. Specific training techniques and established frameworks for fitting the classifier to a data representation are also presented.

4.1. Multivariate Classification

In a high dimensionality, distances between data-points of objects of the same class increase drastically compared to lower dimensions. In machine learning, this is often referred to as *the curse of dimensionality*. Multivariate classification tries to assign an object, defined by different variables (features), to certain classes (targets). This is intended to reduce the dimensionality of a problem while loosing as little information as possible . Thereby correlations between features and targets are being included and lead to a significant boost compared to cut based procedures.
The free parameters of a multivariate classifier (MVC) have to be adapted to a specific data set (sample of a statistical population). The so-called training can be understood as a construction of separation planes in a multidimensional space. The number of features has a significant impact on adaption and separation power.
Multivariate analysis is successfully applied in many areas like engineering, physics and medicine. As described in Section 3.4, in this thesis the distinguishing of neutral B mesons from Anti-B mesons is studied. This is a binary classification process, the two classes are usually defined as signal and background. A multilayer perceptron, which is introduced in Section 4.2 and described in Section 5.6 is employed.
A common problem is the so-called *bias-variance tradeoff*, which describes the danger of *over-fitting* a model on data with increasing its complexity. This indicates that the model complexity of a multivariate classifier has to be carefully adapted according to the given problem.

4.2. Multilayer Perceptron

Inspired by biological information processing systems, several attempts were made to transfer notions for pattern recognition, inference and extrapolation into a mathematical and statistical framework. One of those attempts was the Rosenblatt Perceptron [33] which is one of the predecessors of various established pattern recognition methods, including the widely used multilayer perceptron (MLP) architecture. While in other fields Support Vector Machines (SVM)

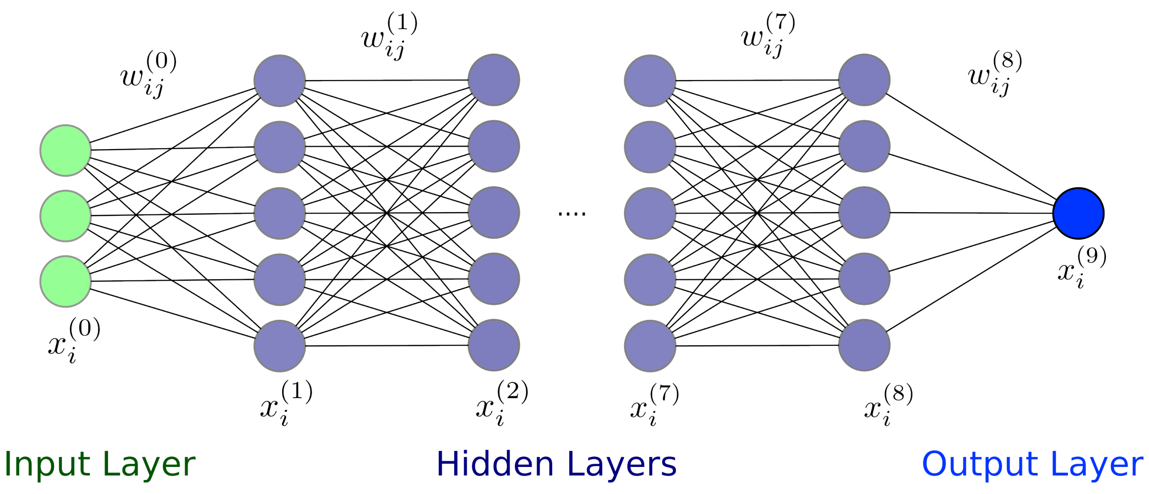


Figure 4.1.: Schematic of a multilayer perceptron

had been very popular, MLPs, which recently regained their popularity in the machine learning sector, have been deployed very successfully in physics [34, 35].
This section follows loosely some aspects in [36, p.225 ff], were a much more detailed description about methods in machine learning and pattern recognition can be found.

4.2.1. Architecture of a Multilayer Perceptron

A MLP contains three different types of layers, an input layer, hidden layers and an output layer. The input layer is defined as an n-dimensional vector $x_i^{(0)}$ of a given feature representation.

The hidden layers and output layer k are represented by the nodes $x_i^{(k)}$ as they are defined in Eq. (4.1). Each node contains a bias w_{i0} and is connected to the preceding nodes with the weights w_{ij}, transformed with the non-linear activation function σ

$$x_i^{(k+1)} = \sigma^{(k+1)} \left(\sum_{j=1} w_{ij}^{(k)} x_j^{(k)} + w_{i0}^{(k)} \right). \tag{4.1}$$

Commonly used activation functions are the sigmoid function

$$\sigma(x) = \frac{1}{1 + \exp(-x)}, \tag{4.2}$$

the Tangens Hyperbolicus function `tanh` and the Rectified Linear Unit

$$r(x) = \max(0, x). \tag{4.3}$$

Since the network output is calculated iteratively from the input nodes to the output nodes, the architecture is called a *feed-forward network*. For a better nomenclature, the last layer of the output for a sample will be defined by $y_n(\vec{x}, \boldsymbol{w})$, where $\vec{x} \in \mathbb{R}^l$ for l features. The weight tensor $\boldsymbol{w}$ is defined by the dimension of the free network parameters. If a multilayer architecture is chosen, where all layers, including the input and the output layer, have the same width d and h hidden layers are deployed, the weight tensors is $\boldsymbol{w} \in \mathbb{R}^d \times \mathbb{R}^{d+1} \times \mathbb{R}^{h+1}$.

4.2.2. Free Parameter Adaption

The convergence of a multivariate classifier training on data is, depending on the algorithm, not necessarily guaranteed by design. For the training process an error function, which compares

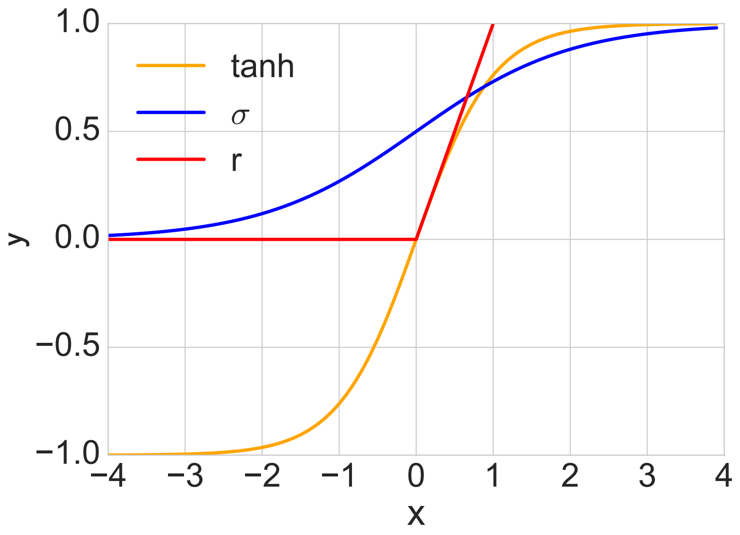


Figure 4.2.: Schematics of different activation functions

the network output to the true value has to be minimized with respect to the weights. Choosing the sigmoid function as non linearity for the output layer for a binary classification problem, with the target (also truth) $t_n \in \{0, 1\}$, the negative log likelihood function is the cross-entropy. With the network output y_n, which is a scalar for binary classification, the error function for N samples is given by

$$E = -\sum_{n=1}^{N} \left[t_n \log y_n + (1 - t_n) \log(1 - y_n)\right]. \tag{4.4}$$

Minimizing this error function corresponds to the calculation of the maxium-log likelihood and allows the interpretation of the network output as a probability. The error function is non convex with the result, that in a multidimensional feature space almost certainly not the global minimum but only local minima are found. It can be shown, that multilayer feed-forward networks can approximate a vast number of different functions [37], but the optimal solutions are usually not found. The adaption of the free parameters on a dataset to approximate the function of interest adequately demands a feasible approach. As described in Section 4.3 a global minimum is not the desired goal to achieve.

A very simple but scalable training algorithm is the *gradient descent*, which uses gradient information of the error function to change the weight parameters according to direction of maximum slope. The magnitude of the change of the weights $\Delta\boldsymbol{w}$ is governed by the parameter η as described in

$$\Delta\boldsymbol{w} = -\eta\nabla E(\boldsymbol{w}). \tag{4.5}$$

Performing the calculation not for the complete data set at once but for each input vector gradually in a random order, can significantly improve convergence of the algorithm. To control statistical fluctuations of the dataset, the weight update can be carried out in mini-batches, the so-called *mini-batch stochastic gradient descent* (SGD).

Furthermore the algorithm can be easily expanded with additional terms, like the *momentum term* (see Eq. (4.6) to prevent oscillations at local minima. Additionally, it helps speeding up while moving in multi-dimensional ravines. The momentum extension uses the weight difference of the previous step

$$\Delta\boldsymbol{w} = -\eta\nabla E(\boldsymbol{w}) + \mu\Delta\boldsymbol{w}_{prev} \tag{4.6}$$

There are more sophisticated training algorithms available that take advantage of the Hessian matrix, or have less hyperparameters to tune, but the SGD is still a widely used method. With the introduction of *error backpropagation* an efficient method to calculate gradient of the error function for the SGD algorithm was provided [38]. It takes advantage of the chain rule to determine the gradients of the hidden layers as described in

$$\frac{\partial E}{\partial w_{ij}^{(k)}} = \frac{\partial E}{\partial a_j^{(k)}} \frac{\partial a_j^{(k)}}{\partial w_{ij}^{(k)}} \qquad a_i^{(k)} = \sum_j w_{ij}^{(k)} x_j^{(k)} \tag{4.7}$$

$$\frac{\partial E}{\partial a_j^{(k)}} = \sigma'^{(k+1)} \left(a_j^{(k)} \right) \sum_i w_{ij}^{(k+1)} \frac{\partial E}{\partial a_i^{(k+1)}}. \tag{4.8}$$

For the output layer, in Eq. (4.8) the gradient of the error function with respect to $a_i^{(k+1)}$ has to be replaced with the gradient with respect to the network output node. Backpropagation can be used to calculate the same expensive computational steps multiple times and reduces the computational complexity to $\mathcal{O}(\dim(\boldsymbol{w}))$ per iteration step. This means that the computational complexity is at linear order to the free parameters and therefore linear to the number of hidden layers. The technique is applicable on various error functions and training algorithms.

4.3. Regularization mechanisms

Besides the problem of not finding a good local minimum, there is also the danger of overfitting the parameters to the training set. There are various methods available and only methods, which are used in this thesis are mentioned in this section. In monitoring the training process on an independent validation dataset, overfitting can be limited with *early stopping*. When the loss function of the validation set does not decrease in a defined interval, the training process will be interrupted. Another method is the introduction of L^2 weight decay. The latter decreases the weights in each iteration as described in Eq. (4.9) with the result, that unused connections between nodes are suppressed

$$E' = E + \frac{\alpha}{2} \sum_{i,j,k} (w_{ij}^{(k)})^2. \tag{4.9}$$

Another method is dropout [39], where the network input for a defined number of random nodes is set to zero. Dropout forces the network to establish different connections and is equivalent of training at multiple networks at once. This technique shows huge improvements on small datasets.

4.4. Deep Neural Networks

A significant boost to machine learning arose from improvements of available hardware for parallel computation. Besides that, the amount of huge labeled datasets increased as well. Jointly, bot developments lead to improved results of already known classification algorithms and formed the foundation for the so-called *deep learning* [8, 40] area of machine learning. Deep learning has the ambition not to use "hand-crafted" high-level features but let the MVC "learn" complex feature representation by itself. The most popular strategies are increasing the depth of the classifier (e.g. the amount of hidden layers of an MLP) or restrict the network architecture to symmetries of the problem. The former results, if the training is successful, in a more abstract feature representation for each network layer. This thesis uses the approach to examine the performance of a deep multilayer perceptron on the classification process of B mesons.

4.4.1. Machine Learning Frameworks

While there is a vast amount of sophisticated tools available for multivariate classification like the TMVA package [22] or Neurobayes [34], a tool which can take advantage of the benefits of GPU parallel computing is preferable. In the main part of thesis the machine learning library Pylearn2 [41] was used, which operates in a python environment and is based on Theano [42]. Theano allows to create symbolic graphs of a mathematical computation and to executing them on the Central Processing Unit (CPU) or Graphics Processing Unit (GPU) via the Compute Unified Device Architecture (CUDA) application programming interface (API) by the company NVIDA. Furthermore Theano allows the symbolic calculation of gradients which makes it a powerful tool for machine learning. Similar functionality is provided by tensorflow [43], which is deployable on a larger scale on multiple machines with multiple CPU and GPU.

5. Deep Neural Network based Flavor Tagger

B meson flavor taggers were improved constantly during the last decades, resulting in a hierarchical, on particle categories based arrangement of classifiers.
In this chapter, a deep neural network as a discriminator is introduced, which shows better classification on Monte Carlo than the category based method (Section 5.6.6). The intention for the deep neural network approach is, that individual categories are not pre-engineered but the corresponding high-level features are learned by the neural network (Section 4.4). The influence of hyperparameters on the results of the MVC is studied. The performance of the MVC is tested on Belle and Belle II Monte Carlo and compared to the established approach.

The process of training and analyzing the artificial neural network can be subdivided into the following steps: At first, Monte Carlo datasets are generated for training, testing and evaluation purpose. Secondly, different input variables, so-called training features have to be selected. These selected variables are preprocessed to a flat distribution before the actual training begins. At last the different models are validated on dedicated test sets. The classifier is compared with the established method on the same test sets.

