
Novel Jet Flavour Tagging Algorithms exploiting Adversarial Deep Learning Techniques with Efficient Computing Methods and Preparation of Open Data for Robustness Studies

Von der Fakultät für Mathematik, Informatik und Naturwissenschaften der RWTH Aachen University zur Erlangung des akademischen Grades einer Doktorin der Naturwissenschaften genehmigte Dissertation

vorgelegt von

Annika Stein, M.Sc. RWTH

aus Hermeskeil

Berichter: Prof. Dr. rer. nat. Alexander Schmidt
Prof. Dr. rer. nat. Michael Krämer

Tag der mündlichen Prüfung: 20.06.2024

Diese Dissertation ist auf den Internetseiten der Universitätsbibliothek verfügbar.



Abstract

Machine learning algorithms are an indispensable tool for science. Precision tests of the standard model of particle physics and searches for processes involving elementary particles are facilitated with novel reconstruction algorithms that exploit complex neural network architectures. Such applications however oftentimes rely on simulated processes, one example being the identification of the flavour of quarks or gluons initiating particle jets (jet flavour tagging). Besides other experimental sources of uncertainties, the efficiency uncertainties stemming from object identification involving neural networks can contribute significantly to final results, expressed for example as uncertainties in a signal strength. Tests with control regions reveal differences in performance between samples obtained through simulation and those from detector data, meaning that calibration is required. With this in mind, this thesis aims at providing not only efficient measures to mitigate this performance gap between data and simulation from the ground up (especially when the algorithm performs very well on simulation), but also derives concepts that assist in understanding why the proposed approaches work. Building up from early versions of adversarial attacks and defenses, a new algorithm, denoted Normed Gradient Method (NGM), is introduced and adapted for physics applications for the first time. This also marks the introduction of a state-of-the-art transformer architecture for small-radius jets for the CMS experiment. In combination with NGM, the currently best performance metrics for this task at CMS are achieved, improving over previous algorithms. The network maintains high performance even under exposition to systematic modifications of inputs. It is thus the first time an (adversarially) robust algorithm is introduced for the official reconstruction software of a high-energy particle detector. Event throughput improves compared to an algorithm that achieves slightly worse performance. The efficient integration was significantly facilitated by a novel software framework specifically developed for jet flavour tagging that is capable of performance studies with data, although the neural network is still in development (training) stage. The time-to-insight from neural network training to reliable performance studies is significantly reduced. This is possible, because the framework is built around only one data tier that serves multiple purposes to study the neural network.

Besides the work within the context of one experiment, another focus is the preparation and utilization of CERN Open Data for robustness studies. This last part of this thesis is dedicated to the conversion of already available open datasets, which can only be used with experiment-specific software, into machine learning-friendly formats. The result is the first open dataset that allows small-radius jet flavour tagging studies with simulation and recorded detector data for a broader audience of data scientists that do not necessarily know how to operate the experiment software. As every experiment provides unique data, tools, and problem statements, the concept of a wrapper structure is introduced, which allows applying a core set of adversarial techniques in a toolbox to the different use cases.

Zusammenfassung

Algorithmen des Maschinellen Lernens sind ein nicht mehr wegzudenkendes Werkzeug für die Wissenschaft. Präzisionstests des Standardmodells der Teilchenphysik und die Suche nach Prozessen, in welchen elementare Teilchen involviert sind, werden durch neue Rekonstruktionsalgorithmen erleichtert, die komplexe Architekturen Neuronaler Netzwerke nutzen. Solche Anwendungen hängen allerdings oft von simulierten Prozessen ab, wobei die Identifikation von Quark-Flavour oder Gluonen, welche Teilchenjets initiieren (Jet Flavour Tagging) ein Beispiel darstellt. Neben anderen experimentellen Unsicherheiten kann die Unsicherheit der Effizienz bei der Objektidentifizierung mithilfe Neuronaler Netzwerke in hohem Maße zu finalen Resultaten beitragen, welche zum Beispiel mit Signalstärken ausgedrückt werden. Tests mit Kontrollregionen machen Unterschiede in der Performance für Stichproben, die durch Simulation gewonnen wurden, und solche, die aus Detektordaten stammen, sichtbar, woraus sich die Notwendigkeit der Kalibrierung ergibt. Dies im Sinn zielt diese Arbeit darauf ab, nicht nur effiziente Mittel bereitzustellen, die die Performancelücke zwischen Daten und Simulation von Grund auf schließen (besonders dann, wenn die Performance für die Simulation sehr gut ist), als auch Konzepte zu entwickeln, die helfen zu verstehen, weshalb die Ansätze wirken. Ausgehend von früheren Versionen für "adversarial" (feindliche) Attacks und Defenses, wird ein neuer Algorithmus, genannt Normed Gradient Method (NGM), eingeführt und zum ersten Mal für eine physikalische Anwendung verwendet. Dies stellt ebenso die Einführung modernster Transformer-Architekturen für Jets mit kleinem Radius in CMS dar. In Kombination mit NGM werden die momentan besten Performance-Metriken für diese Aufgabe in CMS erreicht, besser als mit früheren Algorithmen. Das Netzwerk behält hohe Performance auch dann bei, wenn es systematischen Modifikationen der Eingaben ausgesetzt wird. Es ist das erste Mal, dass ein (adversarial) robuster Algorithmus in der offiziellen Rekonstruktionssoftware eines Teilchendetektors in der Hochenergiephysik integriert wird. Der Durchsatz an Ereignissen steigt im Vergleich zu einem Algorithmus mit etwas geringerer Performance. Die effiziente Integration wurde signifikant durch eine neue, speziell auf Jet Flavour Tagging spezialisierte Software-Umgebung erleichtert, welche Performance-Studien mit Detektor-Daten bereits während der Entwicklungs- (Trainings-)Phase erlaubt. Die Zeit, die vom Training bis zur Verfügbarkeit von zuverlässigen Performance-Studien vergeht, wird signifikant reduziert. Dies ist möglich, da das Entwicklungssystem auf nur einer Stufe der Datenrepräsentation aufbaut, die mehreren Zwecken zur Untersuchung des Neuronalen Netzwerks dient.

Neben der Arbeit im Kontext eines Experiments, ist ein anderer Fokus die Vorbereitung und Nutzung von CERN Open Data für Robustheits-Studien. Dieser letzte Teil der Arbeit widmet sich der Umwandlung bereits vorhandener offener Datensätze, welche ausschließlich mit Experiment-spezifischer Software verwendet werden können, zu Formaten, die sich für Maschinelles Lernen eignen. Das Ergebnis ist der erste offene Datensatz, mit welchem Jet Flavour Tagging Studien für Jets mit geringem Radius mit Simulation und aufgenommenen Detektordaten für ein breiteres Publikum von Data Scientists möglich sind, auch wenn diese nicht notwendigerweise über das Wissen verfügen, die Experiment-Software zu bedienen. Da jedes Experiment individuelle Daten, Hilfsmittel und Problemstellungen bereitstellt, wird das Konzept einer einhüllenden Struktur eingeführt, welche die Anwendung einer Grundmenge von Adversarial-Techniken in einem Werkzeugkasten für verschiedene Nutzungsmöglichkeiten erlaubt.

Contents

Acronyms, Abbreviations	11
1 Introduction	13
I Foundations and Methods	15
2 Theoretical Aspects	17
2.1 The Standard Model of Particle Physics	17
2.1.1 Particles in the SM	17
2.1.2 Mathematical Formulation	19
2.2 Protons and Proton-Proton Collisions	27
2.2.1 The Proton	27
2.2.2 Scattering Processes and Terminology	27
2.2.3 Proton-Proton Collisions	27
2.2.4 Event Generation	30
3 The CMS Experiment at the Large Hadron Collider	33
3.1 The Large Hadron Collider	33
3.1.1 LHC Experiments	35
3.1.2 Luminosity and Pileup	36
3.1.3 Run 3 and HL-LHC Upgrades	37
3.2 The Compact Muon Solenoid	38
3.2.1 Coordinate System	38
3.2.2 Detector Components	39
3.2.3 Trigger System and Data Acquisition	42
3.2.4 Detector Simulation	42
3.2.5 Object Reconstruction	43
4 Jets	45
4.1 Jet Reconstruction	46
4.1.1 Pileup Mitigation and Jet Identification	47
4.1.2 Jet Energy Corrections	49
4.1.3 Missing Transverse Energy	49
4.2 Jet Flavour Tagging	50
4.2.1 Distinguishing Features of Heavy-Flavour Jets	50
4.2.2 Quarks versus Gluons	52
4.2.3 Truth Flavour and Labelling	52
4.2.4 Different Generations of Algorithms and Applications	53
5 Overview of Utilized Machine Learning Methods	59
5.1 Machine Learning and Deep Neural Networks	59
5.1.1 General Aspects	59
5.1.2 DNN Predictions with Forward Passes	60

5.1.3	DNN Training and Performance Evaluation	61
II	Development of Robust Jet Flavour Tagging Algorithms	71
6	Robustness Studies with Simulated Particle Jet Samples and the Loss Surface	73
6.1	Introduction	73
6.2	Dataset and Input Features	75
6.2.1	Input Features	76
6.2.2	Preprocessing	76
6.3	Methods	76
6.3.1	Reference Classifier	76
6.3.2	Evaluation Metrics	77
6.3.3	Adversarial Attacks	77
6.3.4	Adversarial Training	79
6.3.5	Inference	80
6.4	Robustness to Mismodeling	81
6.4.1	Adversarial Attack	81
6.4.2	Vulnerability of the Nominal Training	82
6.4.3	Improving Robustness through Adversarial Training	83
6.4.4	Probing Flavour Dependence of the Attack as a Proxy for Generalization Capability	85
6.5	Robustness in the Context of Other Mismodeling Scenarios	87
6.5.1	Smearing Inputs with a Gaussian Noise Term	87
6.5.2	Transferability of Adversarial Samples as a Black-Box Attack	89
6.5.3	Shifting Inputs Systematically with Up/Down Variations	89
6.6	Loss Surface Study	91
6.6.1	Properties of Loss Manifolds for a Jet Tagging Algorithm Trained on Nominal or Adversarial Samples	91
6.6.2	Discussion	92
6.7	Conclusion	93
7	Probing Adversarial Techniques with Detector Data: Novel Jet Tagging Algorithms for the CMS Experiment	95
7.1	Technical Aspects Enabling Robustness- and Data/MC-Studies with Custom Algorithms and Inputs	95
7.2	The First Robust DeepJet Algorithm	96
7.2.1	The Nominal Algorithm and its Adversarially Trained Companion	96
7.2.2	Performance of DeepJet on Simulated Samples with Nominal or Adversarial Training and Exposing the Models to Adversarial Attacks	100
7.2.3	Performance Comparison of Nominal and Adversarial Training in Three Control Regions	101
7.2.4	Do Adversarial Attacks Mimic or Reduce Input Mismodeling?	111
7.3	A New Era: Development and Integration of a Robust Transformer Network: RobustParticleTransformer	118
7.3.1	A Transformer Architecture for AK4 Jets	118
7.3.2	The Normed Gradient Method	121

7.3.3	Setup for Adversarial Training	124
7.3.4	Tests on Simulated Run 2 Samples	124
7.3.5	Validation with LHC Run 2 Data	134
7.3.6	Implementation for Run 3	136
7.4	Related Approaches	139
7.4.1	Ablation Studies and Interpretability	139
7.4.2	Alternative Sample Distortions for Training, Taking Input Modeling into Account	140
7.4.3	Flavour-Dependent Adversarial Training Strategy to Improve Data-MC Agreement in Three Control Regions	141
7.4.4	Gradient-Reversal for Domain Adaptation	142
III	Open Data Studies	143
8	Preparation of an Experiment-Independent Software Framework for Open Data	145
8.1	Introduction and Problem Statement	145
8.1.1	A Wrapper for Open Data AI Safety Studies	146
8.1.2	Data Preparation	146
8.2	Testing, Deploying, and a Note on the Educational Potential	151
8.2.1	Preliminaries	151
8.2.2	Creating the First Loss Surface Plot with CMS Open Data for Jet Flavour Tagging	152
8.2.3	Deployment on kaggle	154
8.2.4	Comparison with Other CERN / CMS Open Data Frameworks	155
IV	Conclusion	157
9	Summary and Discussion	159
	Appendix	165
A	Further Details for Mathematical Construction of the SM	165
A.1	Generators of Lie Algebras	165
A.2	Dirac-Pauli Representation of the γ -matrices	165
B	Samples, Configuration, Target Definitions and Input Features	167
B.1	Simulated Delphes Dataset	167
B.1.1	Targets	167
B.1.2	Input Variables	167
B.2	CMS Datasets	170
B.2.1	Sample Lists for Run 2 and Run 3 Studies	170
B.2.2	DeepJet	170
B.2.3	RobustParticleTransformer	171

C	Delphes Simulation-Only Cross-Checks	181
C.1	Training Progress and Convergence with Training and Validation Loss	181
C.2	Regular Training with Cross Entropy Loss versus Focal Loss	182
C.3	Output Scores and Discriminant Shapes for Nominal and Distorted Samples . .	182
D	A Modern Framework for Jet Tagger Development: From Training to First Data-to-Simulation Agreement Results in One Go	191
D.1	Existing Jet Tagging Software Tools of the LHC Community and Required Features for Robustness and Generalization Studies with Detector Data	191
D.1.1	Data Tiers and Use Cases	192
D.1.2	Training Stage	192
D.1.3	Integration into Official Reconstruction Pipeline	193
D.1.4	Studying Tagger Performance with Data	193
D.2	Merging Multiple Tools into One with Fewer Required Data Structures	194
D.2.1	Implementation of a Custom Evaluation Pipeline for Algorithms in Development	195
D.3	Conclusion and Recommendations	198
E	Supplementary studies with DeepJet	199
E.1	Alternative Quantification Metrics for Data/MC Agreement	199
E.1.1	Discriminant Distributions with Fine-Grained Binning	199
E.1.2	Jenson-Shannon Divergence and χ^2 Values with 30 or 60 Bins	199
F	Supplementary Studies with RobustParticleTransformer	205
F.1	Correlations Between Nominal and Distorted Samples (Flavour-Dependent) . .	205
F.2	Alternative Training Scenario	206
F.3	Technical Details: Modifying RobustParT for Run 3	211
F.3.1	Architecture and Input Changes compared to Run 2 RobustParT	211
F.3.2	Integration in CMSSW	211
F.3.3	Cross-Check of ONNX Evaluation Pipeline in CMSSW with PyTorch . . .	212
F.3.4	Timing Performance for Feature Production and Inference in CMSSW . . .	213
G	Extended PFNano Samples and Estimated Relevance of Robust Algorithms Applied to Physics Analysis	215
	References	219
	List of Figures	245
	List of Tables	251
	Contributions	253
	Eidesstattliche Erklärung	257
	Acknowledgements	259

Acronyms, Abbreviations

Adam Adaptive Moments Estimation	DY Drell-Yan process
AI Artificial intelligence	ECAL Electromagnetic calorimeter
AK4 small-radius jets, clustered using the anti- k_T algorithm with $R = 0.4$	EM Electromagnetic
ALICE A Large Ion Collider Experiment	EW Electroweak
AOD Analysis Object Data	EWSB Electroweak symmetry breaking
ATLAS A Toroidal LHC Apparatus	FCNN Fully-connected neural network
AUC Area under the curve	FGSM Fast Gradient Sign Method
BBR Beam-beam remnants	FL Focal loss
BCE Binary cross entropy	FPIX Forward pixel layers
BDT Boosted decision tree	FPR False positive rate
BN Batch normalization	FSR Final state radiation
BPIX Barrel pixel layers	GEANT GEometry ANd Tracking
BR Branching ratio	GELU Gaussian Error Linear Unit
BSM Beyond the SM	GEM Gas electron multiplier
CCE Categorical cross entropy	GNN Graph neural network
CERN Conseil européen pour la recherche nucléaire / European Organization for Nuclear Research	GPU Graphics processing unit
CHS Charged hadron subtraction	HB Hadron barrel calorimeter
CKM Cabbibo–Kobayashi–Maskawa (matrix)	HCAL Hadronic calorimeter
CMS Compact Muon Solenoid	HE Hadron endcap calorimeter
CMSSW CMS Software	HEP High energy physics
CNN Convolutional neural network	HF Hadron forward calorimeter
CPF Charged PF candidate	HFT Heavy-flavour tagging
CPU Central processing unit	HL-LHC High-Luminosity LHC
CR Control region	HLT High Level Trigger
CRAB CMS Remote Analysis Builder	HO Hadron outer calorimeter
CSC Cathode strip chamber	ISR Initial state radiation
CSV Combined secondary vertex	IP Interaction point (context: <i>LHC ring</i>) / impact parameter (context: <i>jets</i>)
DL Deep learning	JEC Jet energy correction
DGLAP Dokshitzer–Gribov–Lipatov–Altarelli–Parisi	JER Jet energy resolution
DNN Deep neural network	JES Jet energy scale
DT Drift tube chamber	JSD Jenson-Shannon (JS) divergence
	KLD Kullback-Leibler (KL) divergence
	L1 Level 1 Trigger
	LEP Large Electron-Positron Collider

LHAPDF Les Houches Accord PDF	Generator
LHC Large Hadron Collider	pp Proton-Proton (collision)
LHCb Large Hadron Collider beauty	PP Particle Physics
LHEF Les Houches Event Files	PS Parton shower (context: <i>pp collisions</i>) / Proton Synchrotron (context: <i>CERN ac-</i> <i>celerator complex</i>)
LLO Leading-logarithmic order	PU Pileup
LN Layer normalization	PUPPI Pileup-per-particle interaction
LO Leading order	PV Primary vertex
LS Long Shutdown	QCD Quantum chromodynamics, <i>also</i> QCD multijet sample
LO Leading order	QED Quantum electrodynamics
LSTM Long Short-Term Memory	QFD Quantum flavourdynamics
MC Monte Carlo	QFT Quantum field theory
ME Matrix element	QM Quantum mechanics
MET Missing transverse energy	ReLU Rectified Linear Unit
MHA Multi-head attention	RECO Reconstruction
ML Machine learning	RobustParT Robust Particle Transformer
MLP Multilayer perceptron	ROC Receiver Operator Characteristic (con- text: <i>classifier performance metric</i>) / readout chip (context: <i>detector modules</i>)
MPI Multiple parton interactions	RNN Recurrent neural network
nDOF Number of degrees of freedom	RPC Resistive plate chamber
NGM Normed Gradient Method	SF Scale factor
NLO Next-to-leading order	SIP Signed impact parameter
NN Neural network	SM Standard Model of Particle Physics
NNLO Next-to-next-to-leading order	SR Signal region, <i>also</i> special relativity
NPF Neutral PF candidate	ST Single top quark production
ODP Open Data Portal	SUSY Supersymmetry
ONNX Open Neural Network Exchange	SV Secondary vertex
OS – SS Opposite sign (minus) same sign	TPR True positive rate
ParT Particle Transformer	TTbar top quark-antiquark pair production
PAT Physics Analysis Toolkit	UE Underlying event
PDF Parton distribution function	VEV Vacuum expectation value
PF Particle-Flow	WP Working point
PGD Projected Gradient Descent	
PMNS Pontecorvo–Maki–Nagakawa– Sakata	
PNet ParticleNet	
POG Physics Object Group	
Powheg Positive Weight Hardest Emission	

1

Introduction

“If you step back a little and say we want to do A.I., then you will realize that A.I. needs knowledge, reasoning, and explanation.”

– Oren Etzioni

Several branches of science, with high energy physics (HEP) as one example, profit from novel machine learning (ML) techniques [1–4]. Finding the needle in the haystack or improving precision measurements of the properties of elementary particles as predicted by the standard model of particle physics (SM) (see also Chapter 2) benefits from ever-growing datasets, more complex neural network (NN) architectures, and the availability of large-scale computing infrastructures that exploit high-dimensional information.

Many such applications however rely on simulated samples that can only model the fundamental physics processes up to a given order and can not fully describe all detector effects that arise under real experiment conditions. Before NNs can be used successfully for analysis purposes, their output scores need to be calibrated, which shall yield agreement within uncertainties that are then propagated to a final result. For a diverse set of analyses carried out within the physics programs of the Large Hadron Collider (LHC) experiments [5,6], efficient object identification is crucial to measure properties of elementary particles with high precision, or set more stringent limits for specific parameters (an overview of the experimental environment, including the Compact Muon Solenoid (CMS) detector [7] is given in Chapter 3).

For this thesis, jet flavour tagging* serves as the main ML-assisted object identification example [8–10], further introduced in Chapters 4 and 5. The purpose of such algorithms is to identify the flavour of the elementary particles (in partonic view: quarks, gluons) initiating particle jets, which are collimated objects consisting of many constituents that develop as a result of fragmentation and hadronization processes. Knowledge of the origin of these structures can assist in many analyses for which signal and/or background topologies include jets. Analyses utilizing such algorithms need high efficiency and small misidentification rates for effective separation between signal and background, resilient algorithms which stay effective also with changed conditions, good agreement between distributions obtained from simulation and recorded data (especially if the full outputs are used subsequently in a final score to extract signals), and small uncertainties introduced by these experimental methods. The main goal of this thesis is to develop algorithms with high performance and, at the same time, improved robustness against artificially introduced input modifications and better generalization capability in terms of applying an algorithm to data that has only been trained on simulation, thus improving the agreement.

These challenges are addressed first with studies on simulated samples only (Chapter 6) as

*Sometimes also denoted as jet tagging, tagging, jet flavour identification, heavy-flavour tagging (HFT). All terms will be used synonymously during this thesis.

published in [11], where a NN for jet tagging is improved in terms of robustness by the introduction of adversarial training. Adversarial training is utilized as a defense strategy counteracting the impact of an adversarial attack, in this case, the Fast Gradient Sign Method (FGSM), inspired by Refs. [12–15]. Besides performance studies and the exploitation of additional mismodeling scenarios, it is shown for the first time that adversarial techniques for jet tagging also point at better resilience with respect to physics-motivated input modifications. Explanations are given with the help of the underlying loss surfaces connected to nominal or adversarial training, which were introduced for Ref. [16]. In terms of NN architectures, this chapter only describes studies carried out with fully-connected NN (FCNN) layers with a moderate number of inputs.

A more complete picture can be gained by performing dedicated comparisons of different training strategies not only with simulation, but also when performing inference with detector data. For this purpose, selection criteria to enrich different control regions, as discussed in Refs. [9, 17], are employed for nominal or adversarial training. Starting with one of the popular algorithms for small-radius jet tagging in CMS during the Run 2 period, DeepJet [18], Chapter 7 presents the first dedicated study of the connection between adversarial robustness and data-to-simulation agreement of recent jet tagging algorithms (see also Ref. [19]). The in-depth discussion of input modifications and consequences for output distributions follows.

As a next step, a state-of-the-art architecture, a transformer model based on an attention mechanism [20], as has been adapted for small radius jet tagging in CMS [21], is strengthened with a novel adversarial technique, denoted Normed Gradient Method (NGM) [22]. This algorithm (RobustParticleTransformer) not only yields the currently best performance for this classification task, but it is also the first robust deep learning algorithm supported by the CMS software. Data-to-simulation agreement is studied while still being in the algorithm development process, the advantages of adversarial training are already visible in data before typical commissioning workflows can begin.

