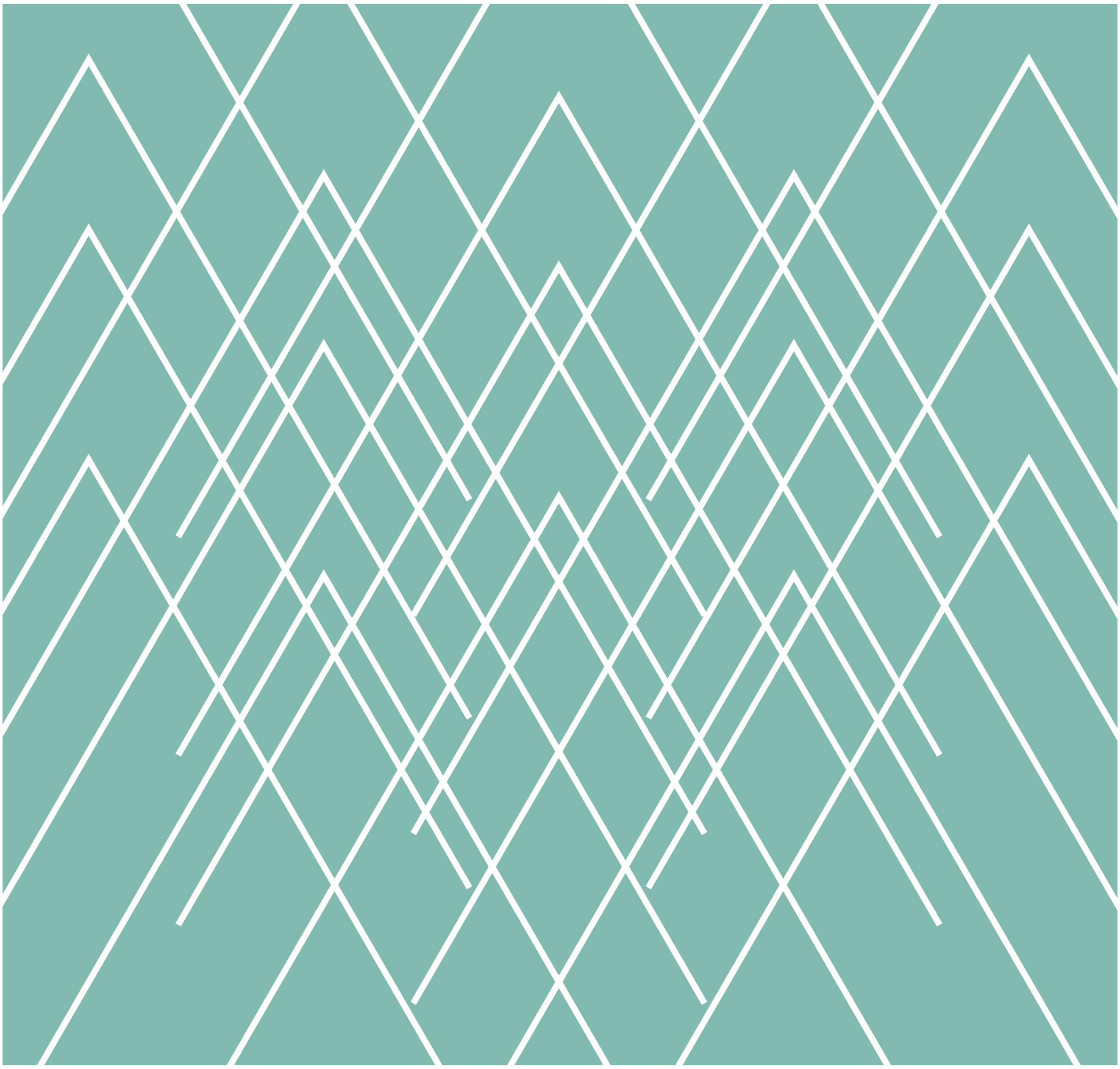
Wernigeröder Automatisierungs- und Informatik-Texte

Ausgabe 01 | 2021

Deep Neural Network Extension to Kinematic Likelihood Fitter

Dirk Neumann



Hochschule Harz
Hochschule für angewandte Wissenschaften

WAIT

Impressum

Inhaltlich verantwortlich

Autor/-in der Abschlussarbeit

Institution

Der Fachbereich Automatisierung und Informatik ist ein Fachbereich der Hochschule Harz. Die Hochschule Harz ist eine Körperschaft des öffentlichen Rechts. Sie wird durch den Rektor Prof. Dr. Folker Roland gesetzlich vertreten: info@hs-harz.de.

Umsatzsteuer-Identifikationsnummer

DE231052095

Adresse

Hochschule Harz
Fachbereich Automatisierung und Informatik
Friedrichstraße 57-59
38855 Wernigerode

Kontakt

Dekanin des Fachbereiches Automatisierung und Informatik
Prof. Dr. Andrea Heilmann
Tel.: +49 3943 659 300
Fax: +49 3943 659 399
E-Mail: dekanin-ai@hs-harz.de

Aufsichtsbehörde

Das Ministerium für Wirtschaft, Wissenschaft und Digitalisierung des Landes Sachsen-Anhalt (MW), Hasselbachstraße 4, 39104 Magdeburg, ist die zuständige Aufsichtsbehörde.

ISSN 2702-2293

Haftungsausschluss

Die Hochschule Harz weist auf Folgendes hin:

Die Hochschule Harz ist lediglich für die Veröffentlichung der einzelnen Werke zuständig, sie übernimmt keinerlei Haftung. Vielmehr gilt Folgendes:

- für den Inhalt der Publikation ist der/die Autor/-in verantwortlich
- mit der Erfassung in der Schriftenreihe Wernigeröder Automatisierungs- und Informatik-Texte verbleiben die Urheberrechte beim Autor/bei der Autorin
- die Einhaltung von Urheber- und Verwertungsrechten Dritter liegt in der Verantwortung des Autors/der Autorin

Vor Veröffentlichung bestätigte der/die Autor/-in,

- dass mit der Bereitstellung der Publikation und jedes Bestandteils (z.B. Abbildungen) nicht gegen gesetzliche Vorschriften verstoßen wird und Rechte Dritter nicht verletzt werden
- dass im Falle der Beteiligung mehrerer Autoren am Werk der/die unterzeichnende Autor/-in stellvertretend im Namen der übrigen Miturheber/-innen handelt
- im Falle der Verwendung personenbezogener Daten den Datenschutz (durch Einholen einer Einwilligung des Dritten zur Veröffentlichung und Verbreitung des Werks) zu beachten
- dass im Falle einer bereits erfolgten Veröffentlichung (z.B. bei einem Verlag) eine Zweitveröffentlichung dem Verlagsvertrag nicht entgegensteht
- dass die Hochschule Harz von etwaigen Ansprüchen Dritter (z.B. Mitautor/-in, Miturheber/-in, Verlage) freigestellt ist

Hochschule Harz

GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

Bachelor's Thesis

Deep Neural Network Extension to Kinematic Likelihood Fitter

Erweiterung des Kinematic Likelihood Fitters durch tiefe neuronale Netze

prepared by

Dirk Neumann

at the Departement of Automation and Computing Science
at Harz University of Applied Science
and the Second Institute of Physics
at Georg-August-University Göttingen

Thesis number:	BA AI 53/2020
Thesis period:	1st June 2020 until 24th August 2020
First referee:	Prof. Dr. Frieder Stolzenburg
Second referee:	Prof. Dr. Arnulf Quadt

Hochschule Harz
Fachbereich Automatisierung und Informatik

Thema und Aufgabenstellung der Bachelorarbeit
BA AI 53/2020
für Herrn Dirk Neumann

Deep Neural Network Extension to Kinematic Likelihood Fitter/ Erweiterung des Kinematic Likelihood Fitters durch tiefe neuronale Netze

Im Rahmen der ATLAS-Kollaboration werden am CERN im Large Hadron Collider Protonen beschleunigt und zur Kollision gebracht. Dabei finden unter anderem sogenannte Top-Antitop-Ereignisse statt, bei welchen aus dem Protonenzusammenstoß ein Topquark und ein Antitopquark entstehen. Diese Teilchen zerfallen dann in weitere Teilchen im Endzustand, welcher beim einfach-leptonischen Zerfallskanal aus sechs Teilchen besteht. Die Zerfallsprodukte müssen im Detektor und mit dedizierten Analysetechniken und Algorithmen rekonstruiert werden, was durch Untergrundprozesse und Messungenauigkeiten erschwert wird. Um eine möglichst gute Rekonstruktion der Top-Quark-Ereignisse zu erhalten, werden mit Hilfe des „Kinematic Likelihood Fitter“ (KLF) verschiedene Teilchenzuordnungen berechnet. Im weiteren Verlauf selektieren Datenanalysten die jeweils wahrscheinlichsten Zuordnungen für ihre Analysen und filtern diese gegebenenfalls weiter. Auf KLF aufbauend wurde in der Doktorarbeit von Dr. Tomas Dado eine Möglichkeit aufgezeigt, mit einem Boosted Decision Tree die Reinheit der korrekten Teilchenzuordnungen zu erhöhen.

Aufbauend auf diesem Ansatz soll ein tiefes neuronales Netz (engl. DNN) entwickelt werden, um die Bewertung der wahrscheinlichsten Teilchenzuordnung in einem ersten Schritt zu rekonstruieren und zu verbessern. Im Anschluss daran soll ein weiteres DNN entwickelt werden, welches Aussagen zu Teilen der Rekonstruktion ermöglicht. Dadurch sollen dessen Nutzer die Möglichkeit erhalten, zugeschnitten auf ihre Problemstellung und mit einer höheren Präzision, einen größeren Datensatz an relevanten Ereignissen aus den ursprünglich gemessenen Daten herauszufiltern.

Die Bachelorarbeit beinhaltet folgende Teilaufgaben:

- Rekonstruktion der bestehenden Lösung
- Entwicklung und Analyse einer vergleichbaren Lösung mit Hilfe eines DNN
- Entwicklung einer erweiterten DNN-Lösung
    - Betrachtung umfangreicherer Ausgangsdaten
    - Erweiterung der Aussagemöglichkeiten
    - Anwendung verschiedener Methoden des maschinellen Lernens
- Bewertung der Ergebnisse

Friedrich Stolzenburg
Prof. Dr. F. Stolzenburg
1. Prüfer

Prof. Dr. A. Quadt
2. Prüfer

Prof. Dr. Arnulf Quadt
Georg-August-Universität Göttingen
II. Physikalisches Institut
Friedrich-Hund-Platz 1
37077 Göttingen
aquadt@uni-goettingen.de

Abstract

As a part of the experiments of the Atlas collaboration, protons are accelerated and brought to collision in the Large Hadron Collider (Lhc) at Cern. So-called top-quark pair production events take place in which a top-quark and an antitop-quark are created from the parton interactions from the underlying proton collision. These particles then decay into new particles in the final state. To study the top-quark, the decay products must be reconstructed in the detector which is complicated by background processes and measurement uncertainties. In order to obtain the best possible reconstruction of the top-quark events, the permutations of possible particle assignments and their probabilities are calculated with the help of the *Kinematic Likelihood Fitter*.

Based on *Kinematic Likelihood Fitter* and a boosted decision tree enhancement to it, in a first step a deep neural network was developed to reconstruct and improve the evaluation of the most probable particle permutation. Subsequently, another deep neural network architecture was developed which allows statements about parts of the reconstruction. This enables users to filter out a larger dataset of relevant events from the originally measured data, which can be tailored to their problem and has a higher purity and precision.

Zusammenfassung

Im Rahmen der Atlas-Kollaboration werden am Cern im Large Hadron Collider (Lhc) Protonen beschleunigt und zur Kollision gebracht. Dabei finden unter anderem sogenannte Topquark-Paarproduktionen statt, bei welchen aus dem Protonenzusammenstoß Partonen und aus deren Interaktion ein Topquark und ein Antitopquark entstehen. Diese Teilchen zerfallen dann in weitere Teilchen im Endzustand. Um das Topquark untersuchen zu können, müssen die Zerfallsprodukte im Detektor rekonstruiert werden, was durch Untergrundprozesse und Messungenauigkeiten erschwert wird. Um eine möglichst gute Rekonstruktion der Top-Quark-Ereignisse zu erhalten, werden verschiedene Teilchenzuordnungen und und deren Wahrscheinlichkeiten mit Hilfe des *Kinematic Likelihood Fitter* berechnet.

Aufbauend auf *Kinematic Likelihood Fitter* und einer Erweiterung durch einen Boosted Decision Tree, wurde ein tiefes neuronales Netz entwickelt, um die Bewertung der wahrscheinlichsten Teilchenzuordnung in einem ersten Schritt zu rekonstruieren und zu verbessern. Im Anschluss daran wurden weitere tiefe neuronale Netze entwickelt, welche Aussagen zu Teilen der Rekonstruktion ermöglichen. Dadurch erhalten dessen Nutzer die Möglichkeit, zugeschnitten auf ihre Problemstellung und mit einer höheren Präzision, einen größeren Datensatz an relevanten Ereignissen aus den ursprünglich gemessenen Daten herauszufiltern.

Contents

1 Introduction

The research field of particle physics is currently based on analyses of very large datasets. Whether simulated or generated in particle accelerators, the volume of data poses increasing information technology problems. Researchers are faced with issues like storing, evaluating and keeping data available. Current trend topics such as grid computing and data science were, to a large extent, developed in the context of particle physics research and others, like machine learning, are used frequently [1–4]. The ATLAS Collaboration, one of the research collaborations at CERN, is involved in many of these research fields and development projects. Top-quark-antitop-quark-events ($t\bar{t}$-events) are part of this research and provide the raw data for the studies presented in this thesis. They will be explained in detail in Chapter 2 together with the *Standard Model of Particle Physics* (SM). Once the data has been collected, the reconstruction of the events is an important step. As much information as possible must be collected correctly in order to have enough data for further analyses like the top-quark decay width measurement. The Kinematic Likelihood Fitter (KLFitter) has been developed for this purpose [5]. It builds the basis for this thesis and will be explained in Chapter 4. Several tailored extensions to KLFitter have been developed whenever a researcher needed an additional step of analysis for his/her work. The boosted decision tree (BDT) developed by T. Dado is one of them, which increases the precision of the KLFitter results [6]. Both KLFitter and the BDT extension aim at finding the completely correct reconstruction of the measured or simulated event. Depending on the subsequent research, completely correct events are not necessarily required. In some cases, however, a partly correct reconstructed event is sufficient. For this purpose, a new extension to KLFitter, based on deep neural networks, is to be developed. After introducing machine learning, especially neural networks, in Chapter 5, the existing BDT approach is reimplemented with a neural network in Section 6.1 and compared in Section 6.2. Subsequently, the new extension to KLFitter is summarised. First, the selection of the new reconstruction hypotheses and then the workflow of the approach and its implementation are discussed. This new extension is evaluated in Section 6.4 and compared with the other approaches. Finally, the results are summarised and the potential for optimisation and further development is shown in Chapter 7.

2 Fundamentals of Particle Physics

To solve a problem in computer science, detailed domain knowledge is often indispensable. Although one might not need and will probably not reach a domain expertise as big as that of the domain experts, it is important to share a common language and to understand logical implications and problems in that particular domain. This chapter will give an overview over the basics of particle physics in terms of the SM. Subsequently, a deeper dive is taken into the kind of events this thesis deals with.

2.1 Standard Model of Particle Physics

The SM is the current attempt to explain the knowledge of the basic components of our universe and most of their interactions within one model. It contains information about three out of four of the elementary interactions and the particles which either mediate them or experience them. The particles which mediate the interactions are called gauge bosons and so far four of them are known. The W-boson and the Z-boson which mediate the so-called weak interaction [7–10], the difference being that the W-boson is charged and the Z-boson is not. The strong interaction is mediated by the gluons [11–14]. It describes the attraction of particles based on their colour charge which can be (anti-)red, (anti-)green or (anti-)blue. The third interaction is the electromagnetic interaction which is transferred by the photons. In addition to the four gauge bosons, there is the Higgs boson. It is the most recently discovered particle [15, 16], although it has already been postulated in 1964 [17–19]. This boson does not mediate any interaction but is needed to explain why W-bosons, Z-bosons and fermions are massive.

The fermions are the 12 basic particles which experience the described interactions and they can be divided into two groups and three generations: The six quarks which then are divided into up-type quarks and down-type quarks are one of the groups. Its first generation consists of the up-quark (u) and the down-quark (d), the second generation of the charm-quark (c) and strange-quark (s) and the third generation of top-quark (t) and bottom-quark (b). u, c and t are the up-type-quarks which have a charge of $+\frac{2}{3}e$ and d, s and b are the down-type-quarks which have a charge of $-\frac{1}{3}e$, where e means the elementary charge $e \approx 1.602 \cdot 10^{-19}$ C [20].

The second group of fermions are the leptons which can be divided into charged leptons and neutral leptons, also called lepton neutrinos. Electron (e), muon (μ) and tau (τ) in this order are the charged leptons of the three generations and their corresponding lepton neutrinos are the electron neutrino (ν_e), the muon neutrino (ν_μ) and the tau neutrino (ν_τ). Neither charged leptons nor their uncharged correspondent have a colour charge and therefore do not experience the strong interaction and while charged leptons have an electric charge of -1 and therefore experience the electromagnetic interaction, the lepton neutrinos are not charged and do not experience the electromagnetic interaction either.

Using this model, many phenomena can be explained. A proton for example consists of one down-quark and two up-quarks. The electro static charges $(+\frac{2}{3}+\frac{2}{3}-\frac{1}{3})$ sum up to one positive charge. The three quarks need to have different colour charges such that red, green and blue sum up to a colourless state (white). For each particle there is an antiparticle with the opposite electromagnetic charge and opposite (negative) colour charge. This gives an explanation why antimatter can exist and how mesons, which consist of one quark and one anti-quark can be stable considering that, for example, a red quark and an anti-red quark build form colourless state (anti-white/ black).

Other phenomena cannot be explained with the SM and are still to be explained. This gives a hint that the SM is not complete yet or might be just another step on the way to a more detailed and sophisticated model to explain nature. Gravity, the fourth interaction, is not explained so far and no boson which could mediate this interaction, e.g. a “Graviton”, has been found yet. Also the question of why the universe consists mainly of matter and not antimatter, so why is there so much more matter than antimatter, cannot be explained with the SM. Another example is that lepton neutrinos are defined as being massless by the SM but other experiments [21–23] proved that they need to have mass in order to make the observations from these experiments explicable.

In conclusion, the SM is an attempt, and the best so far, to explain and define the basic nature of our universe. The model predictions have proven to be accurate several times, e.g. when particles like the top-quark and the Higgs boson were postulated in advance and then were found later, although that, like for the lepton neutrinos, deviations between theory and practice occurred.

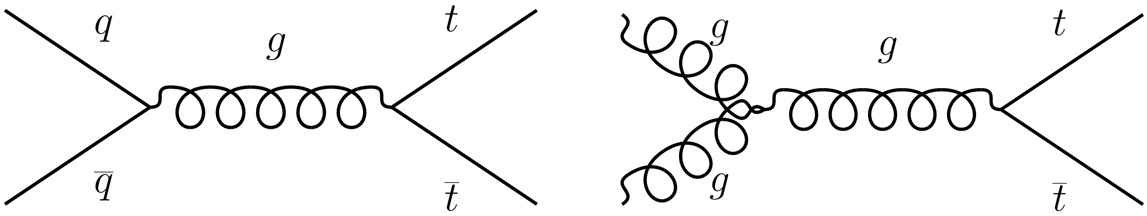
2.2 Top-Antitop Production and Decay

As explained in the previous chapter, there are six quarks and they differ in their properties. The top-quark is the latest found quark and was discovered at the TEVATRON in 1995 [24, 25]. It is of particular interest because of its high mass at approximately 173 GeV [26]. It is so heavy that it decays into other particles before it can hadronise. The same applies to the antimatter version of the top-quark, which is called the antitop-quark ($\bar{t}$).

To work with and do researches on them, they need to be produced in a particle accelerator. From a physics perspective, there are two possibilities to produce a top-antitop-pair. The *quark-antiquark annihilation* is the production of a top-antitop-pair from the collision of a quark and an antiquark and is shown in Figure 2.1(a), whereas Figure 2.1(b) shows the *gluon-gluon fusion* in which the collision takes place between two gluons.

After the top-quark and antitop-quark are produced, they decay almost immediately. The total electric charge of a system is always constant and the top-quark (charge $= +\frac{2}{3}$) decays almost exclusively into a bottom-quark (charge $= +\frac{2}{3}$) and a W^+-boson (charge $= +1$) [27]. Analogous to this the antitop-quark decays almost always into an antibottom-quark ($\bar{b}$) and a W^--boson, where the antimatter versions of the particles are charged oppositely. The bottom-quark and the antibottom-quark build hadrons which decay again. The decay products then build hadrons again and so on which results in the formation of a shower of particles, also called *particle jet*. This is called hadronisation. Each of the two W-bosons can decay either leptonically or hadronically which results in four possible Feynman diagrams which are shown in Figure 2.2.

A hadronic decay which occurs around 67.4% of the time [28] means that a W-boson decays into two quarks of which one is an antimatter-quark. Figure 2.2(a) shows the

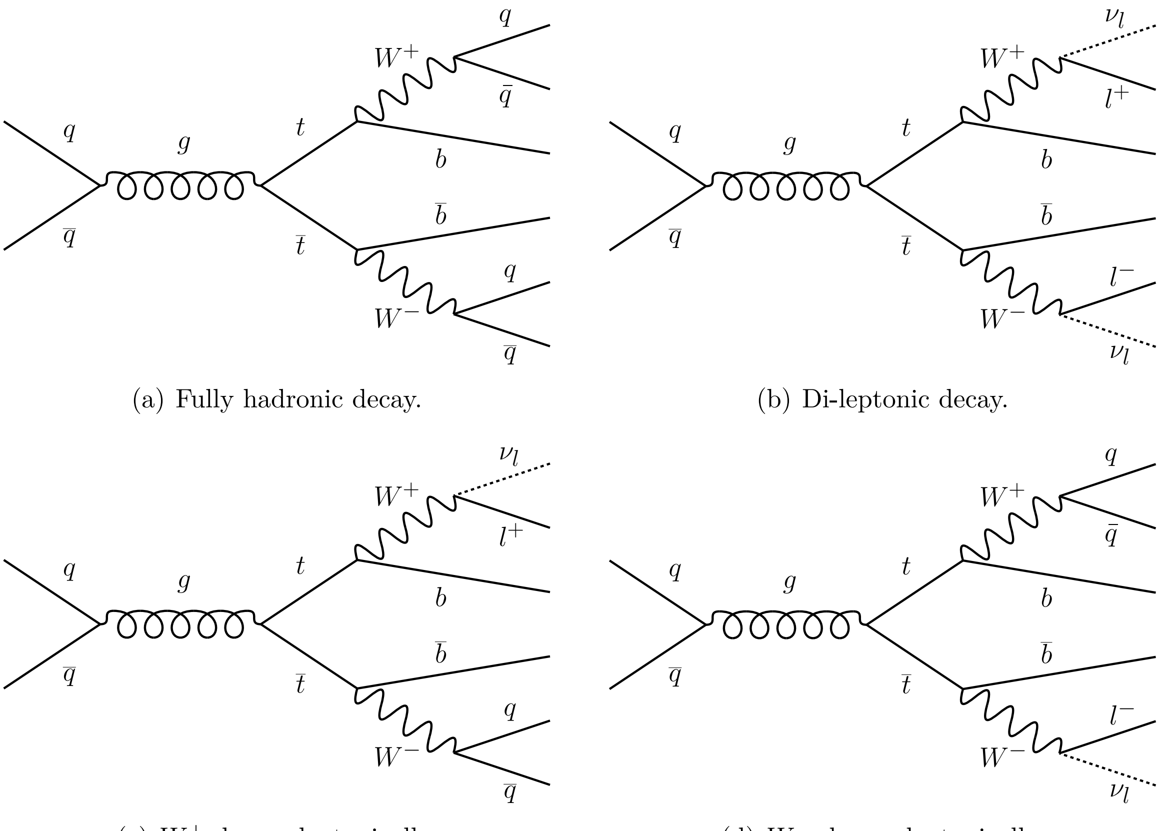


(a) Quark-antiquark annihilation.