5.1. Dataset Generation

Studies in this thesis are based on Monte Carlo generated datasets only. For Belle II Monte Carlo generation and detector simulation, the necessary functions are already available in BASF2. The physical processes of a decay chain are simulated with the package EvtGen [24], the detector simulation is performed with the package GEANT4 [44]. The Belle legacy Monte Carlo production was carried out with an older version of EvtGen and GEANT3. This study was part of the first tests for the conversion features and several problems occurred and had to be considered.

The decay chain describes an $\Upsilon(4S)$ resonance decaying into two B mesons, which are decaying into final state particles, as mentioned in Section 3.4. The tag side B meson is decaying via the $b \to c$ decay chain, according to the decay tables published by the particle data group [13], a so-called generic decay. The products may further decay. Normally, the term generic would include also direct decays without transitions to a charm quark, so-called rare decays. Due to their small fraction they are neglected in this thesis.

Since the tag side is of primary interest, for the selected signal side, a B meson decay into two τ neutrinos is chosen. The signal of this so-called *mono-generic* decay does not cause any

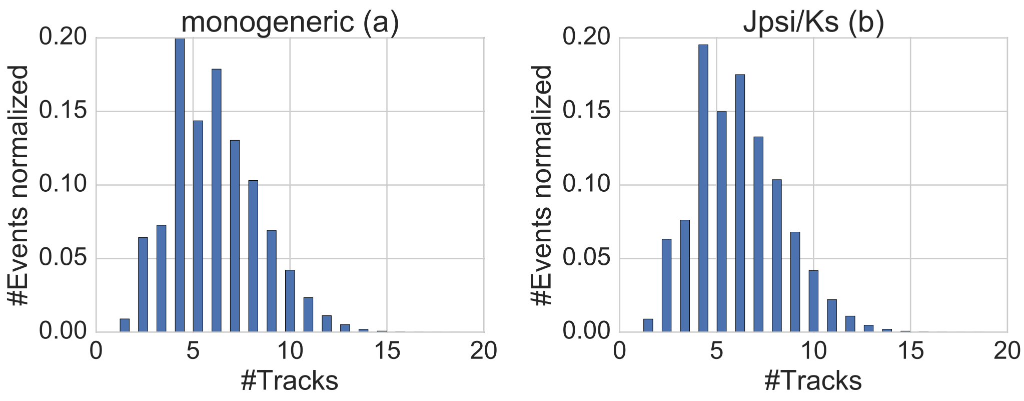


Figure 5.1.: Histogram to show the distribution of tracks for a mono-generic testing (a) set and a testing set with reconstructed $J/\Psi K_s^0$ (b). Mean / standard derivation for: (a) 5.90 / 2.14, (b) 5.89 / 2.40

detector responses. Therefore training and evaluation of the flavor tagger will be independent from the reconstruction efficiency of a specific decay. In this special case, B meson mixing is not included, since this is not necessary for training and testing the flavor tagging algorithm itself.
In a further control sample, the signal side B meson decays into $J/\Psi K_s^0$ with $K_s^0 \to \pi^+\pi^-$ and $J/\Psi \to \mu^+\mu^-$. The tag side decays generically. This allows a comparison with other approaches in literature, since reconstruction efficiency might decrease the MVC performance. Here, B meson mixing is already included in the decay table and allows also a test of the integration of the flavor tagger into the analysis framework. For each signal event, a Monte Carlo matching is performed in order to use only correctly reconstructed signal events for classifier training and evaluation. This can introduce a systematic bias.
The remainder of particle objects in an event after the successful reconstruction of the signal side is called rest of event. Fig. 5.1 shows the distribution of the number of tracks for both signal sides in the rest of event.
Converted Belle Monte Carlo does also include beam background. The beam background was sampled from random trigger data. On the other hand, no measured beam background samples are available for Belle II. Hence no beam background was used in Belle II Monte Carlo generation.

For the **Belle II training datasets**, four datasets with exactly 13 million events were generated and the detector response was simulated. This was performed with a BASF2 version with the revision number 21137. An internal testing and validation set are used to monitor the classifier during the training and to sample the PDFs, required for the transformation of the classifier output to a probability (see Section 5.5.3). Each training dataset was split with a factor 0.92 into an internal training data set and an internal validation and testing set, containing the same amount of events. The internal training set contains approximately 12 million events, the internal testing and internal validation sets approximately 500.000 events respectively.

The **Belle training datasets** include four datasets with 13 million generated events. After passing a skim during the conversion into the BASF2 `mdst` file format, approximately 12.4 million events with no empty `RestOfEvent`, could be successfully reconstructed per training dataset. The sets were divided by the factor 0.92 into an internal training, testing and validation set, as well. Besides that, a training dataset with the signal $J/\Psi K_s^0$ containing 42 million events was generated. After reconstruction and skimming the n-tuple files contained approximately 15.3 million events. The low efficiency is influenced by the limited K_s reconstruction efficiency.

For evaluation, the trained classifiers have to be tested on independent data samples to provide an unbiased comparison between different MVC. These sets differ from the data sets used for internally testing.

5.2. Variable Selection

As described in Section 4.1, choosing the optimal set of variables is crucial for adapting a multivariate classifier to a data representation. The intention is to use a set of features, which are as uncorrelated as possible to each other to gain as much information which as little dimensionality as possible. Variable selection was a process of constant development in this thesis. Here the final selection of all variables is described. The variable selection mainly concentrates on information gained by the reconstructed tracks, which carry most of the information. Linear correlations between those variables are shown in Fig. 5.2. Further improvements will be discussed in the outlook.

Charge

The charge of a track is determined by the curvature of the reconstructed helix, caused by the charged particle traveling through the mostly homogeneous magnetic field. Due to charge conservation, the variable can be an indicator for a missing track or an additionally wrongly reconstructed track. For some particles, it also carries the information about the B meson flavor, as described in Section 3.4. The charge parameter is not provided directly as an input variable but used to impose a certain symmetry on the structure of the input vector. More details are provided in Section 5.4.

Momentum

Kinematic variables of a particle can provide direct flavor information in combination with the charge. As described by the *primary and secondary* particle categories in Section 3.4, high and low momentum decays are an indication, if a track is originating directly from a B meson decay.

The momentum is transformed into the center-of-mass system (CMS) of the $\Upsilon(4S)$ to provide a better representation of the symmetry of the problem. The polar angle ϕ is defined perpendicular to the beam axis, and the azimuth angle θ defined in forward direction with respect to the beam axis. For limited degree, the momentum can be used to infer on common production vertices.

Summarized the

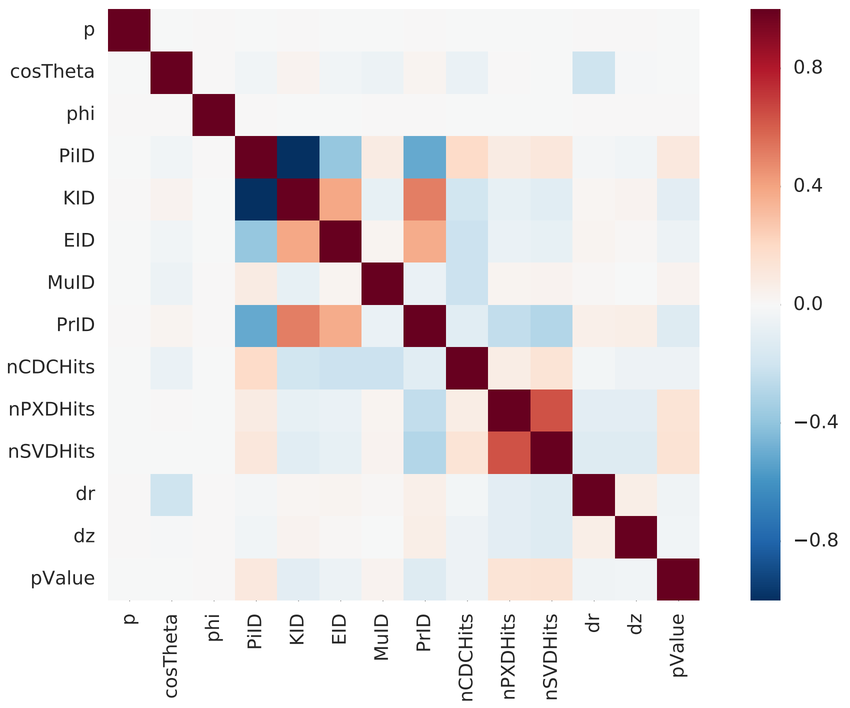
- absolute value of p in CMS,
- cos(θ_p) in CMS,
- ϕ_p in CMS,

are used for the selected particles of the event (Section 5.4).

Particle Identification

As described in Chapter 2, the particle identification (PID) assigns a track to classes of final state particles, using different information from different detector parts. The final result for a PID value is the combined likelihood ratio for a given particle in relation to the pion hypothesis. Since in combination with charge and momentum, the particle type is strongly correlated with the flavor of the B meson, the PID is essential. Knowing the PID is crucial for the reconstruction of mother particles of a subset of final state particles

Following PIDs are explicitly used:



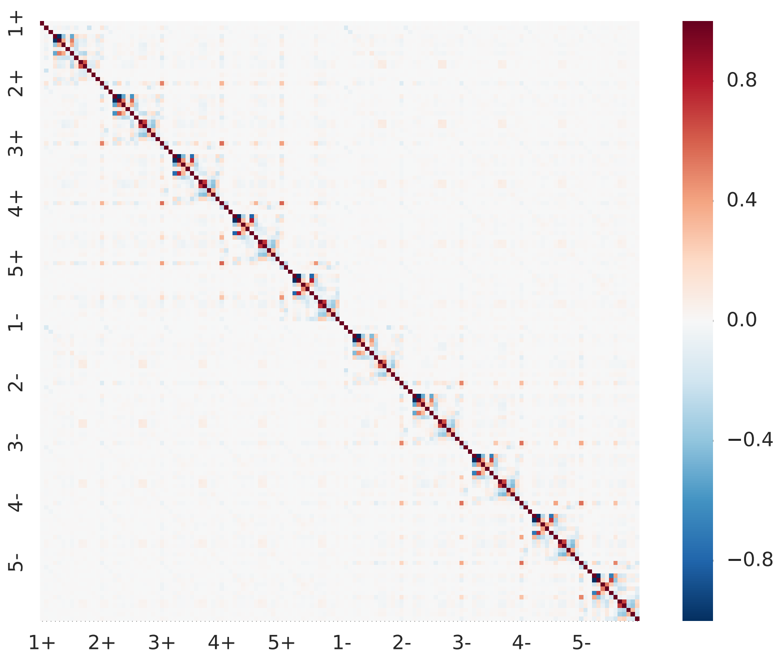


Figure 5.2.: Linear correlations between variables of a track (top) and variables of an event (bottom). The input for an event is described in Section 5.4. A positive sign denotes a positive charged track, a negative sign denotes a negative charged track.

- ElectronID,
- KaonID,
- PionID,
- ProtonID,
- and MuonID.

Number of Hits

As described in Section 2.3, the number of hits has a direct impact on the quality of a track fit. The quality is correlated with the resolution of the track parameters and the PID accuracy. Furthermore, the number of hits, separately for each detector part, might be correlated to the type of a particle and provide hints about the spacial information, e.g. which detector part is reached by the particle. The number of hits is also correlated to the fact, if the track is curling in the detector, which could result in a wrongly reconstructed charge. Following hit counts are used:

- number of hits in the PXD (only Belle II),
- number of hits in the SVD,
- number of hits in the CDC.

Perigee of Tracks

Spacial information provides insight into relations between different tracks. Since only final state particles are detected, knowing the region of their production vertices helps to determine if they originate from primary or intermediate decays and let infer to their mother particles. Furthermore they provide essential information for the reconstruction of mother particles. Unfortunately, the vertices have to be reconstructed under specific hypotheses and recombined to subsets of particle groups. BASF2 would allow such an implementation in general, but this goes beyond the scope of this thesis. Therefore, only the perigee information of a reconstructed track was used, which is also correlated to the production vertex of a particle but far less constraining. However, for particles, which are decaying near the $\Upsilon(4S)$ production vertex, the perigee information might be stronger correlated to their production vertex. Used variables are

- the radial distance of the perigee from the beam axis
- and the distance in z-direction (beam axis) to the interaction point.

This information provides a significant classification quality improvement, as stated in Section 5.6.3.

p Value of the Track Fit

The p value of a track fit is given by the integral over the χ^2 distribution of the corresponding degrees of freedom of the problem. Limits of the integral are chosen from the χ^2_m value of the investigated (fitted) model to infinity. It therefore can be understood as a measure for how successful the fit of a certain model was.