The last part of the thesis, Chapter 8, introduces all necessary technical preparations for creating and deploying an open dataset containing low-level information suitable for jet flavour tagging. In particular, datasets provided through the CERN Open Data portal covering CMS simulation and data (for this purpose, recorded in 2015 during Run 2 of the LHC) offer the relevant inputs to perform robustness studies, but can currently only be used with the help of experiment-specific software. This requires familiarity with the available tools, the relevant computing infrastructure, and reliable understanding of physics objects – which is not necessarily given for the broad audience of data scientists. Combining the domain-specific components and ML techniques on a ML platform (kaggle[†]) to fill the role of a “data physicist” [23], the last sections of this thesis bridge the gap and introduce “ML-friendly” datasets in an experiment-independent way. Their usability is demonstrated, and as similar efforts are conducted for different kinds of experiments, an efficient software framework becomes necessary. The concept of a wrapper is introduced that combines the core adversarial techniques with applications from particle physics and the educational potential of this approach is reviewed as well.

[†]See also <https://www.kaggle.com>.

Part I

Foundations and Methods

2

Theoretical Aspects

“We have seen particle physics emerge as the playground of group theory.”

– Freeman Dyson

Studying the properties of elementary particles and their interactions is of prime interest in particle physics. The interplay between theoretical predictions and experimental probes of these properties is a fundamental concept with which different models are investigated to describe the inconceivable quantum world at smallest scales. One successful model, which does have some limitations after all, is described below. This chapter also introduces selected phenomenological results and gives a brief overview of event generation.

2.1 The Standard Model of Particle Physics

The standard model (SM) of particle physics is a quantum field theory (QFT) that combines quantum mechanics (QM) and special relativity (SR) in a mathematical framework [24, 25]. In the following sections, many explanations and conventions follow Refs. [1, 26]. For more specialized reviews and descriptions focussing on QFT, the reader is referred to literature such as [27–32]. Unless stated otherwise, repeated indices per term imply summing over them, using the Einstein summation convention, and natural units $\hbar = c = 1$ are assumed throughout.

The known elementary particles emerge from field excitations and with the exception of gravity, which is described by the general theory of relativity, the SM includes the remaining fundamental forces that can be associated with mediator particles operating as quanta of gauge fields. For gravity, there currently exists no consistent quantum theory that could describe this fourth force*. Consequently, in the SM, the fundamental interactions and associated charges are electromagnetism / quantum electrodynamics (QED) with electric charge, the strong interaction / quantum chromodynamics (QCD) acting on colour charge, and the weak interaction / quantum flavourdynamics (QFD) [33] with the associated weak isospin. The latter two are discussed under the term electroweak (EW) theory, the unification of QED and QFD [25, 33, 34].

2.1.1 Particles in the SM

The building blocks of particles in the SM are elementary (without any substructure) and pointlike, and can be either matter particles / fermions (with spin 1/2), or bosons, of which

*The relative coupling strength for gravity is negligibly small, compared to the other interactions relevant for HEP. For example, between fundamental particles separated roughly by the radius of a proton, gravity would act 37 orders of magnitude weaker than the strong interaction [26]. A proposed force carrier for gravity, the graviton with zero mass and spin 2, has been hypothesized, but there is no experimental hint for such a boson yet [1, 26].

there are force carriers / gauge bosons (with spin 1), and the Higgs boson (the only spin 0-particle in the SM, thus, a scalar) [1, 26].

Fermions can be further categorized as quarks or leptons. A common property between them is that there are six fundamental particles of each subcategory, which are grouped into three generations with increasing mass. Everyday matter is formed with particles of the first (lightest) generation, into which the heavier particles decay most predominantly. Fermions also have anti-particles with flipped sign for electric, flavour, or leptonic charges (or more generally, the charge-like quantum numbers), but same mass.[†]

There are three types of charged leptons (electron (e), muon (μ), tau (τ)), all with electric charge $Q = -1e$ (with e referring to the elementary charge), and their neutral companions called neutrinos (ν_e, ν_μ, ν_τ). In the SM, charged leptons are massive, while neutrinos are included as massless. But the observation of neutrino oscillations requires at least two neutrino flavours to have mass to explain the oscillation between eigenstates, an effect not described in the SM [1]. Common for all leptons is that they carry weak isospin and thus participate in the weak interaction, while only the charged leptons also participate in the electromagnetic (EM) interaction.

Quarks can be categorized by their electric charge, where $Q = \frac{2}{3}e$ refers to the so-called “up-type” quarks, namely up (u), charm (c) and top (t). The “down-type” quarks with $Q = -\frac{1}{3}e$ consist of the down (d), strange (s) and bottom (b) quarks. The ordering represents increasing mass from first to third generation. Quarks and leptons are summarized in Table 2.1, more details related to their quantum numbers and treatment within the mathematical framework in the SM is added in Table 2.2 later in this chapter.

Table 2.1: Fermions (quarks, leptons) in the SM, split into three generations each, with their masses taken from Ref. [1]. For quarks, the modified minimal subtraction scheme ($\overline{\text{MS}}$) is used for all but the top quark mass. In the $\overline{\text{MS}}$ scheme, the divergent part and a constant are absorbed into counterterms to perform renormalization for perturbative calculations beyond leading order (see also Ref. [1] and Section 2.2.2).

Fermion type	Name (symbol)	Generation	Mass m in GeV	Comment
Quarks	up (u)	1.	$2.16^{+0.49}_{-0.26}$ MeV	$\overline{\text{MS}}$ scheme @ $\mu = 2$ GeV
	down (d)		$4.67^{+0.48}_{-0.17}$ MeV	$\overline{\text{MS}}$ scheme @ $\mu = 2$ GeV
	charm (c)	2.	$1.27 \text{ GeV} \pm 0.02 \text{ GeV}$	$\overline{\text{MS}}$ scheme with $\overline{m} = \overline{m}(\mu = \overline{m})$
	strange (s)		$93.4^{+8.6}_{-3.4}$ MeV	$\overline{\text{MS}}$ scheme @ $\mu = 2$ GeV
	top (t)	3.	$172.69 \text{ GeV} \pm 0.30 \text{ GeV}$	Direct measurement
	bottom (b)		$4.18^{+0.03}_{-0.02}$ GeV	$\overline{\text{MS}}$ scheme with $\overline{m} = \overline{m}(\mu = \overline{m})$
Leptons	electron (e)	1.	0.511 MeV	Uncertainty omitted (small)
	electron neutrino (ν_e)		$< 0.8 \text{ eV}$	@ 90% CL [1, 35]
	muon (μ)	2.	106 MeV	Uncertainty omitted (small)
	muon neutrino (ν_μ)		$< 0.19 \text{ MeV}$	@ 90% CL [1, 36]
	tau (τ)	3.	$1776.86 \text{ MeV} \pm 0.12 \text{ MeV}$	
	tau neutrino (ν_τ)		$< 18.2 \text{ MeV}$	@ 95% CL [1, 37]

Quarks do not only carry electric charge and weak isospin, but also colour charge with possible values red, green or blue. Colour as a new kind of charge is a concept that had to

[†]The “anti-” prefix may be omitted for simplicity. For example, oftentimes one speaks of quark pairs when a quark-antiquark pair is meant.

be introduced to explain the observation of the Δ^{++} resonance, which consists of three up quarks [1, 26]. An odd number of quarks yields half-integer spin and thus a composite particle with fermionic properties (even numbers give rise to bosonic composite particles, correspondingly). Without this additional charge, such a fermionic bound state with otherwise identical quantum numbers would violate the Pauli exclusion principle. Indeed, quarks only occur in bound states by means of colour confinement, where the overall colour charge is neutral [38]. There exist different types of bound states between quarks (hadrons), further classified by the number of quarks. The most simple combination with two quarks (with colour and corresponding anti-colour), is called meson, whereas three quarks form so-called baryons. With LHC data, the LHCb collaboration [39] has also found penta- and tetraquark states with five or four quarks, respectively [40, 41].

All force carriers (gauge bosons, or vector bosons) carry spin 1 and exchange charges related to their respective interactions. The EM interaction with infinite range is mediated by the massless photon (γ). Another type of massless bosons are the gluons (g) which mediate the strong interaction. There are eight independent types of gluons carrying colour charge, and they can self-interact. For the weak interaction, the Z boson ($Q = 0$, $m = 90.19$ GeV) and the $W^\pm$ bosons (same mass of $m = 80.38$ GeV but different charge, $Q = \pm 1$ e), are the mediators with typical range of 10×10^{-18} m. Photons, Z and W bosons are associated with the electroweak theory. The Higgs boson, the only scalar particle in the SM, emerges from the Higgs mechanism related to electroweak symmetry breaking (EWSB), but it is not a force carrier like the other bosons.

2.1.2 Mathematical Formulation

At the core of the mathematical description of the SM [1, 26], a finite and non-abelian gauge theory (Yang-Mills theory [42]), are the Lagrangian density

$$\mathcal{L}_{\text{SM}} = \mathcal{L}_\psi + \mathcal{L}_V + \mathcal{L}_{\text{Higgs}} + \mathcal{L}_{\text{Yukawa}}, \quad (2.1)$$

and the SM gauge group,

$$G = \text{SU}(3)_C \times \text{SU}(2)_L \times \text{U}(1)_Y. \quad (2.2)$$

The fields and interactions covered by the SM are summarized in the SM Lagrangian, namely fermions with the kinetic terms and interactions with gauge bosons ($\mathcal{L}_\psi$), kinetic terms and self-interaction of gauge bosons ($\mathcal{L}_V$), the potential and kinetic terms and self-interaction of the SM Higgs boson ($\mathcal{L}_{\text{Higgs}}$), and finally its interaction with fermions ($\mathcal{L}_{\text{Yukawa}}$). This representation can be gained from rearranging the respective Lagrangians associated with QCD, and the EW / Higgs sector. The SM Lagrangian is renormalizable and invariant under local gauge transformations associated with G . Each continuous and smooth symmetry for differentiable fields contained in the SM is related to a conserved quantity [29], as stated by Noether's theorem [43], or in the Ward-Takahashi identity [44, 45] for QFT. The SM Lagrangian is invariant under transformations of the Poincaré group related to SR, i.e. translations in Minkowski spacetime, Lorentz boosts and rotations, giving rise to more conserved quantities: four momentum (including energy, momentum) and angular momentum. The local gauge symmetry groups included in the SM gauge group and the associated conserved quantities are firstly

the special unitary group $SU(3)_C$ related to strong interactions acting on colour. Secondly, the group $SU(2)_L \times U(1)_Y$ gives rise to the EW interactions related to weak isospin I and electric charge Q (or a combination termed weak hypercharge Y). So far, there exists no consistent Grand Unified Theory [46] to unify the EW sector with QCD and explain all experimental observations [1, 26].

2.1.2.1 Quantum Chromodynamics

As introduced above, $SU(3)_C$ is associated with QCD, the gauge field theory describing quarks and gluons. The corresponding Lagrangian with the gauge covariant derivative already expanded ($\not{D} = (\partial_\mu - ig_s \mathcal{A}_\mu) \gamma^\mu$) is given by

$$\mathcal{L}_{\text{QCD}} = \sum_q \bar{\psi}_{q,a} (i\gamma^\mu \partial_\mu \delta_{ab} - g_s \gamma^\mu t_{ab}^C \mathcal{A}_\mu^C - m_q \delta_{ab}) \psi_{q,b} - \frac{1}{4} \sum_{A=1}^8 F_{\mu\nu}^A F_A^{\mu\nu}, \quad (2.3)$$

where the sum runs over the quark flavours q and the field strength tensor $F_{\mu\nu}^A$ can be expressed as

$$F_{\mu\nu}^A = \partial_\mu \mathcal{A}_\nu^A - \partial_\nu \mathcal{A}_\mu^A - g_s f_{ABC} \mathcal{A}_\mu^B \mathcal{A}_\nu^C. \quad (2.4)$$

The four-derivative is expressed through ∂_μ and γ^μ are the Dirac γ -matrices. The QCD coupling constant $\alpha_S = \frac{g_s^2}{4\pi}$ is encoded via g_s . In this description, the fundamental representation of the $SU(3)_C$ group is associated with quarks $\psi_{q,a}$, where each of them is a quark-field spinor for flavour q and colour a , with $a \in \{\text{red, green, blue}\}$, and mass m_q . Gluon fields are represented by $\mathcal{A}_\mu^A$, and the eight generators t_{ab}^C of the $SU(3)$ group are a concise notation for the 3×3 Gell-Mann matrices [24, 47], see also Section A.1. These generators obey the commutation relations

$$[t^A, t^B] = if_{ABC} t^C, \quad (2.5)$$

with the $SU(3)$ structure constants f_{ABC} .

The expression enclosed in the sum in Eq. (2.3) contains the quark propagators as well as the quark-gluon interactions ($\propto g_s$), whereas the field tensor contraction yields non-linear gluon self-interaction terms proportional to g_s (three-gluon vertex) or g_s^2 (four-gluon vertex). For the different vertices, there are different colour factors that can be obtained from the so-called Casimir operators related to the generators t of the fundamental representation, or the generators T for the adjoint representation of $SU(3)_C$. Relevant colour-algebra relations are given by

$$t_{ab}^A t_{bc}^A = C_F \delta_{ac} \quad , \quad f_{ACD} f_{BCD} = C_A \delta_{AB} \quad , \quad t_{ab}^A t_{ab}^B = T_R \delta_{AB}. \quad (2.6)$$

The factor corresponding to a quark emitting a gluon is $C_F = \frac{4}{3}$, while for the triple gluon vertex one obtains $C_A = 3$, so $C_A > C_F$. For a gluon splitting into $q\bar{q}$, the associated factor is $T_R = \frac{1}{2}$. This has an impact for fragmentation properties of the induced showers and later also for jets initiated by either (light) quarks or gluons.

From the self-interaction of the force carriers, it can be concluded that the coupling “constant”

actually depends on the energy scale or the transferred momentum Q^2 at which it is evaluated. Absorbing constant factors and a reference value $\alpha_S(\mu^2)$ to just focus on the energy dependence, the so-called running of the coupling strength reads

$$\alpha_S(Q^2) \propto \frac{1}{\ln\left(\frac{Q^2}{\mu^2}\right)}. \quad (2.7)$$

This has multiple consequences, firstly, at large energies, α_S vanishes, giving rise to “asymptotic freedom” where the strong interaction becomes small. The other case, at small momentum transfer, the strong interaction is governed by a large coupling constant and quarks can only exist in bound states (hadrons), not as free particles. This yields the already introduced effect of colour confinement pertaining to colour-neutrality [38]. Further phenomenological consequences are revisited in Section 2.2.3.

So far, only a small part of the SM Lagrangian has been described, contributions of one type of fermions (quarks) and one type of gauge bosons (gluons). With that, one can fill the contributions of the strong interaction for $\mathcal{L}_\psi + \mathcal{L}_V$, but the EW theory shall extend these $\mathcal{L}_\psi + \mathcal{L}_V$ terms with particles interacting electromagnetically or only weakly, and add two more terms $\mathcal{L}_{\text{Higgs}} + \mathcal{L}_{\text{Yukawa}}$ via the Higgs mechanism, as explained in the following section.

2.1.2.2 Electroweak Theory

The weak interaction and QED are described in their combination, the electroweak (EW) theory, related to the symmetry group $\text{SU}(3)_L \times \text{U}(1)_Y$. Before symmetry breaking, which will be described later, the focus is now on the Yang-Mills [42] part of the Lagrangian that describes the vector bosons and the chiral theory involving fermions. The electroweak unification as introduced by Glashow [34], Weinberg [25] and Salam [48] distinguishes between left- and right-handed components of a fermion field $\psi = \psi_L + \psi_R$. Indeed, such a description involving chirality arguments (with chirality projectors becoming helicity projectors in the limit of $E \gg m$), as embedded into the vector minus axial-vector (V-A) theory, explains parity-violating interactions as observed in nuclear β decays of polarized ^{60}Co nuclei [49]. Fermions are then further classified by additional quantum numbers, the weak isospin I (with its third component I^3) and weak hypercharge Y . These quantities are related to electric charge via the Nishijima–Nakano–Gell-Mann formula [50–52]:

$$Q = I^3 + \frac{Y}{2}. \quad (2.8)$$

Using the shorthand notation for generators expressed with matrices from Section A.1, the left- / right-handed components are

$$\psi_L = \frac{1 - \gamma^5}{2} \psi, \quad \psi_R = \frac{1 + \gamma^5}{2} \psi, \quad \left(\bar{\psi}_{L,R} = \bar{\psi} \frac{1 \pm \gamma^5}{2} \right). \quad (2.9)$$

Similar to the construction for QCD, one can now compose a Lagrangian involving propagators and interaction terms with gauge bosons for the EW sector, as expressed via

$$\mathcal{L}_{\text{symm.}}^{\text{EW}} = \mathcal{L}_{\text{Dirac}}^{\text{EW}} + \mathcal{L}_{\text{Gauge}}^{\text{EW}} = \bar{\psi}_L i\gamma^\mu D_\mu^L \psi_L + \bar{\psi}_R i\gamma^\mu D_\mu^R \psi_R - \frac{1}{4} \sum_{i=1}^3 F_{\mu\nu}^i F_i^{\mu\nu} - \frac{1}{4} B_{\mu\nu} B^{\mu\nu}. \quad (2.10)$$

In this notation, there appear four fields, three components W_μ^i for $\text{SU}(2)_L$ (with coupling constant g) and one component B_μ for $\text{U}(1)_Y$ (with coupling constant g'), which are used in the covariant derivatives

$$D_\mu^L = \partial_\mu + ig \sum_{i=1}^3 t_L^i W_\mu^i + i\frac{g'}{2} Y_L B_\mu, \quad D_\mu^R = \partial_\mu + i\frac{g'}{2} Y_R B_\mu \quad (2.11)$$

and in the field strength tensors

$$F_{\mu\nu}^i = \partial_\mu W_\nu^i - \partial_\nu W_\mu^i - g\epsilon_{ijk} W_\mu^j W_\nu^k, \quad B_{\mu\nu} = \partial_\mu B_\nu - \partial_\nu B_\mu, \quad (2.12)$$

with the totally antisymmetric Levi-Civita tensor ϵ_{ijk} . Presence of three or four gauge fields per term gives rise to trilinear and quartic couplings in the EW theory.

In analogy to the group generators for the symmetry groups of the strong interaction, for the two symmetry groups introduced in this section, matrices and commutation relations define the algebra. For $\text{SU}(2)$, there are the three generators t_L^i which can be obtained from the 2×2 Pauli matrices via $t^i = \frac{\sigma^i}{2}$ (cf. Section A.1). They obey the commutation relations expressed with the Lie bracket

$$[t_L^i, t_L^j] = i\epsilon_{ijk} t_L^k. \quad (2.13)$$

For the EM part associated with $\text{U}(1)_Y$, the group generators are $Y_{L,R}$. Thus, the coupling constant g' appears in conjunction with weak hypercharge Y , which is described in Eq. (2.8). Comparing the QED component and the one for weak interactions, it can be noticed how the order of operations matters in the $\text{SU}(2)_L$ case, a non-abelian group, while $\text{U}(1)_Y$ has abelian structure. The fermion fields, their chirality and quantum numbers are detailed in Table 2.2. All right-handed quarks (up-type and down-type) and leptons (charged ones only) are weak isospin singlets, with $I = I^3 = 0$, contractions involving the W_μ^i fields in the covariant derivative would evaluate to zero. The left-handed ones do participate in weak interactions, which explains the additional terms when building the respective interaction components in the Lagrangian. Besides the already mentioned neutrino oscillations that were not part of the SM in its original form and which point at non-vanishing neutrino mass, in the SM there are only left-handed, and consequently, massless neutrinos. The $\text{U}(1)_Y$ does apply to both left- and right-handed fields.

The gauge fields introduced above can be mixed or combined linearly to translate them into the physical fields. A common convention is to identify the W bosons as physical fields $W^\pm$ via

$$W^\pm = \frac{(W^1 \mp iW^2)}{\sqrt{2}}. \quad (2.14)$$

For physical photon and Z boson fields, the expression can be gained by demanding vanish-

Table 2.2: Fermions and their quantum numbers for electric charge Q , third component of weak isospin I^3 , and weak hypercharge Y . There are left-handed doublets (index L) of weak isospin I and right-handed singlet states (index R). For down-type quarks, the weak eigenstates with apostrophe ($'$) are used in the doublets, these d', s', b' from the Cabbibo mechanism are more thoroughly introduced after explaining the Higgs mechanism through which quarks acquire mass (and correspondingly also have mass eigenstates). See also Refs. [1, 26].

	Generations			Quantum numbers		
	1.	2.	3.	Q	I^3	Y
Quarks	$\begin{pmatrix} u \\ d' \end{pmatrix}_L$	$\begin{pmatrix} c \\ s' \end{pmatrix}_L$	$\begin{pmatrix} t \\ b' \end{pmatrix}_L$	$+\frac{2}{3}$	$+\frac{1}{2}$	$+\frac{1}{3}$
	u_R	c_R	t_R	$+\frac{2}{3}$	0	$+\frac{4}{3}$
	d_R	s_R	b_R	$-\frac{1}{3}$	0	$-\frac{2}{3}$
Leptons	$\begin{pmatrix} \nu_e \\ e \end{pmatrix}_L$	$\begin{pmatrix} \nu_\mu \\ \mu \end{pmatrix}_L$	$\begin{pmatrix} \nu_\tau \\ \tau \end{pmatrix}_L$	0	$+\frac{1}{2}$	-1
				-1	$-\frac{1}{2}$	-1
	e_R	μ_R	τ_R	-1	0	-2

ing EM coupling to neutrinos carrying no electric charge and fixing the Z boson in such a way that neutral weak currents add to an orthogonal matrix:

$$\begin{pmatrix} A_\mu \\ Z_\mu \end{pmatrix} = \begin{pmatrix} \cos(\theta_W) & \sin(\theta_W) \\ -\sin(\theta_W) & \cos(\theta_W) \end{pmatrix} \begin{pmatrix} B_\mu \\ W_\mu^3 \end{pmatrix}. \quad (2.15)$$

In this representation, θ_W is the weak mixing angle (also: Weinberg angle) [25, 26, 34] that can be expressed through the coupling constants

$$\sin(\theta_W) = \frac{g'}{\sqrt{g^2 + g'^2}}, \cos(\theta_W) = \frac{g}{\sqrt{g^2 + g'^2}} \Rightarrow \tan(\theta_W) = \frac{g'}{g} \quad (2.16)$$

Consistency with predictions from QED requires that $e = g \sin(\theta_W)$, such that the weak mixing angle relates positron charge and weak coupling constant [26]. The experimental measurements yield a world average of $\sin^2(\theta_W) = 0.23121 \pm 0.00004$ [1].