(b) Gluon-gluon fusion.

Figure 2.1: Top-antitop-pair productions, where two different production processes are possible. The quark-antiquark annihilation as shown in Figure 2.1(a) consists of the collision of a quark and its respective antiquark. Figure 2.1(b) shows the gluon-gluon fusion process which is the dominant one at the top-antitop-production at the LHC.

case where both W-bosons decay hadronically. To preserve the electric charges, the W^+-boson has to decay into an up-type-quark and an anti-down-type-quark [1] which both hadronise. If both W-bosons decay leptonically, as shown in Figure 2.2(b), no quarks are produced but, instead, each W-boson decays into one charged lepton and one lepton neutrino. Since the lepton neutrino has no electric charge, the charged lepton carries the whole electric charge of the W-boson. Figure 2.2(c) and Figure 2.2(d) show the cases where one W-boson decays hadronically and the other decays leptonically, following the rules as previously described.



(a) Fully hadronic decay.

(b) Di-leptonic decay.

(c) W^+ decays leptonically.

(d) W^- decays leptonically.

Figure 2.2: Feynman diagrams of top-antitop-decay processes.

These last two, also called semi-leptonic decay processes, are subject of the analysis presented in this thesis. Other decay processes are filtered out and will not be considered further. To note also is that the different possibilities of the production of a top-antitop-pair and the possible decay processes are independent from each other. Therefore, the Feynman diagrams in Figure 2.2 do not imply that certain productions of the top-antitop-pair result in certain decay processes of it or vice versa.

[1] the W^--boson vice versa

3 The Atlas Detector

The dataset which defines the basis for the studies presented in this thesis is artificially produced by a Monte Carlo simulation. This simulation consists of multiple steps and models the technical conditions under which data is normally produced at the Large Hadron Collider (Lhc) at the *European Organization for Nuclear Research* (Cern). Therefore in this chapter a closer look is taken towards the experimental setup to produce and measure $t\bar{t}$-events. Atlas [29] is together with Alice [30], Cms [31] and LhCb [32] a detector at the Lhc, which is currently the largest particle accelerator worldwide. Atlas and Cms are multipurpose detectors. Both are doing research in various fields of particle physics, test the theory and try to disprove the predictions of the SM. LhCb, which stands for "Lhc beauty", is an experiment that is dedicated towards hadron and bottom-quark physics and the charge conjugation parity violation. The Alice experiment focuses on physics at high densities and temperatures, whereas the main focus is on quark-gluon plasma formation. The Atlas experiment is located approximately 200 m underground and has a circumference of 27 km. The colliding particles in this case are protons which are produced and pre-accelerated by older particle accelerators to get the needed energy for the collision in the Lhc. These protons collide at the so-called *Interaction Points* of which four exist, one for each detector. The protons have a center-of-mass energy of 13 TeV when they collide. This collision leads to the processes and decays explained in 2.2. The resulting particle jets and the charged lepton can then be measured with the Atlas detector.

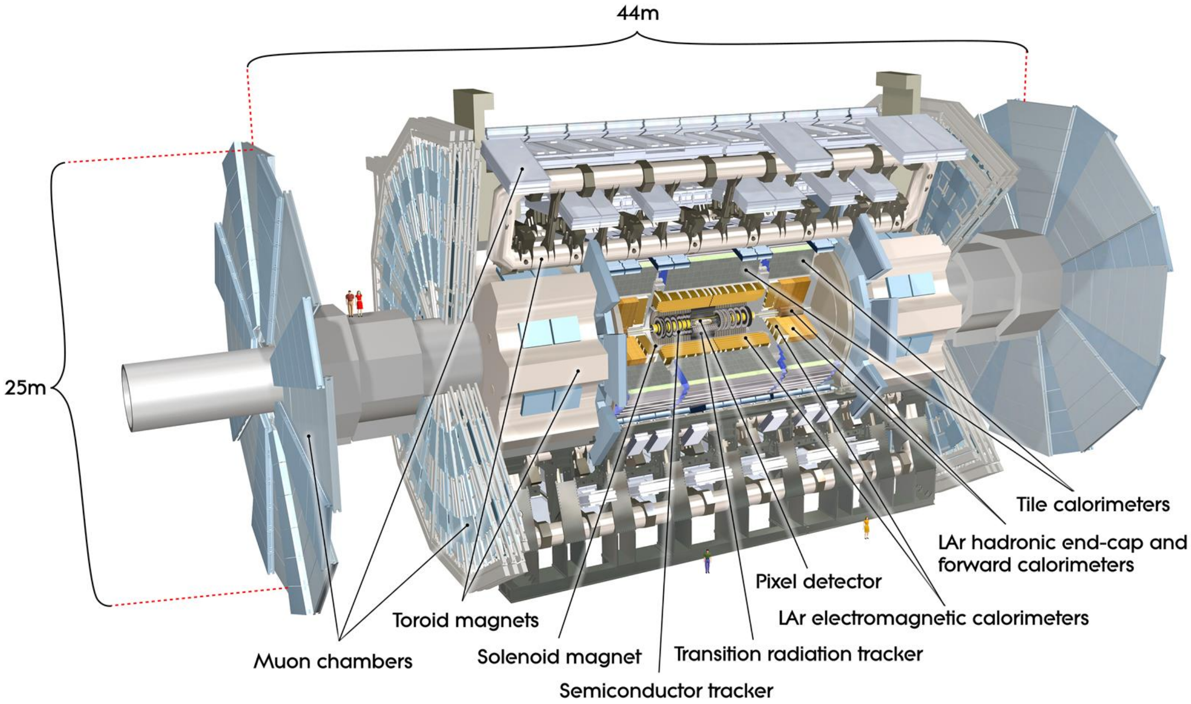

44m
25m
Tile calorimeters
LAr hadronic end-cap and forward calorimeters
Pixel detector
LAr electromagnetic calorimeters
Toroid magnets
Muon chambers
Solenoid magnet
Transition radiation tracker
Semiconductor tracker

Figure 3.1: Cross section of the ATLAS detector © CERN.

The detector, which can be seen in Figure 3.1, consists of multiple layers in concentric arrangement. The *Inner Detector* measures the tracks of particles which pass through it. The movement of charged particles within the strong magnetic field inside the detector generates a Lorenzforce which deflects them. From the diffraction of the track the charge and traverse momentum (p_{T}) can be calculated. The *Calorimeters*, which are further away from the center, are for the measurement of the particle energies. In the case of the $t\bar{t}$-event, this applies to the particle jets, of which the single energies of the particles get combined to the total energy of the particle jet. At this stage, errors can occur due to additional radiation or merging of jets. Additional radiation can cause the production of additional jets and the merging of particle jets leads to less jets measured than present if two or more jets are too close to each other such that they get combined to one large jet by the detector. The outermost part are the *Muon chambers*. Muons barely interact with matter and therefore pass the *Inner Detector* and the *Calorimeters* with almost no energy loss. Their energy is then measured in the *Muon chambers*.

4 Event Selection and Top-Quark Reconstruction

This chapter gives an overview about how the dataset was generated which is used for the studies presented in this thesis. Section 4.1 explains the simulation of the generated events. Subsequently, Section 4.2 introduces the *Kinematic Likelihood Fitter* (KLFitter) which builds the starting point for the further development.

4.1 Monte Carlo Simulation and Event Selection

To precisely measure parameters and whether they are in accordance with their SM prediction, simulations of its physics processes are needed which then can be compared with real data measurements in the ATLAS detector. The dataset which is used for the studies presented in this thesis is not collision data recorded with the ATLAS detector but is simulated by a Monte Carlo (MC) simulation.

The MC simulation consists of multiple steps. At first the $t\bar{t}$-events are simulated at matrix element level. Additional processes like the radiation explained in Chapter 3 are mimicked. Subsequently, the hadronisation and particle showers are simulated. The response of the ATLAS detector to these events is then mimicked with the GEANT4 detector simulation [33] and the events get reconstructed with the same software which is used for real events in the detector. Since this thesis was motivated by a recent top-quark decay width analysis [6], the same dataset is used in this thesis which also keeps the results comparable to those of the top-quark decay width analysis. Therefore, detailed information about the used simulation programs can be obtained from this analysis as well and are not described in detail.

After the dataset has been produced, the relevant events are selected. For the $t\bar{t}$-events in the semi-leptonic decay channel, a lepton and at least four jets are required. All events that do not have exactly one electron or muon are therefore filtered out. In respect to the jets, at least four have to be measured with a transverse momentum $p_{\mathrm{T}} > 25\,\mathrm{GeV}$ out of which two have to originate from bottom-quarks. To determine whether or not a jet originated from a bottom-quark, the ATLAS *MV2c10* b-tagging algorithm is used [34]. It uses a boosted decision tree (BDT) and is calibrated to four different target tagging efficiencies, also called *working points* (WP). For the selection used in this thesis, the 60%-WP is used which means that 60% of the jets which are really bottom-quark-jets (b-jets) pass the selection. In addition, for the $e+$ jets channel, the missing transverse energy, $E_{\mathrm{T}}^{\mathrm{miss}}$, and the transverse mass of the W-boson, $m_{\mathrm{T}}^{\mathrm{W}}$, both have to be larger than 30 GeV, for $\mu+$ jets $E_{\mathrm{T}}^{\mathrm{miss}} + m_{\mathrm{T}}^{\mathrm{W}} > 60\,\mathrm{GeV}$ has to be fulfilled.

4.2 Kinematic Likelihood Fitter

As described in the previous section, a measured event consists of multiple measured jets, a charged lepton and some missing transverse energy from the lepton neutrino. The values which are obtained are energies, angles and momenta and, additionally, some constraints like the conservation of energy and momentum are known. With this information, the goal is to find pairings between the measured jets and the partons in the semi-leptonic decay channel which are expected to produce jets. If four jets are expected to be produced from a $t\bar{t}$-event and in total n jets are measured, then there are $\binom{n}{4} \cdot 4! = \frac{n!}{4! \cdot (n-4)!} \cdot 4! = \frac{n!}{(n-4)!}$ assignments possible for $n \geq 4$. If less than four jets are measured or errors occur as explained in the previous chapter, the event cannot be reconstructed correctly.

KLFitter is a software tool which was implemented to solve this problem [5]. In terms of software, it is based on the Bayesian Analysis Toolkit [35], which is a toolkit for particle physics calculations. From a conceptual perspective, it combines two prior approaches, the ${p_{\mathrm{T}}}^{max}$-method and the χ^2-method. The new approach, which is introduced in KLFitter, is the likelihood based method.

This method calculates for each chosen assignment a likelihood which indicates if the given permutation represents a $t\bar{t}$-event. Since the calculation of the likelihoods is expensive in terms of calculation time in respect to millions of events, the set of jets to choose from can be reduced. Additionally, the number of permutations to calculate is reduced by leaving out the *light jet swapping*. Light jet swapping refers to two permutations which only differ in the assignment of the jets from the hadronically decaying W-boson. Since the algorithm is not sensitive to the difference between these two assignments, the number of permutations which needs to be considered is reduced by a factor of 2.

To calculate the likelihood of one permutation, its kinematics are fitted to the assumed assignment by using a likelihood fit. The likelihood gets stored and the procedure is repeated for the rest of the permutations. Based on these likelihoods, an event probability can be calculated to decide which permutation is then assumed to be the correct one.

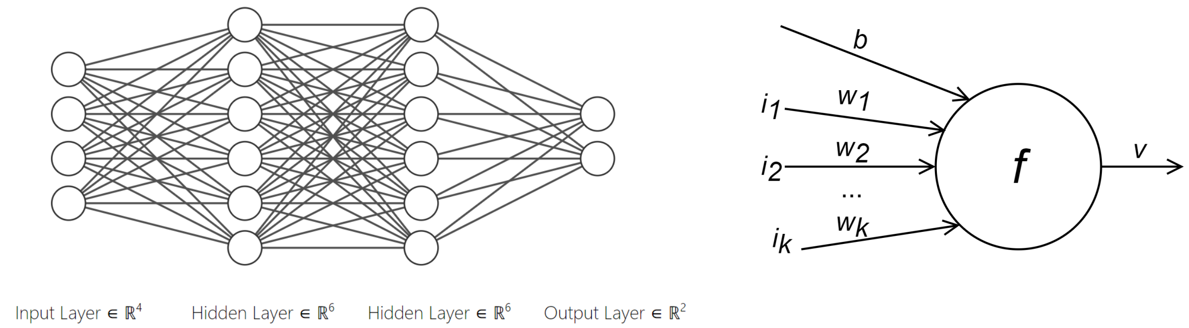
5 Fundamentals of Machine Learning

In classic programming, the program represents a set of rules and instructions to produce outputs for given inputs. In contrast to this, machine learning represents a programming paradigm where the rules are not given explicitly. Instead, an artificial intelligence has to learn the rules from a set of inputs and their respective outputs. One of these machine learning techniques is the neural network (NN). This chapter will give a short overview over neural networks and some additional techniques which are used in the context of the studies presented in this thesis.

5.1 Neural Networks and Deep Learning

In 1943, the first attempt was made to describe an artificial neuron from a mathematical perspective by McCulloch and Pitts [36]. Based on this idea, Rosenblatt developed the *Perceptron* in 1958 which was the first neural network [37] consisting of an *input layer* and an *output layer*. With adding at least two additional *hidden layers*, the *multi layer perceptron*, also called *deep neural network* (DNN), can be created as shown in Figure 5.1(a). As long as the output of a node is not fed into its own layer or a layer before, the NN is called a *feedforward neural network*. Today, DNNs are a widely used machine learning techniques [38–41].

The concept of a DNN is to map an input vector also called a *feature vector* $\vec{x} \in \mathbb{R}^n$ to an output vector $\vec{y} \in \mathbb{R}^m$ where usually $n > m$ with $m, n \in \mathbb{N}$. This is achieved by training the DNN with a training dataset. Each elementary unit in the DNN, *node*, gets a number n of inputs i_k which are weighted with a weight w_k. The sum of these weighted inputs and an optional additional bias term b is then the input for the *activation function* f. The node, shown in Figure 5.1(b), calculates one value v as output as shown in Equation (5.1) which then is either the input for the next hidden layer or part of $\vec{y}$.



(a) Example plot of a neural network. The two hidden layers define it as a deep neural network. It maps the feature vector $\vec{x} \in \mathbb{R}^4$ to the output vector $\vec{y} \in \mathbb{R}^2$.

(b) Example plot of a single node within a neural network. The inputs i_k, weighted with w_k, are in addition with the bias term b the input for the activation function f which returns the output v. v is used as part of the output vector $\vec{y}$ or as input in the next layer.

Figure 5.1: Example plots of a feedforward neural network. Figure 5.1(a) shows a whole neural network architecture while Figure 5.1(b) shows a single node with its in and output values.

$$v = f(\sum_{k=1}^{n}(w_\mathrm{k} \cdot i_\mathrm{k}) + b) \qquad w_\mathrm{k}, i_\mathrm{k} \in \mathbb{R} \text{ for } 1 \geq k \geq n; \quad b, v \in \mathbb{R}; \quad k, n \in \mathbb{N}. \tag{5.1}$$

The goal of the training is to find a set of weights along the edges between the nodes such that the *model* minimises its loss. During the training based on a given input, the *prediction* is calculated which differs from the expected output depending on how well trained the model already is. After a certain number, the so-called *batch size*, of these calculations, an overall error is determined. The overall error is effected by the weights of the different training events, if they are given, and is used to adjust the weights of the model. This algorithm is called *backpropagation* [42]. The *learning rate* effects how large these adjustments are and, therefore, how fast the DNN learns. This procedure is repeated until every data point in the dataset is evaluated once by the model. This is called an *epoch*. After each epoch, the performance of the model can be evaluated which results in some key figures like *loss* and *accuracy*. These key figures can be used to monitor the training over multiple epochs and sum up the training history in learning curves.

Normally the accuracy rises with the number of epochs while the loss decreases. Depending on the *separation power* of the input data and *hyperparameters* such as the number of layers and nodes, an optimum will be reached where further training will not have any beneficial effect on the model's performance. An issue here is the so called *overfitting*. This can occur if the model is trained too long on the same dataset. The model then tends to learn the given training data points by heart which results in a worse performance at predicting data which is not in the training dataset. Therefore validation datasets can be used to check whether or not the model generalises well or starts to overfit.

A prominent misconception in training neural networks is the so-called *curse of dimensionality* which occurs when $\vec{x}$ has many entries. Against the idea that more information gives more separation, it gets harder for the DNN to learn. The more dimensions the phase space has in which the input vectors lie and the more possible values in this dimensions exist, the harder the training is. This is because there are many more possible configurations for an input vector. Without enough data points to populate this phase space, no generalised learning is possible. Generalised learning is mainly based on the finding of nearest neighbours and the assumption that neighbouring data points are alike. With increasing distances between data points and empty spots in the phase space, the model can only remember the results for the training data. The model overfits [43].

5.2 Additional Techniques

Although artificial intelligence algorithms are improving further and the computing resources available increase as well as the training datasets grow quickly, artificial intelligence is often limited by the quality of the data provided and the constraints set. To prevent some of the most common problems during the training, many additional techniques have been developed in the past of which a few will be presented in this section.

Input Normalisation

If the numeric values of one feature are much higher or smaller than for the other features the DNN tends to correct this by adjusting the weights of this feature. This is a disadvantage because big deviations in the weights lead to large deviations in propagating back errors from the loss calculations after one batch was evaluated. *Input Normalisation*, also called scaling of input variables, is an additional step before training in which all input variables are scaled into the same interval [44].

Reweighting

Reweighting is a technique to prevent biases during the training due to highly imbalanced training datasets [45]. Imbalanced training datasets occur when the dataset contains many examples of one category c and only a few of the others. The DNN tends to predict every input vector as a member of the majority class to minimise the loss. However the goal of the training is usually not to minimise the loss. To prevent this behaviour each event gets a weight such that:

$$\sum(w_{c_i}) = \sum(w_{c_j}) \quad \forall i, j. \tag{5.2}$$

As mentioned, the weights are used during the evaluation of the current batch during the training. Therefore, events with a larger weight have a larger impact on the error and therefore on the learning.

Dropout

Dropout is a regularisation technique to prevent overfitting [46]. To apply dropout, some of the edges are selected randomly and blocked. Therefore, the model has to learn correlations and separation redundantly and generally. The dropout rate defines how many of the edges are blocked per epoch. This procedure is repeated for each epoch such that different edges are selected for blocking.

Early Stopping

The loss values per epoch will asymptotically approach a minimum. This means the first few epochs boost the performance of the DNN significantly stronger than later epochs in which the learning effect per epoch decreases and the learning gets slower. As soon as the DNN can not improve any more in terms of generalised learning, it starts to overfit. To prevent this behaviour the training can be stopped as soon as the performance of a DNN does not increase by a predefined amount within a certain number of epochs. This procedure is defined as early stopping [47].

Cross Validation and Bagging

An issue during the training of a DNN is the evaluation of the current model. If data is used for the loss calculation, which is also used for the training, the model has seen the data before and therefore the loss value might be less expressive. On the other hand, it is not desirable to minimise the dataset since normally more training data gives better training results. To solve this conflict, *cross validation* can be used. This is a technique in which the dataset is split into $n = t+v$ parts of which t parts are used for the training and v parts for the validation. In total, n models are trained in which each part of the dataset is t times a part of the training dataset and v times a part of the validation set [48]. In the end, the n models can be evaluated to check if they perform similar. Subsequently, either one of the models can be selected to be used further or *bagging* can be done in which multiple predictions are combined to retrieve one statistically more independent prediction [49].

Weight Initialisation

The goal of a DNN is to adjust its weights along the edges to minimise the loss. These weights have to be initialised in the beginning. During the analysis presented in this thesis, the *Keras* [50] framework is used. Its default initialisation algorithm is the *Glorot Uniform Initialiser* [51]. It assigns the weights per layer to each edge as shown in Equation (5.3) whereas n_j is the number of input nodes for these weights and n_{j+1} the number of output nodes for these weights. The bias node weights are initialised with zeros such that the DNN does not start with any bias term.

$$W \sim U\left[-\sqrt{\frac{6}{n_j + n_{j+1}}}, \sqrt{\frac{6}{n_j + n_{j+1}}}\right]. \tag{5.3}$$

6 Development

Given the base conditions from the previous chapters, this chapter will summarise the development of the extension framework to KLFitter. This can be divided into two main steps. The first step is the development of a binary classifier motivated by previous approaches in Section 6.1. Subsequently, a new extension was conceptualized and implemented. Section 6.3 gives a summary over the different steps of this new extension. Both parts of the development are evaluated, respectively, in Section 6.2 and Section 6.4.

6.1 Implementation of a Binary Classifier

For the implementation of a binary classifier, an already existing solution with a BDT is mimicked. This BDT solution was part of a top-quark decay width analysis [6] of which the framework is used for the analysis in this thesis. As a machine learning technique, the DNN is chosen because a DNN has the advantage that it can take correlations between different features into account. The data set used for the training contains approximately 610,000 $t\bar{t}$-events.

The goal of the DNN is to learn whether the KLFitter permutation with the highest event probability value is actually the correct permutation. Since this is a binary question, a binary classifier is used. The training dataset needs to contain samples where the selected permutation is the correct one and some where it is not. Whether the permutation is correct is determined by checking if all four simulated jets of the $t\bar{t}$-event are paired with one of the measured jets whereas "paired" refers to a ΔR smaller than 0.3 between the two jets. Since, by construction, only one KLFitter permutation can be correct, there are eleven wrong permutations per four-jets-event ($\approx 47\%$) and 59 wrong permutations per 5-or-more-jets-event ($\approx 53\%$). Additionally, events have to be taken into account where none of the permutations is correct because jets are merged or one of the correct jets was not selected or measured in the detector simulation. This leads to a large imbalance between correct and wrong permutations ($\approx$ 1 correct : 36 wrong). Additionally, the problem occurs that most of the wrong permutations differ significantly from the assignment in the correct permutation. This can lead to the problem that the DNN learns to classify

Configuration Parameter	**Choosen Value**
Input Features	19
Hidden Layers (HL)	4 layers á 25 nodes
Output Layer (OL)	1 node
Optimizer	Adam [52]
Activation Function (HL)	ReLu(Equation (6.3))
Activation Function (OL)	Sigmoid(Equation (6.1))
Loss Function	Binary Cross Entropy
Validation Split	0.2
Cross Validation	5-fold
Trained Epochs	50
Batchsize	10000
Early Stopping	None
Dropout	None

Table 6.1: Configuration of the trained binary classifier. The chosen values are the product of multiple different test trainings during the implementation and showed good model performance. More training effort in terms of additional or larger hidden layers showed no significant performance increases which would justify the investment of additional computational resources.

almost correct permutations in the same category as completely correct permutations and only the permutations which differ significantly from the correct one are predicted as wrong permutations. To avoid this, the correct permutation (if present) is selected as signal and the most probable permutation for each number of wrong jet assignments (1 - 4 jets incorrect) are selected as background. As mentioned in Section 4.2, the set of permutations is calculated from a reduced set of jets. If more than five jets are measured, the two jets with the highest b-tagging value are selected as well as the three jets with the highest p_T of the remaining jets. The MC event weights are reweighted to balance signal and background data and the input features are normalized in the interval 0-1. This is done by subtracting the minimum and dividing by the maximum per variable. This dataset is used to train the binary DNN. The details of the model trained are shown in Table 6.1.