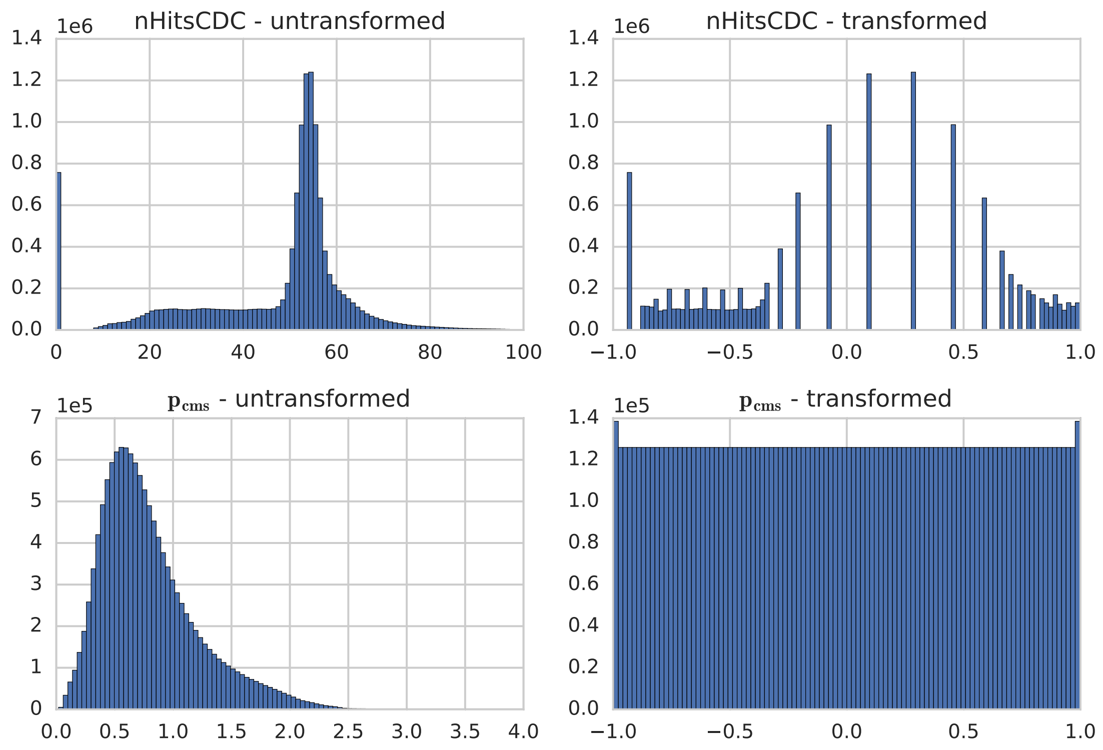


Figure 5.3.: Example of the effect of equal frequency binning. The upper figures show the distribution of the number of CDC hits (discrete) for the highest positively charged momentum particles. The lower figures show the distribution of the momentum in the CMS (continuous) for the highest positively momentum charged particles.

5.3. Input Feature Preprocessing

Regularization and normalization of the input variables can lead to a much better network performance. This is heavily related to the activation functions, used in the first hidden layers. In the vicinity, where a non-linear activation function shows approximately linear behavior, varying input values lead to higher gradients, compared to the high non-linear regime. As a result, a higher sensitivity can be achieved in this region. For the `tanh` function, this is the vicinity around zero.

Normalization can transform most of the values to a desired region, but single outliers still can push the activation functions into the non linear regime. One way to achieve a restriction to a desired interval and furthermore a protection against outliers, is the *equal frequency binning*. Here, the attempt is to transform the input variable into a flat distribution, regulated by the cumulated density function. Since the `tanh` activation functions are used for the hidden layers in this thesis, a uniform distribution in the interval $x_i^{(0)} \in [-1, 1]$ with mean of approximately zero is targeted.
The transformation can be obtained by the following steps. Each bin of a histogram should contain the same amount of entries. An exception is made for a discrete spectrum. If a specific value occurs more often than the targeted number of entries per bin. Then, a larger bin is constructed and the average value of that bin is used in the transfered distribution.
As part of this thesis, this was implemented in the `dft` python-package, containing the necessary functions for training the MVC within and without BASF2. An example of the transformation is given in Fig. 5.3. The flattened distribution can be transformed into a normal distribution, subsequently.

5.4. Feature Representation

Finding a suitable feature representation is key for the success of a multivariate classification problem. If symmetries of the problem are known, they can be put in "by hand" to reduce the complexity of the problem. While the over-engineering of features can result in a loss of useful correlations between input variables, using the raw information increases dimension of the input space significantly and increases the training difficulty of the MVC. In this thesis, not the raw output of sensors of the detector was used, but attributes of the reconstructed `track` objects. Besides that, BASF2 provides also `ECL cluster` and `KLM cluster` objects, which could be potentially used in an extended tagging approach.

One of the major problems of employing a neural network to flavor tagging is the fixed number of input nodes. Since the number of tracks is not equal for each event, the input has to be transformed. It would be beneficial if empty tracks would be separable by the algorithm. Fig. 5.1 shows the distribution of tracks on the tag side for an simulated generic decay of an B^0 meson. A possible solution could be the assignment of a track to a particle type specific input vector. Without increasing the input space significantly, this would not be possible.

Since the decay process obeys charge conservation, an even number of charged particles is produced. Because of errors in recombination and reconstruction, and as well due to the limited detector acceptance, the assumption of an even number of `track` objects is violated. Nevertheless, considering this symmetry could help to decide, if all tracks are found or reconstructed correctly.

Fig. 5.4 shows the feature representation, which was chosen in this thesis. Each track is ranked by its momentum. This is correlated with the attribute of primary and secondary particles and provides a defined order of the input vector. A total number of 10 charged tracks at most is used for each rest of event. This covers roughly 92% of all occurring tracks. For roughly 96% events, no cuts are performed. For events with less than 5 positive or 5 negative tracks, the

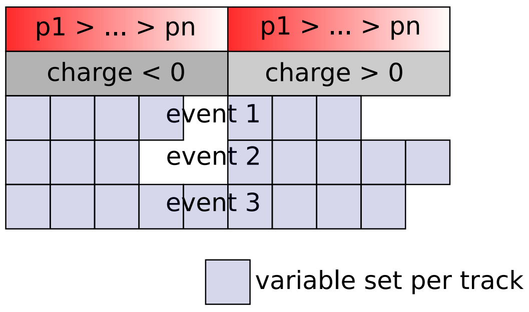


Figure 5.4.: Schematics of the feature representation of the input variables. Each variable set is sampled from a `track` object and contains the variables described in Section 5.2

features of the empty variable sets are set to zero. This value corresponds to the mean of the (uniform) distribution.

Another way to deal with an variable input space can be the introduction of convolutional neural networks which demand spacial symmetries (invariance of patterns against translations or rotations). First tests in the scope of this thesis did indicate that the approach might succeed, but would require further optimization.

5.5. Testing Metrics

In this section, the common metrics used for comparisons in this thesis, are introduced. The Area under the Receiver Operation Curve (AUC) is mainly used in this thesis for benchmarking purpose of the classifier performance, e.g. for different variable sets or hyperparameters. Additionally, it is practical to introduce quantities that directly show the expected impact of the algorithm on measurements. The effective tagging efficiency allows a direct inference of the uncertainty of measurements of the CP asymmetry.

5.5.1. Area under the Receiver Operating Characteristic Curve

The receiver operating characteristics (ROC) curve can be used to determine the performance of a classifier for an arbitrary working point, defined by two quality measures. In this thesis, these two quantities are chosen as the true positive rate (TPR) and the false positive rate (FPR).

They can be expressed with the help of probability density functions (PDF) f, which denote the probability that a random variable $\vec{x}$ takes a value inside a given interval A, with

$$P(\vec{x} \in A) = \int\limits_A f(\vec{x})\ \mathrm{d}\vec{x}. \tag{5.1}$$

The PDF is normalized and always positive. The output of a MVC $y_n \in [0,1]$ can be interpreted as a test-statistic. With the signal hypothesis H_1 and background hypothesis H_0, the PDFs for the test-statistic can be defined as $t_0(y_n)$ and $t_1(y_n)$. For a given threshold parameter T_c, the following cumulative density functions (CDF) P_{TP} (P_{FP}), which define the TPR (FPR), can be obtained

$$P_{TP}(T_c) = \int_{T_c}^{\infty} t_0(y_n)\ \mathrm{d}y_n \tag{5.2}$$

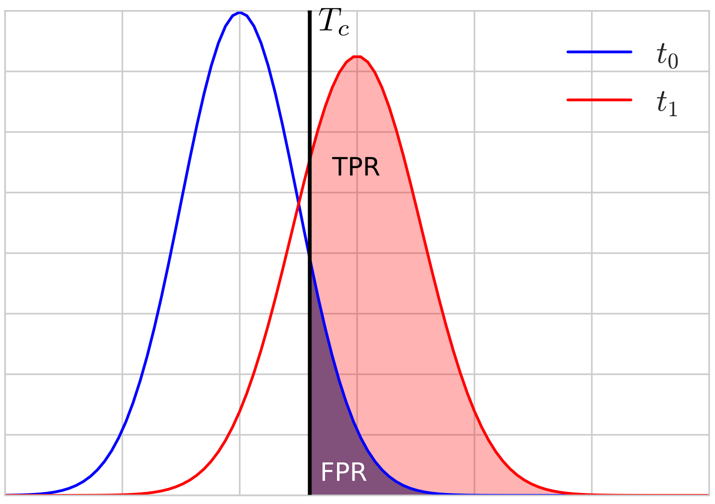


Figure 5.5.: Example for the true positive rate and false positive rate for the two PDFs t_0, t_1 for an arbitrary T_c.

$$P_{FP}(T_c) = \int_{T_c}^{\infty} t_1(y_n)\ \mathrm{d}y_n. \tag{5.3}$$

An example is shown in Fig. 5.5. The area under the ROC curve (AUC), θ, is used as a testing metric, given by

$$\theta = \int_{-\infty}^{\infty} P_{TP}(T_c) P'_{FP}(T_c) \mathrm{d}T_c. \tag{5.4}$$

A value $\theta = 0.5$ would stand for a random decision of an MVC, values smaller than $\theta = 0.5$ can be inverted $(1 - \theta)$ to correspond to values above the threshold.

For estimation of the standard error of the AUC, an approximation of the standard error of the Wilcoxon statistic SE(W), as stated in [45] was used. N_S is defined as the number of true signal events, and N_B as the number of true background events. It is given as

$$\mathrm{SE(W)} = \sqrt{\frac{\theta(1-\theta) + (N_S - 1)(Q_1 - \theta^2) + (N_B - 1)(Q_2 - \theta^2)}{N_S N_B}} \tag{5.5}$$

with

$$Q_1 = \frac{\theta}{2-\theta} \qquad Q_2 = \frac{2\theta^2}{1+\theta}. \tag{5.6}$$

In Fig. 5.6 a comparison between the category based method with the deep neural network approach is shown. For an arbitrary working point, the deep neural network performs better than the category based approach.

5.5.2. Effective Tagging Efficiency

It is useful to find a figure of merit, that allows to directly infer on the impact of the performance of a MVC on a measurement. The purpose of a flavor tagger is to classify the reconstructed B mesons, here defined as signal (B^0) or background ($\bar{B}^0$). It is mainly used for measurements of the CP asymmetry. For this case the *effective tagging efficiency* ϵ_{eff}, as it is described in [32, 30, p.100ff], is a useful metric. The number of reconstructed mesons classified as B^0 ($\bar{B}^0$) is defined as $N_{s,tag}$ ($N_{b,tag}$).

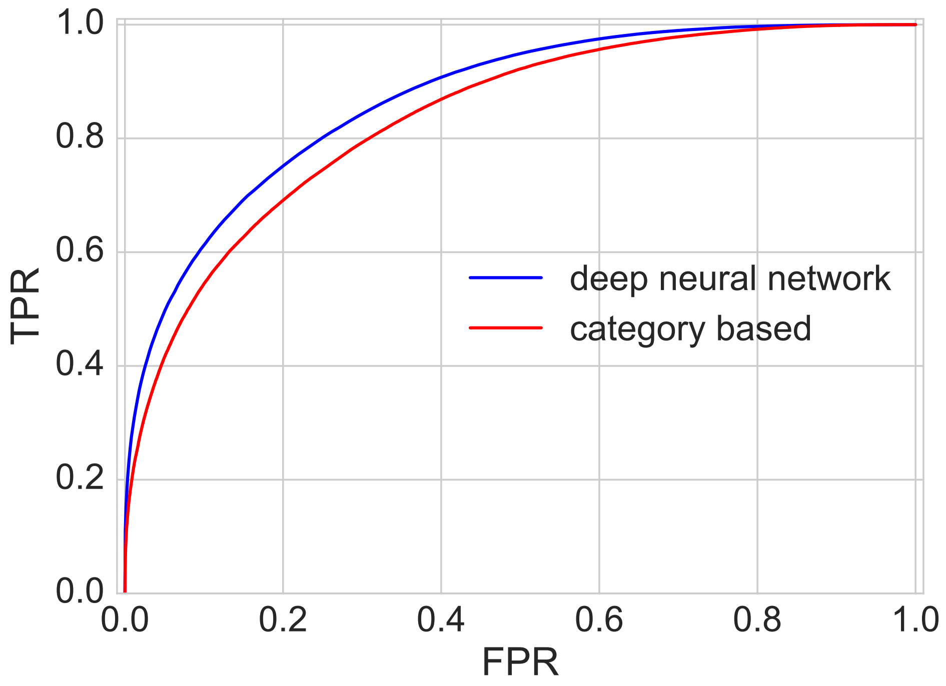


Figure 5.6.: Receiver Operation Curves (ROC) for a comparison of the category based flavor tagger and the deep neural network based approach (default parameters, 8 hidden layers) on Belle II Monte Carlo. The area under the ROC curve (AUC) is used as a quality measure.