Hitherto, the gauge bosons are assumed as massless. For photons and gluons, the description is consistent and local gauge invariance can be achieved with vanishing masses. But for the physical W and Z bosons, experiments have shown that these have actually large masses not compatible with zero, and thus the local gauge invariance with the present Lagrangian can not be upheld for non-zero masses. Besides this contradiction with a missing mechanism to recover the gauge boson masses, another consequence of demanding gauge invariance is that mixing left- and right-handed fermion fields to construct mass terms is not allowed, however, from experiments it is known that fermion masses need to be generated in some way to explain the observations.

2.1.2.3 Electroweak Symmetry Breaking and the Higgs Sector

An ansatz can be made which introduces a complex scalar doublet field ϕ with

$$\phi = \begin{pmatrix} \phi^+ \\ \phi^0 \end{pmatrix} = \begin{pmatrix} \phi_1 + i\phi_2 \\ \phi_3 + i\phi_4 \end{pmatrix}, \quad (2.17)$$

with covariant derivative

$$D_\mu = \partial_\mu + ig \sum_{i=1}^3 t^i W_\mu^i + i \frac{g'}{2} Y B_\mu. \quad (2.18)$$

The Higgs potential V with $|\phi|^2 = \phi^\dagger \phi$ can be constructed as

$$V(\phi) = -\frac{\mu^2}{2} \phi^\dagger \phi + \frac{\lambda}{4} (\phi^\dagger \phi)^2, \quad (2.19)$$

which obeys gauge invariance by being symmetric under $SU(2)_L \times U(1)_Y$. In this picture, μ has mass-dimension one, with $\mu^2 < 0$, and $\lambda > 0$ is the dimensionless quartic coupling constant for self-interaction terms. With this realisation of the Higgs field, the pure ‘‘Higgs’’ part of the SM Lagrangian in Eq. (2.1) can be filled in as

$$\mathcal{L}_{\text{Higgs}} = (D_\mu \phi)^\dagger (D^\mu \phi) - V(\phi^\dagger \phi), \quad (2.20)$$

whereas the ‘‘Yukawa’’ part involving couplings to fermions shall be outlined in a later paragraph.

A consequence of the shape of the scalar potential is that its ground state is reached at a minimum value different from 0, in particular it is reached on all points of a circle around the origin with radius $\frac{\mu}{\sqrt{\lambda}} = |\phi|$. This could be achieved by an infinite number of combinations for the components of the complex doublet, as long as the quadratic equation is fulfilled, and without loss of generality it is possible to choose $\phi_3^2 = v^2 \equiv \frac{2\mu^2}{\lambda}$ and all other components identically zero: $\phi_1 \equiv \phi_2 \equiv \phi_4 = 0$. In this definition, v is referred to as the vacuum expectation value (VEV). With ϕ obtaining a ground-state value, spontaneous symmetry breaking occurs due to explicit choice of the vacuum state residing in one direction in the complex plane. For the ansatz above with the neutral scalar field ϕ^0 , this is parametrized by

$$\langle 0 | \phi | 0 \rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ v \end{pmatrix}, \quad (2.21)$$

and an expansion (or perturbation) around the ground state yields (in unitary gauge for simplicity, i.e. Goldstone bosons [53] do not appear in this representation):

$$\phi(x) = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ v + H(x) \end{pmatrix}. \quad (2.22)$$

The three other degrees of freedom from the original ansatz are now hidden, but they can be associated with the longitudinal components of the physical W and Z bosons.

With the expression for ϕ in Eq. (2.22), the Lagrangian can be expanded. First of all, H can be

identified with an additional boson – the Higgs boson with mass $m_H^2 = \mu^2 = \lambda \frac{v^2}{2}$, one of the consequences of the Brout-Englert-Higgs mechanism [54–56]. Up to a constant, the potential reads

$$V = \frac{\mu^2}{2} H^2 + \frac{\mu^2}{2v} H^3 + \frac{\mu^2}{8v^2} H^4. \quad (2.23)$$

Thus, the potential does not only contain a mass term, but also trilinear and quartic self-interaction terms, all proportional to the Higgs boson mass m_H .

Further, the kinetic terms can be expressed through the parametrization for ϕ and the covariant derivative D_μ from Eq. (2.18). The four-derivatives are first of all applied to the newly introduced Higgs boson:

$$(D_\mu \phi)^\dagger (D^\mu \phi) \supset \frac{1}{2} (\partial^\mu H) (\partial_\mu H). \quad (2.24)$$

Also other terms involving only one type of field can be collected and point at the already introduced gauge bosons which then obtain their mass through the Higgs mechanism:

$$(D_\mu \phi)^\dagger (D^\mu \phi) \supset \underbrace{g^2 \left| \frac{vt^+}{2} \right|^2}_{m_W^2} W_\mu^+ W^{-\mu} + \underbrace{\frac{g^2}{\cos^2(\theta_W)} |vt^3|^2}_{m_Z^2} Z_\mu Z^\mu. \quad (2.25)$$

with

$$m_W = m_{W^\pm} = g \frac{v}{2} \quad \text{and} \quad m_Z = \sqrt{g^2 + g'^2} \frac{v}{2} \Rightarrow \cos(\theta_W) = \frac{m_W}{m_Z}. \quad (2.26)$$

Using the above notation, the associated “mass” of the photon is $m_A = 0$, the physical photon corresponds to the A_μ field and the $U(1)_Y$ symmetry remains unbroken. Another way to phrase this is that physical boson fields propagate as eigenstates of the free Hamiltonian, but when just expanding the Lagrangian in terms of the non-physical bosons, there appear non-diagonal elements associated with weak mixing. This perfectly matches what has been derived for the EW theory.

Just like what was done for the mass terms of the W and Z bosons, also the $W/Z/H$ interactions in the Lagrangian after symmetry breaking can be written out. The interactions can be of the form of vector boson – vector boson – Higgs boson, or two vector bosons coupling to two Higgs bosons. All this is done after associating the physical gauge bosons with the three remaining longitudinal degrees of freedom that are not already absorbed by the Higgs boson. This is the last addition to the $(D_\mu \phi)^\dagger (D^\mu \phi)$ part of the Lagrangian:

$$gm_W W_\mu^+ W^{-\mu} H + gm_Z \frac{1}{\cos^2(\theta_W)} Z_\mu Z^\mu H + \frac{g^2}{4} W_\mu^+ W^{-\mu} H^2 + \frac{g^2}{8 \cos^2(\theta_W)} Z_\mu Z^\mu H^2. \quad (2.27)$$

What remains is to find a renormalizable theory for the $\mathcal{L}_{\text{Yukawa}}$ term to complete the Higgs sector of the SM Lagrangian and introduce masses for fermions. With the help of the Higgs doublet ϕ (Eq. (2.22)) and an ansatz treating left- and right-handed fermions differently (doublets

versus singlets) with a diagonalized mass matrix, the Yukawa term can be written as:

$$\mathcal{L}_{\text{Yukawa}} \supset -y_f \left[\bar{\psi}_L \phi \psi_R + (\bar{\psi}_L \phi \psi_R)^\dagger \right] \quad (2.28)$$

$$= -\frac{y_f v}{\sqrt{2}} \bar{\psi} \psi - \frac{y_f}{\sqrt{2}} \bar{\psi} \psi H \quad (2.29)$$

$$= -m_f \bar{\psi} \psi - \frac{m_f}{v} \bar{\psi} \psi H. \quad (2.30)$$

This description works for the lower components of $\text{SU}(2)_L$ doublets (charged leptons, down-type quarks), a similar construction can be made with the conjugate doublet ϕ_c to give mass to the upper component of the quark doublets[‡]. The first part refers to fermions interacting with the Higgs field (experimentally supported, *cf.* Refs. [57, 58]), while the second term describes the coupling of fermions to the Higgs boson (H) itself. With a VEV of $v = (\sqrt{2}G_F)^{-1/2} \approx 246 \text{ GeV}$ (G_F : Fermi coupling constant), the second contribution is small compared to the mass term [1]. Moreover, the Higgs boson coupling to fermions is proportional to their masses, while the coupling to gauge bosons is proportional to the square of the boson masses. Neither the values of the couplings, nor the parameters λ, μ in the potential are predicted by the SM, thus also the mass of the Higgs boson $m_H = \sqrt{2\lambda}v$ has to be determined experimentally: $m_H = 125.25 \text{ GeV} \pm 0.17 \text{ GeV}$, see [1] and references therein.

When deriving the Yukawa term in Eq. (2.28), the diagonalization process is facilitated by transforming weak eigenstates of quarks into their physical / mass representations, using a combination of unitary matrices $V_{L,R}^{u,d}$. The $W^\pm$ boson interaction then translates between different weak eigenstates, changing the flavour of fermions. By introducing the so-called Cabbibo–Kobayashi–Maskawa (CKM) matrix [59, 60]:

$$V_{\text{CKM}} \equiv V_L^u V_L^{d\dagger} = \begin{pmatrix} V_{ud} & V_{us} & V_{ub} \\ V_{cd} & V_{cs} & V_{cb} \\ V_{td} & V_{ts} & V_{tb} \end{pmatrix}, \quad (2.31)$$

the mass eigenstates for (d, s, b) quarks are related to the weak eigenstates (d', s', b') via

$$\begin{pmatrix} d' \\ s' \\ b' \end{pmatrix} = V_{\text{CKM}} \begin{pmatrix} d \\ s \\ b \end{pmatrix}, \quad \text{with} \quad (2.32)$$

$$|V_{\text{CKM}}| = \begin{pmatrix} 0.97435 \pm 0.00016 & 0.22500 \pm 0.00067 & 0.00369 \pm 0.00011 \\ 0.22486 \pm 0.00067 & 0.97349 \pm 0.00016 & 0.04182^{+0.00085}_{-0.00074} \\ 0.00857^{+0.00020}_{-0.00018} & 0.04110^{+0.00083}_{-0.00072} & 0.999118^{+0.000031}_{-0.000036} \end{pmatrix}, \quad (2.33)$$

where the latter experimental values are taken from a global SM fit to all available measurements [1]. The off-diagonal elements are small, but not zero.[§]

[‡]This formulation does not hold for neutrinos, as there are no right-handed neutrinos to build such mass terms. Observations pointing at non-zero neutrino masses therefore are not explained with this mechanism.

[§]For neutrinos, a similar mixing occurs, and the corresponding relation between weak and mass eigenstates is given by the 3×3 Pontecorvo–Maki–Nagakawa–Sakata (PMNS) matrix [61, 62]. Only in the approximate case with vanishing neutrino masses, the mass matrix is diagonal for charged leptons – mass and flavour eigenstates are the same. For the PMNS matrix, the best fit values for off-diagonal elements are much larger than for the CKM matrix, taking the latest confidence intervals into account. Thus, there is much more mixing between neutrino states than for quarks.

2.2 Protons and Proton-Proton Collisions

A theoretical overview of processes at hadron colliders as well as implications for simulation and the experiment are outlined below, and mainly follow Refs. [26,63].

2.2.1 The Proton

The elementary particles forming the proton are called partons. At valence quark level, a proton is composed of two up quarks and one down quark. Additionally, quantum fluctuations including gluon splitting give rise to sea quarks and gluons in the proton. Instead of merely taking the sum of its parts, the mass of the proton is largely based on the exchange of soft gluons between quarks, with gluons acting as the mediators of the strong force. The involved elementary particles carry colour charge that sums up to a net zero colour, making the proton one realisation of a colour-neutral composite particle, a hadron.

2.2.2 Scattering Processes and Terminology

To model the interactions between relativistic particles, for example, at hadron colliders, there are graphical representations to show possible contributions from different processes that can happen for given incoming and outgoing particles, denoted as Feynman diagrams. These diagrams are accompanied by rules which translate between the visual model and the calculation of scattering amplitudes. From those amplitudes, probabilities for a given process and their translation into cross sections can be obtained. In principle, all diagrams that are possible (not forbidden by conservation laws) need to be taken into account as individual contributions, but in practice, such amplitude calculations are performed to fixed orders in the respective coupling constant. The basic diagrams with which one can already obtain an estimate for the cross section, but with limited accuracy, are denoted as leading order (LO) contributions. Improvements regarding numerical stability and modeling of distributions for kinematic features can be achieved by introducing one or more closed loops of virtual particles (multi-loop), or by adding extra emissions (multi-leg). This separates the “tree-level” diagrams (no loops) from diagrams at next-to-leading order (NLO), or even higher orders like next-to-next-to-leading order (NNLO).

2.2.3 Proton-Proton Collisions

Because of the uncertainty principle, the momentum fraction of partons inside the proton is not known a priori. The probability density for a parton q to carry a momentum fraction x (the Bjorken scaling variable) of the proton’s momentum involved in a proton-proton (pp) collision can be expressed with the parton distribution functions (PDF) [26,64–67], abbreviated as $f_q(x, Q^2)$. Just looking at the normalization with respect to the Bjorken x , for the proton one obtains

$$\int_0^1 (f_u(x) - f_{\bar{u}}(x)) dx = 2 \quad , \quad \int_0^1 (f_d(x) - f_{\bar{d}}(x)) dx = 1 \quad (2.34)$$

$$\text{and} \quad \int_0^1 (f_q(x) - f_{\bar{q}}(x)) dx = 0 \quad \text{for} \quad q \in \{c, s, t, b\} , \quad (2.35)$$

which corresponds to two (valence) up quarks, one (valence) down quark, and otherwise sea quark contributions. These distributions have a second dependency, the momentum transfer Q^2 , at which they are evaluated. Here, $|Q|$ denotes the energy scale. A distinction can be made between high energy interactions (large Q^2 , hard processes) and interactions with small momentum transfer Q^2 (soft processes). While for the former, the coupling constant is small (quasi-free partons in the asymptotic limit) and perturbation theory offers suitable predictions, the latter processes are controlled by a large strong coupling constant yielding tightly bound states requiring phenomenological models. The scale at which the strong coupling constant α_S is evaluated (i.e. the running coupling, thus not really a constant in the literal sense), is denoted as renormalization scale μ_R . At typical energy scales at hadron colliders, parton-parton scattering offers a good, however not exhaustive description of the pp collision. Non-perturbative effects are absorbed into the PDFs, measured directly from experimental data obtained with and then used in global fits or combined with the help of neural networks, yielding specific PDF sets such as NNPDF [66, 68–70]. Among others, measurements from fixed-target experiments, HERA, Tevatron and LHC are taken into account in a recent version [70, 71]. Exemplary PDFs from NNPDF3.1 [66] (corresponding to the version used for CMS studies in this thesis, at NNLO [72]) are given in Fig. 2.1, for small Q^2 and larger Q^2 . While for small Q^2 , gluon and sea quark contributions are only relevant at small x , with

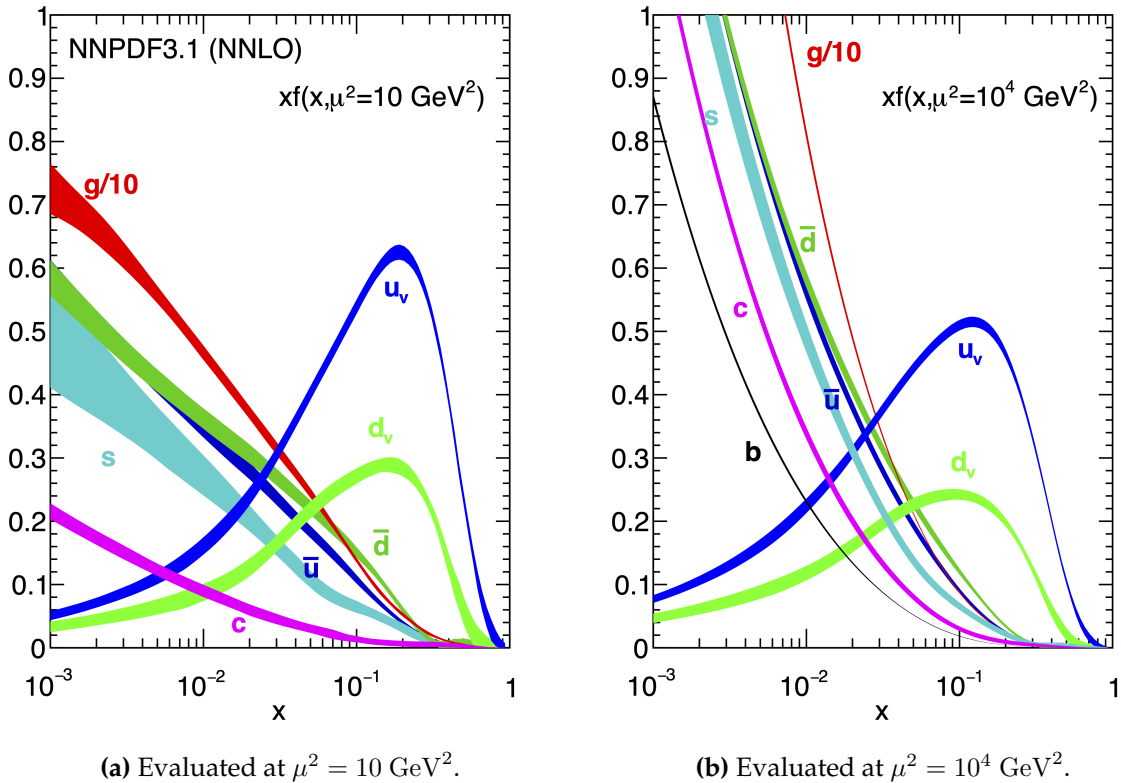


Figure 2.1: NNPDF3.1 NNLO PDFs at two different scales. All figures taken from Ref. [66].

valence quarks dominating at large momentum fractions, the situation changes for high Q^2 where gluons are the most important contribution up to $x \approx 0.2$. Usually, the PDFs assume the so-called five flavour scheme, where all quarks except for the top quark are massless. In the four flavour scheme, also the b quarks are assigned a mass. Via the Dokshitzer-Gribov-

Lipatov-Altarelli-Parisi (DGLAP) evolution equations [73–75], cross section predictions can be obtained by modeling the energy dependence of the PDFs. With the DGLAP equations, the PDFs do not have to be measured at each scale, but an extrapolation is possible starting from a specific Q^2 . The Altarelli-Parisi splitting functions [75] enter the DGLAP equations and denote the probability $P_{\tilde{p} \leftarrow p}$ (short form: $P_{\tilde{p}p}$) for a parton p to radiate a parton $\tilde{p}$. Thus, they describe the radiative patterns associated with $q \rightarrow qg$, $g \rightarrow qq$, and $g \rightarrow gg$ reactions. The parton interaction itself is described by fixed order matrix element (ME) calculations with perturbation theory. The factorization scale μ_F separates perturbative and non-perturbative dynamics used to describe the ME ($Q^2 > \mu_F$) and PDFs ($Q^2 < \mu_F$), respectively. Via the factorisation theorem, the total hadronic cross section σ_{pp} for the process $pp \rightarrow X$ is expressed through an integral involving the universal PDFs f_a, f_b and the partonic cross section $\hat{\sigma}_{ab}$ (summing over all types of partons a, b):

$$\sigma_{pp} = \sum_{a,b} \iint dx_1 dx_2 f_a(x_1, \mu_F^2) f_b(x_2, \mu_F^2) \hat{\sigma}_{ab}(\alpha_S(\mu_R^2), \mu_F^2). \quad (2.36)$$

A single outgoing parton has not been observed, instead, a parton shower (PS) where the partons undergo fragmentation controlled by the splitting functions sets in [76]. Incoming quarks and gluons can emit QCD radiation (initial state radiation from partons in the proton, ISR), but so can the outgoing partons (final state radiation from those partons of the hard scattering, FSR). In the soft (small-energy) and collinear (small-angle) limit, the Altarelli-Parisi splitting functions [75] are suitable to model the PS. As the PS evolves, particles lose energy during a cool-down process culminating in hadronization of the partons. The partons carrying colour charge are subject to colour confinement [38], requiring colour-neutrality. At energies below about 1 GeV, non-perturbative treatment with phenomenological description becomes necessary. There are models describing how the partons form stable hadrons, either directly, or via decay chains of unstable ones (*cf.* cluster model [77–81] or Lund string model [82–84]). In the cluster model, all gluons split into quark anti-quark pairs which are combined into colour neutral clusters. These intermediate objects decay into hadrons. In the Lund string model, massless strings are assumed between every quark anti-quark pair, which break up in the middle to split into pieces, as the potential energy in the string associated with the colour field increases while the original $q\bar{q}$ pair moves apart. Multiple hadrons and their respective decay products (not necessarily only further hadrons, but e.g. also leptons from weak decays, or radiated photons) leave a cone shaped trace and can be grouped into a collimated spray of particles, a so-called jet. This process is further described in the experimental methods part (see Sec. 4) as it involves jet clustering algorithms acting on the reconstructed particles in the detector, with which extrapolation back to properties of the partons that initiated the cascade in the first place becomes possible. At hadron colliders like the LHC, jets are abundant, as there are many quarks and gluons participating via ISR and FSR.

Besides the hard scattering process, there can also be contributions from additional interactions between multiple partons (MPI) or from beam-beam remnants (BBR), referred to as the underlying event (UE). With the available experimental methods, it is not possible to fully distinguish between UE, ISR and FSR [63]. A separate class of interactions mixed into the events is denoted as pileup (PU) and occurs as a result of accelerating and colliding the protons in bunches, thus contaminating the event. These PU collisions between several protons happen simultaneously with the collision of main interest. A sketch of such overlaid processes for pp

collisions is given in Fig. 2.2.