The sigmoid function, which is shown in Equation (6.1) is often chosen as activation function because of its easy derivability shown in Equation (6.2). Additionally, its outputs are always in the interval 0-1 which is helpful since the binary DNN is supposed to predict values between 0 and 1. However, the sigmoid function comes with a problem concerning the backpropagation. It is called the *vanishing gradient problem* [53–55]. Due to the fact that the maximum value of the derivative of the sigmoid function is $\frac{1}{4}$, the backpropagation

$$\sigma(x) = \frac{1}{1+e^{-x}} \quad x \in \mathbb{R}. \tag{6.1}$$

$$\frac{\partial}{\partial x}\sigma(x) = \sigma(x) \cdot (1 - \sigma(x)) \quad x \in \mathbb{R}. \tag{6.2}$$

$$\mathrm{ReLU}(x) = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases} \quad x \in \mathbb{R}. \tag{6.3}$$

$$\frac{\partial}{\partial x}\mathrm{ReLU}(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{if } x < 0 \end{cases} \quad x \in \mathbb{R}. \tag{6.4}$$

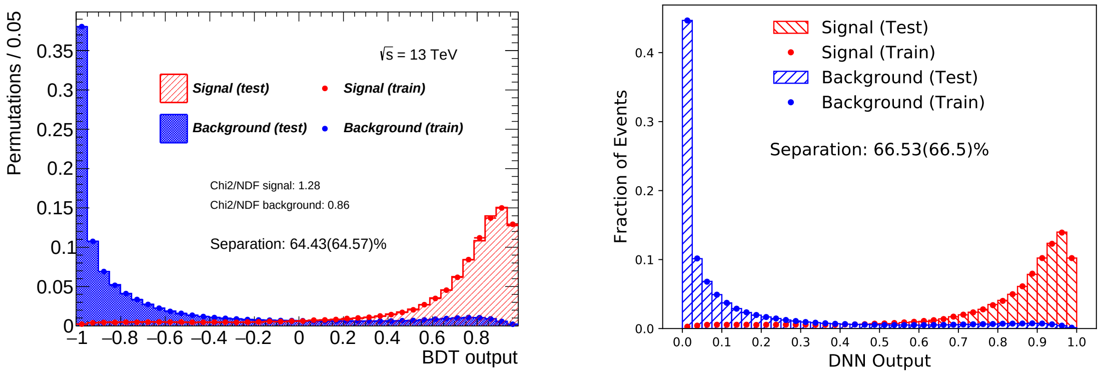
Table 6.2: Activation functions and their derivatives. The sigmoid function, Equation (6.1), is often used in binary classifiers because of its range 0-1 and its simple derivative shown in Equation (6.2). The rectified linear unit (ReLU) function, Equation (6.3), is widely used especially in hidden layers. Due to the property of its derivative, shown in Equation (6.4), of having 0 or 1 as gradient, it is used to avoid the vanishing gradient problem.

of the error gets smaller with each layer. With an increasing amount of hidden layers, the learning of the DNN gets slower. To avoid this problem, the *rectified linear unit* (ReLU) function, which is shown in Equation (6.3), is used in all hidden layers. Its derivative is not limited to a maximum value of $\frac{1}{4}$ as it can be seen in Equation (6.4). Therefore, it does not have the vanishing gradient problem. The ReLU, however, is not limited to output values in the interval 0-1 which is why the combination of both is chosen for the binary classifier.

6.2 Comparison Binary Classifier and BDT

To evaluate a binary classifier, in general, the separation power between signal and background samples is a good key figure. Equation (6.5) shows how the separation is calculated. Figure 6.1(a) and Figure 6.1(b) show the separation power of the BDT and the DNN approach where it can be seen that the DNN outperforms the BDT by approximately 2%.

$$\begin{aligned} &\text{separation} = 0.5 \cdot \sum_{\mathrm{k}=1}^{\mathrm{n}} \frac{(s_{\mathrm{k}} - b_{\mathrm{k}})^2}{(s_{\mathrm{k}} + b_{\mathrm{k}})} \\ &n \text{ ...number of bins} \\ &s_{\mathrm{i}}\text{...fraction of signal events in bin i} \\ &b_{\mathrm{i}}\text{...fraction of background events in bin i.} \end{aligned} \tag{6.5}$$

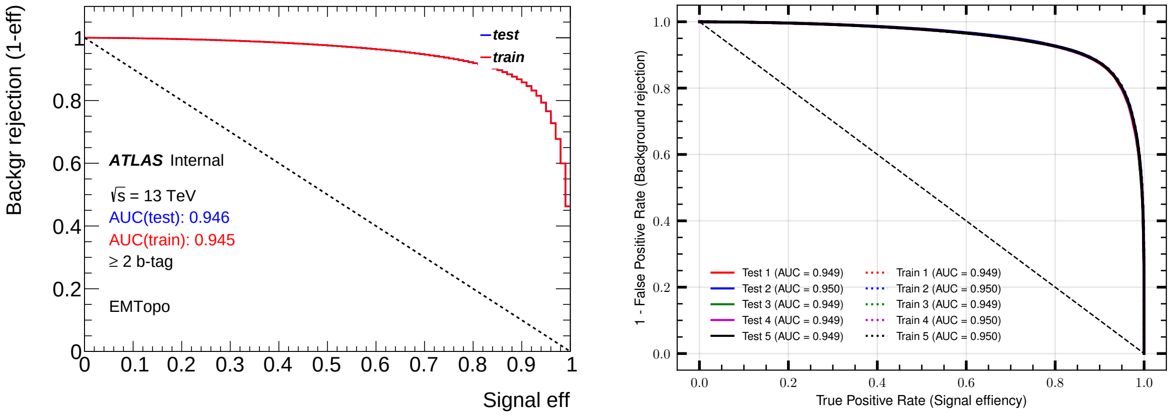


(a) Separation plot of the BDT.

(b) Separation plot of the DNN.

Figure 6.1: Separation plots of the two approaches of the binary classifier. The plots of the BDT (a) and the DNN (b) show the classifier output values grouped in signal and background data. The BDT separation plot is a reconstruction result of the top-quark decay width analysis framework which forms the basis for this thesis [6]. The DNN separation plot is as the DNN itself inspired by its BDT counterpart.

Another option to evaluate the performance of a binary classifier is to use *receiver operating characteristic* (ROC) curves. Figure 6.2(a) and Figure 6.2(b) show the ROC curves of the BDT and the DNN approach. Here the goal is to see how much background is still rejected (specificity) if a certain signal efficiency (sensitivity) is forced by setting a threshold on the output values of the binary classifiers. The *area under curve* (AUC) values shown in the figures are an often used technique to classify the power of a binary classifier on a scale between 0.5 and 1 [56]. The AUC values of the DNN are approximately 0.004 larger which proves that it is slightly better than the BDT. Additionally, the DNN ROC plot shows all five folds from the cross validation. If the curves of training and testing dataset overlap, this is a sign that the models did not overfit. If the different folds look alike, this is a sign that the datasets of training and testing data are evenly distributed in the folds. In this case, both conditions are fulfilled.



(a) ROC curve of the BDT approach. The BDT ROC curve is a reconstruction result of the top-quark decay width analysis framework which builds the basis for this thesis [6].

(b) ROC curve of the DNN approach. The DNN ROC curve is as the DNN itself inspired by its BDT counterpart.

Figure 6.2: *Receiver operating characteristic* (ROC) plots of the two binary classifiers. The plots of the BDT (a) and the DNN (b) set background rejection rate and signal efficiency in relation. The background rejection rate shows how large the fraction is of actual background events which are classified as background. The signal efficiency shows the same situation applied to the signal events. The *area under curve* (AUC) values confirm the superiority of the DNN approach.

6.3 Development of a Multi-Class Approach

Based on the idea of the binary classifier, a multi-class approach is developed. Instead of classifying a single permutation as correct or wrong, the goal is to find the correct one from a set of permutations. This chapter provides an overview over this approach which contains a description of the data used in the different parts, the concept of the approach, the implementation and evaluation.

6.3.1 Definition of Labels and Test Hypotheses

To improve the $t\bar{t}$-event reconstruction, a new approach to classify the data is needed. The goal is to allow partly correct reconstructions to be found in case, that for a certain analysis, a complete reconstruction is not necessary. The idea of the new approach is the prediction and combination of multiple elementary hypotheses to find out whether the event is reconstructed correctly instead of predicting the complete reconstruction directly. The particle jet pairing is the fundamental concept of the reconstruction which is used by KLFitter and therefore also builds the basis for the new approach. This already gives four of the elementary hypotheses which are “The hadronic b-jet (b_{had}) is assigned correctly.”, “The leptonic b-jet (b_{lep}) is assigned correctly.”, “The first light jet (lj_1) is assigned correctly.” and “The second light jet (lj_2) is assigned correctly.”.

One of the input variables is the b-tagging value as explained previously. The data in the used dataset is selected such that two of the jets are tagged as b-jets with the 60% WP. The problem might occur that the two assigned b-jets are swapped. Therefore, the additional hypothesis “The b-jets are swapped.” is introduced.

Another goal of the new approach is to learn the correct assignment of the two light jets. They differ mostly in the flavour of the particles they originated from. One of them originated from a light quark with an up-type flavour. The other light jet originated from a down-type light quark. Both of them are quarks of the lighter first two generations. To train especially the difference between the correct and the swapped light jet assignment, the sixth hypothesis “The light jets are swapped.” is used.

These six hypotheses form the basis for further combined reconstruction hypotheses. For example, the correct reconstruction of the hadronic top-quark t_{had} can be retrieved from logical combinations of the basic labels as shown in Equation (6.6).

$$t_{\text{had}_{\text{correct}}} \equiv b_{\text{had}_{\text{correct}}} \wedge ((lj_{1_{\text{correct}}} \wedge lj_{2_{\text{correct}}}) \vee lj_{\text{swapped}}). \tag{6.6}$$

The dataset has to provide the truth information for each permutation per event such that the classifiers can learn from this truth information. Therefore, for each permutation the six elementary labels are calculated and combined into one label. This label is the integer representation of the sum of elementary labels where each elementary hypothesis represents a power of 2. Table 6.3 shows the labels and their values. This relation will be used during the implementation. In total, 64 labels would be possible but some of these labels cannot occur by definition. Additionally, the permutations are ordered in a special scheme. This allows to retrieve additional information and implies additional restrictions on the possible label vectors. Figure 6.3 shows the distributions of the labels per permutation in percent of all permutations at this position. It can be seen that the labels from 40 onwards do not occur as well as the intervals 5-7, 13-15, 21-23, 29-31 and 37-39. This is due to the fact that, in these cases, either the light jets or the b-jets are swapped but at least one of those which should be swapped is assigned correctly. This cannot happen as shown by the logical implications Equations (6.7) and (6.8).

$$lj_{\text{swapped}} \iff \neg(lj_{1_{\text{correct}}} \vee lj_{2_{\text{correct}}}). \tag{6.7}$$

$$b\text{-jets}_{\text{swapped}} \iff \neg(b_{\text{had}_{\text{correct}}} \vee b_{\text{lep}_{\text{correct}}}). \tag{6.8}$$

Elementary Hypothesis/ Label	**Integer Representation**		
	Binary	**2^x**	**Decimal**
The hadronic b-jet (b_{had}) is assigned correctly.	000001_2	2^0	1
The leptonic b-jet (b_{lep}) is assigned correctly.	000010_2	2^1	2
The b-jets are swapped.	000100_2	2^2	4
The first light jet (lj_1) is assigned correctly.	001000_2	2^3	8
The second light jet (lj_2) is assigned correctly.	010000_2	2^4	16
The light jets are swapped.	100000_2	2^5	32

Table 6.3: The integer representations of the elementary hypotheses. They are summed up per permutation and create a unique label in the interval 0-63. Elementary labels can be extracted later by division and modulo operations.

#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
b_{had}	1	1	1	1	1	1	2	2	2	2	2	2	3	3	3	3	3	3	4	4	4	4	4	4
b_{lep}	2	2	3	3	4	4	1	1	3	3	4	4	1	1	2	2	4	4	1	1	2	2	3	3
lj_1	3	4	2	4	2	3	3	4	1	4	1	3	2	4	1	4	1	2	2	3	1	3	1	2
lj_2	4	3	4	2	3	2	4	3	4	1	3	1	4	2	4	1	2	1	3	2	3	1	2	1

Table 6.4: Permutation scheme of the multi-class approach. Each column represents one permutation and shows which of the measured jets is assigned to which position. The numbers of the measured jets are defined as: 1 = jet with the highest *MV2c10* b-tagging-value; 2 = jet with the second highest *MV2c10* b-tagging-value; 3 = jet with the highest p_{T}-value of the remaining jets; 4 = jet with the second highest p_{T}-value of the remaining jets.

What is also visible in Figure 6.3 are patterns between the different labels and permutations. The values for (permutation 1, label 3) and (permutation 7, label 4), for example, are the same. To explain this, Table 6.4 is needed which shows the assignment order of the measured jets to the slots of the expected jets. In this case, permutation 1 and 7 differ only in the assignment of the two b-jets. Whenever in permutation 1, the two b-jets are assigned correctly (label 3) they have to be assigned swapped (label 4) in permutation 7. These patterns can be found for many of these pairs, which reduces the number of possible label vectors significantly and, additionally, provides a new source of information for a classifier.

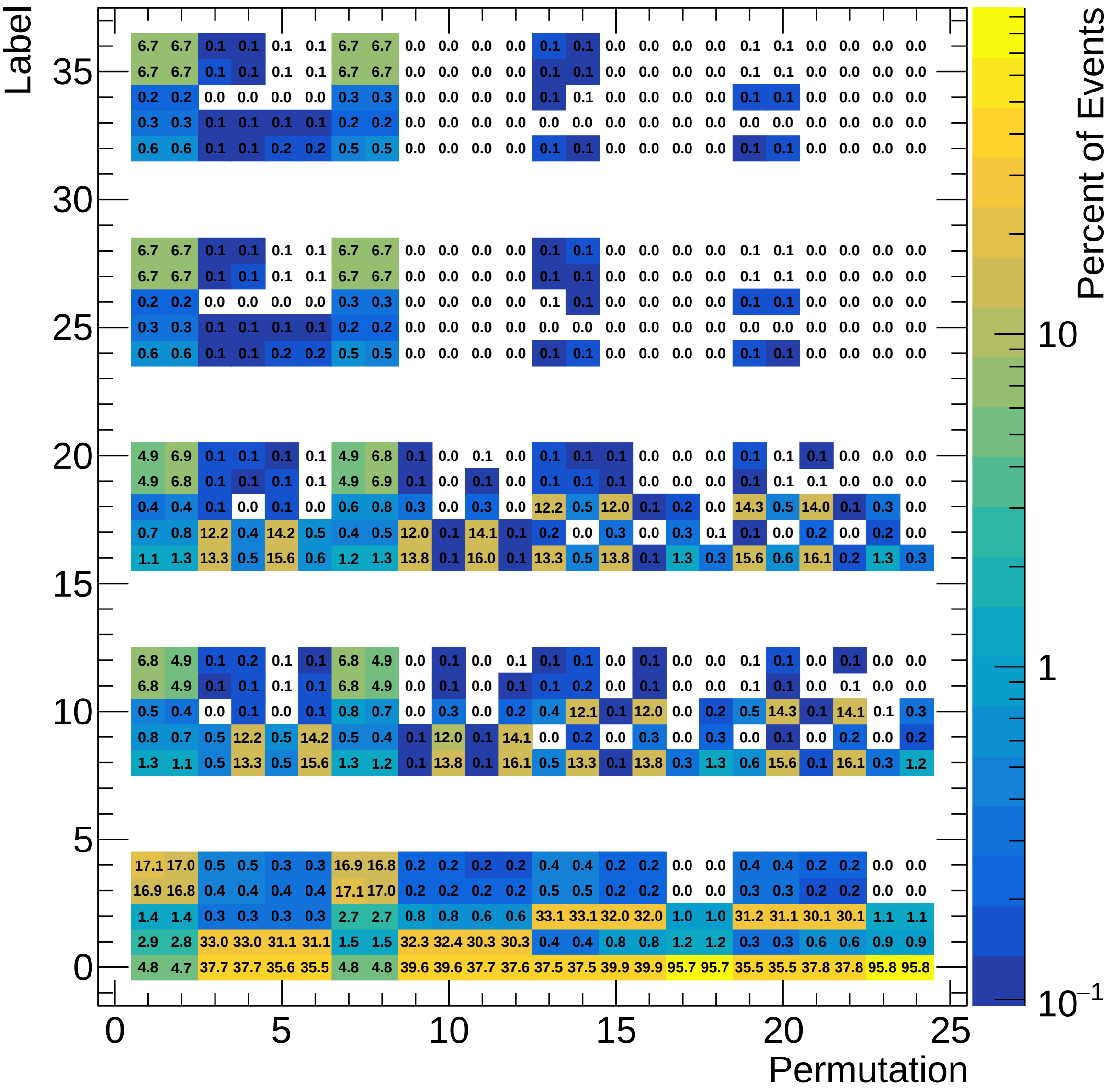


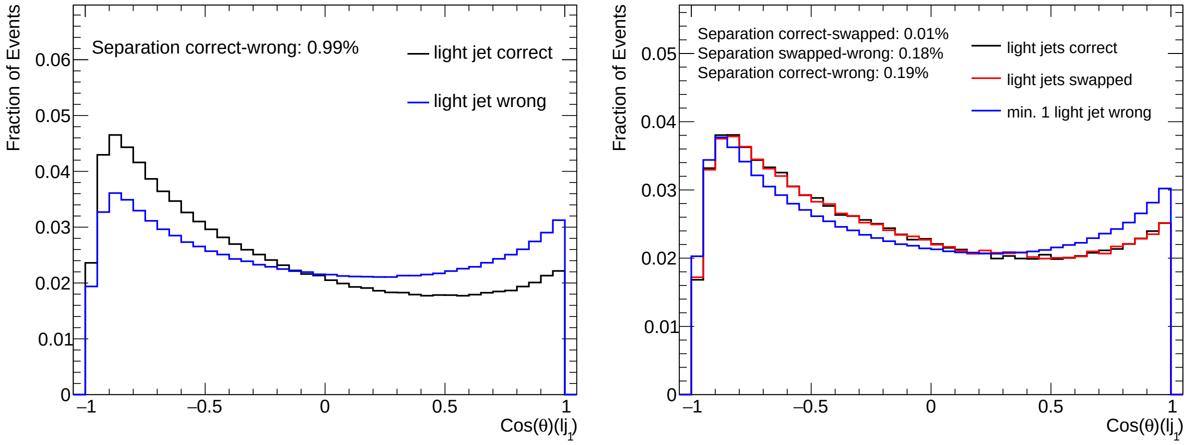
Figure 6.3: 2D-histogram of the label distribution among the different permutations. Each column sums up to 100%. The range of shown labels is limited to 36 since labels from 37-63 cannot appear due to the logical implications within the label definitions. From 40 onwards, always both light jets would be swapped but also at least one of them would be assigned correctly, which cannot happen. The other empty intervals are empty for the same reason applied to the *b*-jet-labels.

6.3.2 Variables

The input features of the multi-class approach are the ones of the binary classifier and in addition some new variables have been added which are described in this chapter. Each event consists of one variable which gives the number of jets measured in the detector simulation and 23 other variables which are included once per permutation.

Table 6.5 gives an overview over the variables which are included multiple times. Motivated by the new labels and hypotheses explained previously, the following variables have been added. The $\cos(\theta)$-values are added to improve the predictions in respect to the correctness of the chosen light jets in the permutation. θ is the polar angle, the angle between $\vec{n}$ and $\vec{p}$ where $\vec{n}$ is the normal of the transversal plane and $\vec{p}$ is the direction vector of the particle θ is calculated for. Figure 6.4 shows the separation power of $\cos(\theta)_{lj_1}$ in two different contexts. In Figure 6.4(a), it is shown that $\cos(\theta)_{lj_1}$ is a variable which can be used to differentiate between the assignment of the correct light jet and the wrong light jet. Although the separation power is just ≈1%, the $\cos(\theta)$-values still can help to improve the performance of the classifier. Additional correlations with other input variables could increase the separation power for the “light jet 1 correct” and “light jet 2 correct” hypotheses. As shown in Figure 6.4(b), $\cos(\theta)_{lj_1}$ is not a good variable to separate between the correct and the swapped light jet assignment. The separation power of $\cos(\theta)_{lj_2}$ does not vary significantly from the ones shown in Figure 6.4.

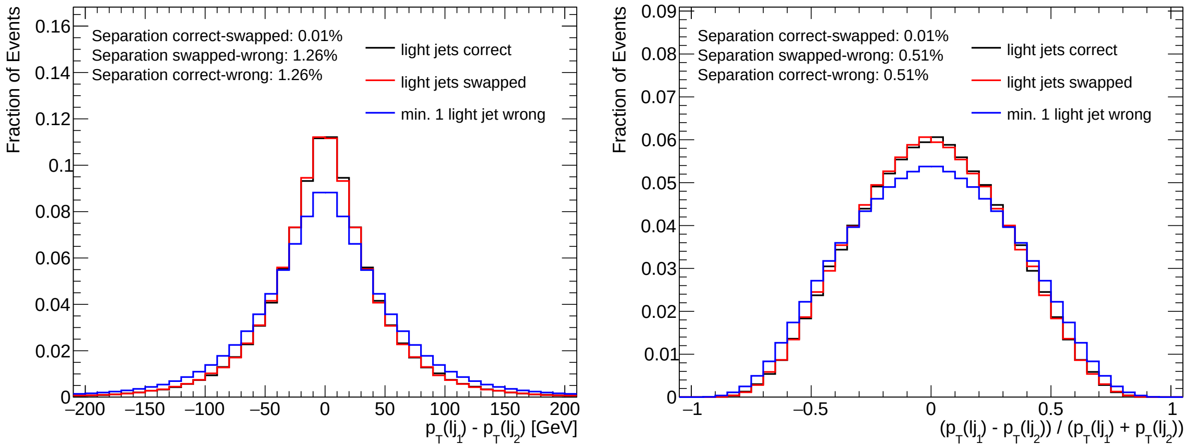
Another variable which has been studied in detail are the transverse momenta of the light jets. Figure 6.5 shows two separation plots of combinations of both light jet p_T-distributions. Since values of both light jets are shown at the same time, the separation categories have to make statements about the assignment of both light jets at the same time as well. Here, the categories “light jets correct”, “light jets swapped” and “min. 1 light jet wrong” are chosen. The results show that the transverse momentum of the light jets has almost no separation power for the correct vs. the swapped assignment. The wrong assignment of at least one light jet can be separated, but only with 1.26% and 0.51% separation power. This information can still be used to determine wrong assignments which would lead to wrong reconstructions of the hadronic W-boson or the hadronic top-quark. The ratio $\frac{p_\mathrm{T}(lj_1)}{p_\mathrm{T}(lj_2)}$ is used as an additional variable as well but shows the same results in terms of separation power as Figure 6.5.



(a) Separation plot of $\cos(\theta)_{lj_1}$. The values are categorised in events in which the first light jet is assigned correctly and in events in which it is not.

(b) Separation plot of $\cos(\theta)_{lj_1}$. The values are categorised in events in which both light jets are assigned correctly, in events with swapped assignment and events with at least one wrong light jet.

Figure 6.4: Separation studies of $\cos(\theta)_{lj_1}$. This plots have been created to study whether or not helpful information can be retrieved from the $\cos(\theta)_{\text{jet}}$-values. Figure 6.4(a) shows that this value can be used to check whether or not a light jet is assigned correctly. For the separation between correct and swapped assignment of the light jets the separation power is too small as shown in Figure 6.4(a).



(a) Separation plot of the difference between lj_1 and lj_2. The difference between the momenta is not a good separation value.

(b) Separation plot of the difference between lj_1 and lj_2. This more complex combination of the light jet transverse momenta does not separate between correct and swapped light jet assignments either.