At Belle and Belle II, a tag can be assigned to almost every reconstructed B meson candidate. Let ϵ be defined as the efficiency for tag-able events

$$\epsilon = \frac{N_{s,tag} + N_{b,tag}}{N_{s,rec} + N_{b,rec}}. \tag{5.7}$$

With the fraction w of wrongly assigned B mesons, for the number of reconstructed B^0 ($\bar{B}^0$) mesons $N_{s,rec}$ and $N_{b,rec}$ follows

$$N_{s,tag} = \epsilon(1-w)N_{s,rec} + \epsilon w N_{b,rec} \tag{5.8}$$

$$N_{b,tag} = \epsilon(1-w)N_{b,rec} + \epsilon w N_{s,rec}. \tag{5.9}$$

The observed CP asymmetry is A_{obs} is

$$A_{obs} = \frac{N_{s,tag} - N_{b,tag}}{N_{s,tag} + N_{b,tag}} = (1-2w)\frac{N_{s,rec} - N_{b,rec}}{N_{s,rec} + N_{b,rec}}. \tag{5.10}$$

The observed asymmetry is diluted by the dilution factor $r = 1 - 2w$ from the actual value A_0

$$A_{obs} = (1-2w)A_0. \tag{5.11}$$

The dilution factor directly influences the statistical uncertainty σ_{A_0} with

$$\sigma_{A_0} \propto \frac{1}{\sqrt{\epsilon}(1-2w)} := \frac{1}{\sqrt{\epsilon_{eff}}} \tag{5.12}$$

In literature, the output of the flavor-tagger is often described in terms of $q \cdot r$, where $q \in \{-1, 1\}$ denotes the assigned flavor and r the dilution factor. Using w obtained from Monte Carlo Studies would introduce a systematic bias on the measurement [32]. The tagging performance on recorded data has to be determined as well. Here the potential danger exists, that Monte Carlo specific attributes are "learned" by the network. This is even more important for the deep neural network approach, since there are no reference values available. The wrong tag fraction w has to be measured on so-called self tagging decays, e.g $B^0 \to D^{*-}l^+\nu$, $D^{*-}\pi^+, D^{*-}\rho+, D^-\pi^+$ [32, p.15]. The charge of the decay products let directly infer to the flavor of the B meson. At the Belle experiment, the wrong tag fraction was measured in 6 regions of r. The latter are defined by $0 < r \leq 0.25,\ 0.25 < r \leq 0.5,\ 0.5 < r \leq 0.625,\ 0.625 < r \leq 0.75,\ 0.75 < r \leq 0.875$ and $0.875 < r \leq 1$. To obtain comparability to Monte Carlo, the wrong tag fraction together with the effective tagging efficiency are calculated bin wise

$$\epsilon_{eff} = \sum_{i=1}^{6} \epsilon_i \langle r_i \rangle^2. \tag{5.13}$$

Here, ϵ_i is the efficiency per bin and $\langle r_i \rangle = (1 - 2\langle w_i \rangle)$ is the average dilution per bin. Evaluation of the flavor tagger on data will be subject of future studies and goes beyond the scope of this thesis. Nevertheless the effective tagging efficiency is a metric of interest for comparability between different Monte Carlo studies.

5.5.3. Transformation to Probability

If the deployed MVC would have been trained perfectly, the network output would correspond to a probability (Section 4.2.2). In this case, for a binned classifier output, the ratio of signal events with respect to all events per bin (bin purity) as a function of the bin number, would be centered around a straight line. Since this is usually not the case, for example the classifier adapted for a certain degree on statistical fluctuations, the network output can be re-weighted on a test set compared to Monte Carlo truth to provide a better representation of the data. This transformation was implemented as part of this thesis. It is applied before evaluating the effective tagging efficiency of the algorithm. An example is shown in Fig. 5.7.

In general, a trained MVC can be used for classification processes with a different signal to background ratio than during the training. In that case, another transformation has to be applied, in order to transform the classifier output to a probability. The presented approach follows [35] and makes use of the Bayes Theorem. The untransformed output of the classifier is transformed to a probability according to the training signal fraction $y_n \to o_t$. Then it is corrected according to the signal fraction of the prediction dataset to o_p. $P_t(S)$ $(P_t(B))$ is the probability to get a specific network output o_t for a signal S (background B). $P_p(S)$ is the probability for the prediction dataset. One obtains

$$o_p = \left(1 + \left(\frac{1}{o_t} - 1\right) \frac{P_p(B)}{P_p(S)} \frac{P_t(S)}{P_t(B)}\right)^{-1}. \tag{5.14}$$

5.6. The Algorithm

The algorithm and its initial hyperparameters used in this thesis are based on a benchmark study for a potential application of deep neural networks and deep learning in high-energy physics [40]. A dataset with 21 low-level and 7 high-level input variables was investigated for different network configurations and learning techniques. Adding the high level-features did only lead to a minor increase of the network performance.

The standard network configuration in this study, which is based on [40] uses `tanh` activation functions in the hidden layers and a sigmoid function in the output layer. An advantage of

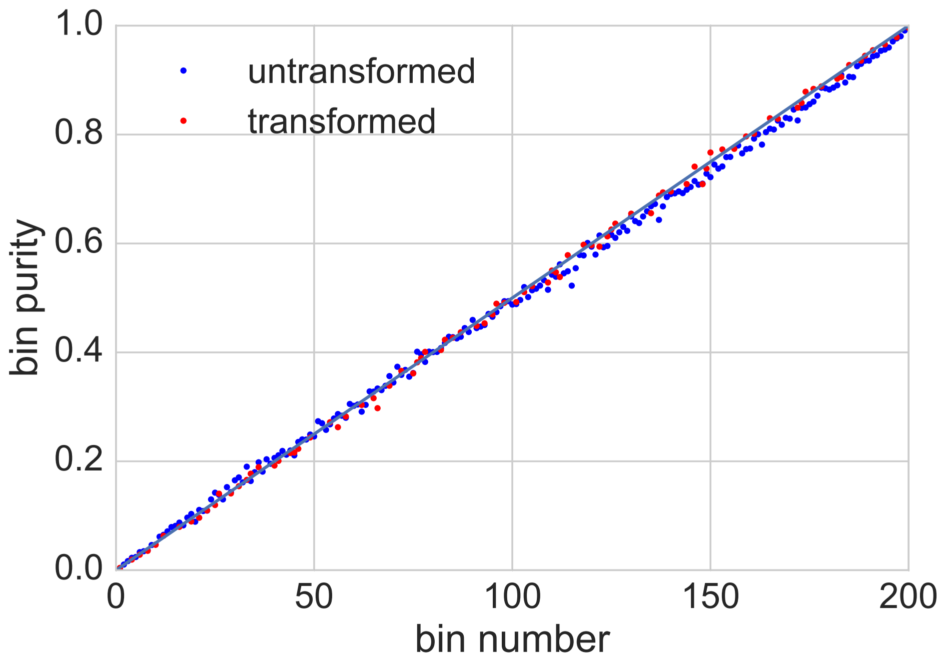


Figure 5.7.: Bin purity with (red) and without (blue) transformation to probability for a training on Belle II Monte Carlo with 8 hidden layers. The straight line trough the origin which corresponds to a perfectly trained algorithm without fluctuations.

the `tanh` function lies in the fact, that the weight symmetry is centered around zero which does not lead to a biased saturation [46]. The classifier is trained with mini-batch stochastic gradient descent with a batch size of 100 events and a binary cross-entropy error function. The weights were initialized with random values drawn from a uniform distribution, with respect to the activations functions according to [46]. The learning rate η was initialized with 0.05 and decreased exponentially with a factor of $1 + 2 \cdot 10^{-7}$ per mini-batch until a value of 10^{-6}. A momentum extension was used with linear increase from 0.9 to 0.99 after 200 epochs. A trainings epoch is defined as a complete iteration on the applied training dataset. The training was monitored on a validation set and stopped when the error function did not decline after 10 epochs with a factor of 10^{-5} to the overall minimum. This parameter setup will be referred to as the *standard parameter set*. The training process for the standard variable set on Belle II Monte Carlo is shown in Fig. 5.7. The error function indicates a slight over-training on the algorithm, although the loss-function on training and validation set shows similar movements on both datasets.

As described in Section 5.6.1, the hyperparameter setup found by [40] was already a good configuration choice for the feature space for the deep neural network used for flavor tagging purpose.

5.6.1. Influence of Hyperparameters

In this thesis the influence of hyperparameters on the employed model was studied. The observed hyperparameters are strongly related to each other and to the feature representation. Varying only some of them while letting the majority constant provides only a limited insight. Nevertheless this approach reveals information about the stability of a minimum, found by the classifier and adjustments can improve the training performance.

For each data point for the comparison of layer width and layer depth, the model has been

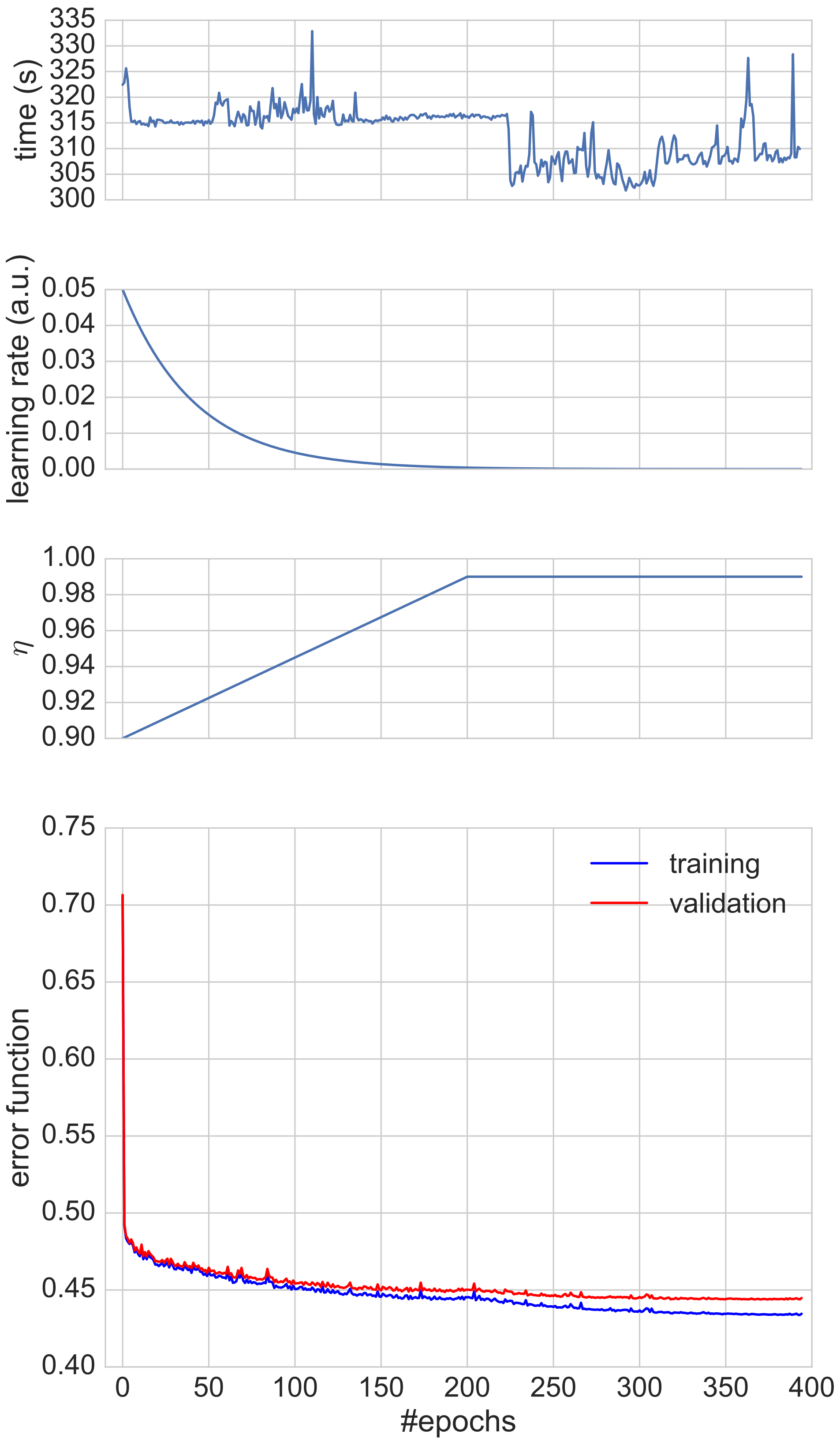


Figure 5.8.: Monitoring of network parameters during the training process for a neural network with 8 hidden layers on Belle II Monte Carlo. The training time, learning rate, momentum parameter η and the error function are shown with respect to the epochs. The error function is monitored for the training dataset and the validation dataset, which is a fraction of 0.04 of the complete dataset defined in Section 5.1.