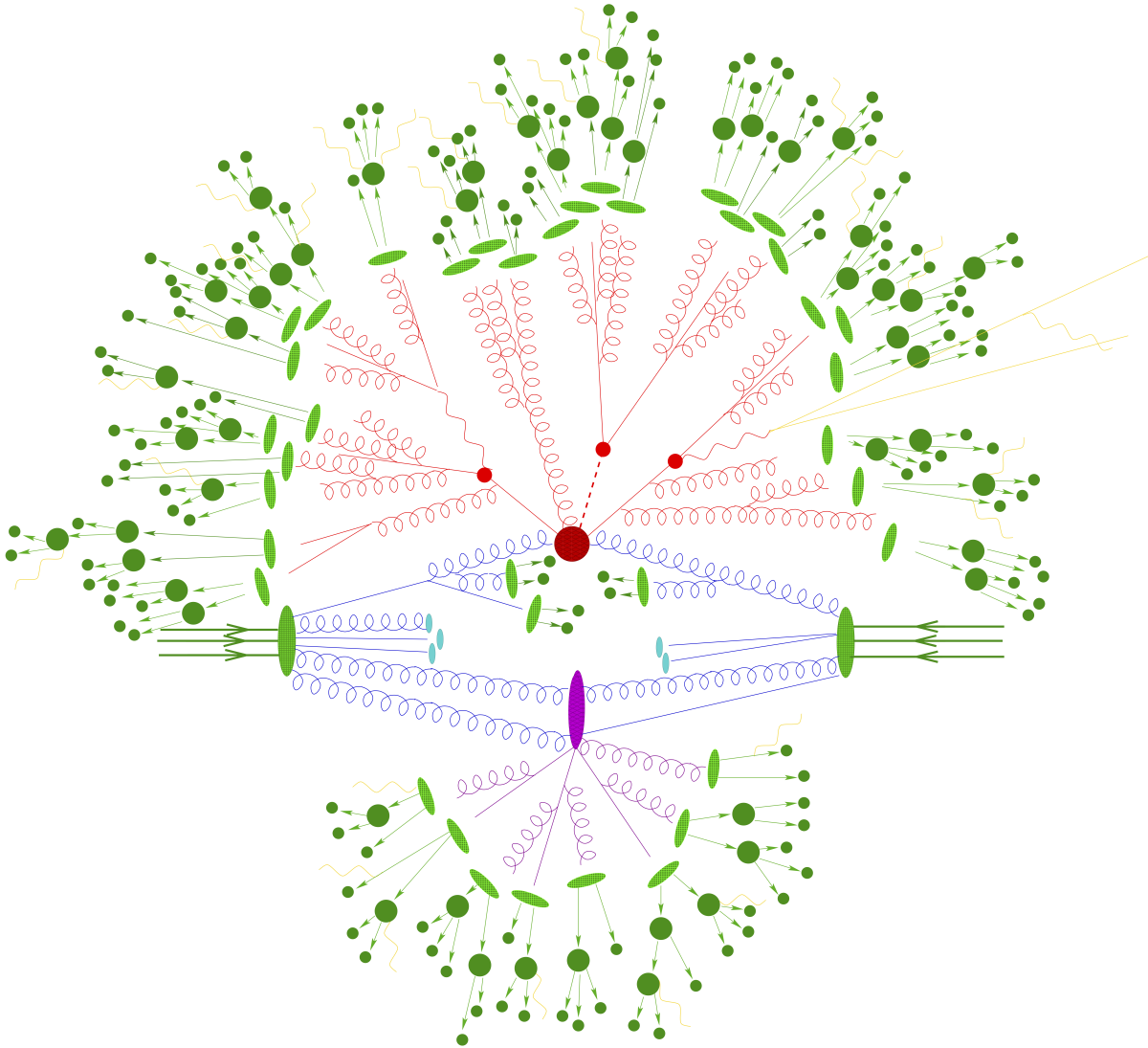


Figure 2.2: Graphical visualization of a hadron collision [76]. Incoming particles (i.e. protons), are shown with three individual lines merged into one green ellipse from the left and from the right. The hard process in the center is represented by a red "blob", the subsequent PS consists of further red components related to multiple splittings and emissions. Light green ellipses show hadrons emerging during hadronization, their decay is displayed with darker green ellipses. A secondary hard scattering event is overlaid in purple, remnants of the beam (strongly related to the initial state comprising quarks and gluons as described by the PDFs) are shown in blue, the UE is therefore shown both as MPI and BBR contributions [1, 63, 76]. Additional soft photon radiation is included as yellow lines [76].

2.2.4 Event Generation

Comparisons of recorded data with predictions from theory require simulated samples of collision events that model the hard process [64] and subsequent steps governed by phenomenology (PS, hadronization, decay) [76, 85] as well as the detector response and object reconstruction. This section describes the detector-independent part of those processes, while

experimental methods are introduced in a later step.

The hard parton level process at fixed-order with corresponding ME is obtained with tools like MADGRAPH5_aMC@NLO [86, 87] or POWHEG (Positive Weight Hardest Emission Generator) [88]. For many physics analyses performed in the LHC context, parton showering and hadronization is done with Monte Carlo (MC) event generators such as PYTHIA [89, 90], HERWIG [91, 92] or SHERPA [93]. Interfacing general purpose generators with the inputs from parton-level can be done using the Les Houches Event Files (LHEF) [94]. The PDFs are accessed via the Les Houches Accord PDF (LHAPDF) interface or with standalone tools [95–97]. To avoid double-counting from additional particles produced in the PS accurate in leading-logarithmic (LL) order and as higher-order ME contributions using multi-loop / multi-leg calculations, there exist PS matching and merging methods [98]. This is necessary, as the MC event generation uses the ME results as inputs. Examples are MLM [99], where an event is rejected if a PS jet is not consistent with one parton in the ME (thus an inefficient technique), or NLO merging via MCatNLO [87], where negative event weights are assigned to specific PS configurations such that taking the sum of weights ensures normalization, or POWHEG [88, 100, 101], where emissions exceeding a threshold are vetoed. For multi-jet merging, schemes like FxFx [87, 102] are used to combine several NLO orders through a combination of MCatNLO for different jet multiplicities and matching similar to MLM. MC generators as mentioned above are highly dependent on free parameters related to phenomenology, whereby correlations between groups of parameters need to be taken into account. For this purpose, so-called tunes are gathered with the help of comparisons to data as they can not be obtained from first principles [103]. For the CMS studies in this thesis, PYTHIA (which uses the Lund string model) in version 8 [90] with the CP5 tune [104] will be used, while PYTHIA 6 [89] is employed for the dataset in Chapter 6. In practice, both inclusive samples containing events from the full phase space, and subsamples for different decay modes / topologies, or for different object multiplicities, or subsamples that are sliced in bins of, e.g. jet p_T or H_T (scalar sum of transverse momenta of hard objects) are in use [63], which can be combined via stitching [105]. In special cases, where spin correlation or off-shell effects are relevant and the decay shall not be handled only by the fast, but inaccurate PS tools, the MadSpin module [106] is used to perform the decay before the shower (in this thesis, for some samples related to top quark production). Which tools are used for which sample is encoded in the names presented in Tables B.4 and B.5 in the appendix, together with the center of mass energy for which they are produced, thus 13 TeV and 13.6 TeV, respectively.

The event simulation does not end with the generator-level particles, but needs to be passed through a detector simulation as well, followed by object reconstruction similar to those steps done with real recorded data. To ensure consistency, PS generators save events for further processing by the experiments in the HEPMC format [107].

The CMS Experiment at the Large Hadron Collider

“The core advantage of data is that it tells you something about the world that you didn’t know before.”

– Hilary Mason

3.1 The Large Hadron Collider

Currently, the Large Hadron Collider (LHC) [5] at the European Organization for Nuclear Research (CERN) near Geneva is the world’s largest particle accelerator with a circumference of 26.7 km. The tunnel and parts of related infrastructure were repurposed from the Large Electron-Positron Collider (LEP). Further descriptions follow Refs. [5,6,108–110].

Hadrons are accelerated to almost the speed of light, and counter-rotating beams travelling in two vacuum beam pipes collide at different points hosting the individual experiments. The LHC can operate in different modes, depending on the injected hadrons: while the majority of so called runs are conducted with proton–proton (pp) beams, there are also dedicated heavy-ion runs where for example Pb-ions are brought to collision.

At CERN, the LHC is embedded into a large accelerator complex, as shown in Fig. 3.1. Several pre-accelerators precede the injection of beams to the LHC, each increasing the energy. In the current setup, the first accelerator for the pp operating mode is Linac 4, replacing Linac 2 which was used during LHC Run 1 (2009 – 2013) and Run 2 (2015 – 2018). The new linear accelerator operates since 2020 and accelerates negative hydrogen ions (a proton and two electrons) before injection into the Proton Synchrotron Booster (BOOSTER), where the two electrons are stripped off, leaving only the proton for further acceleration. For Pb-ions, the route starts in Linac 3, followed by the Low Energy Ion Ring (LEIR). After acceleration in BOOSTER or LEIR for protons or heavy-ions respectively, the hadrons reach the Proton Synchrotron (PS) and Super Proton Synchrotron (SPS) before injection into the LHC. Linear acceleration to 140 MeV is followed by further progressive acceleration to 1.4 GeV, 25 GeV, 450 GeV, and in the LHC the final beam energy of 6.5 TeV (Run 2) or 6.8 TeV (Run 3 after Long Shutdown (LS) 2, as of 2022 and 2023) is reached [110,112–115]. From the beam energy, the center-of-mass energy for the pp collision can be obtained by doubling the beam energy. As the proton is comprised of partons for which the exact energy is not known a priori, fractions of that energy are to be taken into account for the hard scattering process, as outlined in Sec. 2.2.3.

The journey of accelerated particles comprises several rounds through circular accelerators,

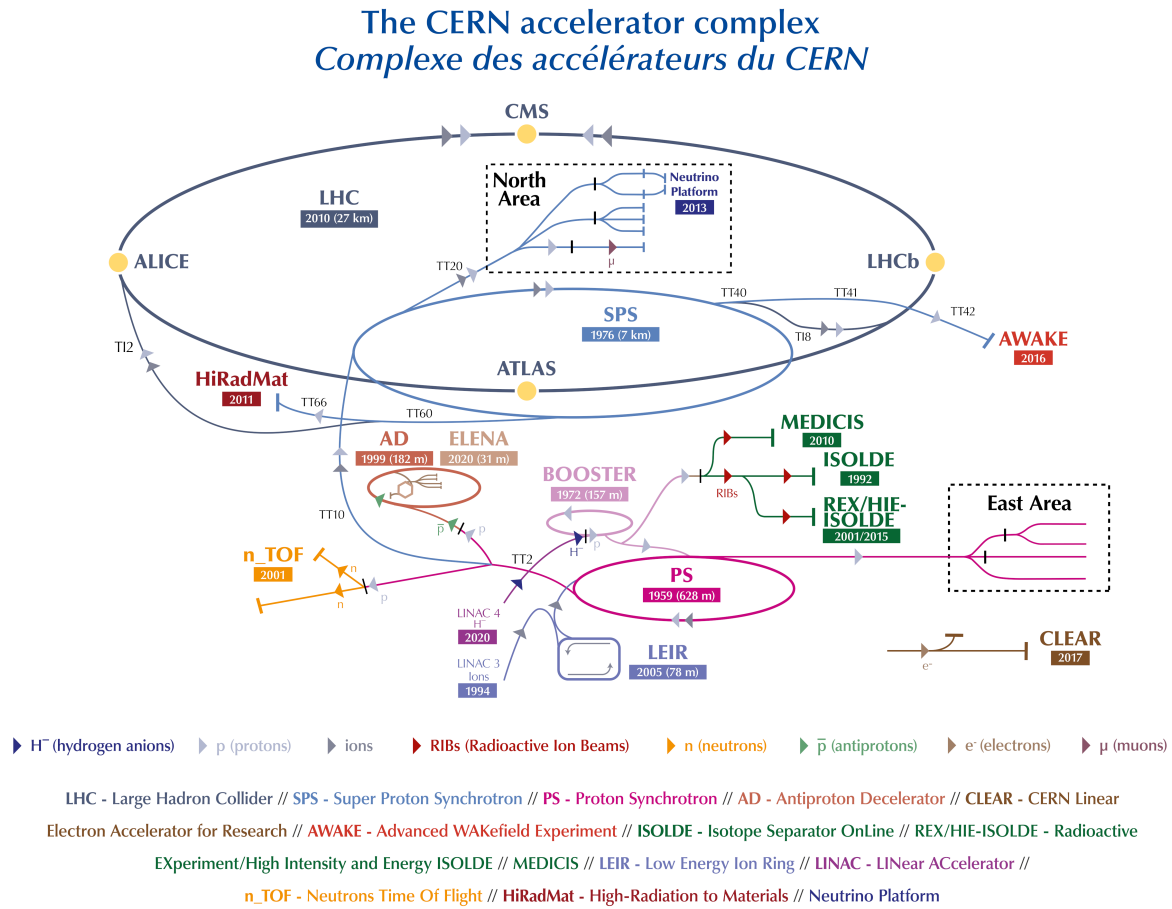


Figure 3.1: Sketch of the CERN accelerator complex with several experiments and accelerators, the launch year and individual circumferences, compiled at the end of LS 2 [111].

allowing them to gradually gain higher energies. As charged particles in circular motion emit energy via synchrotron radiation, inversely proportional to the fourth power of their mass, it is preferable to choose hadrons over leptons to keep the energy loss at acceptable levels [63].

The LHC tunnel is situated 50 m to 175 m below ground, related to the inclination of the geographical site [112]. For acceleration in longitudinal direction, superconducting, high-voltage radio-frequency cavities (with a frequency of 400 MHz) operating at 4.5 K yield a total of 16 MeV energy gain per proton per revolution. Cooled down to 1.9 K, 1232 superconducting NbTi dipole magnets are used to bend the particle beams in circular manner. With quadrupole and higher-order magnets, beam focusing / defocusing and further corrections to keep the beams on the desired trajectory are possible. Each particle beam consists of 2808 so-called bunches in nominal design setup [6], comprised of about 10^{11} protons each. These bunches are placed in so-called buckets, of which not all are filled, leaving thousands of empty “slots” (placeholders that could in principle contain bunches). Individual bunches are separated by 25 ns (bunch spacing), which translates to a rate of 40 MHz (termed bunch-crossing frequency).

3.1.1 LHC Experiments

At each of the four interaction points (IP) of the LHC, there is one main experiment, housed in an underground cavern. The experiments are targeted at different physics goals, thus also their components and geometrical setup vary. While the CMS detector is discussed in Section 3.2 in further detail, an overview of the LHC experiments follows. Several smaller experiments are located in the vicinity of the larger experiments, using the same IP for convenience. Such standalone experiments are listed together with the experiment at the closest interaction point, and are usually placed not more than a few hundred meters away from the main experimental cavern at the IP.

- **IP 1: ATLAS (A Toroidal LHC Apparatus)** [116] As the largest particle detector at an accelerator, ATLAS is one of the general purpose detectors at the LHC. The discovery of the Higgs boson in 2012 at the LHC by the ATLAS and CMS collaborations was a major achievement [117, 118], many properties of this elusive particle are measured with unprecedented precision [58]. Besides SM measurements, the portfolio also comprises analyses related to dark matter candidates and searches for beyond the SM (BSM) particles, and further aim at understanding dark energy. Also heavy-ion collisions (ion-ion or proton-ion) are part of the physics program.
 - **LHCf (Large Hadron Collider forward)** [119] Two sub-detectors, each placed 140 m down- and upstream of IP1, are used to study the number of neutral pions and photons at pseudorapidities close to infinity (zero degree) [120].
 - **FASER (ForWard Search Experiment)** [121] This experiment searches for light and weakly-interacting particles. In 2023, the first direct observation of collider neutrinos has been reported by FASER by detecting charged-current interactions [122].
 - **SND@LHC (The Scattering and Neutrino Detector at the LHC)** [123] Using a complementary pseudorapidity range compared to FASER, this experiment is targeted at neutrino interactions and recently observed collider muon neutrinos [124].
- **IP 2: ALICE (A Large Ion Collider Experiment)** [125] The ALICE detector is targeted at studies of the strongly interacting matter at highest energies to understand the quark-gluon plasma. Thus, the heavy-ion runs of LHC are of great importance for ALICE. With heavy-flavour jets, the dead-cone effect which restricts gluon emission in a cone around heavy quarks, was observed [126].
- **IP 5: CMS (Compact Muon Solenoid)** [7] The physics program of CMS is similar to the one of ATLAS, differences lie for example in a more compact layout with a stronger and different type of magnet. Its sub-detectors and other details are outlined in Section 3.2.
 - **TOTEM (TOTAl Elastic and diffractive cross section Measurement)** [127] Here, the focus is on the measurement of the total pp collision cross section [128].
 - **milliQan experiment** [129] This new detector aims at searches for so-called milli-charged particles, i.e. particles with charges much smaller than the elementary charge e . There is no way of overlooking the name similarity with the Millikan oil drop experiment that once measured the elementary electric charge [130].
- **IP 8: LHCb (Large Hadron Collider beauty)** [39] Evident from its name, the LHCb experiment focuses on b hadron physics. It employs a forward spectrometer to study lepton universality, CP violation and matter-antimatter antisymmetry with high preci-

sion. There is a rich search program targeted at exotic hadrons [40, 41].

- **MoEDAL (Monopole and Exotics Detector at the LHC)** [131] With the help of specialized plastic nuclear-track detectors and trapping techniques, this experiment searches for long-lived highly ionizing particles such as magnetic monopoles [132].

3.1.2 Luminosity and Pileup

A measure of the number of collisions per time at one of the interaction points is termed instantaneous luminosity L [1, 63] and is another key parameter of a particle collider which eventually controls the rate $\frac{dN_P}{dt}$ with which a physics process P with known cross section σ_P occurs:

$$\frac{dN_P}{dt} = \sigma_P \cdot L. \quad (3.1)$$

Integrating both sides of Eq. (3.1) over time t yields a number of (expected) interactions N_P in the $[0, T]$ interval, which is useful for statistical purposes. Just integrating L over time gives rise to the integrated luminosity L_{int} , which is often quoted in inverse femtobarn (fb^{-1}):

$$N_P = \sigma_P \cdot \underbrace{\int_0^T L(t) dt}_{L_{\text{int}}}. \quad (3.2)$$

For the LHC with its two counter-rotating, bunched beams, the instantaneous luminosity can be calculated via:

$$L = \frac{n_1 n_2 n_b f_{\text{rev}}}{4\pi \sigma_x \sigma_y} \mathcal{F}(\theta_c), \quad (3.3)$$

with $n_{1,2}$ the number of protons per bunch, f_{rev} the bunch revolution frequency, $\sigma_{x,y}$ the transverse beam sizes and $\mathcal{F}(\theta_c)$ the correction factor depending on the crossing angle θ_c [1, 110].

For IP1 and IP5, the design luminosity of LHC was $10^{34} \text{ cm}^{-2} \text{ s}^{-1}$, while for IP2 / IP8, it was aimed at $10^{33} \text{ cm}^{-2} \text{ s}^{-1}$. Already during Run 2, this design luminosity was well exceeded (see Fig. 3.2a). To obtain the luminosity scales as recorded by the experiment, beam-separation scans are performed [134]. Integrated over time (as visualized in Fig. 3.2b), all LHC runs with pp collisions until 2023 have delivered 266.42 fb^{-1} to CMS, of which 245.54 fb^{-1} were recorded [133]. Plateaus in the cumulative distribution correspond to technical stops or LS periods. With the total inelastic pp cross section $\sigma_{\text{inelastic}}$, the expected PU μ (collisions per bunch crossing) can be calculated via:

$$\mu = \frac{\sigma_{\text{inelastic}} L}{n_b f_{\text{rev}}}, \quad (3.4)$$

which varied over time, as can be seen in Fig. 3.2c. For example, during 2017, the average PU was $\langle \mu \rangle = 38$ with peaks of up to $\mu = 70$. With higher luminosities during Run 3 and larger inelastic pp cross section at higher center-of-mass energies, the PU consequently surpasses previously quoted values.

A single bunch crossing comes with multiple pp collisions yielding additional interactions that contaminate the event and for the detectors, this results in the difficult task of separating

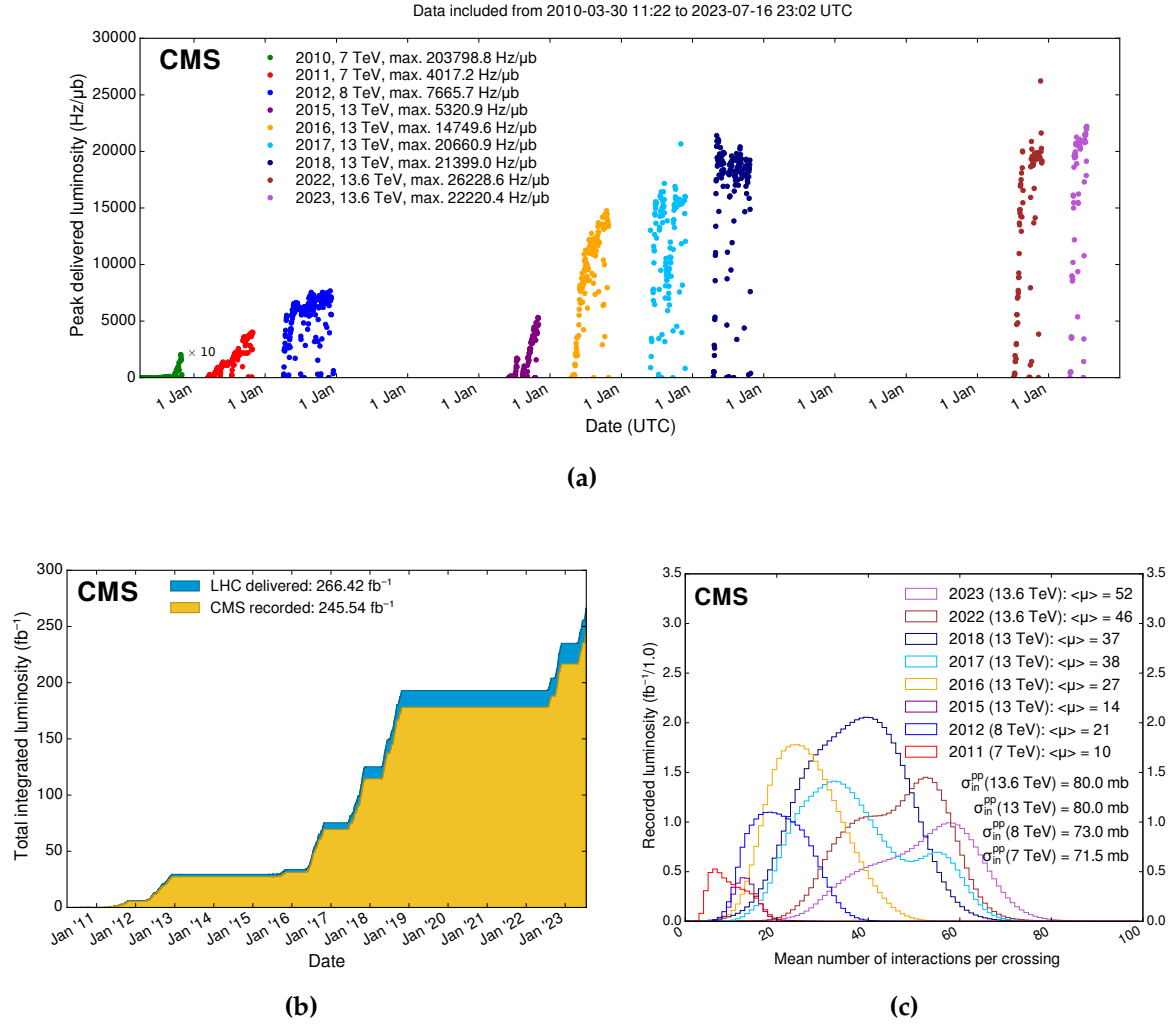


Figure 3.2: Peak delivered luminosity over time per fill (a), total integrated luminosity, cumulatively over time (b), average PU distribution by year (c). The included data are taken under stable beam conditions of the LHC machine, declared as ready for physics analysis. All figures taken from Ref. [133].

the collisions of interest from those that are overlaid. This effect is termed pileup (PU) and can take on the form of in-time PU (denoted μ) from additional collisions taking place in the bunch of interest, or out-of-time PU when the time resolution is not good enough to reject contamination from previous or later bunch crossings. The latter can happen if (sub-)detector systems have integration times larger than the bunch spacing. Sophisticated algorithms are necessary to mitigate PU effects, such that analyses can focus on the interesting hard scatter events [63].