Figure 6.5: Both subfigures show different arithmetic combinations of the light jet transverse momenta. These distributions do not support the separation for the "light jets swapped"-hypothesis but can still be used to identify events with wrong light jet assignments. Otherwise, these wrong assignments would lead to wrong reconstructions of the hadronic W-boson or the hadronic top-quark.

Variable	Meaning	Unit
KLFitter-log-likelihood	The logarithm of the likelihood KLFitter fitted for this permutation.	1
b-tag(jet)	The *MV2c10* b-tagging-WP passed by the given jet represented as integer in 0-4. Calculated for b_{had}, b_{lep}, lj_1 and lj_2.	1
$p_{\mathrm{T}}(jet)$	The transverse momentum of the given jet. Calculated for b_{had}, b_{lep}, lj_1 and lj_2.	GeV
$\Delta R(lj_1, lj_2)$	The angular distance between the light jets in the detector.	1
$\Delta R(b_{\mathrm{lep}}, e\|\mu)$	The angular distance between the leptonic b-jet and the charged lepton.	1
$\Delta R(b_{\mathrm{had}}, b_{\mathrm{lep}})$	The angular distance between the hadronic and the leptonic b-jet.	1
$p_{\mathrm{T}}(t_{\mathrm{had}})$	Reconstructed transverse momentum of the hadronic top quark.	GeV
$\eta(t_{\mathrm{had}})$	η = -ln[tan(θ/2)], where θ in this case is the polar angle between the direction of the hadronic top quark and the normal of the transverse plane.	1
$m(t_{\mathrm{had}})$	Reconstructed mass of the hadronic top quark.	GeV
$m(W_{\mathrm{had}})$	Reconstructed mass of the hadronic W-boson.	GeV
$p_{\mathrm{T}}(b_{\mathrm{lep}} + e\|\mu)$	The transverse momentum of the leptonic **b**-jet and the charged lepton.	GeV
$m(t_{\mathrm{lep}})$	Reconstructed mass of the leptonic top quark.	GeV

Table 6.5: Table of variables per permutation. b_{had}, b_{lep}, lj_1 and lj_2 represent the jets which are assumed to be the jet indicated by the respective index. Correspondingly, W_{had}, t_{had} and t_{lep} stand for the assumed reconstructions of the named partons. The transverse plane is the plane perpendicular to the beam pipe of the LHC. Since the accelerated particles have only energy in the direction of the beam pipe, the sum of transverse momenta must be 0.

6.3.3 Workflow

The workflow of the multi-class approach is shown in Figure 6.6. It looks at two different parts of the approach: the data used and the machine learning stages. The multi-class approach is unlike the binary classifier a *stacked approach* in which multiple stages of machine learning get the data several times as input or outputs of one stage are inputs for the next stage. In theory, they could use different machine learning techniques but in the multi-class classifier presented here only DNNs are used. The workflow describes the different stages of machine learning and what the data between those stages look like. The single steps of the workflow are described below.

Stage 1 – The Input Data

The input data for the whole multi-class approach consist of approximately 610,000 semi-leptonic $t\bar{t}$-decays. Each of these events contains $24 \cdot 23 + 1 = 553$ different input variables. which have been described previously.

Stage 2 – Reshaping to Single-Permutation-Events

The data has to be reshaped such that every single permutation is counted as its own input vector. The size of the input dataset increases by a factor of 24 giving a dataset of 14.68 million input elements. This is larger than the HEMPASS and the HIGGS dataset [57–60]. These events are than fed into the first machine learning stage.

Stage 3 – Six Binary Classifiers

The first machine learning stage of the new approach are six binary classifiers in parallel. Each of these classifiers is used for the prediction of one of the six elementary hypotheses which have been presented in Table 6.3. The goal is to find out for each single permutation whether or not this particular hypothesis is true. The classifiers used are binary DNNs and since they evaluate each permutation on its own, correlations between different permutations cannot be taken into account as shown in Figure 6.3.

Stage 4 – The Prediction-Matrix

The predictions of the six binary DNNs are then combined for each permutation to a prediction vector. Subsequently, the dataset is reshaped back to the representation where one event consists of 24 permutations. Therefore, the new dataset has the size of the number of $t\bar{t}$-events of which each event is represented by a 6×24 matrix.

Stage 5 – The Multi-Class Classifier

This prediction matrix is the input for the second machine learning stage, the multi-class DNN. It is remarkable, that this DNN only gets prediction values as input. The physical variables are not used anymore. This reduces the number of input dimensions significantly. In this stage correlations between the different permutations can also be taken into account, which is necessary to find the correct permutation in the event.

Stage 6 – The ***Best Permutation***

Based on the training of the last classifier, it is possible to decide which of the given 24 permutations is the best one. As motivated at the beginning, the goal is to customize the definition of "the best permutation" such that any reconstruction hypothesis can be combined from the six basic hypotheses. The desired label combination can be retrieved from the compound label as shown in Table 6.3.

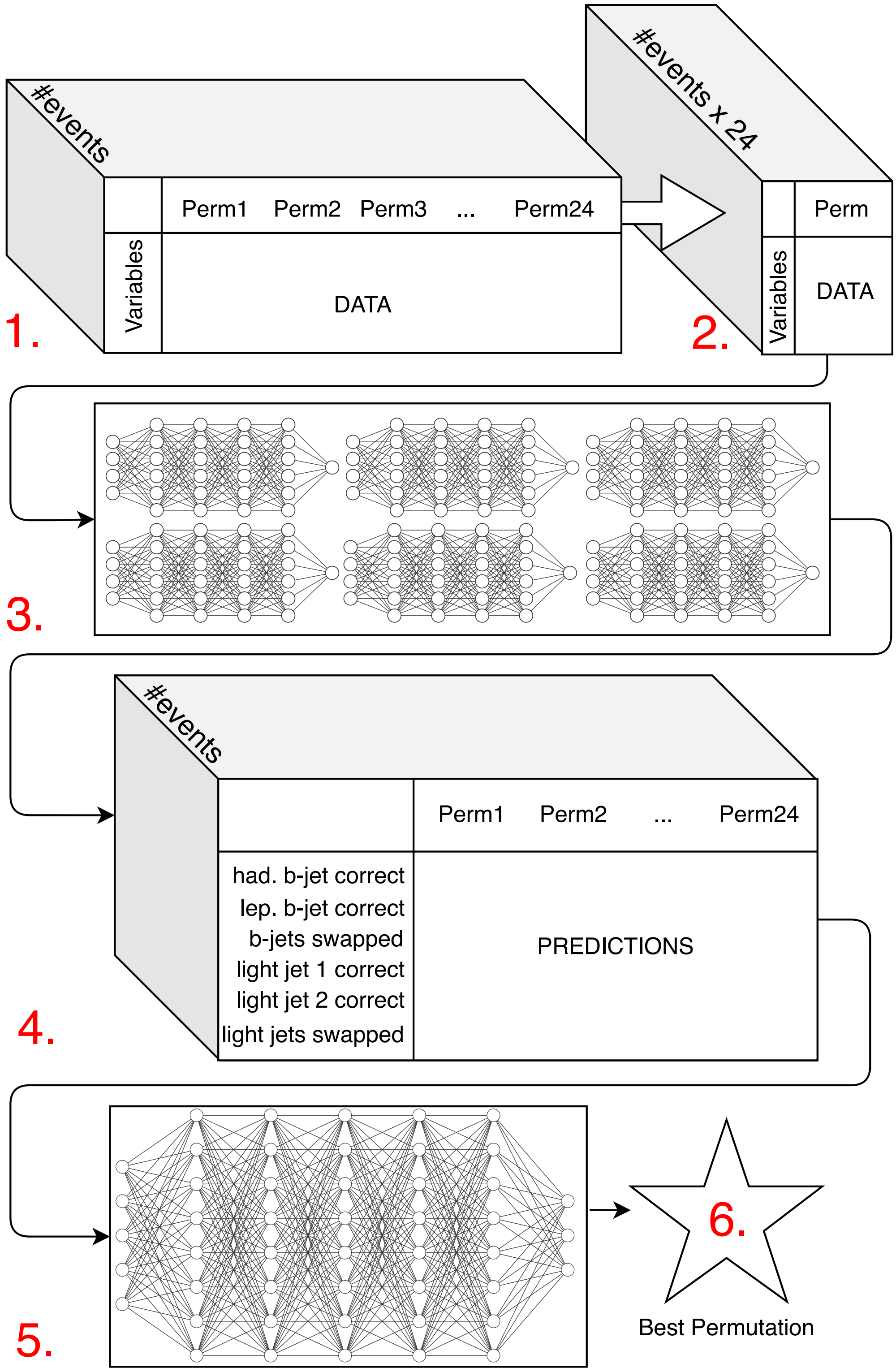


Figure 6.6: Scheme of the workflow of the multi class extension. The dataset which is fed into the classifier (1.) is reshaped to look at one permutation at a time solely. These single permutations (2.) are fed into each of the six binary classifiers (3.) which return the predictions for the elementary hypotheses. The permutations and their predictions are reshaped to the original format (4.). Subsequently, the predictions are input for the multi-class classifier (5.) which then gives the best permutation (6.) as output.

6.3.4 Implementation

In the following, the implementation of the DNN extension is summarised. The starting point is the originally simulated N-tuples set from the top-decay width analysis. The KLFitter algorithm has already been run on these events but taking only a maximum of five jets into account per event. The two jets with the highest *MV2c10* b-tagging values and the three jets with the highest p_{T}-values from the remaining jets are selected. From these five jets, 60 different permutations have been created and fitted. If only four jets have been measured in one event, 12 permutations are calculated, respectively. At this point, the development of this approach starts.

From each event, only four of the reconstructed jets are selected. The two with the highest *MV2c10* b-tagging values and the two jets with the highest p_{T}-values from the remaining jets. Henceforth, these four reconstructed jets will be called the r-jets. Half of the permutations of the r-jets and their KLFitter results have been calculated before. The other half with the swapped light jet assignments is assigned manually and the same KLFitter log likelihoods are assigned. These 24 permutations are ordered based on the ordering scheme in Table 6.4.

Subsequently, for each event, the truth quarks, henceforth called t-quarks, have to be calculated and the r-jets are matched with the t-quarks. This is done recursively. This algorithm finds the best pairing between one r-jet and one t-quark which is then stored and both are deleted from the respective sets. The best pairing is defined as the smallest ΔR between one r-jet and one t-quark which also has to be smaller than 0.3. Therefore, the pairing with the smallest ΔR is selected first. The algorithm is invoked with the reduced set of r-jets and t-quarks until either all t-quarks have been paired or no pairing can be found. If none of the pairings is smaller than 0.3, the pairing algorithm terminates. Therefore, it is possible that some of the t-quarks are not matched by any of the r-jets which have been selected. Figure 6.7 shows an activity diagram of the matching algorithm. As shown, the pairing of t-quarks with r-jets is not allowed if the r-jet originated from a gluon since all truth jets are jets with a quark origin. The truth origin of a jet is given as truth information but is not contained in the reconstructed values of a jet.

For each of the 24 permutations, the variables shown in Table 6.5 are calculated and saved. From the found jet-pairings and Table 6.4, the labels in Table 6.3 are calculated, added up and saved. Additionally, per event, the MC simulation weights and the number of originally measured jets (n_{jet}) is saved. This saved data builds the dataset which is the starting point of the workflow presented in Figure 6.6.

These implementation steps are programmed in `C++11`.

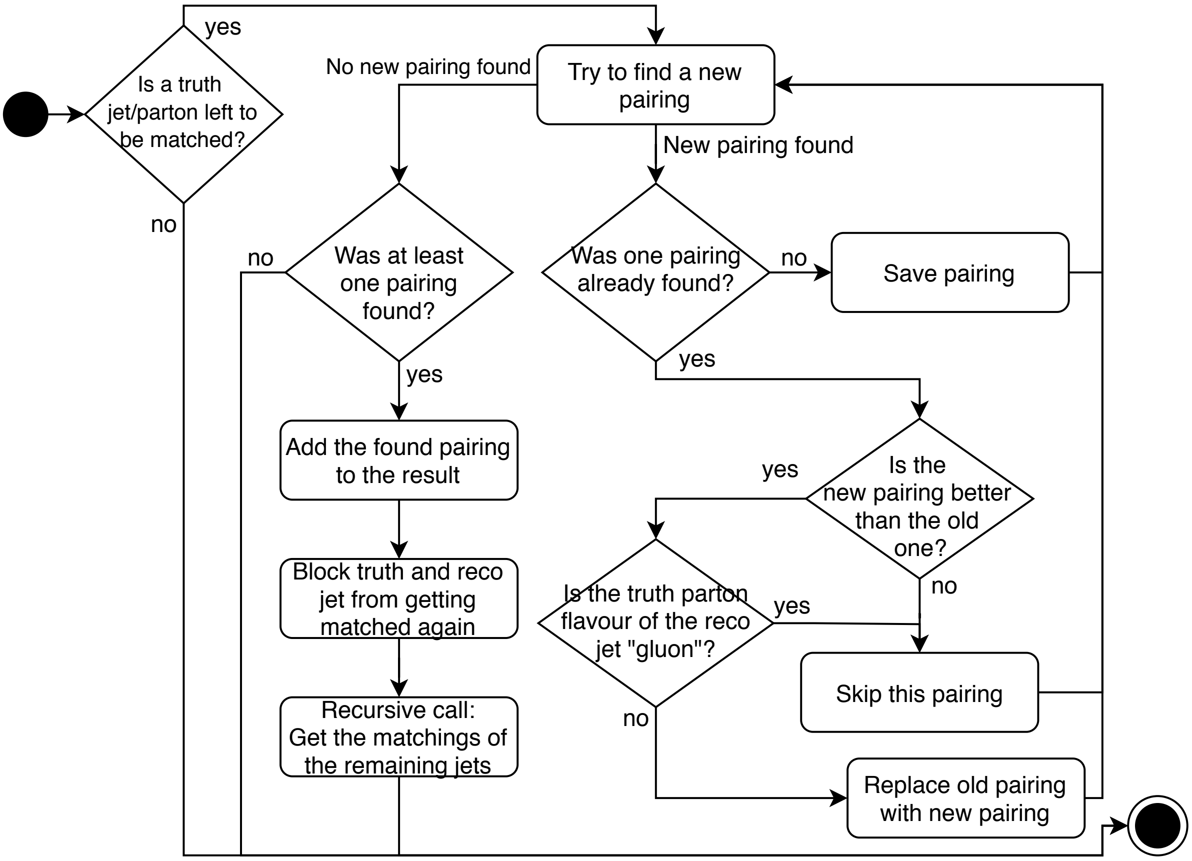

yes
Is a truth jet/parton left to be matched?
no
No new pairing found
Try to find a new pairing
New pairing found
no
Was at least one pairing found?
yes
Was one pairing already found?
no
Save pairing
yes
Add the found pairing to the result
Is the new pairing better than the old one?
yes
no
Block truth and reco jet from getting matched again
Is the truth parton flavour of the reco jet "gluon"?
yes
no
Skip this pairing
Recursive call: Get the matchings of the remaining jets
Replace old pairing with new pairing

Figure 6.7: Activity diagram of the matching algorithm. Truth jets which originated from the truth partons and reconstructed jets are matched based on the ΔR between them. The pairings with smallest ΔR values are taken first.

The data produced in the previous step is saved in a `.root` file. This format is part of the ROOT Framework which is a data analysis framework and provides many predefined classes for scientific work with large amounts of data [61, 62]. It is especially used for particle physics at high energy scales and was and is still developed at CERN. Since the machine learning parts of the new approach are coded in Python with the help of the packages *Tensorflow* [63] and *Keras* [50], the data has to be in a readable format for these libraries. To do so and to ease the work with the dataset in the python environment, the `.root` file is converted into a pandas dataframe and saved as `.hdf5` file [64] with the help of the libraries *Pythonic ROOT* [65], *Numpy* [66] and *Pandas* [67]. Pythonic ROOT is a community driven project to convert `.root` files into numpy arrays which are special array representations and provide many scientific operations with them. These numpy arrays can be used as data sources for pandas dataframes, which is an often used data structure which represents data in a way similar to a database. They also provide the possibility to access and modify data with querys similar to SQL.

After the conversion, the dataset is reshaped as explained in step 2 of the workflow. This is done by concatenating the 24 data columns of the same physical value per event into one column in a new dataset. While this can be done without problems for the 23 different variables which occur once per permutation, the procedure has to be adjusted to reshape n_{jet}. This value exists only once per event instead of once per permutation. To include it, the value is replaced by a vector of length 24 which only contains the value of n_{jet}. Subsequently, the transformation can be applied without problems. This procedure is also applied to the MC simulation weights, whereas, additionally, all negative weights are set to 0 because negative weights cannot be used as training weights and would have ambiguous effects on the loss function. Subsequently, the input variables are normalised by subtracting the minimum value per data column and then divide by the maximum value unless the maximum is 0 which may not occur.

To use this dataset for training, training weights and labels are needed. The training labels for the first machine learning stage need to be in a binary form since all of the six DNNs are supposed to answer a yes-no-question in the form of “Is hypothesis X true for this permutation?”. The corresponding labels can be retrieved from the compound label per permutation, as shown in Table 6.3. For each of the six binary hypotheses, a label column is created. For the four “Jet X is assigned correctly.”-hypothesis, every event gets a 0 or 1 as its label. The “Jets are swapped.”-labels are either 0, 1 or 2. Since the goal is to differentiate between the assignments in which both jets are assigned correctly and in which both jets are swapped, the events where one or both jets are assigned incorrect cannot be considered in these trainings. They get a 2 as label which indicates that they are not used for the trainings. Based on these labels, the training weights are calculated from the MC simulation weights. The reweighting is applied to the subset of training events with the label 0 and the label 1 such that the sum of the weights of these two categories is equal. The events with the label 2 are weighted with 0 in the datasets, respectively, to the DNNs for the swapped assignment trainings.

This creates, in total, one dataset of the input features which is used for all six binary DNNs and six datasets of training labels and weights. Each of them is used for one of the trainings. These trainings are conceptualized and executed the same way with the minor difference, that in the trainings for the swapped-hypotheses, the events with the label two are cut out.

The training of the six binary DNNs starts with loading the datasets. Subsequently, the cross validation is applied. The dataset is split into five parts of which in each iteration one is used as a testing dataset and four as training datasets. These parts are taken from the input datasets and fed into the DNN model which is summarised in Table 6.6. The

Configuration Parameter	Choosen Value
Input Features	24
Hidden Layers (HL)	4 layers á 30 nodes
Output Layer (OL)	1 node
Optimizer	Adam [52]
Activation Function (HL)	ReLu(Equation (6.3))
Activation Function (OL)	Sigmoid(Equation (6.1))
Loss Function	Binary Cross Entropy
Validation Split	0.2
Cross Validation	5-fold
Trained Epochs	40
Batchsize	10000
Early Stopping	None
Dropout	25% Dropout rate between HLs

Table 6.6: Configuration of the six trained binary classifiers for the multi-class approach. The chosen values are the product of multiple different test trainings during the implementation and showed good model performance. More training effort in terms of additional or larger hidden layers showed no significant performance increases which would justify the investment of additional computational resources.

indexes of the validation events are saved and the training is started.

The model and its training are similar to the binary classifier which mimics the BDT. Due to the larger number of input features, the size of the hidden layers has been increased to 30 nodes. At the same time, the number of training epochs is reduced to 40. This is motivated by the fact that the number of training instances is approximately 14.68 million due to the single permutations. Therefore, the model is expected to improve quickly at the beginning of the training, asymptotically reaching a limit after a few epochs. Additionally, in these models dropout is applied between the hidden layers. A 25% dropout rate means that, per epoch, approximately 25% of the edges are blocked randomly.

During the training, the loss and the accuracy are tracked epoch-wise and saved for later. When the training is done, the resulting training history is saved. This enables to later plot learning curves and see whether or not the model improved over time and if it underfitted or overfitted. The next step is the prediction of the events with the final model. Each event is fed into the model once to get the prediction of the model for this event. Subsequently, these predictions and the model itself are saved as well.

These steps are done for all five folds. Although some events have been cut from the training of the swapped-hypotheses, they are classified by the model as well. For the multi-class DNN, predictions for each permutation in each event are needed to have complete feature vectors as input.

The next step is the preparation of the prediction matrices for the final DNN. Therefore, at first the predictions of the six binary DNNs have to be shaped back into a format in which one event consists again of its 24 permutations. Information that is included once per permutation but only needed once per event is dropped and copied only once. This is the information in which of the five cross-validation folds this permutation was used as validation data and n_{jet}. This data is then combined into 6×24 matrices which is the input dataset for the multi-class DNN. For this stage of the training, once again weights and labels have to be calculated.

This time, the label depends on which reconstruction hypothesis is to be found. This reconstruction hypothesis is a combination of the elementary labels. For the example training in the context of this thesis, the "Everything is reconstructed correctly."-hypothesis is used to keep the reconstruction efficiency comparable with the previous approaches. However, the concept of the final training stays the same for all possible combined reconstruction hypotheses.

For each event, a label vector is assigned containing 25 elements. 24 of these 25 elements represent the 24 permutations per event and are set to 1 if the combined reconstruction hypothesis is true for this permutation if not, it is set to 0. The 25th element in this vector represents the case in which none of the permutations meets the requirements.

In the case of the "Everything is reconstructed correctly."-hypothesis, the logical combination of the elementary hypotheses consist of all four "Jet X is assigned correctly."-hypotheses as shown in Equation (6.9). Therefore, the integer label of this hypothesis calculated from Table 6.3 is 1+2+8+16=27.

$$\text{all correct} \equiv b_{\mathrm{had}_{\mathrm{correct}}} \wedge b_{\mathrm{lep}_{\mathrm{correct}}} \wedge lj_{1_{\mathrm{correct}}} \wedge lj_{2_{\mathrm{correct}}}. \tag{6.9}$$

Subsequently, the training weights have to be calculated based on the original MC simulation event weights. Negatively weighted events are reweighted with 0. The other events are reweighted such that each category has an equal sum of weights. At this point an extra analysis is done. Figure 6.3 shows that the "all correct"-label is not equally distributed among the 24 permutations. While the permutations 1, 2, 7 and 8 are correct in ≈7% of all events, the other permutations are correct in less than 0.2% of the events. If the additional possibility is taken into account that none of the permutations represents the completely correct reconstruction, the amount of events which belong to either one of the four 7%-permutations or to the label where none of the permutations is the correct reconstruction is 99%. This means that 99% of all events can be described with five of the 25 possible labels which also draws the conclusion that the other 20 permutations together only account for 1% of all events. Since this dataset is highly imbalanced, the labels of these 20 permutations are not considered during the training and the events are weighted with 0. The other, in this case five, categories are balanced as described previously and the labels and the new weights are saved as well.

This new dataset is the input for the final DNN. Table 6.7 summarises the model trained in this stage. The number of input features increased significantly which is why the number of nodes per hidden layer has been increased as well to 50. At the same time, the complexity of the problem decreased. The final DNN does not have to learn any physical correlations and reconstructions. The elementary hypotheses have already been answered. To be learned are the patterns from Figure 6.3 and the combinations of the labels. Therefore, the number of hidden layers has been decreased to 2. The output layer now has 25 nodes. A problem here could be that an output value, e.g. 0.4, can mean different things depending on the other output values. If the other values are higher, this could mean that other permutations are more likely to be the correct one. If they are smaller than the given 0.4-permutation, this one could be the correct one. To keep the values interpretable some constraints might be helpful. Therefore, the softmax function, shown in Table 6.8, is chosen in the output layer. Its purpose is to return an output vector of which all elements sum up to one. The softmax function is shown in Equation (6.10). softmax(z): $\mathbb{R}^{25} \to \mathbb{R}^{1}$, since for each node all other output node values are taken into account. Therefore, its gradient depends on the index of the input feature as shown in Equation (6.11). The loss function has been changed from the binary cross entropy to the categorical cross entropy since this DNN is no longer binary. Also new for the training of the multi-class DNN is the usage of early stopping. Here its configuration is to check whether or not the loss decreased within the last 20 epochs about at least 0.001. Otherwise the training will stop.