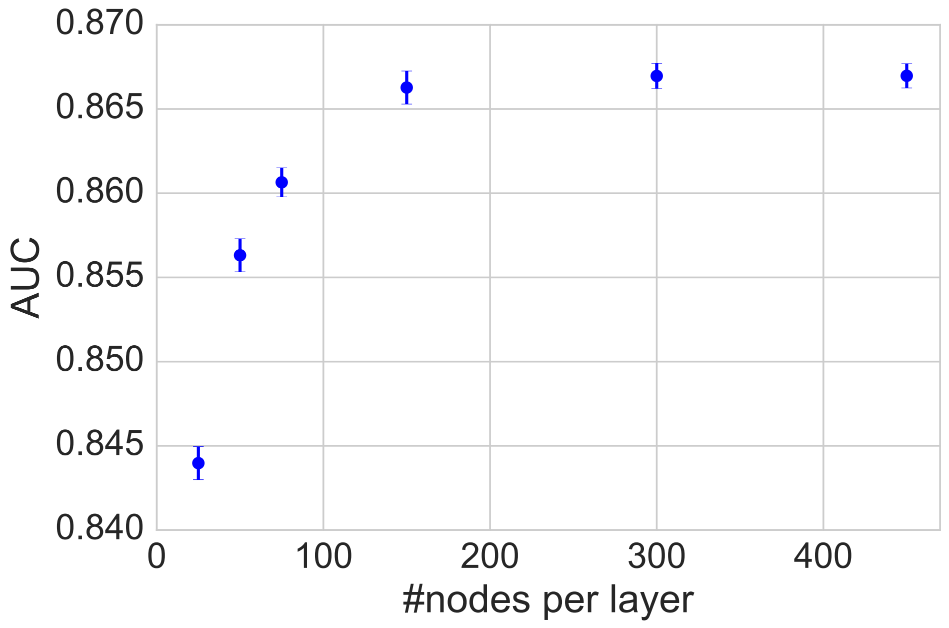


Figure 5.9.: Influence of the layer width for the standard variable set and the standard parameter set on all Belle II Monte Carlo datasets. All hidden layers of a neural network have the same number of nodes.

trained on all 4 datasets with the same hyperparameters. The rather small number of trainings for a classifier was due to the fact, that an average training with the standard parameter set takes in the order of 24 hours on the utilized GTX970 GPU. Eight layer networks require roughly twice as much training time. That is in agreement with the expected computational complexity, as described in Section 4.2.2. For many of the comparisons, therefore only one network training was used, to gain at least a limited overview. A more detailed overview of the used hardware is provided in Appendix A.

If a series of tests for a given hyperparameter setting was available, the median of the AUC or the effective tagging efficiency was calculated. The median was chosen because of its robustness against outliers. For an estimation of the statistical error, the error of the mean is used. For the AUC, this error and the error as stated in Eq. (5.5) is accumulated.

As shown in Fig. 5.9, the classifier performance increases with the **width of layers** and reaches its best value at a width of 300 nodes where it starts to saturate. The number of nodes of a hidden layer defines the dimension of an internal feature representation. In this approach it is intended to chose a network architecture, which is complicated enough to be able to extrapolate all important aspects available in the training data. If the regularization mechanisms (Section 4.3) work as intended, the network performance should be stable inside a certain parameter region, even if the the network contains more degrees of freedom than necessary. Results from Fig. 5.9 indicate that this is the case.

An increase of the **layer depth** allows a better generalization of the feature representation developed by the neural network. However, with an increased number of layers, the gradient of the error function gets smaller in the first layers while training with back propagation [47] (here shown for recurrent neural networks).
For neural networks, this is described as the *vanishing gradient problem*. There are various suggestions to deal with the problem, e.g. increasing the norm of gradient updates in back propagation with respect to the iteration step [48] or the introduction or rectified linear units

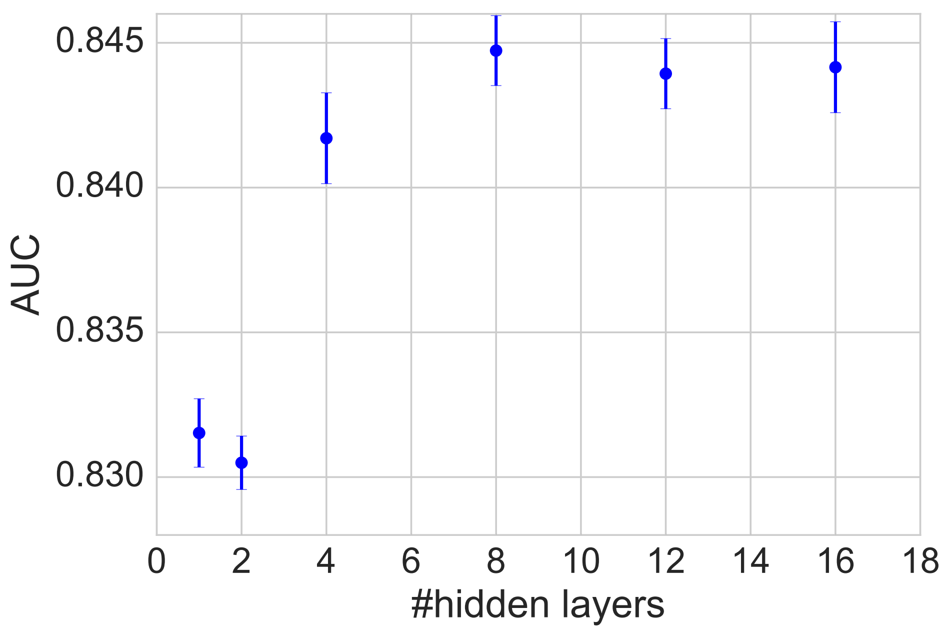


Figure 5.10.: Influence of the number of hidden layers (depth) for the standard variable set and the standard parameter set on all Belle Monte Carlo datasets. All hidden layers of the same set have the same number of nodes.

(ReLu) [49]. Using ReLus comes with the danger of an exploding gradient. This can be controlled with weight decay or clipping the weight tensor to a norm [48]. Limited tests in the scope of this thesis did not show any improvements when using ReLus. In Fig. 5.10 it is shown that the best result on the default parameter set was obtained for a setup with 8 hidden layers. To increase model complexity other approaches like convolutional networks could be deployed.

The **shape of a layer** can influence the developed feature representation of a trained network. Decreasing the number of nodes forces the network to a description of the constructed objects in the feature space with less attributes, increasing could allow a greater variety.
In Table 5.1 the examined models are described. The model

- W0 corresponds to the default network setup,
- W1 and W2 have a declining shape,
- W3 has an increasing layer shape,
- W4 and W5 has an narrowing shape and
- W6 and W7 have an broadening shape.

In Fig. 5.11 can be seen that the shape has only minor effect. Noticeable are the architectures W3 and W7. The former layout has fewest nodes in the input layer, this could indicate that dimension in the first hidden layer is decisive for the development of a good feature representation. The latter layout is increasing to the broadest setup, with the possible effect, that the task, for example the reconstruction of immediate particles of a decay, can be described better with a smaller representation. Another possible reason could be the failure of regulative mechanisms.

In contrast to the previously studied parameters, the regularization mechanism **weight decay** shows a huge influence (see Fig. 5.12). The best result is achieved for the standard parameter setup WD2, whereas a lower and a higher weight decay values for each layer provoke a smaller

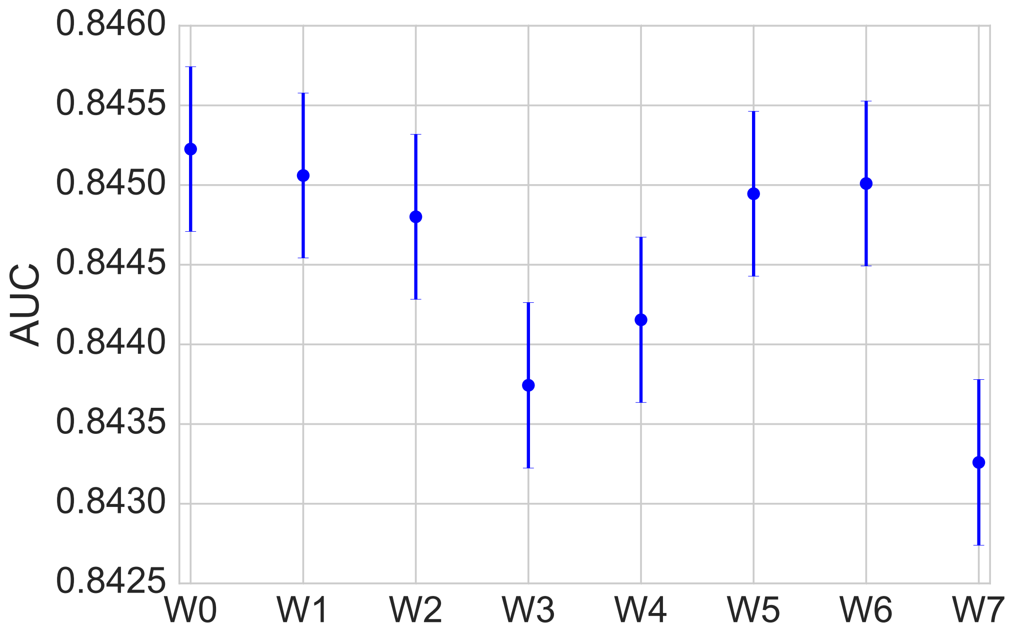


Figure 5.11.: Influence of the shape of hidden layers for the standard variable set and the standard parameter set on a *single* Belle Monte Carlo dataset for a model with *8 hidden layers*. The models W0, ...,W7 are described in Table 5.1.

Model	L 0	L 1	L 2	L 3	L 4	L 5	L 6	L 7	AUC
W0	300	300	300	300	300	300	300	300	0.8452±0.0005
W1	300	275	250	225	200	175	150	125	0.8451±0.0005
W2	600	500	400	300	250	200	175	150	0.8448±0.0005
W3	125	150	175	200	225	250	275	300	0.8437±0.0005
W4	300	200	100	50	50	100	200	300	0.8442±0.0005
W5	300	250	200	150	150	200	250	300	0.8449±0.0005
W6	300	350	400	450	450	400	350	300	0.845±0.0005
W7	300	400	500	600	600	500	400	300	0.8433±0.0005

Table 5.1.: Influence of the shape of hidden layers for the standard variable set and the standard parameter set on a *single* Belle Monte Carlo dataset for a model with *8 hidden layers*. The columns L 0, ..., L 7 describe the number of nodes in the specific hidden layer. The model **W0** corresponds to the default parameter setup.

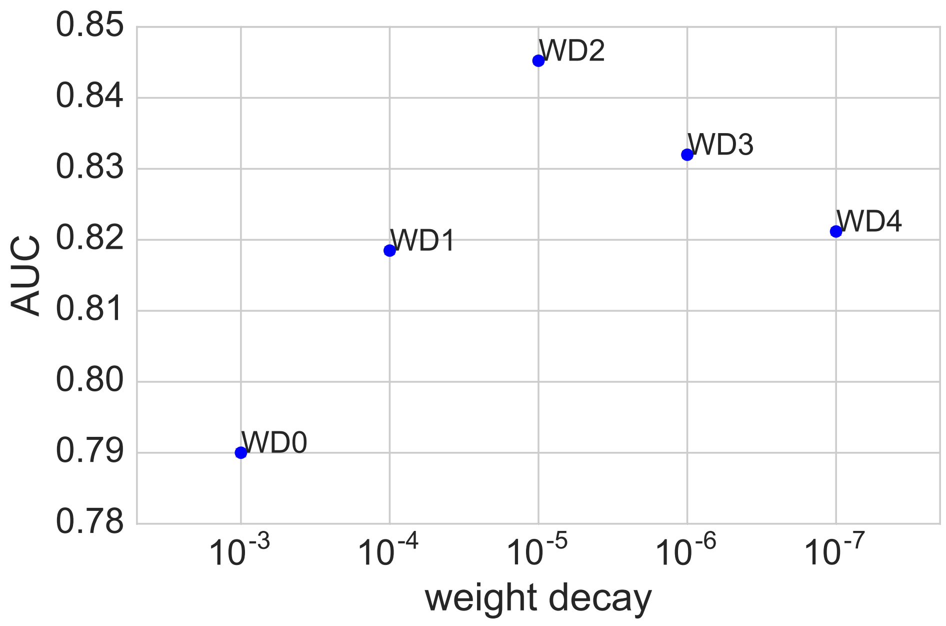


Figure 5.12.: Influence of weight decay for the standard variable set and the standard parameter set on a *single* Belle Monte Carlo dataset for a model with *4 hidden layers*. The models are described in Table 5.2. The errors are too small to be visible.

AUC. Almost similar results could be obtained when linear increasing the weight decay from $\alpha = 10^{-6}.. \ 10^{-4}$ in each layer, respectively (model WD7). This scheme was tested with the vanishing gradient problem but also faster saturation effects in the first layers, shown in [46] in mind.

Varying the **batch size** shows rather interesting results. When using SGD, increasing the mini-batch size can result in a greater generalization and being less susceptible to statistical fluctuations. A smaller mini-batch size results in a more stochastic treatment and a higher difference of batches in each step. Fig. 5.13 shows a maximum for the default parameters and indicate that the minimum, found by the approach in [40] is also applicable for the extended approach in this thesis. It furthermore indicates, that a sophisticated hyperparameter search, e.g with simulated annealing or Bayesian optimization [50] could lead to even better minimum.

Another parameter with a huge impact is the **learning rate**. While the default learning rate already shows good results (Fig. 5.14, model LR0), decreasing the initial learning rate with a factor 5 leads to a slight improvement (LR2). Decreasing the minimum learning rate also leads an improvement compared to the default parameter setup but increases training time significantly (LR6).

The impact of the **momentum term** was object of interest as well (Fig. 5.15). The initial momentum and the slope of the increment was varied. The termination criterion of the minimal number of epochs was altered to the point, when the final value of the momentum was reached. Again, the default parameter (M0) for the momentum is a good choice for the examined hyperparameter set, only setting the initial momentum to 0.5 (M3) leads to a slight improvement.

The best models M3 and LR2 were found during the extensive hyperparameter search. Both values were found on Belle Monte Carlo training data sets. On Belle II Monte Carlo, the superior performance of these compared to the default parameter set, could not be reproduced.