3.1.3 Run 3 and HL-LHC Upgrades

During LS 2, many accelerator and detector upgrades took place, with crucial changes in hardware, software and computing. In 2022, marking the official start of Run 3, the LHC reached stable proton beams with a center-of-mass energy of $\sqrt{s} = 13.6 \text{ TeV}$ for the first time. In fall 2023, the first lead ion beams of Run 3 were accelerated to $\sqrt{s} = 5.36 \text{ TeV}$, however,

the majority of data taken for physics analyses is obtained in the proton–proton setup and the studies conducted in this thesis focus on pp physics.

After LHC Run 3 (2022 – 2025), followed by another upgrade phase, the High-Luminosity LHC (HL-LHC) [135, 136] is set to start in 2027. Research and development activities aim at integrating novel technologies such as crab cavities and powerful 11 T superconducting magnets. With further increase of instantaneous luminosity, anticipated to reach values ranging from $\approx 5 - 7.5 \times 10^{34} \text{ cm}^{-2} \text{ s}^{-1}$ [136], the whole accelerator and detector complex needs to be upgraded to withstand the increased radiation levels. Another consequence will be the large number of simultaneous pp collisions per bunch crossing, reaching average PU of $\langle \mu \rangle = 200$ [136]. As many physics analyses are already quite sensitive to PU, this requires further improvements in reconstruction chains with efficient PU mitigation techniques. The total integrated luminosity after HL-LHC is estimated to be 3000 fb^{-1} , and all those data, together with corresponding simulated samples in different data tiers need to be stored and processed, posing a challenge on its own [137].

3.2 The Compact Muon Solenoid

CMS [7, 138] is a multi-purpose detector at LHC IP5 measuring 22 m in length and 15 m in diameter, covering close to 4π of solid angle [139]. The cylindrically-shaped detector that comprises several sub-detectors in a layered design weighs about 14 000 t and resides 100 m below ground, close to the French village Cessy.

One of the essential components inside the detector is the solenoid that consists of superconducting NbTi coils that are cooled to 4.7 K. It provides a homogeneous magnetic field of 3.8 T inside its internal volume with 6 m in diameter and 12.5 m in length. Charge and momentum of charged particles can be obtained utilizing the curvature of particle tracks, as Lorentz force acts on charged particles in a magnetic field. As the solenoid itself could act as absorber material that modifies the deposited particle energy, it is placed outside all calorimeter sub-detectors. An iron yoke, which returns the magnetic flux, significantly contributes to the total weight with 12 500 t [7, 138, 139].

3.2.1 Coordinate System

Reconstructed objects are described with a right-handed coordinate system with its origin at the primary collision point [7, 63]. The z axis points along the counter-clockwise beam direction when viewed from above, the y axis points vertically upwards and the x axis points horizontally inward to the center of the LHC. The plane spanned by the x and y axes is termed transverse plane. Polar coordinates are defined by the azimuthal angle ϕ with $\phi \in [-\pi, \pi]$ and $\phi = 0$ pointing in positive x axis direction, and the radial coordinate r measuring distances from the primary interaction vertex [7]. For the polar angle $\theta \in [0, \pi]$, the positive z axis serves as reference. The so-called pseudorapidity η is related to θ via:

$$\eta = -\ln \left(\tan \left(\frac{\theta}{2} \right) \right) = \frac{1}{2} \ln \left(\frac{p + p_z}{p - p_z} \right) = \text{arctanh} \left(\frac{p_z}{p} \right), \quad (3.5)$$

where $p = |\vec{p}|$ is the particle momentum with its longitudinal component p_z in z -direction.

For massless particles, the pseudorapidity η is an approximation of the rapidity y , defined as

$$y = \frac{1}{2} \ln \left(\frac{E + p_z}{E - p_z} \right), \quad (3.6)$$

where E refers to the energy of the particle.

Differences in η are invariant under Lorentz boosts along the beam direction. In the relativistic limit where the particle mass is negligible, rapidity and pseudorapidity become equal, such that the statement regarding Lorentz invariance applies to y as well in this special case [1]. Angular separation is measured with (pseudo-)angular distance, defined as

$$\Delta R \equiv \sqrt{(\Delta y)^2 + (\Delta \phi)^2} \quad \text{or} \quad \Delta R \equiv \sqrt{(\Delta \eta)^2 + (\Delta \phi)^2}, \quad (3.7)$$

where the difference in azimuthal angle $\Delta \phi$ is always Lorentz invariant under a boost in longitudinal direction, rendering the first option invariant as well. This remark can be extended to the alternative definition using pseudorapidity differences in the case of massless particles.

Neglecting the crossing angle, the initial sum of transverse momenta $p_T = \sqrt{p_x^2 + p_y^2} = p \sin(\theta)$ of the colliding particles must be zero by means of momentum conservation, which is often used as a constraint for particle reconstruction. A related quantity is the transverse energy $E_T = \sqrt{m^2 + p_T^2}$ with the particle mass m . If the sum of the transverse components differs from zero, some particle(s) must have left the detector undetected. For the longitudinal component, partons carry fractions of the momentum, which are not known in advance, governed by the PDFs introduced in Sec. 2.2.3. Because of that, it is impossible to determine the longitudinal component of the particle(s) that escaped.

3.2.2 Detector Components

A brief overview over the different components, ordered from the one closest to the beam pipe radially outwards, is given below and visualized in Fig. 3.3. Selected upgrades for Run 3 are described as well.

3.2.2.1 Tracking Detector

The innermost part of CMS is the tracking system close to the beam pipe [142]. Being placed inside the magnetic field introduced above, trajectories of charged particles can be measured, which is also essential for vertex reconstruction. With its silicon pixels and strip modules (doped semiconductors), the tracking system offers high granularity and fast response for charged particles up to pseudorapidities of $|\eta| < 2.5$ (tracker acceptance as given by its 2.5 m diameter and 5.8 m length). Track information is utilized by several physics object reconstruction and identification steps for offline processing, but is also already an input for the High-Level Trigger (HLT) [7, 138, 141–143].

Until the end of 2016, the pixel detector consisted of three barrel pixel layers (BPIX) and two endcap disks on each forward / backward end (FPIX). With the so-called Phase 1 upgrade [141] in the beginning of 2017, a fourth BPIX layer was installed and the whole pixel detector moved closer to the beam pipe. The building blocks of BPIX and FPIX modules consist of 1184 and 672 silicon sensors respectively, each a size of $100 \times 150 \mu\text{m}^2$ and with 150×416

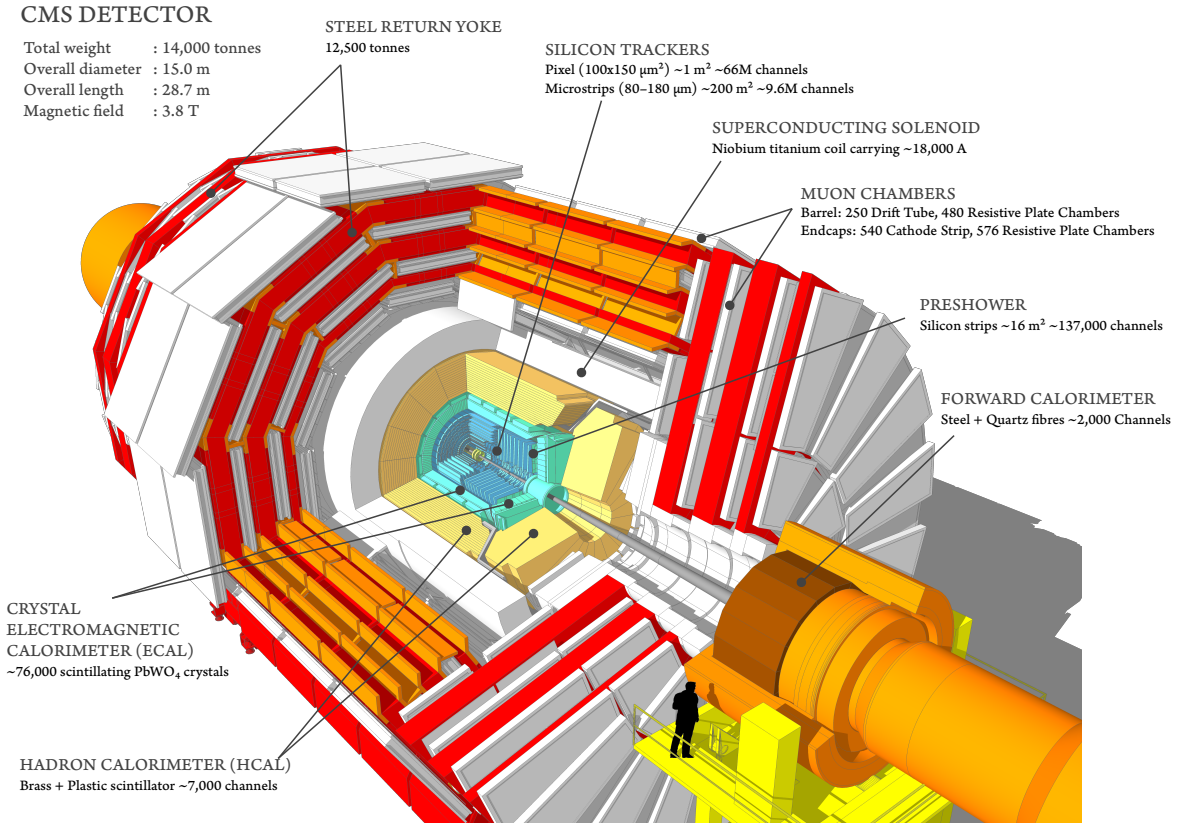


Figure 3.3: A 3D model of the CMS detector after the Phase-1 pixel upgrade 2017 – 2018 [140, 141]. Dimensions and installed sub-detectors have changed since the original configuration [7].

pixels, connected to 16 readout chips (ROCs). This Phase 1 pixel detector offers 124 million readout channels. Also during LS 2, detector components had to be replaced to ensure further operability, as especially the innermost parts need to work in an environment with high radiation doses and high hit rates [139, 144]. The pixel detectors are enclosed with multiple strip layers, which serve as the outer tracking system. Based on the location of the individual layers, they are further specified as tracker outer barrel (TOB), tracker inner barrel (TIB), supplemented by tracker inner disks (TID), or tracker endcap (TEC), comprised of multiple rings. In the barrel region, mainly the coordinates r, ϕ are measured, while in the endcap region, the strip modules aim at a dedicated r, z measurement. Some of the layers utilize so-called stereo modules (with secondary modules mounted back-to-back), such that the third coordinate not mentioned in the respective regions can be inferred as well. The 15148 strip modules offer 198 m^2 of active silicon area, with 9.3 million micro-strips [139, 144].

3.2.2.2 Calorimeters

The highly granular electromagnetic calorimeter (ECAL) is made of lead tungstate (PbWO_4) crystals and forms the inner calorimetry system [7, 138, 145, 146]. When charged particles (especially electrons) and photons interact with the homogeneous crystal material, an electromagnetic shower is created, during which the particles lose energy. Proportionally to the energy, the scintillator material subsequently emits scintillation light that can be detected

with photodetectors or avalanche diodes, such that the deposited energy can be inferred. The ECAL resolution is given by

$$\frac{\sigma_E}{E} = \frac{2.8\%}{\sqrt{E}} \oplus \frac{12.0\%}{E} \oplus 0.3\%, \quad (3.8)$$

where $\oplus$ refers to addition in quadrature and E must be given in units of GeV [7, 145–147]. For example, in the barrel region, the crystals have surface areas from $22 \times 22 \text{ mm}^2$ – $26 \times 26 \text{ mm}^2$ and extend for 230 mm in length.

Four sub-parts comprise the hadron calorimeter (HCAL): the hadron barrel (HB), hadron end-cap (HE), hadron forward (HF), and hadron outer (HO) calorimeters [7, 148, 149]. The HCAL is constructed as a sampling calorimeter which mainly uses brass as absorber material inducing hadronic showers (complemented with steel in some layers of the HB), and interleaved plastic scintillators to produce the signals. The solenoid itself acts as additional absorber material for the HO calorimeter. Segmented in the $\eta - \phi$ space, the resulting tower structures are recorded with hybrid photodetectors or lately with silicon photomultipliers [7, 139, 148, 149]. The hadronic energy component of physics objects can be measured with high granularity (ranging between 0.087×0.087 and 0.17×0.17), and with an energy resolution (in the barrel region) of

$$\frac{\sigma_E}{E} = \frac{85\%}{\sqrt{E}} \oplus 7\%, \quad (3.9)$$

where the notation is similar to Eq. (3.8) [150, 151].

3.2.2.3 Muon System

The muon system comprises different types of gas-ionization chambers which are placed in the outermost layer of the detector, interleaved by the steel flux-return yoke that reverses the magnetic field orientation [7, 152]. For muons with a mass about 200 larger than the one of electrons, the interaction with detector material in general and energy loss via bremsstrahlung in particular are much less. With the placement of the muon system behind all calorimeters and the solenoid, for particles different from muons and neutrinos there is only a small probability to penetrate the detector and reach the gaseous detectors. With the muon system, these particles are identified, their momenta measured, and signals for the trigger system are composed as so-called trigger primitives. For the barrel region, drift tube (DT) chambers offer a good spatial resolution ($100 \mu\text{m}$) at a time resolution of 2 ns. In the endcaps, cathode strip chambers (CSC) provide fast response (3 ns) and with precision of $50 - 140 \mu\text{m}$. Resistive plate chambers (RPC) especially support the trigger system with immediate (1.5 ns) measurement of the muon momentum with a resolution of $0.8 - 1.3 \text{ cm}$ for the barrel and endcap section [7, 152]. For Run 3, a new type was installed, the gas electron multiplier (GEM) chambers in front of the first CSC station, which offer fast response ($< 10 \text{ ns}$) and good spatial resolution ($100 \mu\text{m}$) [139].

3.2.3 Trigger System and Data Acquisition

At the LHC, bunch crossings happen at a rate of 40 MHz, as introduced in the accelerator description. Only a tiny fraction of these events contain interesting physics processes suitable for standard physics analyses, which fortunately can be extracted with sufficient efficiency without overstraining the storage system. This is possible, because the individual cross sections of these processes are in general orders of magnitude smaller than the total pp cross section, such that many events can be discarded [63]. For this purpose, a two-tiered trigger system is employed [7, 153, 154].

The first stage, the so called Level 1 (L1) trigger, is a specialized hardware trigger and reduces the event rate to 100 kHz. With the help of calorimeter information related to energy deposits and inputs from all types of muon chambers, a fast architecture on the hardware level with weak selection and sorting criteria is performed. A full event is built after this first level. While the event size during Run 1 was 1 MB, during Run 2 and 3 it has already doubled [139], posing new challenges for the data acquisition (DAQ) system at a throughput rate of 100 GB/s, which pulls information from about 700 detector backend boards [139].

After the L1 trigger, the software-based High-Level Trigger (HLT) reduces the event rate to approximately 1 kHz with the help of a fast reconstruction step, i.e. a simplified version of the full reconstruction chain [7, 154, 155]. Tracker information and object identification steps that can nowadays already apply dedicated HLT versions of machine learning algorithms [139] are used to save potentially interesting events for analysis. A step-by-step description of different criteria is summarized with dedicated HLT paths. At least one path needs to evaluate to True for the event to be subsequently transferred to the Tier 0 computing center at CERN [7, 139, 153, 154, 156]. Such paths and corresponding threshold criteria (examples: lepton isolation, transverse momenta) are always a tradeoff between efficiency and rate reduction, such that the appropriate choice is analysis specific.

Trigger rates can be further reduced for specific processes by applying so-called pre-scales which in general depend on luminosity and are therefore sometimes given as dynamic pre-scales [7, 63, 139, 154].

3.2.4 Detector Simulation

The output of MC generators is used in detector simulation, where the passage and interaction of the produced particles with the different subdetector elements are simulated [63]. Particles with long enough decay times then leave signals in the detector components. Each experiment thus has its own detector simulation, which needs to be adapted to varying run conditions. Detailed geometrical objects encode the detector in programs such as GEANT4 (GEometry ANd Tracking) [157], where interactions for different particle types are modeled, for example bremsstrahlung, ionisation, or scattering. Each subdetector, including sensitive material and readout electronics, infrastructure for cooling or other structures, is taken into account. Every particle type is transported differently, and electromagnetic or hadronic showers consistent with the electric and magnetic fields are modeled independently in the full simulation [158]. Approximations are used to reduce the required computing time and scale and smearing factors are employed in programs like DELPHES [159]. Certain parts of the detector and shower models are only parametrized, making DELPHES a suitable light-weight tool to

estimate detector effects that runs significantly faster than full simulation. Both the fast and full simulation have their purpose, and in this thesis, both variants are used for their respective applications in separate chapters.

PU contributions and the UE are added during the detector simulation stage as well, the later modeling for analysis use follows the distribution observed in data through a standard procedure called PU reweighting [63]. Energy deposits are recorded as so-called hits in the sensitive material. At the end of the detector simulation chain, digitization takes those hits and converts them into electronic signals [63]. To ensure comparability with detector data, simulation employs the same triggering and reconstruction algorithms.

3.2.5 Object Reconstruction

The different signatures left in the detector need to be assigned to particle hypotheses, and the process of physics object reconstruction is one of the essential ingredients implemented in the CMSSW framework [160] by the CMS collaboration.

A sophisticated three-step algorithm termed Particle-Flow (PF) [147] combines information from the tracking, calorimeter and muon systems.

1. First, hits from pixel and strip modules in the tracking system are combined to form tracks of charged particles. For this task, which is implemented with the combinatorial track finder (CTF) that is based on a Kalman filter, techniques such as pattern recognition and track fitting are applied [142, 161–163]. For Run 3, the mkFit algorithm is explored, which yields a speedup of the track reconstruction [164]. With the iterative tracking process, efficiency and low misidentification rates are optimized by prioritizing easily recognizable tracks first. For both the ECAL and HCAL separately, cells with energy maxima exceeding a threshold serve as seeds for so-called clusters. To obtain energy deposits in the case of close-by particles, a Gaussian mixture model serves as underlying fitting function for topological clusters. The output of this first step are PF elements.
2. Clusters and extrapolated tracks undergo a linking procedure to build PF blocks from PF elements, ensuring linked objects do not exceed a pre-defined geometrical distance in (η, ϕ) or are enclosed in an envelope around the involved cluster [147]. In algebraic terms, PF elements are linked transitively: if element A is linked to B, and B is linked to C, then A is also linked to C.
3. In sequential manner, physics objects are identified, starting with high-quality muons [147]. After subtraction of estimated energy deposits from these muons, electrons and isolated photons are identified, employing a combination of track and calorimeter information. While electrons are identified via tracks linked to ECAL clusters, the (neutral) isolated photons are extracted from the PF blocks as ECAL clusters without tracks. Neutral hadrons / non-isolated photons show a signature in the HCAL, but no track, and charged hadrons are identified from the remaining HCAL clusters with one or more tracks. To find the primary vertex (PV) corresponding to the collision of interest, all tracks consistent with the interaction point are grouped via deterministic annealing [165] and further processed by an adaptive vertex fitter [166]. Of all reconstructed vertex candidates along the beam axis, the one with the largest sum of squared transverse momenta (p_T^2) is taken as the PV [9, 147].

For the obtained physics objects from PF candidates, there are dedicated isolation criteria and identification (ID) techniques that come with tradeoffs between efficiency and purity, and specific choices for these parameters are denoted as working points (WPs). Different Physics Object Groups (POGs) develop dedicated performance measurement techniques such that a calibration for ID algorithms between simulation and detector data becomes available to the collaboration. Derived so-called scale factors along with related uncertainties need to be applied in physics analyses.

A non-exhaustive list of tools related to reconstruction and identification of physics objects is given below, while Chapter 4 contains more detailed information on jets and related topics.

- **Muons:** Muon tracks can be seeded from the inner tracker and propagated to the muon detectors (tracker muons), or vice versa (global muons) [147, 152, 167]. Combining both improves momentum resolution. The goal of the PF muon algorithm is to identify muons from the PV or from decays of heavy hadrons, using loose / medium / tight WPs. Additionally, there are isolation criteria depending on the absence of energy deposits in a cone around the muon candidate which are likely to stem from hadrons that punched through the solenoid [152, 167].
- **Electrons and photons:** Unlike muons which leave a characteristic signature obtained with the muon system, the reconstruction of electrons is more difficult, as they share some properties with several other kinds of particles. Electrons lose a significant fraction of their energy by emitting bremsstrahlung (photons), which needs to be recollected in the ECAL [63, 168, 169]. Additionally, continued photon emission with a change of electron energy and momentum can result in perceived disconnection or kinks in their track. This effect is mitigated by the gaussian sum filter (GSF) technique, where a combination of Gaussian distributions estimates the energy loss [168, 170]. A multivariant algorithm (MVA) utilizing a boosted decision tree (BDT)* is used to identify electrons, and takes track, cluster shape, and more information relating tracks and clusters into account [169, 171]. As for muons, different WPs are available, and the candidate information is complemented with isolation criteria. Distinguishing electrons from photon conversion into electron-positron pairs is facilitated by requirements on a maximum number of missing hits in the inner tracker and absence of conversion vertices [172].
- **Jets and Missing Transverse Energy:** See Chapter 4.
- **Tau leptons:** For hadronically decaying tau leptons, the hadrons-plus-strips (HPS) algorithm combines calorimeter information and the availability of one or three tracks to infer the decay mode [173–175]. The ML algorithm termed DeepTau [175] is used to distinguish hadronically decaying tau leptons from jets (initiated by partons that underwent hadronization) and from muons or electrons. Structurally, it shares some architecture components with the DeepJet [18] algorithm, which will be introduced in Chapter 4.

*A BDT can be understood as an ensemble of weak learners, each obtained from standard decision trees and combined sequentially to build a more powerful algorithm [63].

4

Jets

“I think you can have a ridiculously enormous and complex data set, but if you have the right tools and methodology then it’s not a problem.”

– Aaron Koblin

As explained in Sec. 2.2.3, quarks and gluons from pp collisions undergo parton showering followed by hadronization resulting in a collimated spray of particles, which, when grouped together, are referred to as a jet. The emergence of jets is not limited to nowadays’ hadron colliders, early studies [176–179] date back many decades.

While the exact energy fraction depends slightly on the flavour of the initiating parton / hadron, on average, the largest fraction of the total jet energy is carried by charged hadrons, followed by photons and neutral hadrons. Exemplary distributions which can answer the question “What is (in) a jet?” are given in Fig. 4.1. The reconstruction of such constituents into a complex physics object is done with jet clustering algorithms, which are described in the following.