After the model is trained, it calculates predictions for all events and categorises them in training and validation data predictions. Subsequently, these predictions are saved and the next iteration through the cross validation starts. From the prediction vector of an event, a prediction can be retrieved of which permutation is the best one for the selected hypothesis. In this case, the overall prediction can either be in which permutation all jets are assigned correctly or that none of the permutations reconstructs the $t\bar{t}$-decay completely correctly.

Configuration Parameter	**Choosen Value**
Input Features	144
Hidden Layers (HL)	2 layers á 50 nodes
Output Layer (OL)	25 nodes
Optimizer	Adam [52]
Activation Function (HL)	ReLu(Equation (6.3))
Activation Function (OL)	Softmax(Equation (6.10))
Loss Function	Categorical Cross Entropy
Validation Split	0.2
Cross Validation	5-fold
Trained Epochs	dynamic
Batchsize	10000

Early Stopping	modus = minimising loss
	patience = 20 epochs
	$\Delta_{\mathrm{min}} = 0.001$

Dropout	25% Dropout rate between HLs

Table 6.7: Configuration of the multi-class classifier of the DNN extension to KLFitter. The chosen values are the product of multiple different trainings and showed good model performance. More training effort in terms of additional or larger hidden layers showed no significant performance increases which would justify the investment of additional computational resources.

$$\mathrm{softmax}(z)_{\mathrm{i}} = \frac{e^{z_{\mathrm{i}}}}{\sum_{j=1}^{\mathrm{n}}(e^{z_{\mathrm{j}}})} \qquad \mathrm{i},\mathrm{j} \in \mathbb{N}_{\mathrm{n}};\ z = (z_1, ..., z_{\mathrm{n}}) \in \mathbb{R}^{\mathrm{n}}. \tag{6.10}$$

$$\frac{\partial}{\partial z_{\mathrm{j}}}\mathrm{softmax}(z, \mathrm{i}) = \begin{cases} z_{\mathrm{i}}(1 - z_{\mathrm{j}}) & \text{if i = j} \\ -z_{\mathrm{i}} z_{\mathrm{j}} & \text{if i} \neq \text{j} \end{cases} \qquad \mathrm{i},\mathrm{j} \in \mathbb{N}_{\mathrm{n}};\ z = (z_1, ..., z_{\mathrm{n}}) \in \mathbb{R}^{\mathrm{n}}. \tag{6.11}$$

Table 6.8: The softmax function shown in Equation (6.10) and its derivative shown in Equation (6.11).

6.4 Evaluation

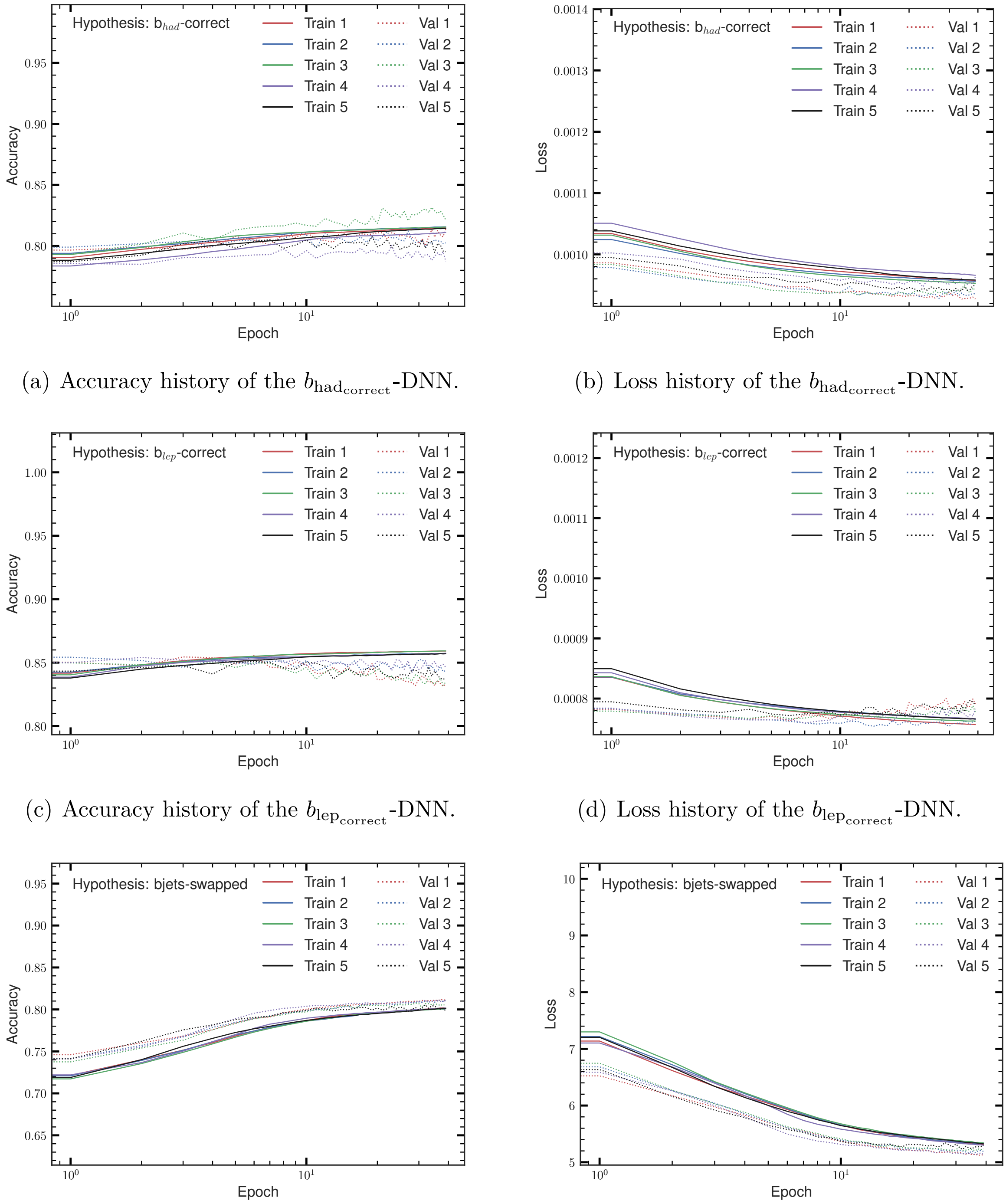
This chapter gives an overview over the performances of the different steps of the new extension to KLFitter. This evaluation is split into two parts, each of them evaluating one of the two machine learning stages of the multi-class approach.

6.4.1 Evaluation of the 6 Binary DNNs

The trainings of the six binary DNNs and the performances of the models which originated from them can be evaluated with multiple plots and benchmarks. In the context of the binary DNNs three parts have to be considered: the training of the models, the performance of the trained classifier and a glance at the traceability of the predictions.

Training of the Models

To evaluate the training performance of the models, the training histories are saved. Different values can be tracked of which in this thesis the accuracy and the loss are chosen. Figures 6.8 and 6.9 show the training histories of the different hypotheses. An untrained classifier predicts random values. This means that it is in approximately 50% of all cases correct if the classification task is binary as it is for the six binary DNNs. The accuracy should rise over time and reach 100% in the ideal case. The loss should gradually decrease. It is expected that the classifier is slightly biased towards the training dataset. Figure 6.8 shows that the trainings of the b-jet hypotheses produced better results than those of the light jet hypotheses shown in Figure 6.9. The accuracy plots show the effect of the large training dataset. The first values are tracked after the first epoch was trained. Therefore the plots show that the accuracy rises within the first epoch significantly from 0.5 to 0.68 for the light jets and 0.5 to 0.84 for the leptonic b-jet. The only training in which the values do not improve is the training of the “light jets swapped”-hypothesis. Its models stay at the initial values which indicates no improvements during the training and, therefore, no learning of the classifier.



(a) Accuracy history of the $b_{\text{had}_{\text{correct}}}$-DNN.

(b) Loss history of the $b_{\text{had}_{\text{correct}}}$-DNN.

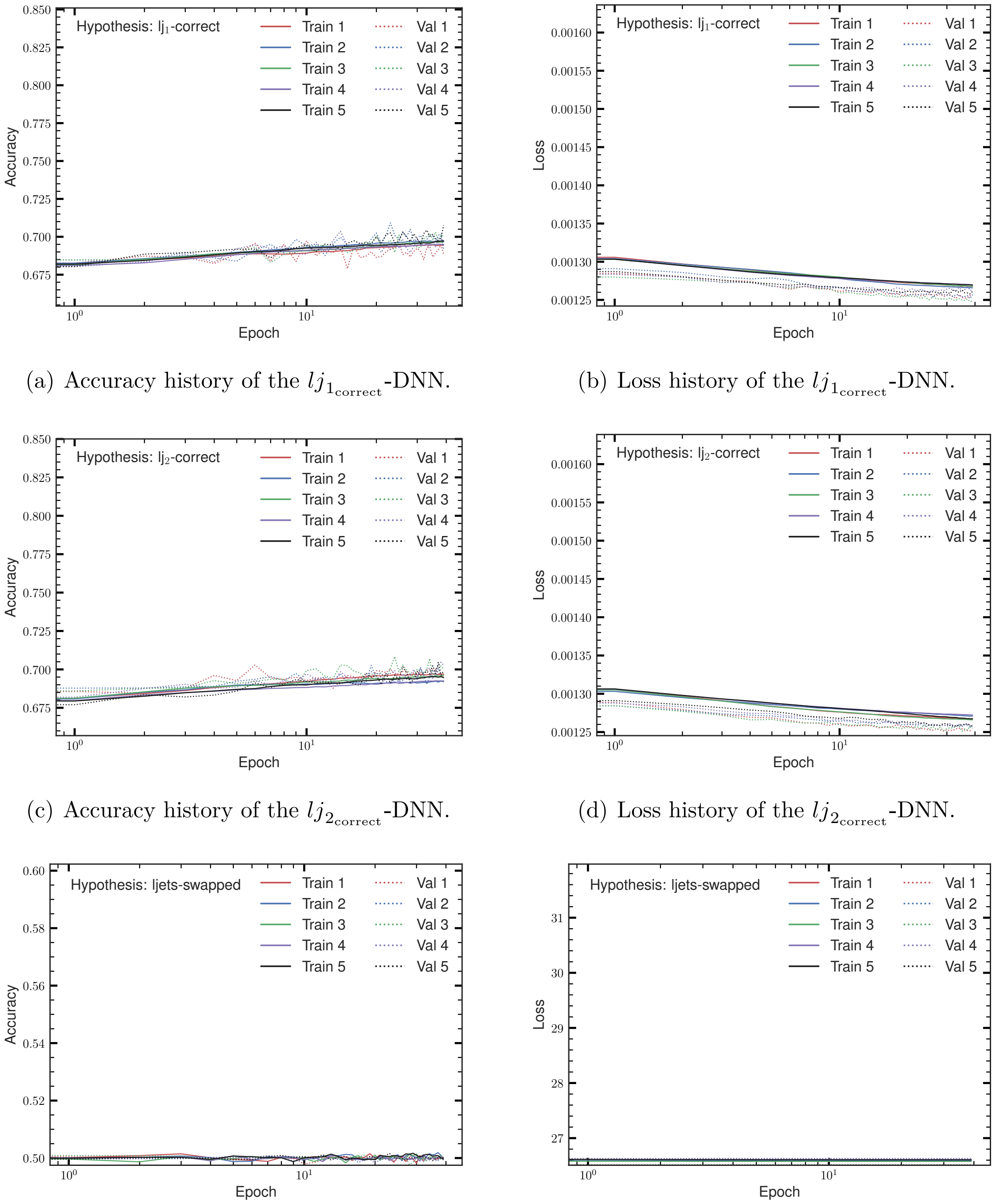
(c) Accuracy history of the $b_{\text{lep}_{\text{correct}}}$-DNN.

(d) Loss history of the $b_{\text{lep}_{\text{correct}}}$-DNN.

(e) Accuracy history of the $b-jets_{\text{swapped}}$-DNN.

(f) Loss history of the $b-jets_{\text{swapped}}$-DNN.

Figure 6.8: Training history plots for accuracy and loss values of the binary DNNs which predict a hypothesis connected to the b-jets. Shown are always the results of training and validation datasets of all five cross-validation-folds.



(a) Accuracy history of the $lj_{1_{\text{correct}}}$-DNN.

(b) Loss history of the $lj_{1_{\text{correct}}}$-DNN.

(c) Accuracy history of the $lj_{2_{\text{correct}}}$-DNN.

(d) Loss history of the $lj_{2_{\text{correct}}}$-DNN.

(e) Accuracy history of the *light jets*$_{\text{swapped}}$-DNN.

(f) Loss history of the *light jets*$_{\text{swapped}}$-DNN.

Figure 6.9: Training history plots for accuracy and loss values of the binary DNNs which predict a hypothesis connected to the light jets. Shown are always the results of training and validation datasets of all five cross-validation-folds.

Performance of the Trained Classifier

After the training ended, the training dataset and the testing dataset are classified by the trained model. The resulting predictions for the different hypotheses are saved and used afterwards. As explained in Section 6.2, two possibilities to evaluate the performance of a trained classifier are histograms of the classifier output values and ROC curves. They are produced for the six binary DNNs as well. The data has been divided by hypotheses, cross-validation-folds and permutations since these dimensions have an impact on the performance. This results in a total of 720 histograms and 144 ROC curves of which, in this thesis, only a few are shown to explain the different impacts and compare the performances. Additionally, to the ROC curves, *precision recall curves* (PRCs) are used to evaluate the performances of the models [68, 69]. PRCs show the recall, which is the same as sensitivity or true positive rate, and the precision. The precision is calculated as $\frac{\text{true positives}}{\text{true positives} + \text{false positives}}$. Therefore, this type of diagram shows the relation between the fraction of events that are actually signal, if predicted so and the fraction of classified-as-signal-events if they actually are signal events. In PRCs, the untrained classifier is represented as a horizontal line at $y_{\text{no skill}} = \frac{\langle\text{number of signal events}\rangle}{\langle\text{total number of events}\rangle}$.

For the evaluation of the binary DNNs, the first and 24th permutation are chosen. The first permutation (perm1) represents a case which is more likely than the 24th permutation (perm24). This is as shown in Table 6.4 because of the assumptions made in the different permutations. In perm1, the two jets with the highest b-tagging values are assumed to be the b-jets from the event and the other two the light jets. In perm24, these assignments are swapped. It is less likely that the two jets which are actually the b-jets have neither the highest b-tagging value nor the second highest. Therefore, perm24 represents the case which is highly unlikely. This is particularly interesting to check if the binary classifiers can perform well on these permutations too.

The evaluation of the $b_{\text{had}_{\text{correct}}}$-DNN and the $b_{\text{lep}_{\text{correct}}}$-DNN are shown in Figures 6.10 and 6.11. The histograms Figures 6.10(a) and 6.11(a) show mainly how good the classifiers are in distinguishing the correct b-jet from the incorrect one. The background events in these diagrams consist almost entirely of the respective other b-jet. The jet which is assumed to be the expected b-jet is most likely either this particular jet or the respective other b-jet. The signal fraction is $\approx 45\%$ which indicates an almost balanced dataset. However the $b_{\text{lep}_{\text{correct}}}$-DNN performs better which can be seen from the separation values of both plots whereas the $b_{\text{had}_{\text{correct}}}$-DNN has a separation power of $\approx 39\%$ and the $b_{\text{lep}_{\text{correct}}}$-DNN of $\approx 44\%$.

In contrast to perm1, perm24 is highly imbalanced in terms of the correct assignment of the b-jets. As shown in Figures 6.10(b) and 6.11(b), the jets are now more likely to actually be the true light jets or jets that do not belong to the event at all rather than being one of the assumed b-jets. Less than 2% of all events have the b-jet assignment of perm24 as the correct one. Under this circumstance, it is remarkable that the $b_{\mathrm{lep}_{\mathrm{correct}}}$-DNN performs approximately 10% better than on perm1 while the $b_{\mathrm{had}_{\mathrm{correct}}}$-DNN has a separation power which is approximately 27% worse than on perm1.

This difference can be seen in Figure 6.10(f) and Figure 6.11(f) as well. The PRC AUC values are higher for the $b_{\mathrm{lep}_{\mathrm{correct}}}$-DNN than for the $b_{\mathrm{had}_{\mathrm{correct}}}$-DNN. However, in the ROC plots, Figures 6.10(c), 6.10(d), 6.11(c) and 6.11(d), these differences and also the differences between perm1 and perm24 are not visible. This is due to the fact that the ROC curves are not sensitive to imbalanced datasets. The $\approx 2\%$ of actual correct b-jet assumptions in perm24 do not have a large effect on the true positive rate and the false positive rate. However, this is covered by the precision of the PRCs which makes them useful in this context. In the context of perm1, they are not necessarily needed. Here the datasets are almost perfectly balanced and the PRCs (Figures 6.10(e) and 6.11(e)) do not set the respective ROC curves in a different context.

The binary DNNs for the “light jet correct”-hypotheses show a smaller separation power and, therefore, performance of the classifiers than the performance of the b-jet DNNs. Figure 6.12(a) and Figure 6.13(a) show the separation and the prediction distribution. Additionally, the background distribution is shown which shows the positions to which the as-light-jet-assumed jets actually belong to. In perm1, the two jets which do not belong to the jets with the two highest b-tagging values are assumed to be the light jets. This is more likely than the correct assignment of a strongly b-tagged jet (60%-WP) to a light jet which can be seen in the fractions of the signal events for perm1 of approximately 31% for the first light jet and approximately 26% for the second light jet. In contrast to this, in perm24, the fractions of signal events for the first and second light jet are approximately 1.7% and 0.4%.

Figure 6.12(b) and Figure 6.13(b) show that these DNNs cannot differentiate whether or not a jet is the correct light jet for perm24. Even with applying training weights during the training, the imbalance cannot be resolved. This is remarkable since the $b_{\mathrm{had}_{\mathrm{correct}}}$-DNN has $\approx 8\%$ separation power and the $b_{\mathrm{lep}_{\mathrm{correct}}}$-DNN even $\approx 54\%$. Although the b-jet DNNs are distinguishing b-jet and light jets from each other at these separation powers for perm24, the light jet DNNs perform much worse in distinguishing b-jet and light jets. One possibility to explain this is that the binary DNNs are trained on all permutations at once and that the overall signal-background-ratio is different for the jets. For the light jet DNNs in total, approximately 15% of all permutations are signal permutations where the light jet is assigned correctly. For the b-jet DNNs the permutations represent a correct assignment in approximately 24% of all events. The imbalance here has to exist by definition. Each possible expected jet can only be assigned correctly in 6 out of 24 permutations and, therefore, the fraction of signal permutations can be at most 25%.

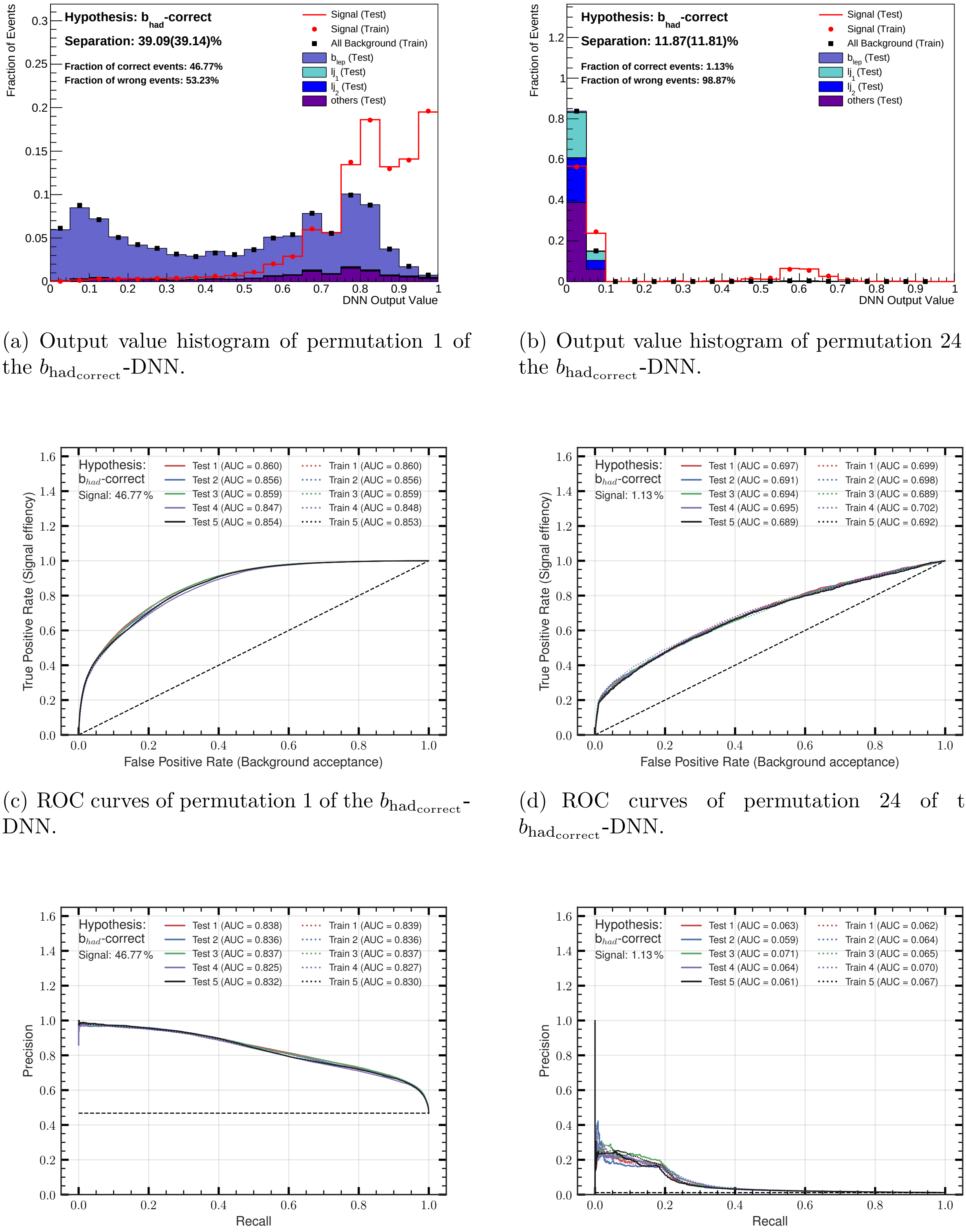
The ROC curves and the PRCs of perm24 of the $lj_{1_{\text{correct}}}$-DNN (Figures 6.12(d) and 6.12(f)) and the $lj_{2_{\text{correct}}}$-DNN (Figures 6.13(d) and 6.13(f)) show again the effect of the large imbalance of the dataset on its performance. The PRCs are almost identical with those of an untrained classifier which shows the small performance in respect to the large imbalance in the datasets. Additionally, it is shown that the ROC curves have increasing differences between the cross-validation-folds which originate from the small number of signal events which causes the statistics to fluctuate more.

The ROC curves and PRCs of perm1 can be used to analyse where thresholds could possibly be set to decide in which category an event is sorted. This decision has consequences on the different statistics which can be seen in the diagrams. E.g. if for lj_1 80% of all correct assignments in perm1 are to be found (sensitivity = recall = 0.8) this implies that, of all assignments which are then predicted as correct, 40% (precision = 0.4) are actually correctly assigned. This is shown in Figure 6.12(e). Additionally, from Figure 6.12(c), it can be concluded that this also implies that approximately 55% (false positive rate = 0.55) of all wrong assignments are also predicted as correct assignments.

The last two binary DNNs are the “jets swapped”-DNNs for which the evaluation is different compared to the other DNNs. Their hypotheses split the dataset in three parts: the correct assignment, the swapped assignment and the wrong assignment. This is why they are not trained on the complete datasets. ROC curves and PRC plots are not meant to be applied for a multi-class problem. Here, they could only be used between two of the three classes at once. However, histograms can still be used. Figures 6.14 and 6.15 show the histograms of the two “jets swapped”-DNNs for perm1 and perm24.