Model Name	first layer	last layer	type	AUC
WD0	0.001	0.001	constant	0.79±0.0006
WD1	0.0001	0.0001	constant	0.8185±0.0006
WD2	1e-05	1e-05	constant	0.8452±0.0005
WD3	1e-06	1e-06	constant	0.832±0.0005
WD4	1e-07	1e-07	constant	0.8212±0.0006
WD5	5e-05	5e-06	linear decrease	0.8289±0.0005
WD6	5e-06	5e-05	linear increase	0.8403±0.0005
WD7	1e-06	0.0001	linear increase	0.8422±0.0005

Table 5.2.: Influence of weight decay for the standard variable set and the standard parameter set on a *single* Belle Monte Carlo dataset for a model with *4 hidden layers*. The weight decay factor for the first and for the last layer is given, the state of the intermediate layers is described by their type. The model **WD2** corresponds to the default parameter setup.

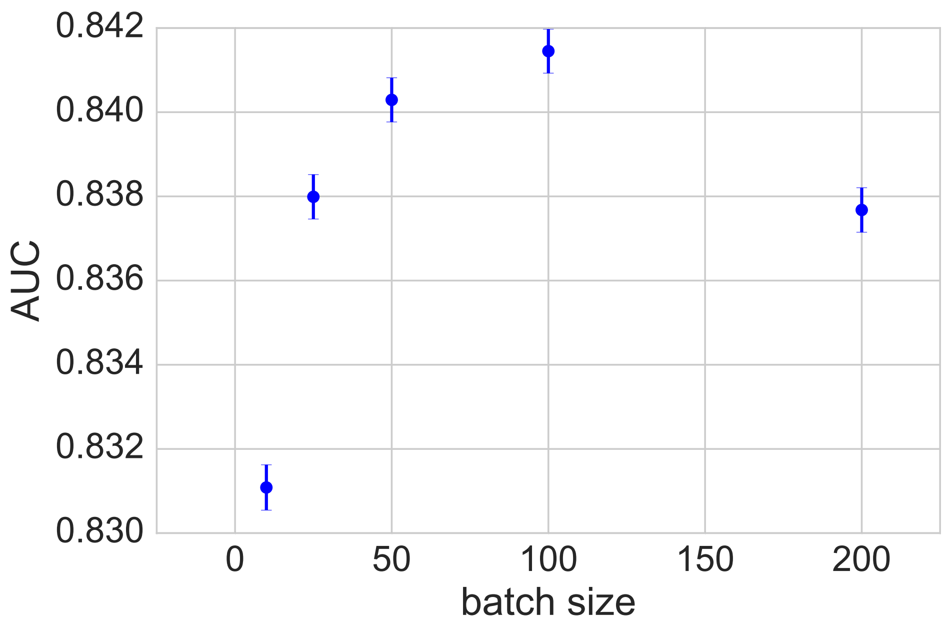


Figure 5.13.: Influence of the mini-batch size for standard variable set and standard parameter set on a *single* Belle Monte Carlo dataset.

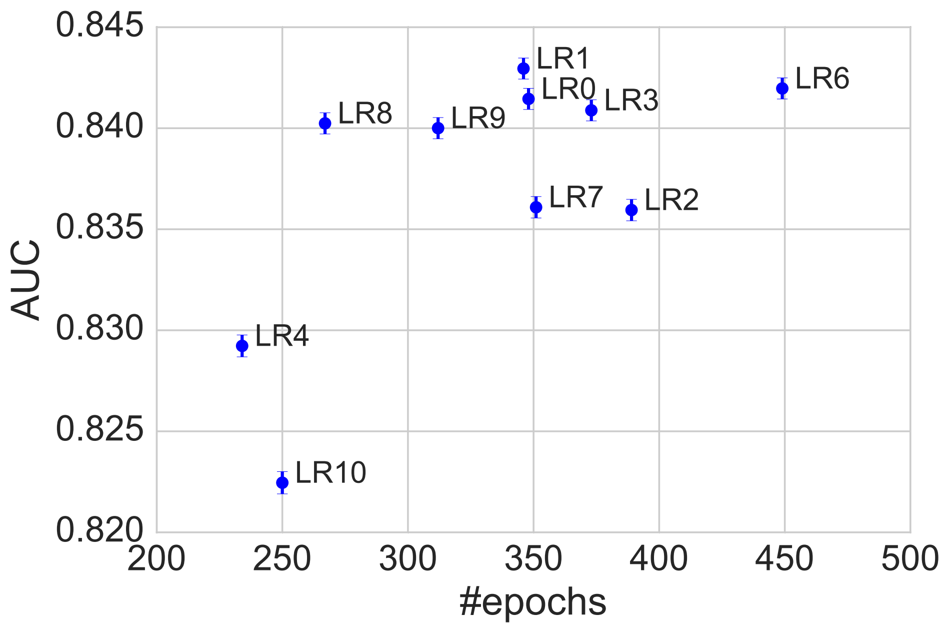


Figure 5.14.: Influence of the learning rate for standard variable set and standard parameter set on a *single* Belle Monte Carlo dataset with *4 hidden layers*. The models LR0, ..., LR10 are described in Table 5.1. The model LR5 is not shown due to its large relative number of training epochs (1297).

Model	Initial LR	Final LR	LR decay factor	AUC
LR0	0.05	1e-06	1.0000002	0.8415±0.0005
LR1	0.01	1e-06	1.0000002	0.843±0.0005
LR2	0.001	1e-06	1.0000002	0.836±0.0005
LR3	0.05	1e-06	1.0000004	0.8409±0.0005
LR4	0.05	1e-06	1.0000001	0.8292±0.0005
LR5	0.05	1e-06	1.000008	0.831±0.0005
LR6	0.05	1e-08	1.0000001	0.842±0.0005
LR7	0.001	1e-08	1.0000001	0.8361±0.0005
LR8	0.01	1e-06	1.0000001	0.8402±0.0005
LR9	0.01	1e-06	1.0000004	0.84±0.0005
LR10	0.01	1e-06	1.0000008	0.8225±0.0006

Table 5.3.: Influence of the learning rate for standard variable set and standard parameter set on a *single* Belle Monte Carlo dataset for a model with *4 hidden layers*. The initial learning rate, the final learning rate and the learning rate decay factor per batch is shown. The model **LR0** corresponds to the default parameter setup.

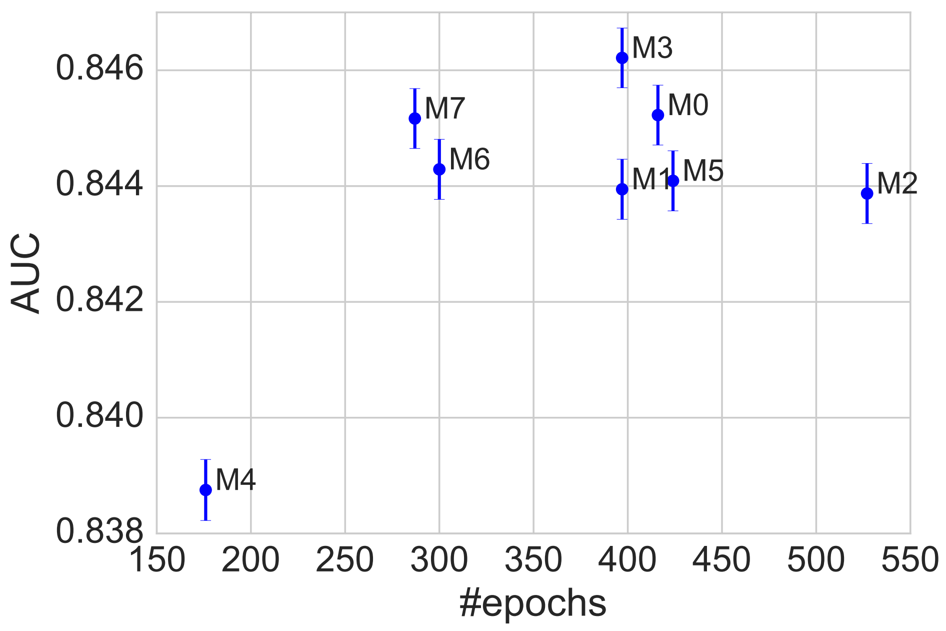


Figure 5.15.: Influence of momentum for the standard variable set and the standard parameter set on a *single* Belle Monte Carlo dataset. The models M0, ..., M7 are described in Table 5.4.

Model	Initial Momentum	Increase until Epoch	AUC
M0	0.9	200	0.8452±0.0005
M1	0.9	100	0.8439±0.0005
M2	0.9	300	0.8439±0.0005
M3	0.5	200	0.8462±0.0005
M4	0.5	100	0.8388±0.0005
M5	0.5	300	0.8441±0.0005
M6	0.6	300	0.8443±0.0005
M7	0.8	300	0.8452±0.0005

Table 5.4.: Influence of momentum for the standard variable set and the standard parameter set on a *single* Belle Monte Carlo dataset for a model with *4 hidden layers*. The models are described in Table 5.4. The model **M0** corresponds to the default parameter setup.

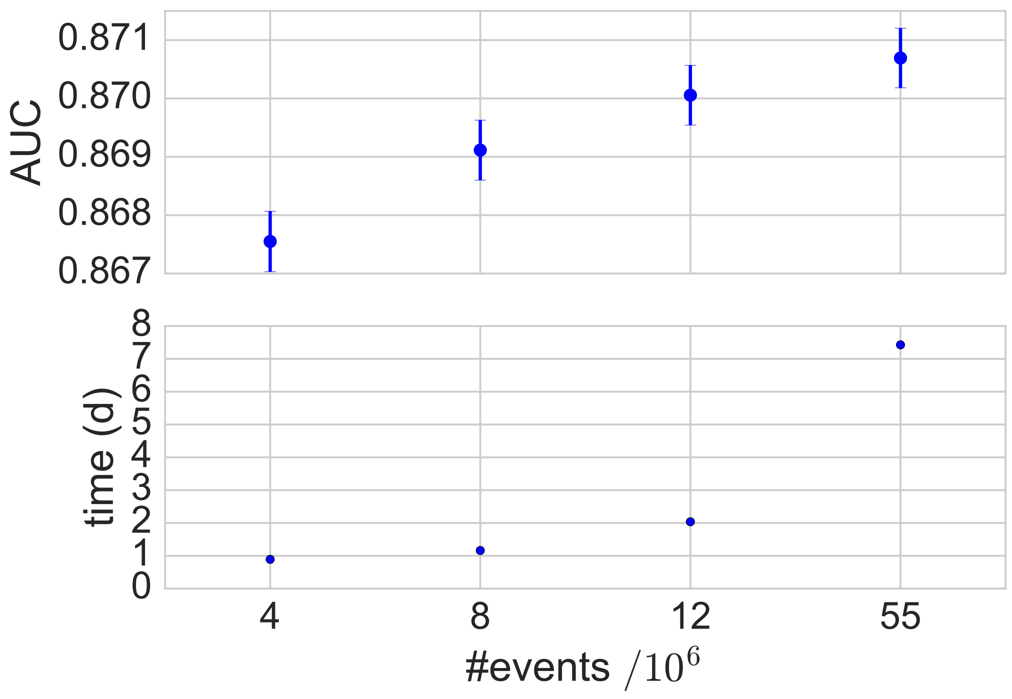


Figure 5.16.: Influence of size for the standard variable set and the standard parameter set on a *single* Belle II Monte Carlo dataset for model with *8 hidden layers*.

Furthermore, first studies indicated, that the hyperparamters, found by the approach of [40] provided stable results. Therefore this set was chosen as default parameter set.

5.6.2. Dataset Size

A key requirement of deep learning is the massive amount of (labeled) data. In this thesis, an algorithm with large scalability was examined. Therefore it is of interest, how scaling has an effect on the results. In Fig. 5.16 the impact of four different dataset sizes is shown. The best result can be achieved on a dataset with 55 million training events. Unfortunately the benefit comes with increased training and dataset generation time. Nevertheless the results indicate, that an improvement is still possible.

5.6.3. Variable Influence

Influence of additional variables is of interest, especially the impact of the hit variables. In this section the most interesting discriminations between the variables of the used standard variable sets are shown. In Fig. 5.17 the differences between the standard parameter sets, with and without any combination of perigee information and hit information are shown. Without the perigee information, using the hit information provides a significant increase classification performance. After adding the impact parameters, no significant difference is notable. There are many more variables including ECL- and KLM cluster variables which could be of interest for flavor tagging. A restriction for the usage of variables was the limited amount of memory that was available for a large part of this thesis. This did not allow an arbitrary large amount of variables to test the limitations of the network setup with respect to the variable number.

5.6.4. Decay Channel Dependency

In Table 5.6 a comparison for training and testing on different Monte Carlo sets is shown. Due to the very limited statistics, this can only be seen as a hint, that a training on mono-generic Monte Carlo is preferable.