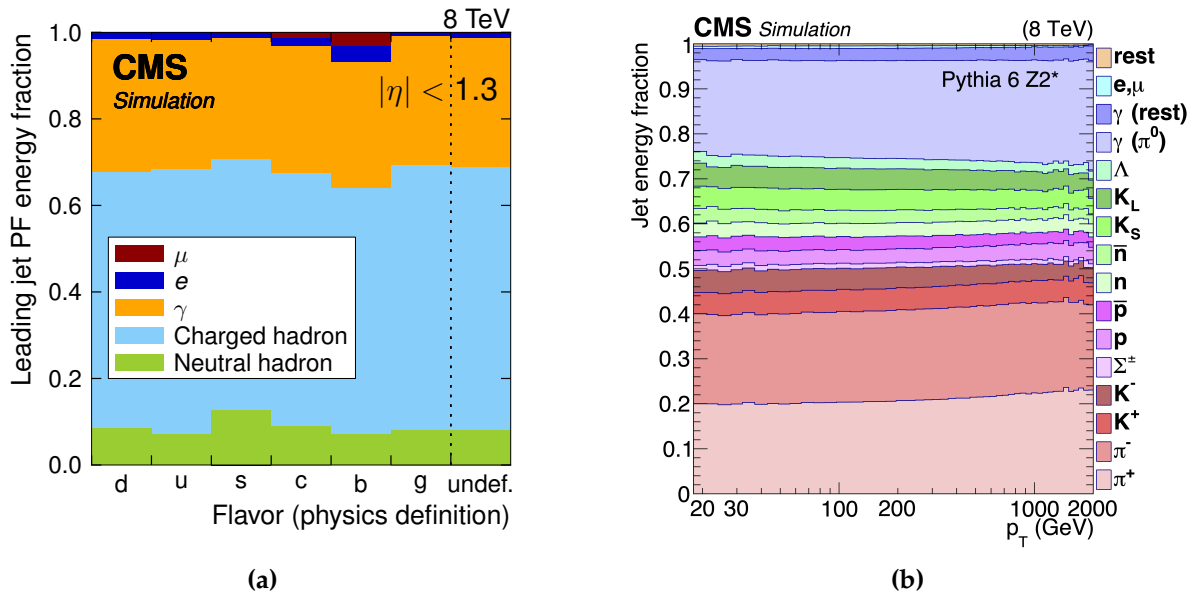


Figure 4.1: Jet energy fraction carried by different types of particles with simulated samples, split by flavour and grouping individual hadrons for Z+jet events (a), as a function of jet transverse momentum and at the particle level for a QCD dijet sample (b). The so-called physics definition of flavour that was used by the CMS collaboration during Run 1 will be introduced in Sec. 4.2, where the notion of jet flavour (tagging) is explained. All figures taken from Ref. [180].

4.1 Jet Reconstruction

At the detector level, energy deposits and hits in the tracking system or their combination via the Particle-Flow algorithm into PF objects (see Sec. 3.2.5) serve as a proxy to study jet constituents and their origin [26,63]. Commonly used techniques to cluster jets can be categorized into two classes, which are briefly described below.

At the Tevatron, so-called cone algorithms were used to cluster the available input objects [181, 182]. Starting with the highest energy object as a seed, an intermediate jet is constructed from all objects within a cone of fixed radius R around that seed. With these constituents, the jet axis is recalculated as a four-momenta sum over the chosen constituents. This process is repeated using that new axis as seed, until the newly obtained axis no longer changes. Constituents that were used in this process are removed from the overall list, and the cone algorithm continues from scratch with the remaining objects. All combinations are done in coordinate space. Besides the iterative cone algorithm with progressive removal, there are also other variants that do not remove already used objects but apply splitting and merging to deal with overlapping stable cones [182, 183], and an infrared-safe companion denoted SIScone [184, 185], or XCone [186].

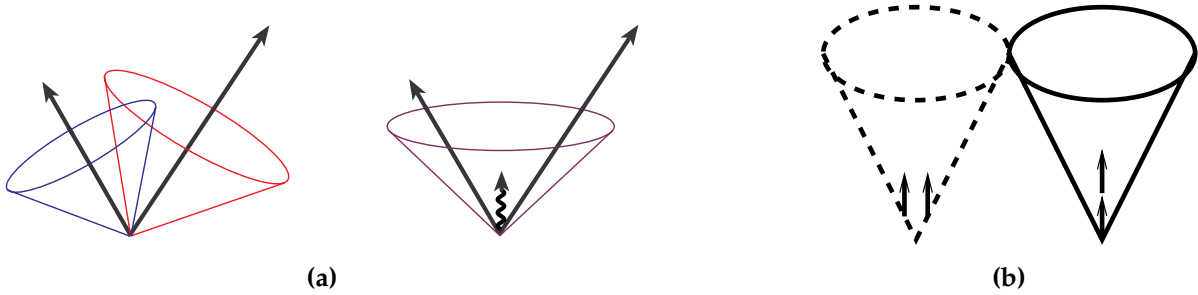


Figure 4.2: Schematic display of infrared- (a) and collinear-safety (b). Infrared-safety means that the results of a jet clustering algorithm should not change by the addition of a soft gluon (in this example, from two separate jets in blue and red into one violet large-radius jet). For collinear-safety, the requirement is that splitting one parton into two collinear partons should have no effect on the jet clustering. All figures taken from Ref. [182].

With the emergence of new clustering techniques obeying infrared- and collinear-safety criteria (see Fig. 4.2) and a computationally efficient implementation [187], especially with the FASTJET [188, 189] package, at the LHC, cone algorithms have been superseded by sequential recombination algorithms. Early adaptations of sequential clustering were used by experiments at the HERA electron-positron collider [190, 191].

At the LHC experiments, the most commonly used technique for jet clustering can be identified as one sub-type of the generalized k_T algorithm, which combines geometrical properties with distance measures related to transverse momentum (this description uses k in place of p , following the naming convention in Ref. [188]). In the following distance measure, $k_{T,i}$ refers to the transverse momentum of the i -th particle, y_i to its rapidity and ϕ_i to its azimuthal angle.

Two entities i, j are compared in iterative manner with distances defined as:

$$d_{ij} = \min \left(k_{T,i}^{2p}, k_{T,j}^{2p} \right) \frac{\Delta R_{ij}}{R} \quad \text{and} \quad d_{iB} = k_{T,i}^{2p}, \quad (4.1)$$

$$\text{where } \Delta R_{ij} = \sqrt{(y_i - y_j)^2 + (\phi_i - \phi_j)^2} \quad \text{and} \quad p = \begin{cases} 1, & \text{for } k_T; \\ 0, & \text{for Cambridge / Aachen;} \\ -1, & \text{for anti-}k_T. \end{cases} \quad (4.2)$$

In Eq. (4.1), three different options are considered, the k_T algorithm [192], the Cambridge / Aachen algorithm [193, 194], and the anti- k_T algorithm [188]. The jet radius parameter R needs to be defined in advance. In each clustering step, the distance between object i and the beam, d_{iB} , as well as the pairwise distance are computed and compared. If a pairwise distance d_{ij} is smaller than d_{iB} , those two objects i, j are combined by summing their four momenta. Then, the algorithm proceeds with the updated collection of physics-objects or pseudojets. If instead the minimum of the two distances corresponds to d_{iB} , this object i is called a jet and will not be used for the next iteration, by removing this clustered physics object from the overall list of entities. The algorithm terminates if no more entities are left for further clustering iterations [188].

For this thesis, the anti- k_T jet clustering algorithm with a jet cone size of $R = 0.4$ will be applied for all studies*. In CMS, such jets are conventionally referred to as AK4 jets, whereas another commonly used jet collection (“Fat Jets”, as in tools like NanoAOD [195] and PFNano [196] and described in Ref. [197]) is clustered using $R = 0.8$ (AK8), or $R = 1.5$ (AK15), containing substructure variables. There exist specialized applications, for example analyses described in Refs. [198–201], where such larger radii are suitable to identify physics objects with high efficiency in boosted / merged-jet topologies.

4.1.1 Pileup Mitigation and Jet Identification

Jet clustering can be contaminated with PF objects related to PU. The main PU mitigation technique for AK4 jets in Run 2 is charged hadron subtraction (CHS) [202, 203]. Entire objects are kept or removed, a representative weight can only take values 0 (remove) or 1 (keep) which in case of the jet flavour tagging algorithms presented later in this thesis enters as input variable. That way, neutral PU particles not tracked to a vertex are handled by an overall per-event correction, charged constituents associated with a PU vertex are excluded from jet clustering. An additional ID algorithm, the so-called pileup jet identification (PUJETID) [204], can be used to suppress jets by employing a BDT which takes the jet shape and its constituents such as tracks as inputs. The PUJETID algorithm offers different levels to apply loose or tight criteria.

For Run 3, the removal of PU contributions for AK4 jets is handled with the pileup-per-particle identification (PUPPI) algorithm [203, 205, 206]. For AK8 jets, this has been the default PU mitigation technique already during Run 2. Instead of purely subtracting particles arising from other vertices related to PU, neutral particles carry a weight between 0 and 1 that is used to encode the relative contribution of this object to the jet under investigation. Besides the local particle distribution, this algorithm also takes tracking information and further PU

*The convention introduced above is the default since Run 2, while during Run 1, $R = 0.5$ was used for small-radius (AK5) jets.

properties into account.

The PUPPI algorithm calculates an α_i value per particle i in the event [203,205,206], according to the formula

$$\alpha_i = \log \sum_{j \neq i, \Delta R_{ij} < R_0} \left(\frac{p_{T,j}}{\Delta R_{ij}} \right)^2, \quad (4.3)$$

with a pre-defined radius R_0 , the distance between particles i and j , ΔR_{ij} (j refers to charged particles from the PV, if $|\eta_i| < 2.5$, all reconstructed particles else), and the transverse momentum of particle j , $p_{T,j}$. Using the median ($\bar{\alpha}_{\text{PU}}$) and the root-mean-square value ($\alpha_{\text{PU}}^{\text{RMS}}$) of the charged PU α distribution, a signed χ^2 is calculated:

$$\text{signed } \chi^2 = \frac{(\alpha_i - \bar{\alpha}_{\text{PU}}) |\alpha_i - \bar{\alpha}_{\text{PU}}|}{(\alpha_{\text{PU}}^{\text{RMS}})^2} \quad (4.4)$$

which is then used to define a cumulative distribution function that serves as the individual particle's PUPPI weight w_i :

$$w_i = F_{\chi^2, \text{NDF}=1}(\text{signed } \chi^2). \quad (4.5)$$

Different so-called PUPPI tunes exist [206], for example the “v15” tune that improves the resolution for jets with high transverse momentum by a p_T - η -based weight protection for charged and neutral particles is used for Run 3 jet flavour tagging studies in this thesis.



In the jet reconstruction software and derived steps, the default small-radius jet collection in MiniAOD therefore changed from `slimmedJets` in Run 2 to `slimmedJetsPuppi` in Run 3. The author of this thesis adapted the flavour tagging software to incorporate this change and consistently use the newly weighted contributions in all steps from training to commissioning, in close collaboration with the JetMET Algorithms and Reconstruction (JMAR) group. As listed in Tables B.9 and B.10, these variables become input features for flavour tagging algorithms. Besides that (which in itself is not a problem for the neural networks, except that they need to be trained on these new samples for Run 3), the four-momenta of the affected jet constituents are scaled by their PUPPI weight. This can, in exceptional cases, lead to vanishing values or division by zero in subsequent steps, for example related to jet substructure variables. The author of this thesis applied and reviewed protective measures for tools such as secondary vertex producers, to ensure the RecoBTag package [160] becomes robust against PUPPI-induced abort signals in the reconstruction chain (where weighted PseudoJets enter the FASTJET package [189]). No more issues coming from this step in PromptReco or re-reco^a have been reported since integration, facilitating processing and enabling commissioning during the new data-taking period in Run 3 [160].

^aPromptReco is the first reconstruction that is mainly used for validation purposes, re-reco takes place afterwards and already includes a set of corrections and calibrations.

Another effect overlaid on top of PU contamination is misidentification of objects that are not actual jets (such as charged leptons), hadronic taus, or electronic noise. The rejection of such entities colloquially referred to as “fake jets” is done with the JETID algorithm [180,207]. With pseudorapidity-dependent criteria related to multiplicities of jet constituents and their energy

fractions in the ECAL and HCAL, different WPs are available to balance efficiency and purity.

4.1.2 Jet Energy Corrections

With multiple jet energy correction (JEC) steps, the jets are calibrated in a pre-defined order to better estimate the jet energy, i.e. to get closer to the “true” jet energy [180,208]. Such jets with corrections already applied should be used in physics analyses, while for flavour tagging, uncorrected jets are used as inputs to the different algorithms [9,180]. There are corrections related to the jet energy scale (JES) and the jet energy resolution (JER), because taking the sum of four-momenta of the constituents [147] is just an approximation [180,209,210]. The different steps are briefly described below, introducing multiple effects with variable impact dependent on kinematic properties of the jet.

First, using simulation of dijet events (i.e. multijet events with two jets) with and without overlaid PU contribution, the PU energy contamination is removed with corrections depending on jet shape, pseudorapidity and transverse momentum [211,212]. Furthermore, using recorded data and simulation of so-called Zero Bias events [180,208–210], additional corrections are provided.

In the second step, again based on simulated dijet events and provided as a function of jet η and p_T , the detector response is corrected. The transverse momentum of the jet is compared at reconstruction- and at particle-level, and the target is to achieve reconstructed quantities matching those at particle-level, with ratios close to 1. Besides the MC correction of calorimeter response, residual differences are measured in data with Z/γ + jet and multijet events.

Third, also the different responses to different jet flavours are considered in the JECs, using the closest (ΔR) generated parton as a proxy for jet flavour. Jets with differing flavour show variations in composition and the constituent’s energy distributions, as seen in Z/γ + jet simulation. Therefore, this correction is given as a function of jet η , p_T , and flavour.

A jet resolution smearing (via scale factors or stochastically) is applied to simulated jets, as the resolution in data (here obtained in dijet events) is worse than for simulated samples [180,209,210].

For Run 3, the steps are slightly altered, as the default PUPPI jet collection already reduces a significant amount of remnant PU contaminations, especially in the forward region. It was found that the first step (removing additional PU energy contributions in the JEC procedure, together with a residual correction) only has negligible effect and does not need to be used [213].

4.1.3 Missing Transverse Energy

The imbalance of transverse momenta measured by the detector is referred to as missing transverse momentum or MET [63,214–216]. As explained in Sec. 3.2.1, from momentum conservation and the initial state where partons (approximately) only have a longitudinal momentum component, for transverse momenta, a sum of zero is expected, if all particles were measured in the detector. For the longitudinal component, however, no such handle is available, because the initial value is unknown (cf. Sec. 3.2.1). From the SM particle content, neutrinos are the candidate particles that can escape the detector without leaving signals in the sub-detectors, as they only participate in the weak interaction. MET is then calculated after PF objects and

jets are reconstructed and identified, via

$$\vec{E}_T^{\text{miss}} = - \sum_i^{\text{particles}} \vec{p}_{T,i} \quad \text{and} \quad E_T^{\text{miss}} = |\vec{E}_T^{\text{miss}}|. \quad (4.6)$$

As MET depends on all measured particles, similar corrections as the ones for jets can be applied [208, 216]. The JECs of the previous section, which yield corrected jets (with $\vec{p}_{T,j}^{\text{corr}}$), can be re-purposed to also calibrate MET with the formula

$$\vec{E}_T^{\text{miss,corr}} = \vec{E}_T^{\text{miss}} - \sum_j^{\text{jets}} (\vec{p}_{T,j}^{\text{corr}} - \vec{p}_{T,j}). \quad (4.7)$$

4.2 Jet Flavour Tagging

The identification of jets to get a handle on the flavour of the originating parton is an essential tool for numerous analyses in the physics program of the LHC experiments [8, 9, 217, 218]. Historically, the identification of b jets (initiated by b hadrons containing b quarks) or b-tagging was the most prevalent task, jet heavy-flavour tagging thus mostly focussed on distinguishing b from other jets[†]. More recently, especially with the rise of machine learning-based algorithms, jet flavour tagging also incorporates charm tagging and the separation between light (up / down / strange) quarks and gluons. There are studies targeted at strange jet tagging, and a whole branch of boosted or heavy object identification techniques. The focus for this thesis are small-radius jets (AK4), which are frequently used whenever multiple jets can be resolved up to moderate values of transverse momentum / H_T . An overview of applications and different generations of algorithms is presented in Sec. 4.2.4. A more general picture of discriminating variables and a definition of truth labels is presented in the following.

4.2.1 Distinguishing Features of Heavy-Flavour Jets

With the help of several properties distinguishing heavy-flavour from light-flavour jets, it is possible to define input features for classification algorithms later on. The origin of these properties from a physics perspective is outlined below.

- The heavy b (c) hadrons have **long lifetimes** of about 1.5 ps (1 ps) in their rest frames, which corresponds to displacements of few mm to one cm before they decay, giving rise to a secondary vertex (SV) [1, 9]. The distance between the PV and the SV is termed “flight distance” and inherently has discriminating power, due to the correspondence between distance and lifetime via $\gamma c\tau$, with the speed of light c and the relativistic factor γ [1]. At the SV, charged-particle tracks emerge which are displaced with respect to the PV. The impact parameter (IP) vector for a given track points from the PV to the point of closest approach and is therefore a measure of the displacement. Its magnitude (impact parameter) has high discriminating power, and oftentimes, a variant, the signed IP (SIP)

[†]For the top quark (with a mass of $m_t \approx 173$ GeV close to the electroweak scale at $v \approx 246$ GeV, and a mean lifetime of roughly 5×10^{-25} s), it should be noted that it does not hadronize, but that it decays directly through the weak interaction, as evident from the CKM matrix mainly via $t \rightarrow Wb$ [1, 219]. The signature contains top quark-specific properties that are exploited in top-tagging techniques.

is used as input to classification algorithms. The sign is obtained from the scalar product between the IP vector and the jet axis, a positive scalar product corresponds to positive SIP values [9, 220]. The terms “downstream” (“upstream”) are sometimes used to refer to positive (negative) IP. The idea behind this is that if there was a hypothetical plane perpendicular to the jet axis and fixed at the PV, $SIP > 0$ would correspond to a point of closest approach in direction of the jet, while $SIP < 0$ would be “behind” that plane when taking the jet as a reference. Typical processes involve the weak decay of a b hadron into a W boson and a c hadron, which may then decay subsequently, resulting in a tertiary vertex inside the jet [1].

- Also the relatively **high masses** of b (c) quarks of about 4.2 GeV (1.3 GeV) and correspondingly the high masses of b and c hadrons (e.g. ≈ 5 GeV for b hadrons) are a useful property to identify heavy-flavour jets [1, 9]. When such heavy hadrons decay at the SV, the invariant mass of the reconstructed particles that emerge from the SV can be computed (“sv_mass”), directly correlated to the mass of the decaying particle [9, 17]. Another property is related to the **hard fragmentation function** for heavy-flavour jets, where the heavy hadron carries a large fraction of the total jet energy, and comparably large track transverse momentum relative to the jet axis [9]. Also the **constituent multiplicities** for heavy-flavour jets are higher on average than for light-flavoured jets [9].
- As a consequence of weak b (c) hadron decays, about 20 % (10 %) of b (c) jets contain **charged leptons**, namely electrons or muons [1, 9, 17]. The presence of soft leptons yields additional features for example in neural network training of combined taggers, but also facilitates subsequent calibration methods that exploit this property to define selective criteria for different phase space regions enriching heavy-flavour jets [9, 17].

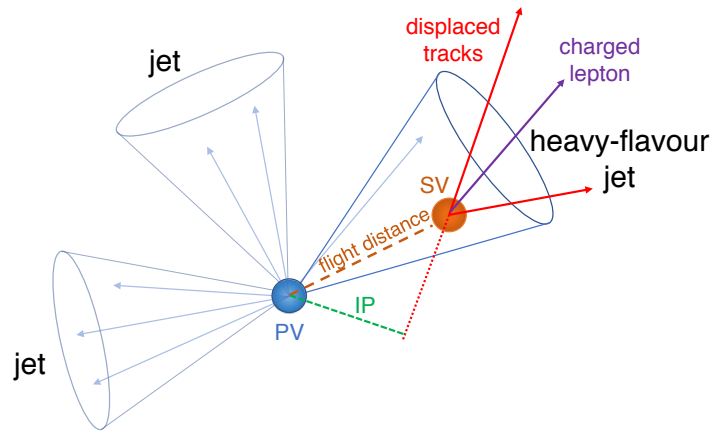


Figure 4.3: Schematic view of three jets that emerge from a common primary vertex (PV). Characteristic for the heavy-flavour jet (darker blue edges) is the presence of a secondary vertex (SV) from which multiple displaced tracks emerge, each with large impact parameter (IP) when extrapolated back to the origin. In this view, the shown displaced tracks have positive IP (signed IP = SIP > 0). [9, 220]. Figure taken from Ref. [9].

Many of the properties that are specific to bottom jets can be extended to charm jets as well. However, with the mass of the charm quark sitting in between the mass of u , d , s and b quarks (correspondingly for light / c / b hadrons and jets), the task of c jet identification is inherently more difficult, and so is the calibration [9, 17].

Different algorithms use information obtained from collections of charged or neutral particles, (secondary) vertices, or features describing the jet as a whole. The number and choice of variables depends strongly on the exact architecture in use, of which multiple generations are introduced below in Sec. 4.2.4. Inputs are listed with descriptions in Chapter. B.

4.2.2 Quarks versus Gluons

The different colour factors ($C_F = \frac{4}{3} < 3 = C_A$, see Sec. 2.1.2.1) result in higher particle / constituent multiplicities (higher emission rates) for gluon-initiated jets, but more uniform energy fragmentation, compared to light quark jets. In quark jets, fewer constituents carry a larger fraction of the total energy, and these jets are narrower than gluon jets. Correspondingly, typical features with discriminating power for quark–gluon tagging are constituent multiplicities or features describing energy and angular distributions [221–224].

4.2.3 Truth Flavour and Labelling

Every classification algorithm needs a proper definition of the categories into which a given object should be placed. For jets, such a definition can not be found in an unambiguous manner, as the detector does not “see” the initial partons from hard scattering, but only the evolved picture after hadronization and decay. There are conventions which tackle this problem using different viewpoints. Such labelling procedure is only possible for simulated jet samples, as the generator level information is necessary to assign a truth flavour as the target.

- During LHC **Run 1**, CMS employed a primarily parton-based jet flavour definition that came in two main sub-variants:
 - The so-called “physics” definition (as utilized in Fig. 4.1a) performs parton matching between the reconstructed jet and particles found in a cone around the jet axis. The radius of that parton-matching cone does not need to coincide with the jet clustering radius, and the choice will always be a tradeoff between efficiency and purity. A typical value, as utilized by CMS, is $\Delta R_{\text{matching}} = 0.3$. In consequence of this definition, the primary parton takes precedence, e.g. if a gluon splits into $b\bar{b}$, the jet will be labelled gluon jet.
 - A different view is delivered by the “algorithmic” definition, where instead of the original parton, the (heaviest) flavour, i.e. the one that most likely defines the jet’s properties, takes precedence. For this definition, if there is a b quark, no matter whether it originates from gluon splitting or not, this jet will be labelled as b jet.

A less common definition is termed “energetic”. Here, the jet flavour is obtained from energy fractions assigned to b or c hadrons, at first only using generator-level jets. While matching this to the reconstructed quantities (e.g. with calorimeter information), the flavour can be provided. Such a definition is possible (although not a prevalent choice), because jet clustering algorithms do not necessarily need to be applied to PF candidate objects, but can also be used with generator level inputs directly (without any detector simulation applied).