The b-jets-swapped DNN has a separation power of approximately 50% for perm1, where this separation is calculated between the correct and the swapped light jet assignment predictions. As shown in Figure 6.14(a), the fractions of either correct or swapped assignments is approximately 84%. This large value is caused by the assumption that both highly b-tagged jets are assumed to be the correct b-jets. The distributions for perm2, perm7 and perm8 would look similar. However, by definition, only two out of the 24 permutations can represent a correct and two others a swapped b-jet assignment. Therefore, on average, only four out of 24 permutations can be not-wrong. If the additional possibility is taken into account that some of the hadronic and leptonic b-jets have not been reconstructed correctly and are, therefore, not existing in the event this fraction is reduced to approximately 15%. The other 85% represent wrong *b*-jet assignments. This, and the unequal distribution of the remaining 15% among the permutations, lead to the results shown in Figure 6.14(b). Here, almost all *b*-jet assignments are wrong. From the approximately 610,000 events the 24$^{\text{th}}$ permutation is only correct in approximately 122 events and swapped in 122 different events. This also explains the differences between the training and testing datasets, in this plot.

The trained classifier for the swapping of the light jets has not improved in comparison with the untrained classifier. This was already shown in the history plots (Figures 6.9(e) and 6.9(f)) of the classifier. Therefore, the classifier has almost no separation power which can be seen in Figure 6.15. The phenomena explained for Figure 6.14(b) is visible in Figure 6.15(b) as well. The number of interesting assignments is so small that the already small separation becomes negligible. If the number of correct and swapped assignments is larger, the separation power does not increase either, although the interval of the histogram has already been reduced to [0.48, 0.52], as shown in Figure 6.15(b). The problem here is probably missing input features which provide separation power between the correct and swapped assignments of the two light jets.



(a) Output value histogram of permutation 1 of the $b_{\text{had}_{\text{correct}}}$-DNN.

(b) Output value histogram of permutation 24 of the $b_{\text{had}_{\text{correct}}}$-DNN.

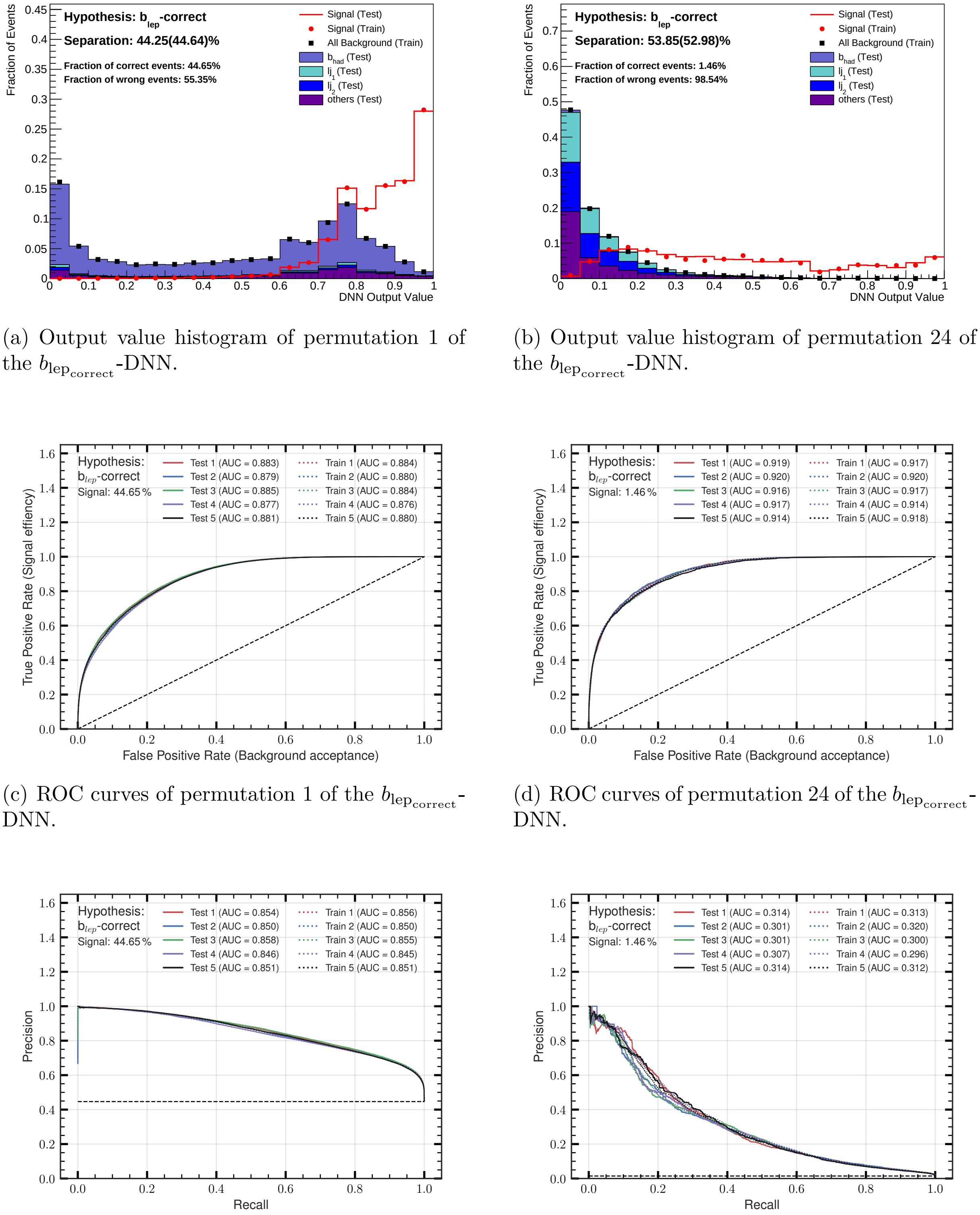
(c) ROC curves of permutation 1 of the $b_{\text{had}_{\text{correct}}}$-DNN.

(d) ROC curves of permutation 24 of the $b_{\text{had}_{\text{correct}}}$-DNN.

(e) Precision recall curves of permutation 1 of the $b_{\text{had}_{\text{correct}}}$-DNN.

(f) Precision recall curves of permutation 24 of the $b_{\text{had}_{\text{correct}}}$-DNN.

Figure 6.10: Evaluation plots of the b_{had}-DNN. Shown are the permutations 1 and 24. The different fractions of signal events is a consequence of the assignment of a b-jet and a light jet to the position of the hadronic b-jet. A b-tagged jet is more likely to be the correct choice.



(a) Output value histogram of permutation 1 of the $b_{\mathrm{lep}_{\mathrm{correct}}}$-DNN.

(b) Output value histogram of permutation 24 of the $b_{\mathrm{lep}_{\mathrm{correct}}}$-DNN.

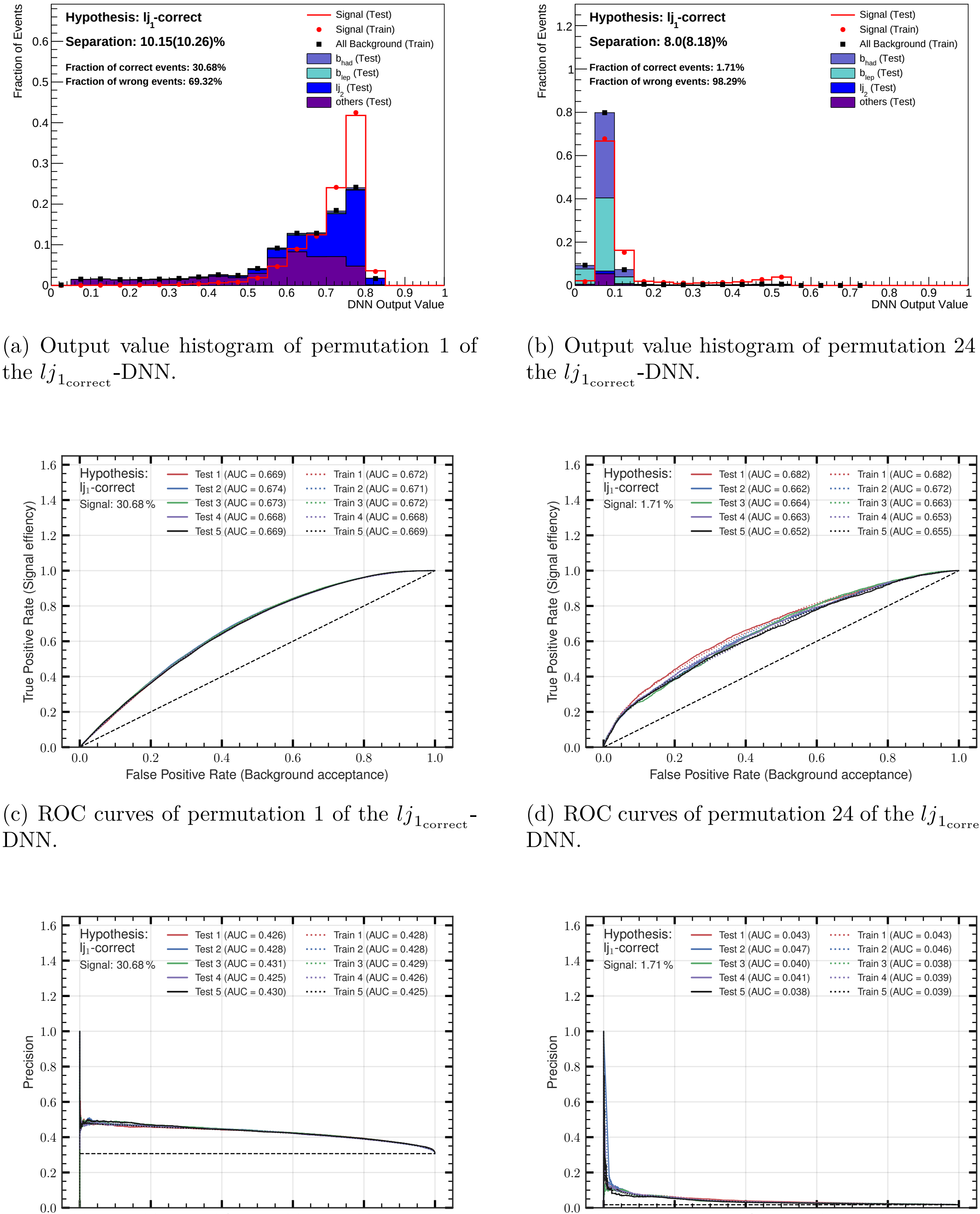
(c) ROC curves of permutation 1 of the $b_{\mathrm{lep}_{\mathrm{correct}}}$-DNN.

(d) ROC curves of permutation 24 of the $b_{\mathrm{lep}_{\mathrm{correct}}}$-DNN.

(e) Precision recall curves of permutation 1 of the $b_{\mathrm{lep}_{\mathrm{correct}}}$-DNN.

(f) Precision recall curves of permutation 24 of the $b_{\mathrm{lep}_{\mathrm{correct}}}$-DNN.

Figure 6.11: Evaluation plots of the b_{lep}-DNN. Shown are the permutations 1 and 24. The less likely assignments of jets with a low b-tagging to the position of the leptonic b-jet cause the differences in the fractions of signal events.



(a) Output value histogram of permutation 1 of the $lj_{1_{\text{correct}}}$-DNN.

(b) Output value histogram of permutation 24 of the $lj_{1_{\text{correct}}}$-DNN.

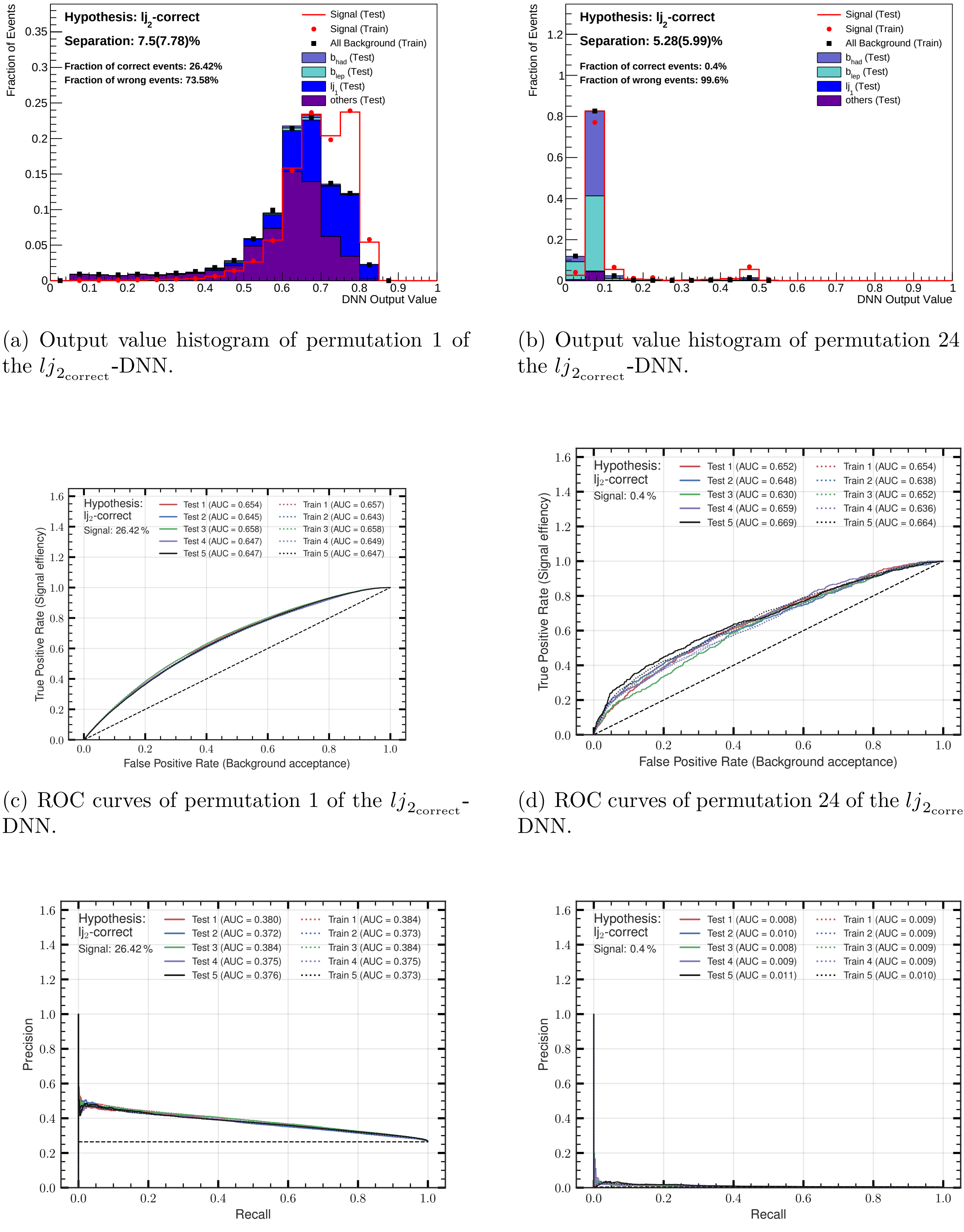
(c) ROC curves of permutation 1 of the $lj_{1_{\text{correct}}}$-DNN.

(d) ROC curves of permutation 24 of the $lj_{1_{\text{correct}}}$-DNN.

(e) Precision recall curves of permutation 1 of the $lj_{1_{\text{correct}}}$-DNN.

(f) Precision recall curves of permutation 24 of the $lj_{1_{\text{correct}}}$-DNN.

Figure 6.12: Evaluation plots of the $lj_{1_{\text{correct}}}$-DNN. Shown are the permutations 1 and 24. The different fractions of signal events is a consequence of the assignment of a b-jet and a light jet to the position of the first light jet. Assignments of not-b-tagged jets are more likely than others.



(a) Output value histogram of permutation 1 of the $lj_{2_{\text{correct}}}$-DNN.

(b) Output value histogram of permutation 24 of the $lj_{2_{\text{correct}}}$-DNN.

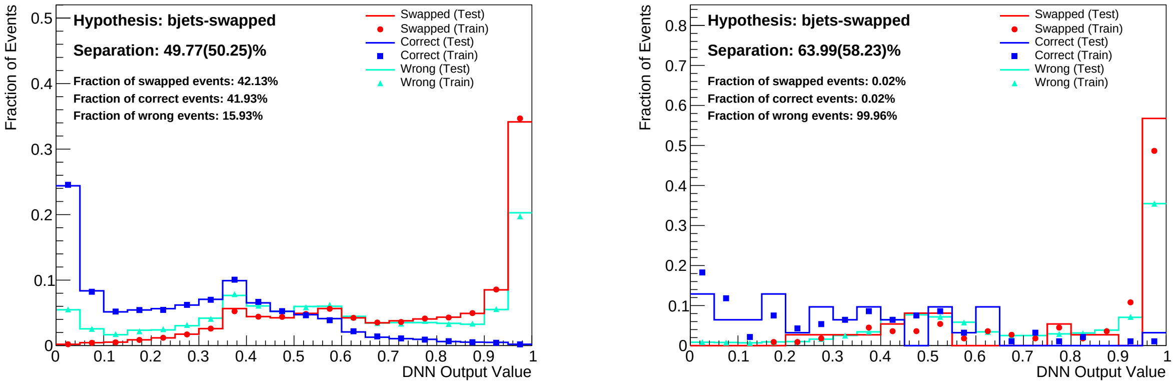
(c) ROC curves of permutation 1 of the $lj_{2_{\text{correct}}}$-DNN.

(d) ROC curves of permutation 24 of the $lj_{2_{\text{correct}}}$-DNN.

(e) Precision recall curves of permutation 1 of the $lj_{2_{\text{correct}}}$-DNN.

(f) Precision recall curves of permutation 24 of the $lj_{2_{\text{correct}}}$-DNN.

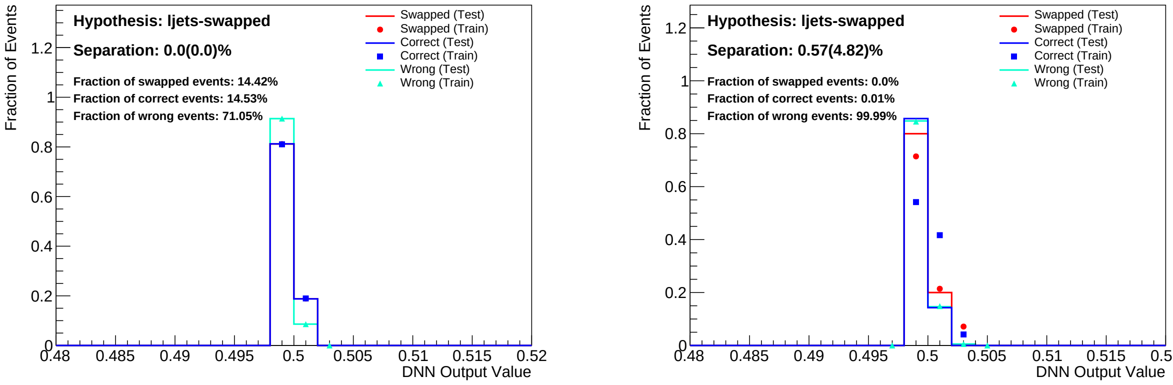
Figure 6.13: Evaluation plots of the $lj_{2_{\text{correct}}}$-DNN. Shown are the permutations 1 and 24. Due to more unlikely assignment of b-tagged jets to the second light jet, the permutations show different fractions of signal events.



(a) Output value histogram of permutation 1 of the "*b*-jets swapped"-DNN.

(b) Output value histogram of permutation 24 of the "*b*-jets swapped"-DNN.

Figure 6.14: Evaluation plots of the "*b*-jets swapped"-DNN. Shown are the permutations 1 and 24. The different fractions of signal events are a consequence of the assignment of *b*-jets and light jets to the *b*-jet positions. The majority of the events have the assignment of *b*-tagged jets to the *b*-jet positions as correct assignments.



(a) Output value histogram of permutation 1 of the "light jets swapped"-DNN.

(b) Output value histogram of permutation 24 of the "light jets swapped"-DNN.

Figure 6.15: Evaluation plots of the "light jets swapped"-DNN. Shown are the permutations 1 and 24. The different fractions of signal events is a consequence of the assignment of *b*-jets and light jets to the light jet positions. Less events have the *b*-tagged jets assigned correctly to the light jets.

Traceability of the Trained Classifiers

A common problem with DNNs is that they function like a black box. Although the trained weights are readable in the end, this does not allow the reconstruction of a readable function which would allow an analysis of how input features influence the results. Therefore, it is not clear what the DNN learned exactly and it is not predictable how a trained DNN will respond to data it has not been trained on. However, some attempts have been made to get a glance of how the network deals with the input features. One of them, which is presented here, is the *permutation importance.* This attempt is based on the shuffling of the input features. Particularly, one of the input features is selected as whole column and is shuffled across all events. The shuffled column is then saved for the events. Subsequently, the events with the shuffled input feature are fed into the trained classifier, the predictions are calculated and the performance is evaluated again. This is done for every feature in the input vector. The corresponding assumption is that the higher the importance of one feature is, the more the performance of the classifier should drop if this input feature is shuffled across all events.

The permutation importance has been applied for each of the binary DNNs. At first, the selected feature is shuffled. Secondly, per fold 100,000 elements from the testing dataset are selected and evaluated. The selection of 100,000 elements is done 5 times to compensate for statistical fluctuations. From the resulting predictions, the ROC curve AUC score is calculated and kept for plotting.

Figure 6.16 shows the results of the feature importance calculations.

As shown in Figure 6.16(a) the $b_{\mathrm{had}_{\mathrm{correct}}}$-DNN relies strongly on the b-tagging value. Its shuffling decreases the ROC AUC value of the classifier by approximately 0.35. The features concerning the hadronic top-quark were expected to be important as well since the hadronic b-quark is needed for its reconstruction and the top-quark has a certain mass, which is a constraint.