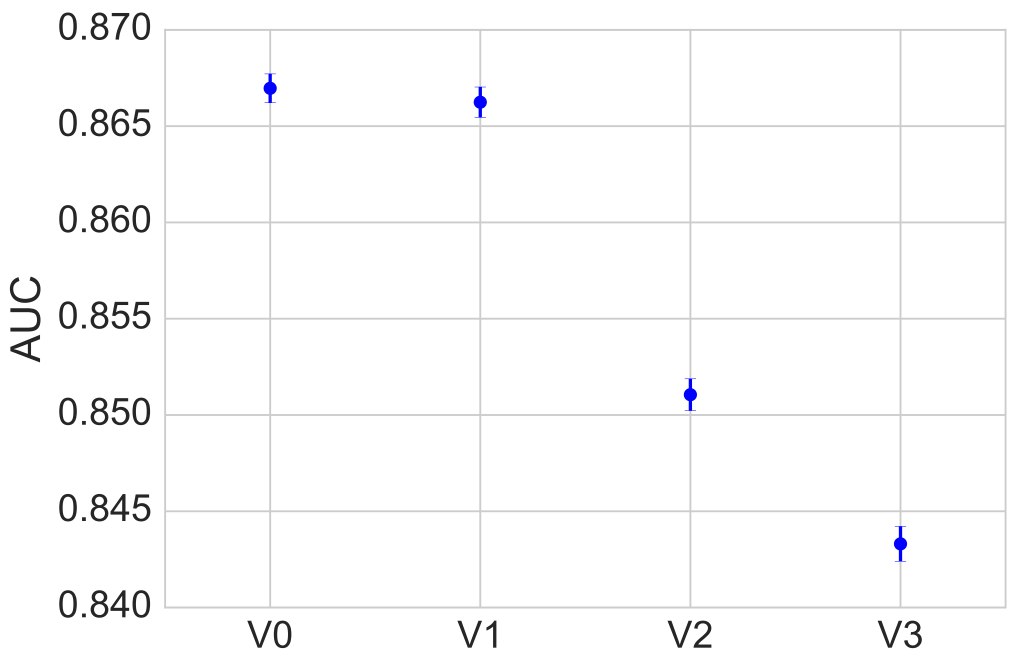


Figure 5.17.: Influence of size for different variable sets for the standard parameter set on a network with *4 hidden layers* on all Belle II Monte Carlo datasets. The variable sets are described in Table 5.5

Model	hit variables	perigee variables	AUC
V0	x	x	0.867±0.0007
V1		x	0.8662±0.0008
V2	x		0.8511±0.0008
V3			0.8433±0.0009

Table 5.5.: Influence of chosen variables for the standard parameter set on all Belle Monte Carlo II dataset for a model with *4 hidden layers*. In for this selection, the usage of the hit variables and the perigee variables was altered. The model **V0** corresponds to the default parameter setup.

	mono-generic training	$J/\Psi K_S^0$ training
mono-generic test set	0.3492±0.0006	0.3438
$J/\Psi K_S^0$ test set	0.3387±0.0008	0.3372

Table 5.6.: Comparison of the effective tagging efficiency for training and testing on mono-generic and $J/\Psi K_S^0$ Belle Monte Carlo with a 4 layer deep neural network. For training on $J/\Psi K_S^0$ this was only performed on one dataset. Therefore no error is calculable.

		Category Based	Deep Neural Network
Belle II	mono-generic	0.3407±0.0001	0.4091±0.0005
	$J/\Psi K_S^0$	0.3329±0.0001	0.4069±0.0003
Belle	mono-generic		0.3549±0.0008
	$J/\Psi K_S^0$		0.3442±0.0009

Table 5.7.: Comparison of the effective flavor tagging efficiency of the category based and the deep neural network based approach with *8 hidden layers* on different Monte Carlo test sets. Both taggers were trained on mono-generic Monte Carlo. At the time of evaluation the category based flavor tagger was not available for converted Belle Monte Carlo. The flavor tagger of the Belle experiment showed an effective tagging efficiency of 0.293 ± 0.01[32] on Monte Carlo. The evaluation was performed on different test sets, as used for the hyperparameter search.

5.6.5. Implementation and Execution in BASF2

As part of this thesis, the feature selection, feature preprocessing and network training was implemented to be accessible within BASF2. At first a TMVA plugin was developed but discarded due to the high internal memory usage of TMVA for the rather large training datasets. Due to the fact of the constant development of BASF2, several changes and adaptions of the interface had to be carried out. Feature selection is performed on the steering file level and can access and rank according to every variable of the BASF2 variable manager. The variable preprocessing and network training which is carried out in `Python` is accessed by the multivariate analysis package `MVA`, which was influenced by this thesis. The `Python` code is executed via `boost::python` in `C++`. The training itself is performed via the `basf2_mva_teacher` which can be executed as a stand-alone program. The required `bash` command and the training variables file is provided by the `DeepFlavorTagger` function from the `dft` package. During the training process, a weight file is generated which is stored in a local database and can be uploaded to a global weight file database. To use the trained network for classification, the identifier of the weight file has to be passed, besides the parameters regarding the specific input and output files. The network can be also used internally via the `MVAExpert` module. A basic example for the usage of the flavor tagger is given in Appendix B.

5.6.6. Comparison on Belle and Belle II Monte Carlo

As shown in Section 5.6.1, the standard parameter set was among the best parameters and resulted in a reliable and stable tagging performance. The results on a mono-generic and on a $J/\Psi K_s^0$ test sample are provided. Table 5.7 shows that on Monte Carlo, the deep neural network flavor tagger shows a significant improvement. The output of the trained classifier for the corresponding test dataset is shown in Fig. 5.18 for Belle II and in Fig. 5.19 converted Belle Monte Carlo. It is noticeable, that for Belle II Monte Carlo, more tagging decisions are mapped with to high probability.

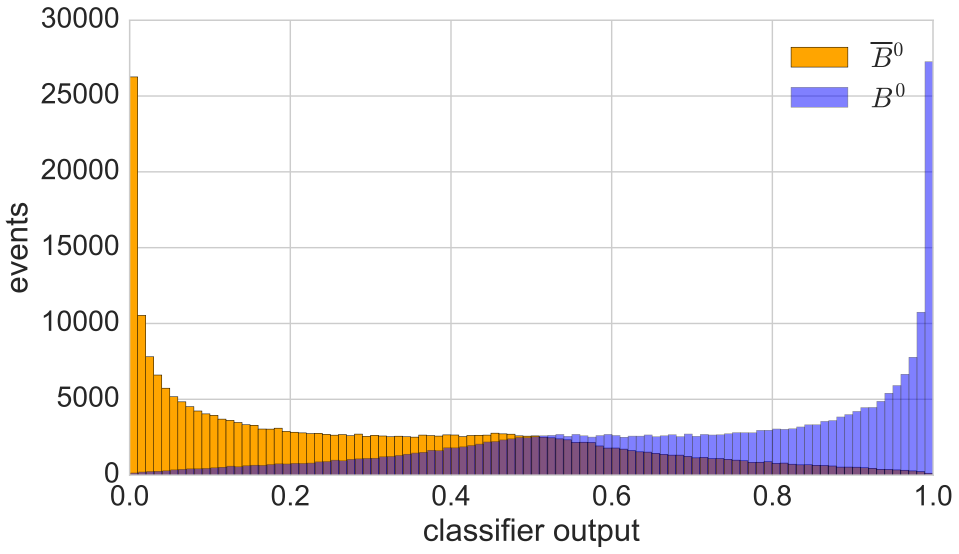


Figure 5.18.: Classifier output on the standard parameter set and *8 layers* for Belle II. Signal events (B^0 mesons) and background events ($\bar{B}^0$) are separated by color.

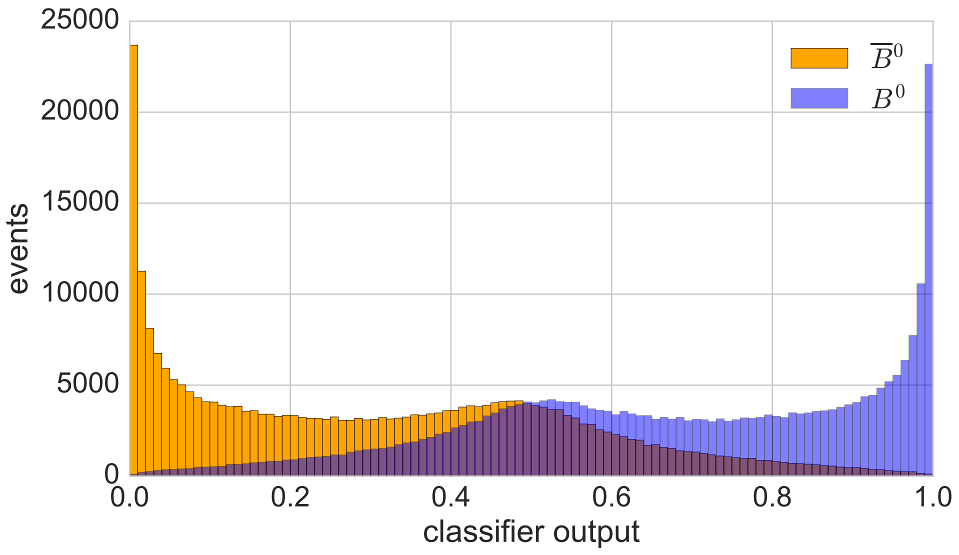


Figure 5.19.: Classifier output on the standard parameter set and *8 layers* for Belle. Signal events (B^0 mesons) and background events ($\bar{B}^0$) are separated by color.

6. Summary and Outlook

In this thesis, a deep neural network was applied to the task of flavor tagging. Instead of assigning single tracks with dedicated classifiers to single categories and combining them afterwards, this approach considers each event at once. It uses attributes of tracks, like particle identification, momentum, and spacial variables. An effective tagging efficiency of 0.4069 ± 0.0003 on Belle II Monte Carlo and of 0.3442 ± 0.0009 on Belle Monte Carlo is achieved. Compared to the established method, this corresponds to relative increase of the effective tagging efficiency on Belle II Monte Carlo of approximately 22%. On Belle Monte Carlo, the relative improvement of the effective tagging efficiency with respect to the established flavor tagger is approximately 17%.
The necessary pre-processing and post-processing transformations were implemented. The approach was integrated into BASF2 and is now also available for converted Belle Monte Carlo. A classifier, which was trained on a GPU, can also be applied on machines with only CPUs. This study was part of the first tests of the `b2bii` conversion feature and had to cope with continuous changes of BASF2, which is still under heavy development.
A study of the performance dependence of the classifier on various hyperparameters has been carried out. The parameter set found by Baldi, Sadowski and Whiteson [40] was among the best parameters. Maxima, which was found on Belle Monte Carlo did not show a significant improvement on Belle II Monte Carlo.

The performance of the presented approach heavily depends on the quality of the Monte Carlo. Therefore an evaluation on data is mandatory to make a final conclusion on the flavor tagging quality of the presented approach. This evaluation can be performed with so-called self-tagging decays. However, since no intermediate particles or sub-categories are introduced as classification targets, there is the possibility to train the deep neural network directly on data. This can be achieved with advanced event reweighting techniques [51]. Depending on the quality of the Monte Carlo, it might be possible to start a network training on Monte Carlo and finish the parameter adaption on data, allowing for a better fine tuning on data specific features.
Although the approach of this thesis already shows a significant improvement of a method, which was optimized over decades, further progression seems likely.

Better minima might be found when carrying out a sophisticated hyperparameter search based on Bayesian methods [50] or simulated annealing. This can also lead to a better classifier or a faster convergence of the training process while taking less computational resources into account. On the other hand, it is reasonable to decrease training time in order to increase the rate of testable models or to use a higher amount of data efficiently. A promising way to speed

up the training process is by the introduction of the batch normalization technique [52], where each mini-batch is normalized by its variance.
While GPUs became very popular in machine learning because of the high possibility of parallelization, the deployed hardware might shift to engineered systems for the specific task. Computation on application-specific integrated circuits (ASICs), where calculations are performed with reduced precision, like the Tensor Processing Unit (TPU) [53] utilized by google, might decrease the training time of a neural network drastically.

Rectified linear units were only considered very briefly in this thesis. In combination with an altered hyperparameter setup, especially for the weight decay parameters, improvement may be achieved. In combination with different training methods like Adadelta, Adagrad [54] or Adam [55], which adjust the learning rate with respect to the gradients of the preceding steps or take the Hessian matrix into account, the training process can be sped up and stabilized.
Another approach could be the introduction of a convolutional neural network, to cope with the altering input space of an event. For example, arranging tracks according to their production vertices in the $dr - dz$ plane could allow the MVC to construct an easier connection between single tracks and allow inference to a common mother particle. It should be noted, that these vertices would have to be found by a vertex fitter under a certain particle hypothesis or with the utilization of a vertex finder.
First tests with using the perigee information as a reference system, indicated that this approach might succeed, but will require significant work in tuning, testing and further optimization.

Bibliography

[1] **ATLAS** Collaboration, G. Aad *et al.*, "Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC," *Phys. Lett.* **B716** (2012) 1–29, `arXiv:1207.7214 [hep-ex]`.

[2] **CMS** Collaboration, S. Chatrchyan *et al.*, "Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC," *Phys. Lett.* **B716** (2012) 30–61, `arXiv:1207.7235 [hep-ex]`.

[3] A. D. Sakharov, "Violation of CP invariance, C asymmetry, and baryon asymmetry of the universe," *JETP lett.* **5** (1967) 24–27.

[4] M. Kobayashi and T. Maskawa, "CP-Violation in the Renormalizable Theory of Weak Interaction," *Progress of Theoretical Physics* **49** no. 2, (1973) 652–657.

[5] B. Aubert *et al.*, "Observation of CP Violation in the B^0 Meson System," *Physical Review Letters* **87** no. 9, (2001) 091801, `arXiv:1201.5897`.

[6] **Belle** Collaboration, K. Abe *et al.*, "Observation of Mixing-induced CP Violation in the Neutral B Meson System," `arXiv:0202027 [hep-ex]`.

[7] "Physics achievements from the Belle Experiment," *Progress of Theoretical and Experimental Physics* **2012** no. 1, (2012) 4D001, `arXiv:1212.5342`.

[8] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature* **521** no. 7553, (2015) 436–444.

[9] **Belle-II** Collaboration, T. Abe *et al.*, "Belle II Technical Design Report," `arXiv:1011.0352 [physics.ins-det]`.

[10] A. Abashian, K. Gotow, N. Morgan, L. Piilonen, S. Schrenk, and K. Abe, "The Belle Detector," *Nuclear Instruments and Methods in Physics Research* **479** no. A, (2002) 117–232.