- Starting from **Run 2**, the main defining feature of jet flavour is obtained from the hadrons clustered inside the jet. This was necessary to avoid having to write specific jet clustering algorithms per utilized MC generator, as they may treat partons slightly differently (e.g.

decay before or during showering, different convention for particle status codes). More precisely, instead of hadrons in the usual sense, the MC truth for jets in CMS is based on “ghost” hadrons [9,211,225]. A re-clustering of jets is done using the same algorithm and parameters as during the original reconstruction, but with a process termed “ghost association” [9,225]. In this step, the four-momenta of the immediately decaying hadrons are rescaled, usually by the factor 10^{-18} [225]. These soft “ghosts” which have negligible effect on the final re-clustered jet are added to the list of jet constituents [9,63,225]. Then, a hierarchical definition based on the number of “ghost” hadrons of a certain type follows: a jet is labelled b jet if at least one b ghost hadron is found, c jet if there is at least one c ghost hadron (but no b ghost hadron), and light jet if neither b nor c ghost hadrons have been found. The mapping in terms of the currently used `hadronFlavour` branch in the reconstruction software is $(b, c, \text{light}) \rightarrow (5, 4, 0)$. By taking the actual number of b (c) hadrons into account, the heavy-flavour categories b / c can be further split (e.g. b into b, bb). In practice, a hybrid labelling that also uses “ghost” parton information allows further distinguishing the light flavour category by the type of the hardest parton (u , d , s or g). For modern machine learning based algorithms, fine-grained truth-level information is available and can be grouped conveniently to form more coarse labels. A recent set of possible labels, which also includes information on close leptons, is given in Table B.7.

4.2.4 Different Generations of Algorithms and Applications

Over the last years, many different algorithms for jet flavour tagging have been developed, with rising importance of machine learning based algorithms at the LHC experiments [2,4]. In this thesis, and in particular also in this brief review section, the focus lies on those algorithms for small-radius jets. While most of the descriptions are related to CMS algorithms, a few examples in the ATLAS context, which play a role in Chapter 6, are described as well.

Independent of the generation, training of the classifiers for this thesis is always done on uncorrected jets, i.e. JECs as presented in Sec. 4.1.2 are not applied until comparisons with detector data are made downstream when performing analyses [9,180]. Details on their implementation, training and validation follow in the sections dedicated to the respective generation.

Early approaches before the deep learning era were track-based and mainly relied on the SIP significance[‡]. With this feature, distinguishing displaced from non-displaced tracks is possible (for example when sorting tracks by SIP significance as in the Track Counting algorithm [226]), and thus, a proxy for compatibility of tracks with the PV or not, has been defined [9,226]. This in turn has separation power when discriminating heavy-flavour from light-flavour jets, as outlined above. To combine multiple absolute track IP significances in an overall tagger score, jet probability (JP) taggers compute a likelihood over the individual track probabilities [9,226]. In ATLAS, the JetProb [227] algorithm employed a similar technique, consistency of tracks with the primary interaction is given as p-value for the jet (i.e. b jets have a low p-value). For example, such algorithms were used in early top quark pair production cross section measurements with simplified parametrization of b-tagging efficiencies [228,229]. As

[‡]Significance refers to value divided by its uncertainty, rendering the significance a unitless quantity. This concept can be generalized also to SV information.

jet probability algorithms only rely on track-based quantities, they can be used in efficiency calibrations of more complex algorithms that use SV or soft lepton information [9].

Using SV-related information, further features with discriminating power become available, for example the flight distance significance. For SV finding, the inclusive vertex finder (IVF) is the default algorithm that provides the SV collection for subsequent use in the RecoBTag package [160], an alternative is called adaptive vertex finder (AVF) [9, 142]. Flight distance significance alone can be used to tag heavy-flavour jets (Simple Secondary Vertex (SSV) algorithms), or, using both SV- as well as track-based information, so-called Combined Secondary Vertex (CSV) algorithms become more powerful tools [9, 226]. The first CSV version already included numerous variables on top of SIP significance and flight distance significance, such as track multiplicities, energy ratios or vertex mass, and multiple likelihood ratios were used for the discriminant [226]. To better exploit correlations between input features, multivariate techniques such as boosted decision trees gained attention [9]. In the CSVv2 tagger, a sparse neural network was used (employing the feed-forward technique for a perceptron with one hidden layer). Besides the architecture, a reweighting technique to mitigate unwanted correlations with jet kinematic properties (pseudorapidity and transverse momentum) distinguishes this second version from its Run 1 counterpart [9]. A differential cross section measurement for top quark pair production used CSVv2 [230]. With soft lepton information (available for a small fraction of heavy-flavour jets and thus not suitable for standalone usage), another meta-tagger was created on top of the previously introduced JP and CSVv2 algorithms, denoted cMVA_{v2}. A BDT with gradient boosting was employed. For charm jet tagging, a dedicated BDT was trained, with similar input features and reweighting techniques as already introduced for CSVv2. However, for c jets, two trainings were necessary, one to distinguish c from b jets (CvsB), and a second one for c versus light-flavour jet (CvsL) separation [9].



The tagging algorithms presented so far paved the way for modern deep neural network architectures, which are the focus of robustness studies in this thesis. In current versions of the reconstruction software in CMS [160], only the DNN algorithms are supported. The author of this thesis took part in the phase-out campaign associated with Run 3, where the remaining algorithms of the “old” generation before the “Deep” era were removed from the list of maintained taggers while simultaneously introducing new versions.

With the Deep Combined Secondary Vertex (DeepCSV) algorithm, ML techniques are taken one step further with more hidden layers and more nodes per layer, in a fully-connected, deep neural network (DNN). The inputs are similar to CSVv2, but track features are given for up to six tracks in DeepCSV. Input preprocessing to obtain distributions centered at zero with root-mean-square of one is applied to speed up the training, missing values are replaced with zeros after preprocessing [9]. A variant of this algorithm, for a non-CMS application, will be studied in more detail in Chap. 6. The CMS-variant that was integrated during Run 2 is a multiclassification algorithm with four output nodes representing probability scores for the four classes b, bb, c and udsg, respectively, in its latest version running in the experiment software [160].

In physics analyses, DeepCSV was used for example to identify b jets produced in the $H \rightarrow b\bar{b}$ process (with branching ratio (BR) of 5.82×10^{-1}) as a tool to reconstruct the Higgs boson by requiring selected jets to pass certain DeepCSV WPs [231]. Combining the vector-boson

associated production mode with existing measurements, this led to the observation of the Higgs boson decay to bottom quarks [231]. Also, DeepCSV was chosen for the resolved-jet topology in the first direct search for the SM Higgs boson decaying to a pair of charm quarks (with a smaller BR of 2.89×10^{-2}) by the CMS collaboration [198]. For charm jet identification, a calibration in two dimensions (CvsL and CvsB) was necessary to integrate the discriminant distributions for the final signal extraction [17,198,199], which will also play a role in Chapter 7 when discussing data to simulation agreement. Also in supersymmetry (SUSY) searches such as the one presented in Ref. [232], DeepCSV is employed as b-tagging algorithm.

The full list of input features for the CSV, CSVv2, and DeepCSV algorithms can be found in Ref. [9], for DeepCSV also with a mapping to available keys in the so-called PFNano tool and with re-implementation details in Ref. [233].

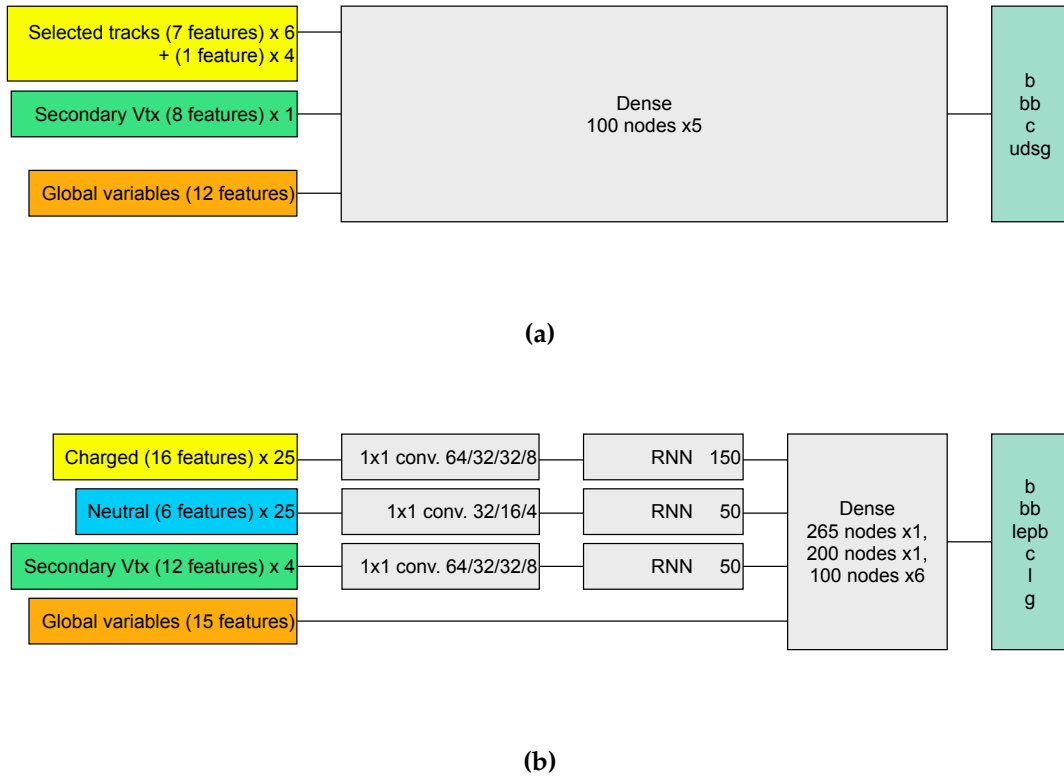


Figure 4.4: Architectures of the DeepCSV (a) and DeepJet (b) taggers. Figure (b) inspired by Ref. [18], but adapted to the version studied in Ref. [19].

The DeepJet algorithm includes, besides several global jet-related features similar to DeepCSV, a multitude of low-level input variables, directly related to up to 25 charged or neutral PF candidates, respectively, and takes up to four SVs with several features into account. With multiple convolutional NN layers for the jet constituents, followed by recurrent / LSTM layers and dense layers where the constituent information is merged with global jet information, DeepJet applies a significantly more complex feature extraction than DeepCSV. Additionally, the last layer has two more components, as DeepJet further splits some categories into separate nodes [18,19]. All inputs and all classes or, correspondingly, all truth labels are described in Sec. B.2.2. A comparison of the DeepCSV and DeepJet architectures is presented in Fig. 4.4 and Table 4.1.

With DeepJet, sensitivity improved for many analyses relying on jet flavour tagging. This includes for example SM searches and measurements [199, 234, 235], or searches targeting SUSY / BSM phase spaces [236, 237]. There are also analyses for which DeepCSV is used at the HLT, and DeepJet is used in the offline object selection [238].

That being said, whenever such identification algorithms are applied in physics analyses that compare expected with observed quantities, calibrations need to be applied [9, 17], which is not limited to flavour tagging. Such calibration is necessary to map the efficiency and mistag rates as seen in simulation to those observed in data, which are usually lower due to imperfect physics and detector modeling in simulation. The accompanying uncertainties related to the flavour-tagging efficiency measurements need to be propagated for the final reported results. This is a common property for the algorithms employed for jet flavour tagging, as shown also in the grey section of Table 4.1. While supervised learning only uses simulation in the training process, the inference is done with simulation and data, where differences are observed that are in turn mitigated with scale factors for calibration. For DeepJet, it has been found that larger scale factors are necessary, when comparing them with those for DeepCSV [17, 239, 240]. This correlates with the higher performance of the DeepJet tagger, compared to DeepCSV. A more detailed study of this property and possible ways to mitigate the effect are a key goal of the data/MC studies in Chapter 7. Recent updates of the performance studies with legacy data reveal differences regarding effects of SF application for DeepCSV and DeepJet when studying receiver operator characteristic (ROC) curves for performance [239, 240], an example for b-tagging is provided in Fig. 4.5. More details on obtaining ROC curves are added in the corresponding section describing DNN evaluation metrics, see Sec. 5.1.3.7. The graphic for DeepCSV shows that after application of the derived b-tagging SFs, the calibrated curve imitating performance in data touches the uncertainty band for the nominal ROC curve, for DeepJet, the calibrated curve is not covered by the uncertainties anymore. A similar conclusion can be drawn for the CvsB separation part shown in Ref. [240]. The more pronounced effect of discrepancies between simulation and data for a better performing model is one of the central aspects to be mitigated with methods developed and evaluated in this thesis, as will be detailed in Chapters 6 (with simulation only) and 7 (with CMS data).

Table 4.1: Comparison between the DeepCSV and DeepJet algorithms: differences and similarities. More details on the ML techniques will be given in Chapter 5.

DeepCSV	DeepJet
Only fully-connected (dense) layers	Convolutional layers, recurrent layers (LSTMs), dense layers
66 features , from up to six tracks, one secondary vertex, and high-level jet features	613 features , of which many are low-level features directly from up to 25 ParticleFlow candidates (charged & neutral) and four secondary vertices; high-level features from DeepCSV
Four outputs: b, bb, c, udsg	Six outputs: b, bb, lepb, c, uds, g
<i>Typical workflow:</i> train on simulation evaluate on simulation & data observe differences and correct by calibrating via scale factors	

More recently, complex graph neural networks (GNNs) such as ParticleNet (PNet) [241] and

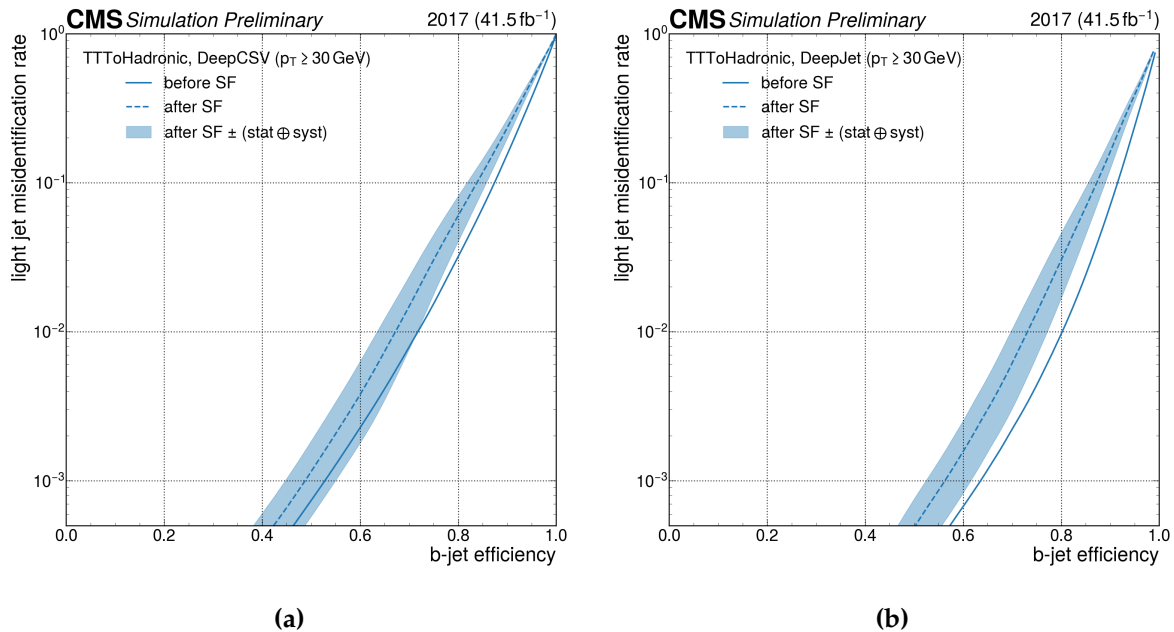


Figure 4.5: Performance of DeepCSV (a) and DeepJet (b) expressed via b-jet identification efficiency (x axis) versus light-jet misidentification rate (y axis), using rates in simulation and comparing them with results obtained with the help of data recorded in 2017. The solid line refers to simulation-only results (for hadronic top-quark pair production), the dashed line is obtained after SF application, surrounded by associated SF uncertainties. All figures taken from Ref. [239].

transformer architectures as utilized in Particle Transformer (ParT) [20] have been proposed for jet flavour tagging. While the former has mainly been used for large-radius jets in CMS [201], where the high number of inputs and their correlations can be exploited effectively with such a network[§], the latter was only recently integrated in CMS for Run 3, in a variant that is the subject of Chapter. 7. Correspondingly, details on the so-called “RobustParTAK4” tagger, which includes the author’s improvements on model robustness to enhance a first CMS AK4 implementation discussed in Ref. [21], are given in Sec. 7.3. In the literature, for this new generation of AK4 algorithms, ParticleNet and (Robust)ParticleTransformer, there are no performance results with detector data yet. A study comparing a nominal setup with a more robust one (see Sec. 7.3) also marks the first result of that kind for taggers superseding the DeepJet era in CMS, and the first performance result in data for a transformer network applied to AK4 jet tagging.

For a broader picture on jet flavour tagging at the multipurpose detectors at the LHC, a brief overview is given for ATLAS b-/c-tagging. There, the path from (IP- and vertex-based-) low-level to high-level taggers was similar to CMS, and included the MV2 (using a BDT) and DL1 (DNN) algorithms [10,218]. Novel architectures utilize recurrent NNs (RNNIP, used in DL1r), treat constituents as unordered sets (DIPS), or employ Graph NNs (GN1) with vertexing and track classification as auxiliary tasks [218,242–245]. Besides scale factors related to data-MC

[§]A new AK4 version was made available in parallel to the author’s contributions for the Jet and PUPPI Jet tables of NanoAOD in CMSSW [160]. A version without extensions regarding simultaneous transverse momentum regression will be used for comparisons with the novel developments that are covered in the main part of this thesis.

efficiency / mistag calibration, there are also MC-to-MC scale factors to correct for different hadronization schemes of different MC generators utilized in ATLAS [246].

5

Overview of Utilized Machine Learning Methods

“The key to artificial intelligence has always been the representation.”

– Jeff Hawkins

5.1 Machine Learning and Deep Neural Networks

5.1.1 General Aspects

The goal for many machine learning (ML) applications that are used in the *supervised* learning context is to learn an abstract representation from the given data, containing inputs and expected outputs [247, 248], to answer questions for new inputs such as

- What type of class does this sample belong to, given some characteristic features? (Classification)
- Which is the most likely realisation of a property for an object, given certain other properties? (Regression)

Neural networks (NNs) are one realisation of ML algorithms that map inputs to outputs, while internally approximating some function f between the two domains [247]. Jet flavour tagging is one application of classification algorithms that involve high dimensional input spaces, which are subsequently mapped to a low dimensional output space (for example, reducing the dimensionality from about 600 inputs to 6 outputs [18]). For this task, flavour acts as the ground-truth label or target for which the NN outputs are aiming for.

Besides these examples for supervised learning, there are also two other forms of ML that are mentioned for completeness, but will not be the focus of this thesis: *unsupervised* learning, which aims for structures in unlabeled data, or *reinforcement* learning, where a reward mechanism is used to condition the model in question to yield the desired outcome [247, 248].

The fundamental ingredients of NNs are the neurons (or nodes). Such neurons, inspired by neuroscience, can be connected with several other neurons, and are oftentimes arranged in multiple layers [247]. While the input layer (first layer) and the output layer (last layer) are essential to map the inputs from $\mathbb{R}^n$ to k outputs, the number of hidden (intermediate) layers between them (the depth) is configurable. Early adoptions of NNs used one hidden layer, but especially with the rise of multilayer perceptrons (MLPs), the term deep (feedforward) neural network (DNN) as a realisation of deep learning (DL) algorithms gained traction [247, 248]. This is also what happened for small-radius jet tagging in CMS, where for example the (sparse) CSV algorithm was superseded by DeepCSV [9], see also Section 4.2.4. With increased

depth of the model, there are more layers that each use the output of the previous layer, which distinguishes such DNNs from shallow NNs, the simplest example for the latter being a linear combination of inputs to yield the output layer. If all neurons are connected with all neurons of adjacent layers, one speaks of a fully-connected neural network (FCNN); DeepCSV is such a FCNN [9, 248].

5.1.2 DNN Predictions with Forward Passes

The forward process to translate inputs into outputs with a NN is composed of linear as well as non-linear operations. For the linear pre-activation within a layer l , a weight matrix $W_{l,l-1}$ is multiplied with the outputs of the previous layer $\vec{x}_{l-1}$. A bias term $\vec{b}_l$ is added [247]. In the matrix $W_{l,l-1}$, there can be entries for every neuron of layer l , and every neuron of layer $l-1$ (unless some connections are effectively zeroed out, like in DropConnect [249], a regularization method generalizing the Dropout [250] technique that randomly drops entire neurons during training). To perform matrix multiplication, the number of columns in $W_{l,l-1}$ must match the number of neurons in layer $l-1$, while the number of rows is the width of the l -th layer. The pre-activation vector $\vec{x}_l'$ can be computed via [247]

$$\vec{x}_l' = W_{l,l-1} \cdot \vec{x}_{l-1} + \vec{b}_l. \quad (5.1)$$

All steps so far are linear mappings, non-linearity is introduced with the help of a so-called activation function f_a . The output of a given layer l is then [247]

$$\vec{x}_l = f_a(\vec{x}_l') = f_a(W_{l,l-1} \cdot \vec{x}_{l-1} + \vec{b}_l). \quad (5.2)$$

In DNNs, multiple such layers are stacked and each layer may be equipped with a dedicated activation function. This eventually yields a complex function to calculate the overall network output, recursively obtained from the activations of the d individual non-input layers (d being the depth of the model) [247, 248]:

$$\vec{x}_d = f_a^{(d)} \left(W_{d,d-1} \cdot \left(\dots f_a^{(2)} \left(W_{2,1} \cdot \left(f_a^{(1)} \left(W_{1,0} \cdot \vec{x} + \vec{b}_1 \right) \right) + \vec{b}_2 \right) \dots \right) + \vec{b}_d \right), \quad (5.3)$$

where weight matrices $W_{l,l-1}$ as well as bias terms $\vec{b}_l$ are learned parameters of the network (sometimes combined into the notation $\vec{w} = (\vec{W}, \vec{b})$), activation functions $f_a^{(l)}$ are defined in advance; $\vec{x}$ are the original inputs and $\vec{x}_d$ are the final outputs of the network. Examples of activation functions operating on the linear pre-activation $z_i = (W \cdot \vec{x} + \vec{b})_i$ used for this thesis are:

- Rectified Linear Unit (ReLU) [247, 248, 251, 252], computed via $f_{\text{ReLU}}(z_i) = \max(0, z_i)$. Many dense hidden layers of the algorithms studied in this thesis use ReLU activation.
- Gaussian Error Linear Unit (GELU) [253], defined as $f_{\text{GELU}}(z_i) = z_i \cdot \Phi(z_i)$ with $\Phi(z_i)$ being the cumulative distribution function of the standard normal distribution. Introduced in the context of transformer networks, this activation is used for the RobustParT studies in Sec. 7.3.
- Softmax [254], for some output node i with original linear pre-activation z_i given as $f_{\text{Softmax}}(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$. This is often used to transform the final layer network logits into

predictions which are interpreted as the probabilities per class, $\hat{y}$, as this function is bound between 0 and 1 and all individual scores sum to 1 (normalization) [13,248].