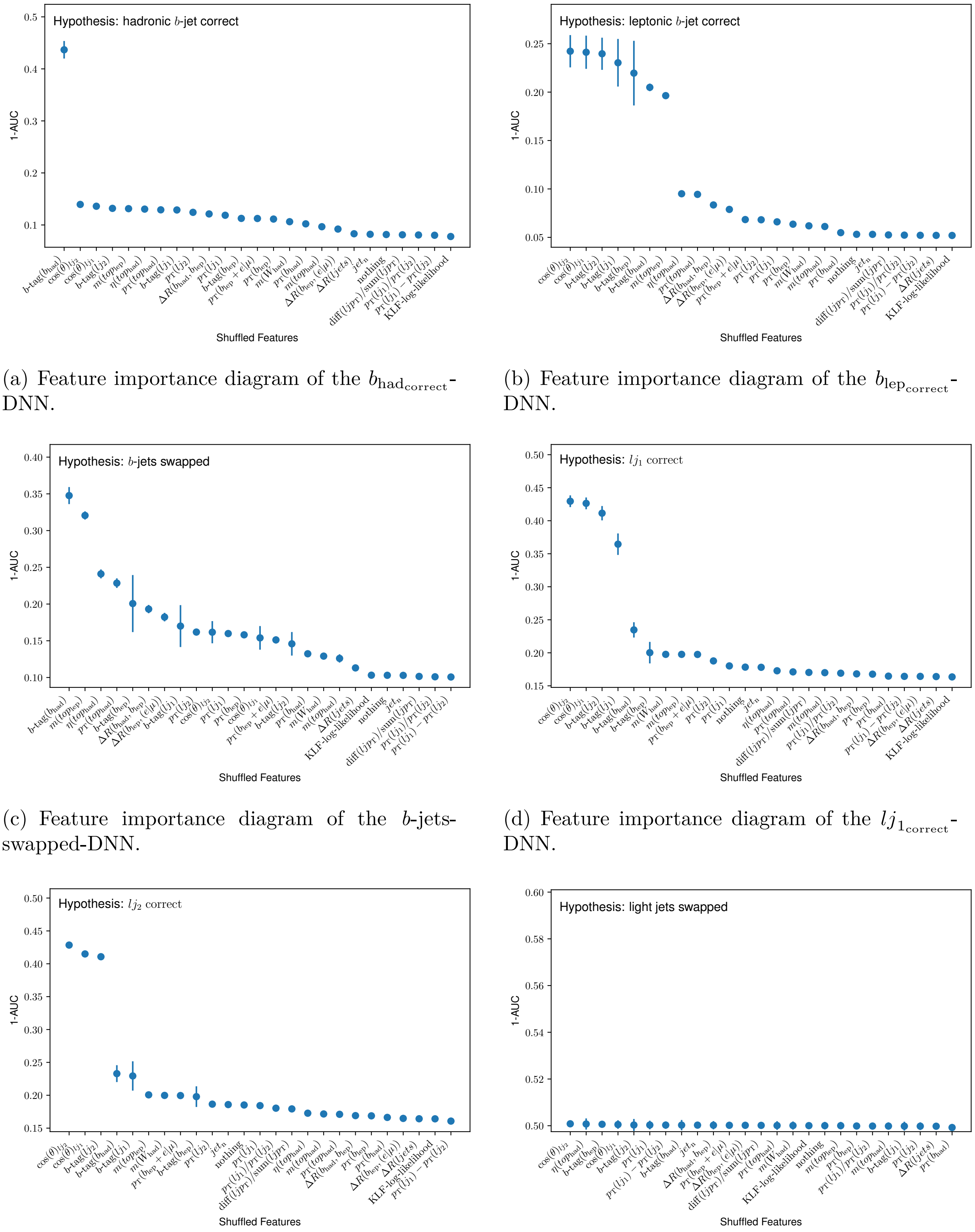
The classifier for the leptonic b-jet is remarkable in the context that the most important features belong to the light jets. These do not belong to any reconstruction in combination with the leptonic b-quark or the leptonic hemisphere. One possible explanation could be that the DNN checks whether the light jets look like they are correctly assigned. Depending on this, the number of possible other correct assignments is reduced and the classification in correct or wrong assignments is simplified. Here again, the features of the reconstructions connected to the leptonic b-jet seem to be less important.

The two light jet DNNs (Figures 6.16(d) and 6.16(e)) seem to rely both on the $\cos(\theta)$ values of the light jets and the b-tagging values, which seems reasonable. The features of the reconstructed partons, like the hadronic W-boson or the hadronic top-quark, are important as well since the AUC values decrease with the shuffling of them.

For the b-jets swapped-DNN, multiple features are important: the b-tagging values of the two $b$jets as well as the reconstructed values of the hadronic and the leptonic top-quark. For this DNN, many different features have a high impact on the AUC values as shown in Figure 6.16(c). This could indicate the necessity for this DNN to combine multiple values and learn from the combinations and correlations of these values.

The “light jets swapped”-DNN (Figure 6.16(f)) does not vary in its performance no matter which of the features was shuffled. This indicates, again, that this DNN learned nothing and outputs approximately 0.5 no matter which input is fed into it.

Although results of the feature importance are not complete proofs, they are a useful possibility to check if physical and logical expectations are met by the learning of the classifier. The existence of correlations cannot be proofed completely, but, if multiple shufflings let the performance drop to the same value, this can be a hint that these two, or more, features have only been in combinations meaningful for the classifier.



(a) Feature importance diagram of the $b_{\text{had}_{\text{correct}}}$-DNN.

(b) Feature importance diagram of the $b_{\text{lep}_{\text{correct}}}$-DNN.

(c) Feature importance diagram of the b-jets-swapped-DNN.

(d) Feature importance diagram of the $lj_{1_{\text{correct}}}$-DNN.

(e) Feature importance diagram of the $lj_{2_{\text{correct}}}$-DNN.

(f) Feature importance diagram of the light-jets-swapped-DNN.

Figure 6.16: Feature importance plots of the binary DNNs. In the figures, the performance of the binary DNNs are shown depending on which feature has been selected for the permutation importance algorithm. In this algorithm, one of the input features is shuffled over all events. A stronger decreasing performance is associated with a higher importance for the classification.

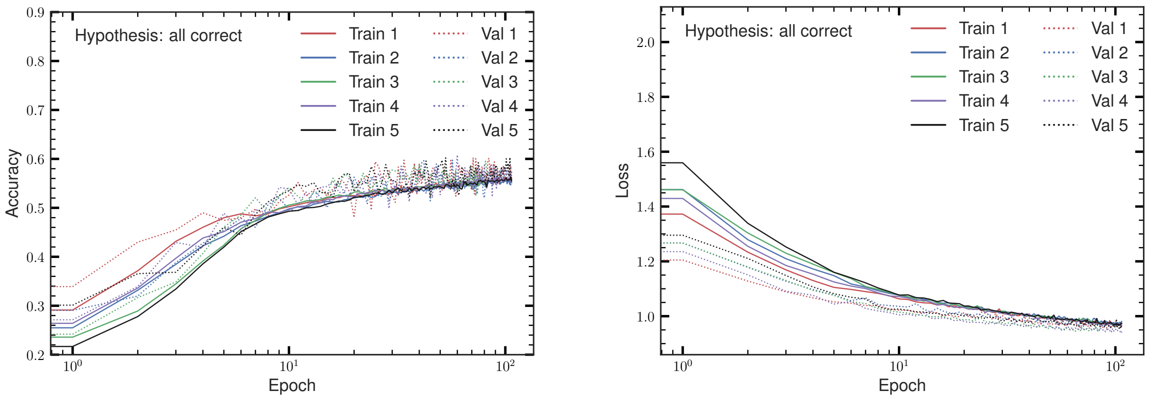
6.4.2 Evaluation of the Multi-Class DNN

The last step of the new approach is the evaluation of the multi-class classifier, which is supposed to find the best permutation in the end. To do so, training history and the performance of the trained classifier are evaluated. The importance of input features is not dealt with since the different logical combinations and reconstructions of more complex hypotheses are already known.

Training of the Model

Figure 6.17 shows the training history of the multi-class DNN. As listed in Table 6.7, the training of the multi-class DNN is terminated by early stopping. Therefore, the different folds stopped after a different number of epochs, where the longest fold took 136 epochs and the shortest 86 epochs. This gives a maximum relative difference of approximately 0.63. As shown in the plots, all folds converge to the same values of accuracy and loss. Therefore, the different training lengths have no significant impact on the convergence of the tracked values.

Figure 6.17(a) shows the accuracy history. As explained previously, the multi-class DNN was trained on five different but balanced labels. Therefore, an untrained classifier which guesses randomly the correct class would be correct in 20% of all cases, which gives a start accuracy of approximately 0.2. Within the first ten epochs, the accuracy rises up to approximately 0.5 and starts to asymptotically approach an accuracy of 0.55 within the rest of the training. The loss decreases exponentially, as shown in Figure 6.17(b). Additionally, it can be seen that the validation results are slightly better than the training results, which indicates that the model did not start to overfit. The accuracy fluctuations indicate that the model parameters reached a point near the global optimum.



(a) Accuracy history of multi-class DNN.

(b) Loss history of multi-class DNN.

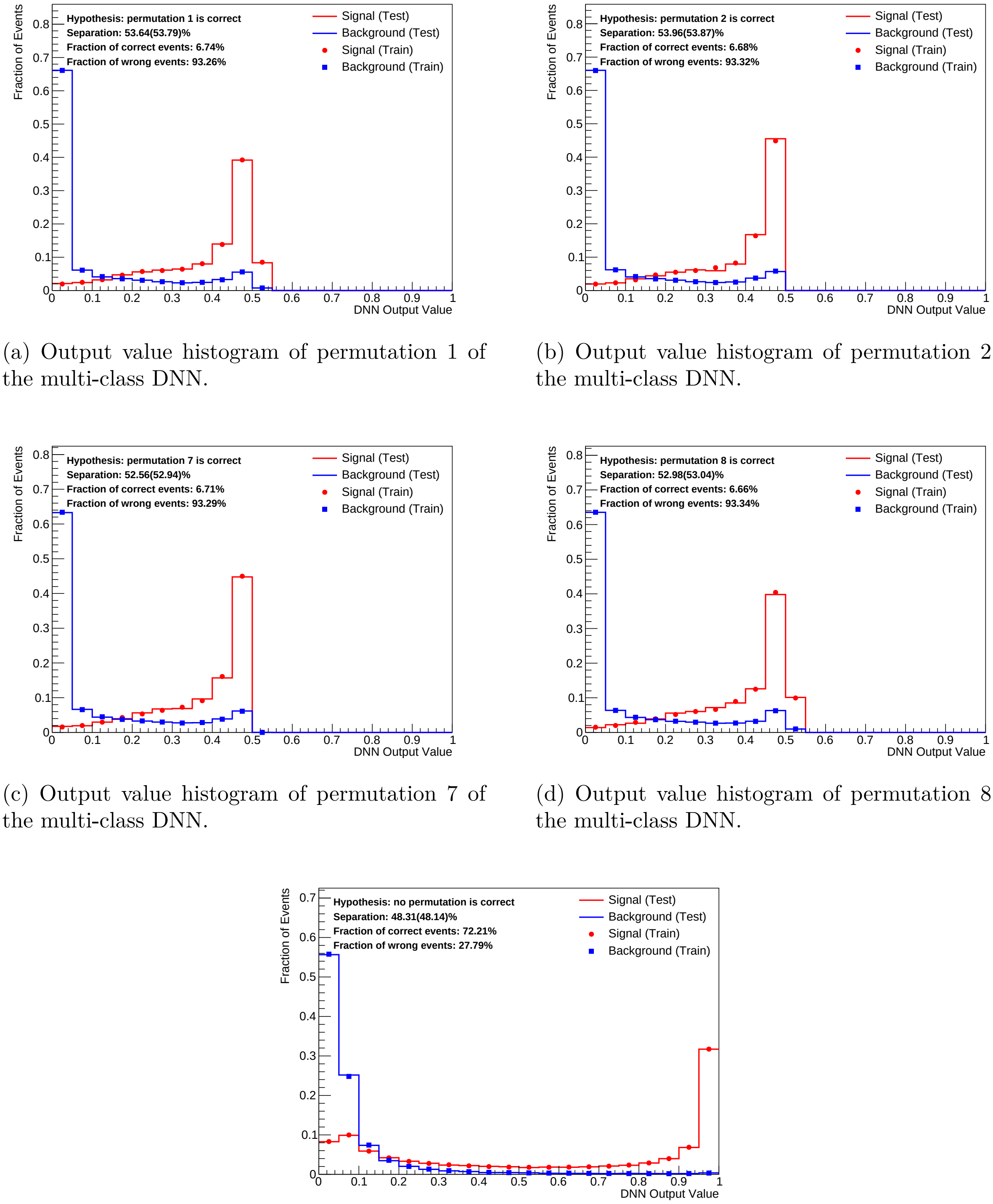
Figure 6.17: Training history plots for accuracy and loss values of the multi-class DNN which predicts whether or not a correct permutation exists among the given 24. If yes it predicts which of the given 24 permutations is the wanted.

Performance of the Trained Classifier

The results of the multi-class DNN are shown in Figure 6.18. The separation powers of the DNN for the different permutations (Figures 6.18(a) to 6.18(d)) are all approximately 53%. The separation of label 25 (Figure 6.18(e)), which stands for “no correct permutation present”, is approximately 48%. Conspicuous features in these plots is the distribution of the output values of the labels 1, 2, 7 and 8. Here, the DNN predicts only values less than 0.6, in contrast to the last histogram which uses the full range between 0 and 1. This is caused by the softmax activation function of the output layer in the multi-class DNN. It forces the values of the output vector to sum up to 1.

Therefore, the values of the output vector can be interpreted as probabilities, which concludes that the DNN is never more confident than 60% that a certain permutation is the correct one. However, it can predict that no permutation in the events represents the correct assignment with 100% confidence. This can be explained by looking at the correlations between different output vector elements. These correlations are shown in Figure 6.19.

The predictions of permutation 1 and 2 and permutation 7 and 8 are highly correlated. This indicates that the problem is the swapped assignment of the light jets. The binary DNN, which is responsible for this distinction was not able to learn anything and, therefore, the multi-class DNN cannot use any information about these assignments. Due to this, the DNN predicts both permutations of one of these pairs with almost equal probabilities, which leads to the distribution stopping at 0.6.



(a) Output value histogram of permutation 1 of the multi-class DNN.

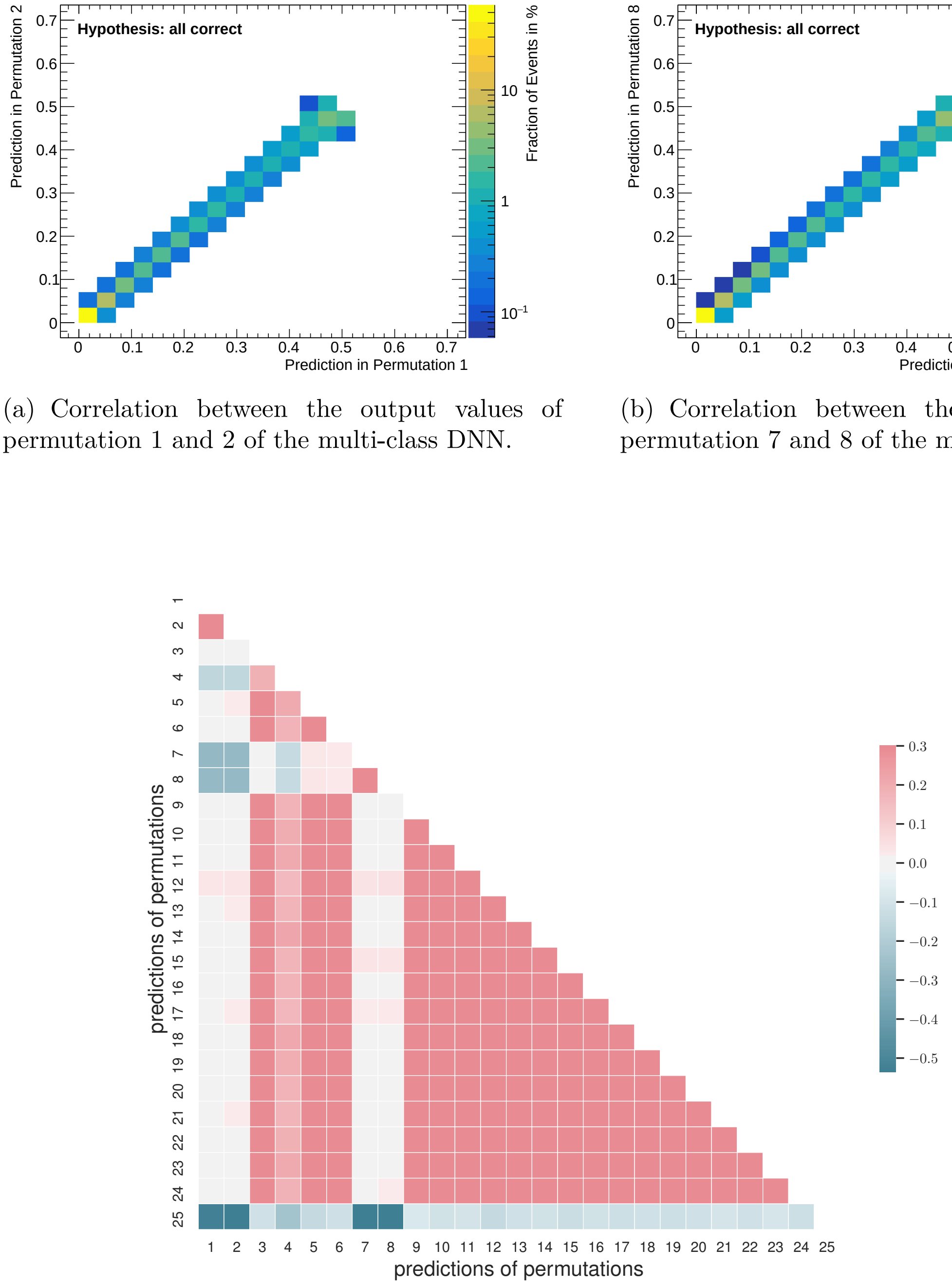
(b) Output value histogram of permutation 2 of the multi-class DNN.

(c) Output value histogram of permutation 7 of the multi-class DNN.

(d) Output value histogram of permutation 8 of the multi-class DNN.

(e) Output value histogram of the "no permutation is correct"-label of the multi-class DNN.

Figure 6.18: Evaluation plots of the b_{had}-DNN. Shown are the permutations 1 and 23. The different fractions of signal events is a consequence of the assignment of a b-jet and a light jet to the position of the hadronic b-jet. A b-tagged jet is more likely to be the correct choice.



(a) Correlation between the output values of permutation 1 and 2 of the multi-class DNN.

(b) Correlation between the output values of permutation 7 and 8 of the multi-class DNN.

(c) Correlation matrix between the output values for the different labels of the multi-class DNN.

Figure 6.19: Correlation plots of the multi class DNN. Shown are the correlations between the output values of permutation 1 and 2 and permutation 7 and 8. These pairs of permutations only differ in the swapped assignment of the light jets. The b-jets are the same. Additionally, the correlation coefficients between all labels are shown whereas label 25 symbolises the case in which no permutation is correct.

The correlation factors of all other permutation pairs are shown in Figure 6.19(c). The predictions of the permutations which have not been trained are correlated (correlation factor ≈ 0.3). Since no training data element was used where one of these permutations was correct, the DNN learns that these permutations are never correct. Therefore, the reason for these correlations is not so much based on the similarity between those permutations but rather on the common property of never being correct. The untrained and trained permutations are not correlated. The trained permutations are anti-correlated (correlation factor between -0.3 and -0.5) with each other except for the pairings shown in Figures 6.19(a) and 6.19(b)(correlation factor ≈ 0.3). These anti-correlations give a hint that the b-jet swapping values improve the predictions, although the anti-correlations between the groups of permutations 1 and 2 and permutations 7 and 8 are not as large as those between one of the groups and the "no permutation is correct"-label.

Finally, the performance of the multi-class approach is evaluated in terms of the reconstruction efficiency of the different hypotheses. Table 6.10 shows these results, in addition to the performances of the KLFitter approach, where the permutation with the highest event probability is assumed to be the correct one, and the BDT approach, which, based on the KLFitter approach, does one additional check if this permutation is the correct one. As it is shown the DNN approach outperforms the previous approaches in every category. The reconstruction efficiencies in respect to events where reconstructions could have been found correctly increased by approximately 11% in the b-jet categories and by approximately 15% for the combined reconstructions.

The evaluation contains some problematic categories. One of them is the definition of "present". This definition has been altered and split such that it corresponds better with the intuitive understanding of "present" and the new category "matched". For KLFitter and the top-decay width analysis "present" is calculated by checking for each t-parton if a r-jet exists such that they can be paired with a $\Delta R < 0.3$. The order here is $b_{\mathrm{had}} \rightarrow b_{\mathrm{lep}} \rightarrow lj_1 \rightarrow lj_2$ which concludes that the true hadronic b-jet has the highest chance of getting matched. All of the following t-partons can only be paired with one of the remaining r-jets. This is unintuitive since there is no natural order in these jets. The new definition of "present" is calculated the same way but without the constraint that r-jets that have been used for a pairing already cannot be used again. This causes the slightly higher values for the light jets. This, of course, is not ideal either. Therefore, the category "matched" has been created. This category is calculated by the algorithm represented in Figure 6.7. Based on the present or matched jets, the reconstructed particles can also be present or matched. The training is based on the "matched"-labels.

The other problematic definition is whether or not a light jet is assigned correctly. The

Fractions[%]	b_{lep}	b_{had}	**1st light jet**	**2nd light jet**	W_{had}	top_{had}	**All**
KLF + BDT present	96	96	74	74	53	51	49
DNN present	96	96	75	75	54	52	50
DNN matched	93	93	59	59	31	29	28

Table 6.9: In this table the fractions of events are shown in which a certain reconstruction could possibly be found correctly by a classifier. The definition of “present” was altered and slightly loosened to a more intuitive understanding. Additionally, a new category “matched” is introduced which represents a stricter definition of whether jet assignments can be found correctly.

Reco eff[%]	b_{lep}	b_{had}	**1st light jet**	**2nd light jet**	W_{had}	top_{had}	**All**
KLF correct vs. total	57	58	63	58	39	30	28
KLF correct vs. present	59	60	85	79	74	39	57
BDT correct vs. total	75	74	68	58	41	36	35
BDT correct vs. present	79	77	91	79	77	71	71
DNN correct vs. total	81	82	51	49	29	25	24
DNN correct vs. matched	**87**	**88**	**97**	**93**	**93**	**85**	**85**

Table 6.10: Performance comparison of the DNN approach with previous approaches. The reconstruction of the hadronic W-boson, the hadronic top-quark and the whole event is not sensitive to the light jet swapping to keep the results comparable with the previous approaches.

Definition	**correct vs. matched**	**correct vs. total**
lj_1 correct (strict)	49.31%	26.13%
lj_2 correct (strict)	46.87%	24.84%
One light jet correct (strict)	49.82%	26.4%
Two light jets correct (strict)	46.36%	24.57%
One light jet correct (swapping ok)	96.52%	51.16%
Two light jets correct (swapping ok)	93.05%	49.32%

Table 6.11: Different definitions of correct light jet reconstruction. The first two rows represent the most intuitive definition of a correct assignment. The previous approaches use a definition similar to the last two rows. They are of special interest when it comes to the combined reconstructions like the hadronic W-boson since a swapped assignment does not matter for this hypothesis.

definition of the previous approaches is constructed such that, at first, it is attempted to pair the true first light jet with one of the two assumed light jets from the “best permutation” and then the true second light jet. Therefore, the true second light jet has a smaller chance to be matched correctly. This procedure is an artefact caused by the fact that, in previous approaches, the correct reconstruction of light jets was not part of the research interest and, therefore, neglected. Table 6.11 gives an overview over different possible definitions of a correct light jet assignment, where the last row is equivalent to the hypothesis "W_{had} reconstructed correctly" in Table 6.10. This definition is applied for the DNN as well to keep the results comparable with those of the previous approaches.

However, this is not the most intuitive definition. The other rows show other possible definitions and how the DNN performs if they are applied. The first two rows represent the most intuitive definition.

Figure 6.20 shows the predictions divided into several categories, depending on the actual truth label of the event and the label vector the multi-class DNN assigned to the event. The label which is predicted to be the correct one is calculated by finding the trained label with the minimal summed cross-entropy to the predicted label vector. The true negatives are the events where no permutation with all jets assigned correct could have been found and the DNN predicted this correctly. The false positives are the cases where the DNN determined a "best permutation", although there was none. If the DNN predicted that no permutation exists although one could have been found, this is categorised as a false negative. The other segments represent the predictions of the correct permutations, the permutations with the swapped light jet assignment and the wrong predicted permutations. The near by identical fractions of correctly predicted permutations and predicted permutations with the swapped light jet assignment reflect the problem with the untrained "light jets swapped"-assignment and resulting correlations between permutation in the prediction vector of the multi-class DNN.