[11] M. Ziegler, "Search for the decay $B^0 \to \tau^+\tau^-$ with the Belle detector," *Internal Belle Note* no. 1390, (2016) .

[12] D. Besson and T. Skwarnicki, "Upsilon Spectroscopy: Transitions in the Bottomonium System," *Annual Review of Nuclear and Particle Science* **43** no. 1, (1993) 333–378.

[13] **Particle Data Group** Collaboration, K. A. Olive *et al.*, "Review of Particle Physics," *Chin. Phys.* **C38** (2014) 090001.

[14] C. Schwanda, "SuperKEKB machine and Belle II detector status," *Nuclear Physics B - Proceedings Supplements* **209** no. 1, (2010) 70–72.

[15] T. E. Browder, T. Gershon, D. Pirjol, A. Soni, and J. Zupan, "New physics at a super flavor factory," *Reviews of Modern Physics* **81** no. 4, (2009) 1887.

[16] **sBelle Design Group** Collaboration, I. Adachi *et al.*, "sBelle Design Study Report," `arXiv:0810.4084 [hep-ex]`.

[17] N. Braun, "Momentum Estimation of Slow Pions and Improvements on the Track Finding in the Central Drift Chamber for the Belle II Experiment," Master's thesis, KIT, 2015. `https://ekp-invenio.physik.uni-karlsruhe.de/record/48740`.

[18] M. Prim, "Study of material effects in track fitting and improvement of the K_S^0 reconstruction at Belle II," Master's thesis, KIT, 2015. `https://ekp-invenio.physik.uni-karlsruhe.de/record/48795`.

[19] J. Rauch and T. Schlüter, "GENFIT — a Generic Track-Fitting Toolkit," *J. Phys. Conf. Ser.* **608** no. 1, (2015) 012042, `arXiv:1410.3698 [physics.ins-det]`.

[20] A. Moll, "The Software Framework of the Belle II Experiment," *Journal of Physics: Conference Series* **331** no. 3, (2011) 032024.

[21] I. Antcheva, M. Ballintijn, B. Bellenot, M. Biskup, R. Brun, N. Buncic, P. Canal, D. Casadei, O. Couet, V. Fine, *et al.*, "ROOT—A C++ framework for petabyte data storage, statistical analysis and visualization," *Computer Physics Communications* **182** no. 6, (2011) 1384–1385.

[22] A. Hoecker, P. Speckmayer, J. Stelzer, J. Therhaag, E. von Toerne, H. Voss, M. Backes, T. Carli, O. Cohen, A. Christov, *et al.*, "TMVA-Toolkit for multivariate data analysis," *arXiv preprint physics/0703039* (2007) .

[23] C. Pulvermacher, T. Keck, M. Feindt, M. Heck, and T. Kuhr, "An automated framework for hierarchical reconstruction of B mesons at the Belle II experiment," `arXiv:arXiv:1410.3259v1`.

[24] D. J. Lange, "The evtgen particle decay simulation package," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **462** no. 1, (2001) 152–155.

[25] C. Pulvermacher, *Analysis Software and Full Event Interpretation for the Belle II Experiment*. PhD thesis, KIT, 2015. `https://ekp-invenio.physik.uni-karlsruhe.de/record/48741`.

[26] L. Wolfenstein, "Parametrization of the Kobayashi-Maskawa matrix," *Physical Review Letters* **51** no. 21, (1983) 1945.

[27] A. J. Buras, M. E. Lautenbacher, and G. Ostermaier, "Waiting for the top quark mass, K+ $\to$ pi+ neutrino anti-neutrino, B(s)0 - anti-B(s)0 mixing and CP asymmetries in B decays," *Phys. Rev.* **D50** (1994) 3433–3446, `arXiv:hep-ph/9403384 [hep-ph]`.

[28] CKMfitter Group (J. Charles et al.), "CP violation and the CKM matrix: Assessing the impact of the asymmetric B factories," *Eur. Phys. J. C* **41** (2005) 1–131. updated results and plots available at: `http://ckmfitter.in2p3.fr`.

[29] T. Kuhr, *Flavor Physics at the Tevatron: Decay, Mixing and CP-violation Measurements in Pp-collisions*, vol. 249. Springer, 2012.

[30] **Belle, BaBar** Collaboration, A. J. Bevan *et al.*, "The Physics of the B Factories," *Eur. Phys. J.* **C74** (2014) 3026, `arXiv:1406.6311 [hep-ex]`.

[31] M. Gelb, "Neutral B Meson Flavor Tagging for Belle II," Master's thesis, KIT, 2015. `https://ekp-invenio.physik.uni-karlsruhe.de/record/48719`.

[32] H. Kakuno *et al.*, "Neutral B flavor tagging for the measurement of mixing-induced CP violation at Belle," *Nuclear Instruments and Methods in Physics Research, Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **533** no. 3, (2004) 516–531, `arXiv:0403022 [hep-ex]`.

[33] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain.," *Psychological review* **65** no. 6, (1958) 386.

[34] M. Feindt and U. Kerzel, "The NeuroBayes neural network package," *Nuclear Instruments and Methods in Physics Research, Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **559** no. 1 SPEC. ISS., (2006) 190–194.

[35] M. Feindt, F. Keller, M. Kreps, T. Kuhr, S. Neubauer, D. Zander, and A. Zupanc, "A hierarchical NeuroBayes-based algorithm for full reconstruction of B mesons at B factories," *Nuclear Instruments and Methods in Physics Research, Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **654** no. 1, (2011) 432–440, `arXiv:1102.3876`.

[36] C. M. Bishop, *Pattern recognition and machine learning*. Information science and statistics. Springer, 2009.

[37] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks* **2** no. 5, (1989) 359–366.

[38] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature* **323** no. 6088, (1986) 533–536, `arXiv:arXiv:1011.1669v3`.

[39] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580* (2012) .

[40] P. Baldi, P. Sadowski, and D. Whiteson, "Searching for exotic particles in high-energy physics with deep learning.," *Nature communications* **5** (2014) 4308, `arXiv:1402.4735`.

[41] I. J. Goodfellow, D. Warde-Farley, P. Lamblin, V. Dumoulin, M. Mirza, R. Pascanu, J. Bergstra, F. Bastien, and Y. Bengio, "Pylearn2: a machine learning research library," *arXiv preprint arXiv:1308.4214* (2013) .

[42] Theano Development Team, "Theano: A Python framework for fast computation of mathematical expressions," *arXiv e-prints* **abs/1605.02688** (May, 2016) . `http://arxiv.org/abs/1605.02688`.

[43] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467* (2016) .

[44] **GEANT4** Collaboration, S. Agostinelli *et al.*, "GEANT4: A Simulation toolkit," *Nucl. Instrum. Meth.* **A506** (2003) 250–303.

[45] A. P. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," *Pattern Recognition* **30** no. 7, (1997) 1145–1159.

[46] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)* **9** (2010) 249–256.

[47] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks* **5** no. 2, (Mar, 1994) 157–166.

[48] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," *Proceedings of The 30th International Conference on Machine Learning* no. 2, (2012) 1310–1318, `arXiv:arXiv:1211.5063v2`.

[49] X. Glorot, A. Bordes, and Y. Bengio, "Deep Sparse Rectifier Neural Networks," *Aistats* **15** (2011) 315–323.

[50] J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian Optimization of Machine Learning Algorithms," *ArXiv e-prints* (June, 2012) , `arXiv:1206.2944 [stat.ML]`.

[51] D. Martschei, M. Feindt, S. Honc, and J. Wagner-Kuhr, "Advanced event reweighting using multivariate analysis," *Journal of Physics: Conference Series* **368** no. 1, (2012) 012028.

[52] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," *arXiv:1502.03167* (2015) 1–11, `arXiv:1502.03167`.

[53] N. Juouppi, "Blogpost: Google supercharges machine learning tasks with tpu custom chip." `https://cloudplatform.googleblog.com/2016/05/Google-supercharges-machine-learning-tasks-with-custom-chip.html`, 2016. [Online; accessed 10-July-2016].

[54] M. D. Zeiler, "ADADELTA: An Adaptive Learning Rate Method," *arXiv* (2012) 6, `arXiv:1212.5701`.

[55] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980* (2014) .

Appendix

A. Available Hardware

IEKP (Institut für experimentelle Kernphysik) working station – Hardware available during almost the complete time of the study

- Intel Core i7-5820K @3.30 GHz (6 cores)
- 32 GB DDR4-2133
- Nvidia GeForce GTX 970 Phantom, 4GB GDDR5
- Mainboard: Gigabyte GA-X99-UD4 Quad Channel
- Internal Storage: 1 TB HDD, 256GB SSD

IEKP GPU server – Hardware available for a limited time

- 2x Intel Xeon CPU E5-2630 v4 @2.20 GHz (10 cores)
- 378 GB DDR4-2133 LRDIMM
- 4x GeForce GTX TITAN X (Maxwell), 12 GB GDDR5
  7.00 TFlops Single Precision floating point performance (peak)
- Mainboard: Supermicro X10 DGQ - Dual Socket

Institut für Prozessdatenverarbeitung und Elektronik (IPE) – Hardware available for a limited time

- 2x Intel Xeon CPU E5-2640 @2.50 GHz (12 cores)
- 258 GB RAM
- 2x Tesla K20Xm, ∼6GB GDDR
- 4x GeForce GTX TITAN 6GB GDDR

B. Tutorial for using the Deep Flavor Tagger with BASF2

```
from dft import steering_training_data
import os
import subprocess
import pickle

# Basic tutorial for the training of the deep flavor tagger.
# make sure that you have installed Theano (pip3 install theano),

# you also need pandas, nose_parametriezed, root-numpy
# and pylearn2 (for training, only)

# examples for data set generation, testing and training
# are provided in steering_training_data

# monogeneric monte carlo. a decay into nu_tau and anti-nu_tau for classifier
# classifier training is recommended

# enter a list of strings for the filenames
filenames = []

# name of the identifier in the database
uniqueIdentifier = 'standard'

# insert here a list of basf2 variable names
# None is standard
variable_list = None

# use the current directory as working directory
working_dir = ''

# path of the extern train command will be
extern_train_command_path = os.path.join(working_dir,
        uniqueIdentifier + '_teacher_prefix')

# write out the variable file, and the training command for extern training
# internal training could be implemented, too

# if necessary, you can pass keyword-arguments for the mlp training,
# for example the number of 8 hidden layers and uniform weight intialization
# classifier_args = {'nhid': 8, 'weight_init':'uniform'}

steering_training_data.create_train_data(
    working_dir,
    filenames,
    uniqueIdentifier,
    variable_list,
    convert_to_cpu=False,
    overwrite=True)

# run the external bash command, just with ./bash standard_teacher_prefix
# <prefix>_teacher_prefix

# or execute via subprocess
# or rewrite the function, just as described in mva/tutorials

# in this example, read the written file from disk
with open(extern_train_command_path, 'r') as f:
    command = f.readlines()[0]
    print(command)

# and start the actuall training
job = subprocess.Popen(command, shell=True)
job.wait()


##########################################
# Basic tutorial for the expert

# after the training, you can use the expert with

# basf2_mva_expert --datafile <training_data_file> --treename dft_variables
# --weightfile <uniqueIdentifier> --outputfile <outputfile>.root

# make sure the data_file is in the same format as the training data

# the expert can operate at basf2 steering file level
# an example is provided in test_expert
steering_training_data.test_expert(working_dir, filenames, uniqueIdentifier)
```

Danksagung

Zunächst möchte ich mich bei Herrn Prof. Dr. Michael Feindt für die Möglichkeit, am Institut für Experimentelle Kernphysik über dieses interessante Thema eine Masterarbeit anfertigen zu können und die hilfreichen Ratschläge bedanken.

Außerdem möchte ich mich bei Herrn Prof. Dr. Günter Quast für die Übernahme des Korreferats bedanken.

Bei Herrn Dr. Martin Heck bedanke ich mich für die stets hervoragende Betreuung und hilfreichen Diskussionen.

Ganz herzlich möchte ich mich bei Herrn Thomas Keck für die anregenden Diskussionen, präzisen und fundierten Antworten auf Fragen, sowie für das Korrekturlesen von Teilen dieser Arbeit bedanken.

Des Weiteren möchte ich bei Herrn Dr. Pablo Goldenzweig, Herrn Nils Braun und Herrn Moritz Gelb für das Korrekturlesen von Teilen dieser Arbeit bedanken.

Für weiteres Korrekturlesen bedanke ich mich bei Herrn Karsten Wolff.

Zudem bedanke ich mich bei allen, hier namentlich nicht erwähnten Mitgliedern der B-Physik-Arbeitsgruppe am IEKP für eine angenehme Arbeitsatmosphäre und interessante Diskussionen.

Außerdem möchte ich mich bei Herrn Dr. Suren Chilingaryan vom Institut für Prozessdatenverarbeitung und Elektronik (IPE) für die Möglichkeit bedanken, einen ihrer GPU-Server zu verwenden.

Zuletzt möchte ich mich bei meinen Eltern und meiner Familie, für ihre uneingeschränkte Unterstützung vor und während des gesamten Studiums bedanken.

Ich versichere wahrheitsgemäß, die Arbeit selbständing verfasst, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Abänderung entnommen wurde sowie die Satzung des KIT zur Sicherung guter wissenschaftlicher Praxis in der gültigen Fassung vom 17.11.2014 beachtet zu haben.
Karlsruhe, den 3. August 2016

....

(Jochen Gemmler)