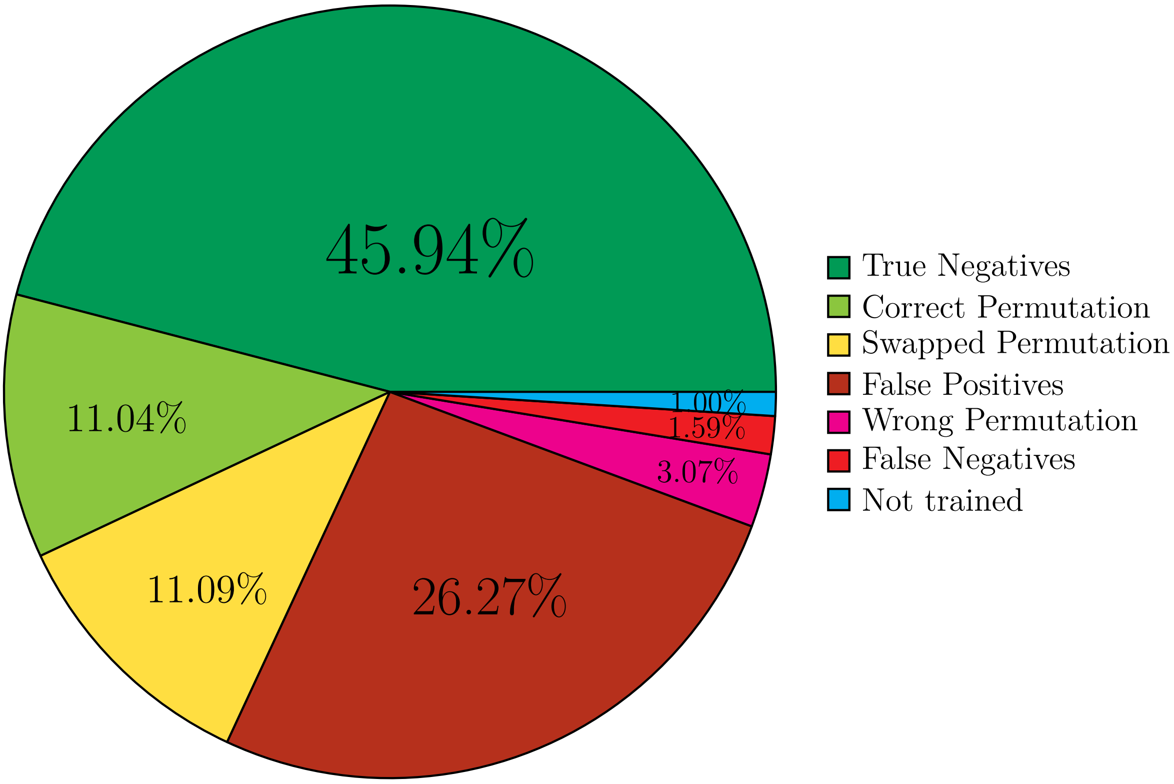


Figure 6.20: Pie chart of the prediction of the multi-class DNN. The segments of the true negatives and the correct permutations represent the fraction of events which are predicted correctly. The prediction of a hypothesis with swapped light jet assignment is separated from the wrong predictions since these permutations still can be used for the reconstruction. An example for the "Wrong Permutation" segment is an event in which permutation 1 would be the correct permutation but a different permutation than 1 or 2 are selected. False positives are the events in which no correct permutation exists but the DNN predicts one. False negatives are the events in which the DNN predicted that no correct permutation exists although one could be found.

7 Summary and Outlook

Concluding this thesis, a brief summary and conclusion are given in this chapter. Subsequently, possible ways to continue the studies presented in this thesis are shown.

7.1 Summary

This thesis presented the development of a new extension framework to KLFitter using a stacked machine learning approach consisting of DNNs. Furthermore, previous approaches have been explained which built the basis for the studies presented in this thesis. The predecessor approach was rebuilt and reimplemented forming a stepping stone towards concepts of the new approach. The concept of compound reconstruction hypotheses motivated the architecture of the stacked multi-class classifier. The implementation has been documented and carried out. The resulting classifiers were evaluated in respect to multiple properties and the performance results were interpreted.

The reconstruction efficiency for b-jets is improved to 87% for hadronic and 88% for leptonic b-jets. This is an 11% increase with respect to the BDT and a 28% increase to KLFitter. Using a similar definition of correctly reconstructed light jets the performance has reached 97% for one and 93% efficiency for two light jets. Especially the case of two correctly reconstructed light jets is improved by 14% with respect to both: the BDT and KLFitter. These improvements allow an increased efficiency for the combined reconstructions as well. The hadronic W-boson can be found correctly 16% more often than with the BDT and 19% more than with KLFitter. This, again, improves the efficiency for hadronic top-quarks as well. Here the performance improved from KLFitter to the BDT by 32% from 39% to 71% and from the BDT again by 14% to 85% of the DNN approach. Complete $t\bar{t}$-decays can be reconstructed correctly in 85% of all cases with the DNN approach which is 14% more compared to the BDT and 28% compared to KLFitter.

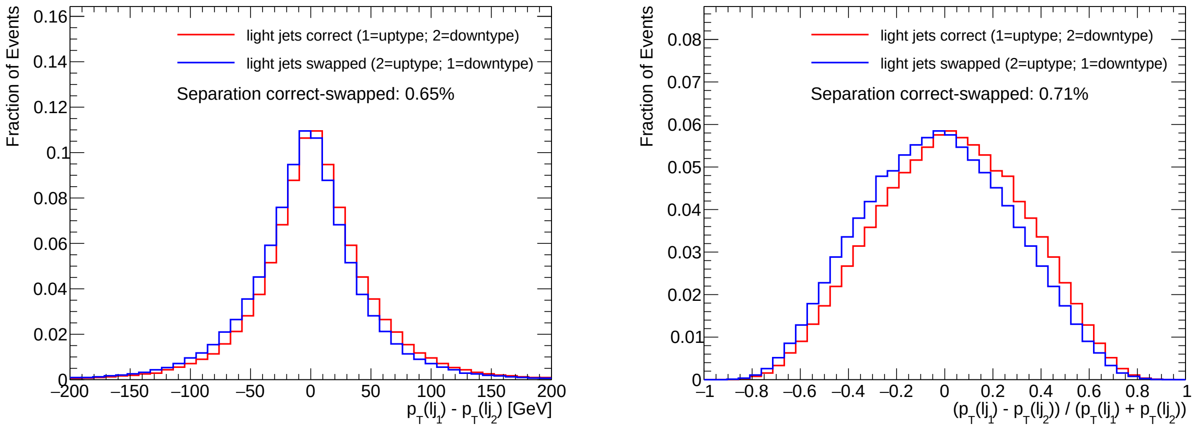
This new classifier approach provides the possibility of a higher reconstruction efficiency of $t\bar{t}$-decays. Therefore, using this tool can improve studies of the top-quark and its properties. Additionally, the multi-class DNN can be retrained to predict a different hypothesis. The hadronic W-boson, for example, can be reconstructed in 93% of all cases if the DNN is trained to reconstruct the complete event. With a retraining to especially this hypothesis, the reconstruction efficiency could improve even more. Therefore, the new approach does not only provide a better $t\bar{t}$-decay reconstruction efficiency, but a basis for multiple other trainings as well. Furthermore, it provides a blue print for other possible machine learning tasks based on the prediction of one or more hypotheses combined from several elementary hypotheses which can be predicted on their own. Several particle physics processes ,including reconstructions of decay products, could be addressed with an approach based on the one successfully presented here.

7.2 Outlook

Based on the results from the presented approach, recommended actions can be identified. The primary problem during the reconstruction of the events is the assignment of the light jets. The reconstruction efficiency of single light jets in terms of a strict assignment has to be larger and, additionally, the light jet swapping detection should be improved. One way to do so could be to use flavour information of the jets. A constraint that one light jet has to originate from an up-type-flavour-quark and one from a down-type-flavour-quark might be useful. Figure 7.1 shows potential separation power between light jets based on this idea. The values shown in this figure are already present on the simulation level. So far, no sufficient way of reconstructing these values has been developed. Currently, development is done towards c-tagging to classify whether or not a jet originated from a charm-quark. If this development is successful, at least cases in which both chosen light jets have large c-tagging values assigned could be rejected. Further development towards tagging algorithms for other quark flavours will be helpful as well.

An additional analysis that could be conducted and might be successful, is using convolutional neural networks which are especially aimed at image recognition. The 6×24 prediction matrices of each event can be interpreted as a “picture of an event”.

In general, other input features could improve the model performance. As shown in Figure 6.16, not all currently used variables increase the performance of all binary DNNs. A more individualised input variable selection could lead to better performances.



(a) Separation plot of the p_{T}-differance between the light jets.

(b) Separation plot of a more complex p_{T}-combination between the light jets.

Figure 7.1: Separation plots of p_{T}-distributions of up-type-flavour-quarks and down-type-flavour-quarks. Using and improving flavour reconstruction, could increase reconstruction efficiencies towards the correct light jet assignment.

Bibliography

[1] P. Speckmayer, et al., *The toolkit for multivariate data analysis, TMVA 4*, *in* J. Phys. Conf. Ser. **219**, 032057 (2010), doi:10.1088/1742-6596/219/3/032057

[2] P. Baldi, et al., *Jet substructure classification in high-energy physics with deep neural networks*, *in* Physical Review D **93(9)** (2016), doi:10.1103/physrevd.93.094034

[3] ATLAS Collaboration, *A neural network clustering algorithm for the ATLAS silicon pixel detector*, *in* JINST **9**, P09009 (2014), doi:10.1088/1748-0221/9/09/P09009

[4] L. Breiman, *Random Forests*, *in* Machine Learning **45(1)**, 5–32 (2001), doi:10.1023/A:1010933404324

[5] J. Erdmann, et al., *A likelihood-based reconstruction algorithm for top-quark pairs and the KLFitter framework*, *in* Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **748**, 18–25 (2014), doi:10.1016/j.nima.2014.02.029

[6] T. Dado, *Direct measurement of the top-quark decay width with the ATLAS detector*, Ph.D. thesis (2019)

[7] S. Glashow, *Partial Symmetries of Weak Interactions*, *in* Nucl. Phys. **22**, 579–588 (1961), doi:10.1016/0029-5582(61)90469-2

[8] A. Salam, *Weak and Electromagnetic Interactions*, *in* Conf. Proc. C **680519**, 367–377 (1968), doi:10.1142/9789812795915_0034

[9] S. Weinberg, *A Model of Leptons*, *in* Phys. Rev. Lett. **19**, 1264–1266 (1967), doi:10.1103/PhysRevLett.19.1264

[10] S. Glashow, J. Iliopoulos, L. Maiani, *Weak Interactions with Lepton-Hadron Symmetry*, *in* Phys. Rev. D **2**, 1285–1292 (1970), doi:10.1103/PhysRevD.2.1285

[11] D. J. Gross, F. Wilczek, *Ultraviolet Behavior of Nonabelian Gauge Theories*, *in* Phys. Rev. Lett. **30**, 1343–1346 (1973), doi:10.1103/PhysRevLett.30.1343

[12] H. Politzer, *Reliable Perturbative Results for Strong Interactions?*, *in* Phys. Rev. Lett. **30**, 1346–1349 (1973), doi:10.1103/PhysRevLett.30.1346

[13] H. Politzer, *Asymptotic Freedom: An Approach to Strong Interactions*, *in* Phys. Rept. **14**, 129–180 (1974), doi:10.1016/0370-1573(74)90014-3

[14] W. J. Marciano, H. Pagels, *Quantum Chromodynamics: A Review*, *in* Phys. Rept. **36**, 137 (1978), doi:10.1016/0370-1573(78)90208-9

[15] Atlas Collaboration, *Observation of a new particle in the search for the Standard Model Higgs boson with the Atlas detector at the LHC*, *in* Phys. Lett. B **716**, 1–29 (2012), doi:10.1016/j.physletb.2012.08.020

[16] Cms Collaboration, *Observation of a New Boson at a Mass of 125 GeV with the Cms Experiment at the LHC*, *in* Phys. Lett. B **716**, 30–61 (2012), doi:10.1016/j.physletb.2012.08.021

[17] F. Englert, R. Brout, *Broken Symmetry and the Mass of Gauge Vector Mesons*, *in* Phys. Rev. Lett. **13**, 321–323 (1964), doi:10.1103/PhysRevLett.13.321

[18] P. W. Higgs, *Broken Symmetries and the Masses of Gauge Bosons*, *in* Phys. Rev. Lett. **13**, 508–509 (1964), doi:10.1103/PhysRevLett.13.508

[19] P. W. Higgs, *Broken symmetries, massless particles and gauge fields*, *in* Phys. Lett. **12**, 132–133 (1964), doi:10.1016/0031-9163(64)91136-9

[20] H. Bachmair, et al., editors, *Units and Fundamental Constants in Physics and Chemistry: Subvolume b: Fundamental Constants in Physics and Chemistry*, chapter The elementary charge, 79–82, Springer-Verlag (1992)

[21] Super-Kamiokande Collaboration, *Evidence for Oscillation of Atmospheric Neutrinos*, *in* Phys. Rev. Lett. **81**, 1562–1567 (1998), doi:10.1103/PhysRevLett.81.1562

[22] SNO Collaboration, *Measurement of the Rate of $\nu_e + d \to p + p + e^-$ Interactions Produced by ^8B Solar Neutrinos at the Sudbury Neutrino Observatory*, *in* Phys. Rev. Lett. **87**, 071301 (2001), doi:10.1103/PhysRevLett.87.071301

[23] SNO Collaboration, *Direct Evidence for Neutrino Flavor Transformation from Neutral-Current Interactions in the Sudbury Neutrino Observatory*, *in* Phys. Rev. Lett. **89**, 011301 (2002), doi:10.1103/PhysRevLett.89.011301

[24] CDF Collaboration, *Observation of Top Quark Production in $\overline{p}p$ Collisions with the Collider Detector at Fermilab*, *in* Phys. Rev. Lett. **74**, 2626–2631 (1995), doi:10.1103/PhysRevLett.74.2626

[25] DØ Collaboration, *Observation of the Top Quark*, *in* Phys. Rev. Lett. **74**, 2632–2637 (1995), doi:10.1103/PhysRevLett.74.2632

[26] Atlas Collaboration, *Measurements of the top-quark mass with the Atlas detector*, *in* PoS **ICHEP2018**, 656 (2019), doi:10.22323/1.340.0656

[27] Cms Collaboration, *Measurement of the ratio $\mathcal{B}(t \to Wb)/\mathcal{B}(t \to Wq)$ in pp collisions at $\sqrt{s}$ = 8 TeV*, *in* Phys. Lett. B **736**, 33–57 (2014), doi:10.1016/j.physletb.2014.06.076

[28] Particle Data Group, *Review of Particle Physics*, *in* Phys. Rev. D **98**, 900 (2018), doi:10.1103/PhysRevD.98.030001

[29] Atlas Collaboration, *The Atlas Experiment at the Cern Large Hadron Collider*, *in* Journal of Instrumentation **3(08)**, S08003–S08003 (2008), doi:10.1088/1748-0221/3/08/s08003

[30] Alice Collaboration, *The Alice experiment at the Cern Lhc*, *in* Journal of Instrumentation **3(08)**, S08002–S08002 (2008), doi:10.1088/1748-0221/3/08/s08002

[31] The Cms Collaboration, *The Cms experiment at the Cern Lhc*, *in* Journal of Instrumentation **3(08)**, S08004–S08004 (2008), doi:10.1088/1748-0221/3/08/s08004

[32] Lhcb Collaboration, *The Lhcb Detector at the Lhc*, *in* Journal of Instrumentation **3(08)**, S08005–S08005 (2008), doi:10.1088/1748-0221/3/08/s08005

[33] S. Agostinelli, et al., *Geant4—a simulation toolkit*, *in* Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **506(3)**, 250–303 (2003), doi:10.1016/S0168-9002(03)01368-8

[34] The Atlas Collaboration, *Measurements of b-jet tagging efficiency with the ATLAS detector using $t\bar{t}$ events at $\sqrt{s}$ = 13 TeV*, *in* Journal of High Energy Physics **2018(8)**, 89 (2018), doi:10.1007/JHEP08(2018)089

[35] A. Caldwell, D. Kollár, K. Kröninger, *BAT – The Bayesian analysis toolkit*, *in* Computer Physics Communications **180(11)**, 2197–2209 (2009), doi:10.1016/j.cpc.2009.06.026

[36] W. S. McCulloch, W. Pitts, *A logical calculus of the ideas immanent in nervous activity*, *in* The bulletin of mathematical biophysics **5(4)**, 115–133 (1943), doi:10.1007/BF02478259

[37] F. Rosenblatt, *The Perceptron: A Probabilistic Model for Information Storage and Organization in The Brain*, *in* Psychological Review **65(6)**, 386–408 (1958), doi:10.1037/h0042519

[38] D. Cireşan, et al., *Multi-column deep neural network for traffic sign classification*, *in* Neural Networks **32**, 333–338 (2012), doi:10.1016/j.neunet.2012.02.023

[39] S. De, et al., *Predicting the Popularity of Instagram Posts for a Lifestyle Magazine Using Deep Learning*, *in 2017 2nd International Conference on Communication Systems, Computing and IT Applications (CSCITA)*, 174–177 (2017), doi:10.1109/CSCITA.2017.8066548

[40] A. Zhavoronkov, et al., *Deep learning enables rapid identification of potent DDR1 kinase inhibitors*, *in* Nature Biotechnology **37(9)**, 1038–1040 (2019), doi:10.1038/s41587-019-0224-x

[41] D. Silver, et al., *Mastering the game of Go without human knowledge*, *in* Nature **550(7676)**, 354–359 (2017), doi:10.1038/nature24270

[42] D. E. Rumelhart, G. E. Hinton, R. J. Williams, *Learning representations by back-propagating errors*, *in* Nature **323(6088)**, 533–536 (1986), doi:10.1038/323533a0

[43] K. Beyer, et al., *When Is "Nearest Neighbor" Meaningful?*, *in Database Theory — ICDT'99*, 217–235, Springer Berlin Heidelberg, Berlin, Heidelberg (1999), doi:10.1007/3-540-49257-7_15

[44] J. Sola, J. Sevilla, *Importance of input data normalization for the application of neural networks to complex industrial problems*, *in* IEEE Transactions on Nuclear Science **44(3)**, 1464–1468 (1997), doi:10.1109/23.589532

[45] R. Anand, et al., *An improved algorithm for neural network classification of imbalanced training sets*, *in* IEEE Transactions on Neural Networks **4(6)**, 962–969 (1993), doi:10.1109/72.286891

[46] N. Srivastava, et al., *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*, *in* Journal of Machine Learning Research **15(56)**, 1929–1958 (2014)

[47] C. Bishop, *Regularization and Complexity Control in Feed-forward Networks*, *in Proceedings International Conference on Artificial Neural Networks ICANN'95*, vol. 1, 141–148, EC2 et Cie (1995)

[48] D. Allen, *The Relationship Between Variable Selection and Data Agumentation and a Method for Prediction*, *in* Technometrics **16**, 125–127 (2012), doi:10.1080/00401706.1974.10489157

[49] L. Breiman, *Bagging Predictors*, *in* Machine Learning **24(2)**, 123–140 (1996), doi:10.1023/A:1018054314350

[50] F. Chollet, et al., *Keras*, *in* (2015), URL `https://keras.io`

[51] X. Glorot, Y. Bengio, *Understanding the difficulty of training deep feedforward neural networks*, *in* Journal of Machine Learning Research - Proceedings Track **9**, 249–256 (2010)

[52] D. P. Kingma, J. Ba, *Adam: A Method for Stochastic Optimization* (2014)

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

[54] S. Hochreiter, et al., *Gradient Flow in Recurrent Nets: The Difficulty of Learning LongTerm Dependencies*, 237–243, IEEE Press (2001)

[55] G. B. Goh, N. O. Hodas, A. Vishnu, *Deep learning for computational chemistry*, *in* Journal of Computational Chemistry **38(16)**, 1291–1307 (2017), doi:10.1002/jcc.24764

[56] J. Hanley, B. Mcneil, *A Method of Comparing the Areas Under Receiver Operating Characteristic Curves Derived from the Same Cases*, *in* Radiology **148**, 839–43 (1983), doi:10.1148/radiology.148.3.6878708

[57] P. Baldi, et al., *Searching for Exotic Particles in High-Energy Physics with Deep Learning*, *in* Nature communications **5(1)**, 4308 (2014), doi:10.1038/ncomms5308

[58] P. Baldi, et al., *Enhanced Higgs Boson to $\tau^+\tau^-$ Search with Deep Learning.*, *in* Physical review letters **114(1)**, 111801 (2015), doi:10.1103/physrevlett.114.111801

[59] P. Baldi, et al., *Parameterized Machine Learning for High-Energy Physics*, *in* The European Physical Journal C **76(5)**, 235 (2016), doi:10.1140/epjc/s10052-016-4099-4

[60] C. Adam-Bourdarios, et al., *The Higgs Machine Learning Challenge*, *in* Journal of Physics: Conference Series **664(7)**, 072015 (2015), doi:10.1088/1742-6596/664/7/072015

[61] R. Brun, F. Rademakers, *ROOT — An object oriented data analysis framework*, *in* Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **389(1)**, 81–86 (1997), doi:10.1016/S0168-9002(97)00048-X, new Computing Techniques in Physics Research V

[62] R. Brun, et al., *ROOT — Reference Guide*, *in* (2020), URL `https://root.cern/doc/master/`

[63] M. Abadi, et al., *TensorFlow: A system for large-scale machine learning*, *in* (2016)

[64] B. Fortner, *HDF: The hierarchical data format*, *in* Dr Dobb's J Software Tools Prof Program **23(5)**, 42 (1998)

[65] Scientific Python Community, *Pythonic ROOT*, *in* (2017), URL `http://www.rootpy.org/`

[66] S. Van Der Walt, S. C. Colbert, G. Varoquaux, *The NumPy array: a structure for efficient numerical computation*, *in* Computing in Science & Engineering **13(2)**, 22 (2011)

[67] W. McKinney, *Data structures for statistical computing in python*, *in* Stéfan van der Walt, Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference1*, 51–56 (2010), doi:10.25080/MAJORA-92BF1922-00A

[68] J. Davis, M. Goadrich, *The Relationship between Precision-Recall and ROC Curves*, *in Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, 233–240, Association for Computing Machinery, New York, NY, USA (2006), doi:10.1145/1143844.1143874

[69] P. A. Flach, M. Kull, *Precision-Recall-Gain Curves: PR Analysis Done Right*, *in Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, 838–846, MIT Press, Cambridge, MA, USA (2015), doi:10.5555/2969239.2969333

Hiermit versichere ich, dass ich die vorliegende Arbeit bisher bei keiner anderen Prüfungsbehörde eingereicht, sie selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe.

Ort, Datum Wernigerode, 17.08.2020

Eigenhändige Unterschrift D Neumann

Im Rahmen der Atlas-Kollaboration werden am Cern im Large Hadron Collider (Lhc) Protonen beschleunigt und zur Kollision gebracht. Dabei finden unter anderem sogenannte Topquark-Paarproduktionen statt, bei welchen aus dem Protonenzusammenstoß Partonen und aus deren Interaktion ein Topquark und ein Antitopquark entstehen. Diese Teilchen zerfallen dann in weitere Teilchen im Endzustand. Um das Topquark untersuchen zu können, müssen die Zerfallsprodukte im Detektor rekonstruiert werden, was durch Untergrundprozesse und Messungenauigkeiten erschwert wird. Um eine möglichst gute Rekonstruktion der Top-Quark-Ereignisse zu erhalten, werden verschiedene Teilchenzuordnungen und und deren Wahrscheinlichkeiten mit Hilfe des *Kinematic Likelihood Fitter* berechnet.

Aufbauend auf *Kinematic Likelihood Fitter* und einer Erweiterung durch einen Boosted Decision Tree, wurde ein tiefes neuronales Netz entwickelt, um die Bewertung der wahrscheinlichsten Teilchenzuordnung in einem ersten Schritt zu rekonstruieren und zu verbessern. Im Anschluss daran wurden weitere tiefe neuronale Netze entwickelt, welche Aussagen zu Teilen der Rekonstruktion ermöglichen. Dadurch erhalten dessen Nutzer die Möglichkeit, zugeschnitten auf ihre Problemstellung und mit einer höheren Präzision, einen größeren Datensatz an relevanten Ereignissen aus den ursprünglich gemessenen Daten herauszufiltern.