Network initialization utilizes pseudo-random numbers for the model parameters, although these parameters can also be fixed with predefined seeds for the chosen ML framework (in the case of this thesis, in PyTorch [255]) to start with identical initialization [248] (for Chapter 6, this is done to mitigate any dependence on randomness and only compare different training strategies for the same model).

With the universal approximation theorem [247,256,257], many (Borel-measurable) functions can be approximated with NNs, given enough hidden layers and availability of non-linear structures with suitable activation functions. Examples for such functions where this theorem holds are continuous functions defined on a closed and bounded subset of $\mathbb{R}^n$ [247]. For applications such as HEP, this is a useful property, as without non-linearity, any composition of linear functions would only be able to approximate linear processes or linear mappings [247,248].

5.1.3 DNN Training and Performance Evaluation

During training, the network is updated frequently. Usually, the dataset for training is split into smaller parts, the batches of batch size N , and with each such batch, the updates are performed. How exactly the parameters are modified and which figure of merit is used during this stage is subject of the following section.

5.1.3.1 Loss Function

While the previous paragraphs show how inputs are processed in forward direction, the crucial aspect of (D)NNs is that during training, the parameters can be modified to yield better results. For this, first of all one needs to define a suitable metric to measure how well the network is performing. In practice, so-called loss functions are used as objective and the typical strategy involves minimizing this figure of merit with updates of the model parameters during the training process. This criterion can be interpreted as an error rate when using network predictions $\hat{y}$ to approximate the known labels y . Loss functions are one way to measure discrepancies between two probability distributions, those from the network (predictions), and those related to the known, true labels of the samples in question [248]. For classification problems, which are subject of this thesis, cross entropy loss is a popular choice, either in binary cross entropy (BCE) form [13,248] (if there are only two classes, for example, when differentiating a signal ($y = 1$) from a background process in HEP ($y = 0$) [3]), or in a multi-class variant that is explained thereafter and denoted categorical cross entropy (CCE) [247,258].

Given a signal probability x and label y , BCE is defined as [247,248]

$$L_{\text{BCE}}(x, y) = -(y \log(x) + (1 - y) \log(1 - x)) = \begin{cases} -\log(x), & \text{for } y = 1; \\ -\log(1 - x), & \text{for } y = 0. \end{cases} \quad (5.4)$$

Similarly, with more than two possible classes and performing operations batch-wise (with

output and target vectors, or Tensors as in PyTorch), CCE loss can be computed via [247, 258]

$$L_{\text{CCE}}(\vec{x}, \vec{y}) = -\frac{1}{N} \sum_{i=1}^N \sum_{j=1}^M y_{j,i} \log(\hat{y}_{j,i}), \quad (5.5)$$

where M is the number of classes, N denotes the batch size, $y_{j,i}$ is the truth label (defined as 1 if the i -th sample is part of class j , 0 otherwise) and where $\hat{y}_{j,i} = y_j(x_i|W, b)$ is the predicted value of class j for the i -th sample, given the weights W and bias terms b associated with the model.

Within PyTorch, the implementation directly works with the Softmax activation of the last layer, x is the prediction tensor for M possible target classes for an individual sample (see [259]):

$$\text{loss}(x, \text{class}) = -\log\left(\frac{e^{x[\text{class}]}}{\sum_{j=0}^{M-1} e^{x[j]}}\right) = -x[\text{class}] + \log\left(\sum_{j=0}^{M-1} e^{x[j]}\right). \quad (5.6)$$

Another way to interpret this definition is that a negative log likelihood is combined with a softmax function, where, as for other applications of likelihood functions that shall be minimized, the logarithm improves numerical stability. Here, the probability associated with the correct class is termed likelihood [248].

If the likelihood is low (bad prediction), the loss is high, for high likelihoods (good prediction), the loss is small [248]. As already introduced in Eq. (5.5), the loss function is computed in batch-wise manner, the division by batch size $\text{length}(\text{batch}) = N$ guarantees suitable normalization when averaging over each batch during dimensionality reduction, which is implemented as [259]:

$$\text{loss}(\text{batch}) = \frac{1}{N} \sum_{i=0}^{N-1} \text{loss}(\text{batch_predictions}[i], \text{batch_classes}[i]). \quad (5.7)$$

There are more variants of CCE that will be introduced for specific algorithms in this thesis, while for many algorithms under investigation, CCE loss [248, 259] is sufficient. On the one hand the weights of individual samples (jets) can be modified based on physics criteria to ensure learning from kinematic distributions associated with different flavours in unbiased way (loss reweighting with static weights $\rightarrow$ Chapter 6, or keeping the loss fixed as CCE and performing resampling instead $\rightarrow$ Chapter 7). The second advanced modification of CCE is focal loss (FL) [11, 260–262], which is a generalized variant of CCE that dynamically updates each sample's importance based on how well it is already classified by the algorithm. This technique is used in Chapter 6, supplementary studies have been conducted and documented in Sections C.2 and F.2.

In multi-class focal loss (with an additional static weight for the i -th sample in a batch (w_i) from pre-processing the training set as introduced above), there is an additional term to down-weight easy-to-classify samples during training. This puts more focus on samples for which the algorithm is not yet doing as well, hence the term focusing parameter for the exponent γ that controls the severity of this focusing technique [11, 260]. For a batch of length N , focal

loss is computed via

$$\frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^N w_i \sum_{j=1}^M \left(- \underbrace{(1 - \hat{y}_{ij})^\gamma}_{\text{Focusing term}} \cdot y_{ij} \log(\hat{y}_{ij}) \right), \quad (5.8)$$

where $\hat{y}_{ij}$ and y_{ij} are defined as above. More on the individual choice of parameters and why this loss function is used in the respective cases can be found in the above-mentioned sections.

5.1.3.2 Training Loop and Backpropagation

This paragraph is strongly inspired by Ref. [248], as within PyTorch [255] unlike some other ML frameworks, many of the steps are written explicitly. Within the so-called training loop, the network is presented all training data, for which the model parameters are optimized via backpropagation to yield small error rates. Each iteration through this loop is termed epoch, and within each epoch, there is some variant of an inner loop which iterates over batches (disjoint subsets of the training data). The network is updated once per batch (i.e. once per calculated batch-wise criterion), thus, multiple times per epoch. Both the number of epochs and batch size are configurable parameters, the latter is oftentimes chosen as a power of 2 when using graphics processing units (GPUs) to better exploit the internal architecture. The training loop itself has two parts, one dealing with the training set only and working with gradients of the loss function per batch, a second part utilizing a separate, disjoint set of validation samples to cross-check the model performance on “unseen” data, i.e. data from which it did not learn. Validation is done after the inner loop over training batches is finished. As the investigated algorithms for this thesis all perform supervised training, any so-called `DataLoader` in PyTorch yields not only the inputs of the given samples, but also the truth labels to compare against. A third return value obtained from a custom `DataLoader` (building on top of [263–265]) is the static sample weight to perform weighted averaging over individual samples to maintain certain pre-defined constraints for some input or target distributions (e.g. kinematic quantities of jets, reweighted to a reference distribution or corrections related to class imbalance).

For each iteration through the inner loop with training data, the inputs are passed through the network (forward step), the loss is computed, and the gradient of the loss with respect to the model parameters is accumulated using the backpropagation technique. With this new information, model parameters θ are optimized via gradient-based algorithms. Backpropagation is the ML analogon of the chain rule for derivatives known from calculus. Starting with the gradient of the loss on the output layer, the backpropagation algorithm moves recursively backwards through all layers of the network, accumulating gradients for each output activation:

$$\frac{\partial L}{\partial \theta} = \frac{\partial L}{\partial \vec{x}^l} \underbrace{\frac{\partial \vec{x}^l}{\partial \theta}}_{\frac{\partial \vec{x}^l}{\partial \vec{x}^{l-1}} \frac{\partial \vec{x}^{l-1}}{\partial \theta} \dots} \quad (5.9)$$

Eventually, the final gradient serves as a tool to update the weights and bias terms, thus, the learned parameters of the network [247, 248].

5.1.3.3 Optimizer and Learning Rate

The minimization of the loss function is performed with optimizers [247,248,266], for example using gradient descent:

$$\theta_{t+1} = \theta_t - \eta \nabla_{\theta}(L), \quad (5.10)$$

where η is the learning rate (configurable hyperparameter), and t is the respective step for optimization of the parameter θ . This optimizer takes all training data into account, however, as already introduced for the training loop, the whole dataset is oftentimes split into smaller, randomly chosen parts, the batches. This technique then corresponds to stochastic gradient descent (SGD) [267], where the model parameters are updated after each iteration through the inner loop, i.e. for each batch individually. Therefore, more model updates per iteration through the whole training set are performed, yielding quicker convergence, and as the random batches introduce a variance, the algorithm is less prone to approach local minima only [247,266]. An extension is given by the Adaptive Moments Estimation (Adam) optimizer [268], which employs a dynamic adaptation of the learning rate to control the weight updates. Instead of keeping the learning rate fixed or scheduling modifications of the learning rate *a priori* without monitoring potential oscillations around or between local minima, convergence is facilitated with Adam by adapting the step size for each update individually [247,266,268].

A model parameter update for the $t + 1$ -th batch, using the value of the previous, t -th batch in the calculation, follows the iteratively applied instruction [247,266,268]

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t \quad (5.11)$$

with $\hat{m}_t = \frac{\beta_1 m_{t-1} + (1 - \beta_1) \nabla_{\theta_t} J(\theta_t)}{1 - \beta_1^t}$ and $\hat{v}_t = \frac{\beta_2 v_{t-1} + (1 - \beta_2) (\nabla_{\theta_t} J(\theta_t))^2}{1 - \beta_2^t}$,

where $\hat{m}_t$ refers to the mean (first moment) and $\hat{v}_t$ refers to the variance (second moment) of the gradients of the loss function with respect to the model parameters [266]. This definition includes tuneable parameters, which are listed with their default values in PyTorch: $\beta_1 = 0.9$, $\beta_2 = 0.999$ (decay parameters), $\eta = 0.001$ (initial learning rate) and $\epsilon = 10^{-8}$ (small constant for numerical stability) [269].

A learning rate schedule [270] may be employed to prevent “overshooting” while approaching a (global) loss surface minimum. In PyTorch, the implementation of optimizers allows defining an initial lr parameter (equivalent to η in Eq. (5.11)) and accessing this parameter dynamically at every stage of the training, such that a custom schedule can be implemented with a function of the form

$$\text{lr}(\text{epoch}) = \frac{\text{lr}(\text{epoch} = 0)}{1 + \frac{\text{epoch}}{n_e}} \quad (5.12)$$

where n_e is the total number of epochs for the training and $\text{lr}(\text{epoch} = 0)$ is a configurable initial learning rate.

5.1.3.4 Regularization

The performance of a NN on data it was trained on needs to be compared with performance on a disjoint set of samples (unseen data) that are not used to fit the model parameters to investigate generalization of the model. If the model predicts observations of the training set well but shows higher generalization error for a validation set, i.e. on the unseen samples, overfitting occurs. Mitigation techniques aiming at keeping the generalization gap between training and validation small, thus preventing overfitting, are denoted as regularization techniques [247, 248].

For all networks in this thesis, the dropout [250] technique is used as a countermeasure against overfitting. The activation of individual, randomly chosen nodes in the hidden layers may be set to 0 during the training stage, with a pre-defined probability. When inference is performed, these nodes are active again [271]. While averaging over ensembles of different NNs would be theoretically possible to achieve regularization, in practice, time and memory are a limiting factor, such that dropout is considered a suitable alternative for ensemble learning [247, 272]. A generalization of this method is DropConnect [249], where instead of entire nodes, individual connections (weights) between nodes are dropped.

Another method, applied for the CMS tagging algorithms studied in this thesis, is batch normalization [273]. This can be understood as a dynamically learned scaling and shifting of the outputs of a given layer by computing a batch-wise mean and standard deviation for a given layer, and applying a parametrized linear transformation thereafter.

A simple technique that does not involve any modifications of the network architecture itself is early stopping [247, 248, 274, 275]. It requires splitting all available data into three sets, the training set (from which the model is learning), validation set (to probe generalization capabilities), and the test set (to report performance metrics with an unbiased set of samples), for example with the `sklearn.model_selection.train_test_split()` function of `scikit-learn` [276]. Using validation loss after each epoch or model update, the convergence is probed and the training eventually stopped if the performance metric does not improve anymore after a configured amount of steps. The model with best validation loss so far is usually chosen as the final model to be studied with the test set, and to be deployed to production.

5.1.3.5 Convolutional Neural Networks

Inspired by biological pattern recognition processes involved in vision [277–279], convolutional neural networks (CNNs) exploit correlations in images or image-like data [247, 248, 279]. Unlike simple MLPs for which the number of parameters increases quickly for high-dimensional input spaces, CNNs are able to leverage information in a parameter-efficient way by encoding shared weights in multidimensional convolution layers in which parts of said layer are only connected to a small receptive field. Local patterns in the input are passed through the CNN with the help of learned filters (or kernels) that scan the input for important features helping with the recognition process. When performing the convolution operation as implemented in PyTorch, a weighted sum between, for example, the pictures of an image, and the kernel is calculated using matrix operations [248]. The kernel with configurable size is translated along the input in configurable step size (stride), using the same weights (trans-

lation invariance). After “scanning” a given layer with a kernel, the outputs or feature maps may be aggregated by pooling layers which derive a summary quantity of neighboring cells in a rectangle, for example by noting the maximum activation of the cells under investigation, or their average. With this pooling process, the dimensionality of the feature maps is reduced. Eventually, the final convolved layers can be unpacked into a linear layer with which standard MLP operations are performed, ending for instance with output layers to classify an image [247,248].

5.1.3.6 Recurrent Neural Networks

For recurrent neural networks (RNNs), inputs are handled in sequential order and processed in such a way that the input for each forward pass includes the output of the previous forward pass [247,248]. This makes them suited to perform natural language processing (NLP) tasks [248]. In fact, recent RNN architectures scale to variable length inputs and their computational graph can be unfolded into steps in recursive manner, such that a RNN layer is sometimes also associated with a step in time. A popular class of RNNs is termed Long Short-Term Memory (LSTM) [280]. Besides the outer recurrence, LSTMs also feature an inner self loop and gates that control forgetting. This helps when recalling long-term dependencies is necessary, and old states of information do not need to be retained [247].

5.1.3.7 Performance Metrics

Studying the performance of DNNs, specifically for classification tasks, requires defining suitable evaluation metrics when passing test data through the network in the inference step after training. In principle, the loss function acts as a measure of performance (usually for training or validation sets), but its magnitude is non-universal, given that the number of classes, samples and their class distribution balance, and calculation strategy (for example CCE or FL) play a role. To derive accurate measures of performance, suitable for jet flavour tagging, some definitions following Refs. [233,281] are introduced, starting with binary classification as an example.

The first concept refers to building a so-called confusion matrix that relates predicted classes to truth classes. For the binary classification case, the four different cases are summarized in Table 5.1, which can be extended to multiclassification by increasing the dimensionality of the matrix accordingly.

Table 5.1: Confusion matrix of a binary classification problem, where predictions are compared against the actual classes, taken from Refs. [233,282].

	Positive prediction	Negative prediction
Positive class	True Positive (TP): correctly identified examples of the positive class as part of the positive class	False Negative (FN): wrongly identified examples of the positive class as part of the negative class
Negative class	False Positive (FP): wrongly identified examples of the negative class as part of the positive class	True Negative (TN): correctly identified examples of the negative class as part of the negative class

The four quantities TP , FN , FP and TN may be understood as absolute numbers or rates with respect to the total number of samples. If these are used as components in fractions, the denominator cancels out leaving just the absolute numbers. One example of such a new quantity that is calculated from the confusion matrix components is accuracy [281]:

$$\text{Accuracy} = \frac{TP + TN}{TP + FN + FP + TN}. \quad (5.13)$$

This quantity counts how many samples are correctly classified and relates this value to the total number of samples, the size of the population. Comparing across classification algorithms, this metric is independent of the overall size of the population, but it is inaccurate for highly imbalanced datasets and classifiers that, in the worst case, only assign the most frequent (majority) class to any possible input configuration. Accuracy would be high in such a setting, although, again in the worst case, not a single sample of the less frequent class is assigned correctly, rendering the classifier redundant. Better metrics need to be defined to also quote reliable measures of performance on minority classes [281, 282]. The distribution of classes can be taken into account by quoting other measures of performance which are normalized to the fraction of samples of a certain truth class as summarized in Table 5.2, instead of normalizing to the total population:

Table 5.2: Terminology for selected performance metrics for binary classification tasks and alternative naming schemes for object identification such as jet flavour tagging [9, 17, 233, 282].

True Positive Rate (TPR, sensitivity, efficiency): fraction of positive instances correctly classified as positive $TPR = \frac{TP}{TP + FN}$	False Negative Rate (FNR, miss rate): fraction of positive instances misclassified as negative $FNR = \frac{FN}{TP + FN}$
False Positive Rate (FPR, mistagging rate): fraction of negative instances misclassified as positive $FPR = \frac{FP}{FP + TN}$	True Negative Rate (TNR, specificity): fraction of negative instances correctly classified as negative $TNR = \frac{TN}{FP + TN}$

With Receiver Operating Characteristic (ROC) curves [282–284], performance of binary classifiers is visualized in a two-dimensional representation with axes not biased by class imbalance [281]*. Choosing the example of jet flavour tagging, the creation of ROC curves is explained in Fig. 5.1. The definitions of Table 5.1 are visualized for an example binary classifier (blue box and lines) and a random classifier (red box and lines) that separate the two truth classes on the x axis matching the classifier score P . With configurable precision, this axis is scanned with a varying threshold value above which a sample would be classified as positive, otherwise as negative. This threshold describes the inherent tradeoff of high efficien-

*Besides ROC curves, performance of classifiers may also be reported with precision-recall (PR) curves [283]. The axes of PR curves are built from recall (also known as sensitivity) and precision (true positives over all samples predicted to be positive). There are cases where PR curves offer a more nuanced view on performance for samples with large class imbalance [281].

cies that also introduce high mistagging rates. Different classifiers perform differently well for this discrimination task. For each threshold value, the TPR and FPR scores are determined according to Table 5.2 and shown on the resulting ROC curve as individual points. In the standard visualization, TPR and FPR form the y and x axis, respectively [282], for jet flavour tagging in CMS, the axes are swapped and the y axis shown on a logarithmic scale (making it easier to read off efficiencies at fixed working points related to negative powers of ten[†]) [9, 17, 63, 239, 240].

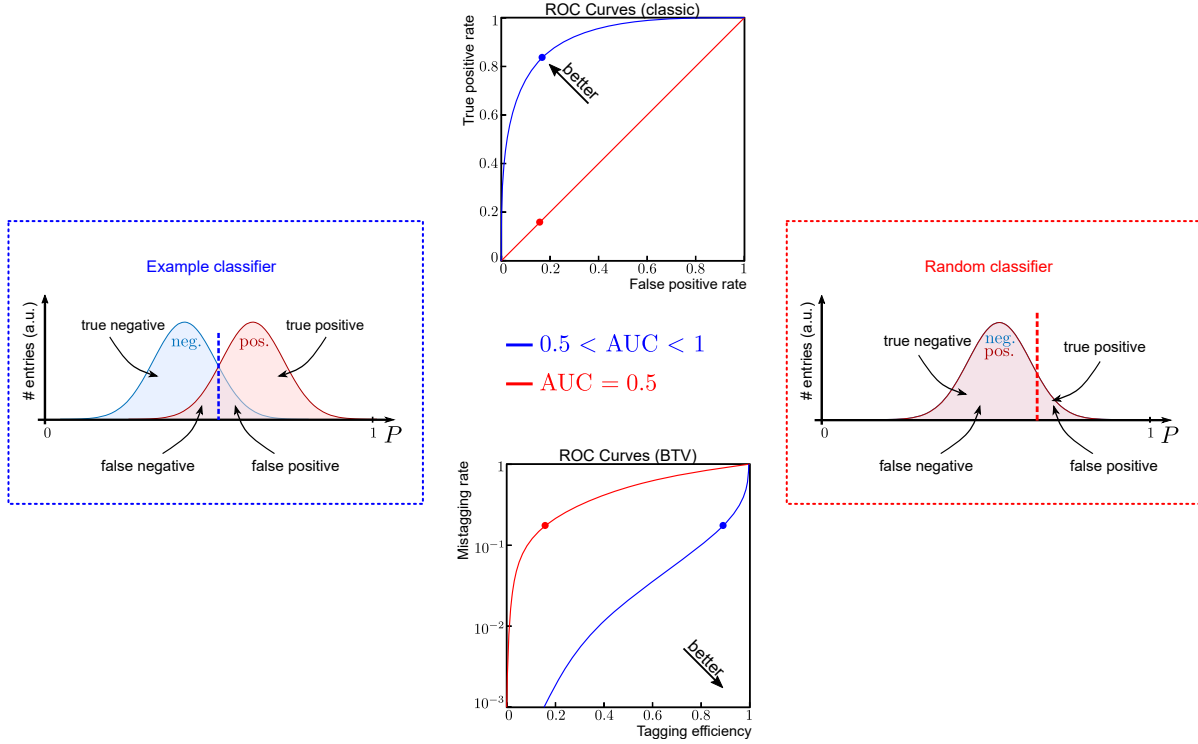


Figure 5.1: Sketch of classifier score distributions for positive and negative classes, for a typical classifier, compared with a random classifier, and corresponding ROC curves. On the top, the ROC curves are shown with general ML labelling and axes definitions, the lower panel represents the style that is used in the b-tagging and vertexing (BTV) group in CMS, for example as shown in Refs. [9, 239, 240]. The blue box and correspondingly blue lines represent a typical example classifier, while the respective red objects are associated with a random classifier practically showing exactly overlaid positive and negative distributions. Exemplary thresholds at which to evaluate performance are shown as dashed lines for the scores or points on the coloured ROC curves, respectively. Taken from Ref. [233], Gaussian distribution adapted from Ref. [285].

In Fig. 5.1 (top), better performance is shown by ROC curves extending further towards the upper left, on the lower panel, the lower right corner hints at better performance.

Besides the twodimensional full representation of TPR and FPR , the integral can be calculated to yield the area under the curve (AUC) as a scalar quantity. This measure can be chosen if information loss with respect to the full ROC curve is uncritical, for example to show evolution of performance over training iterations, not having to rely on quantities like loss or accuracy [281]. Best performance would yield an AUC of 1 [283, 284], a value of 0.5 corres-

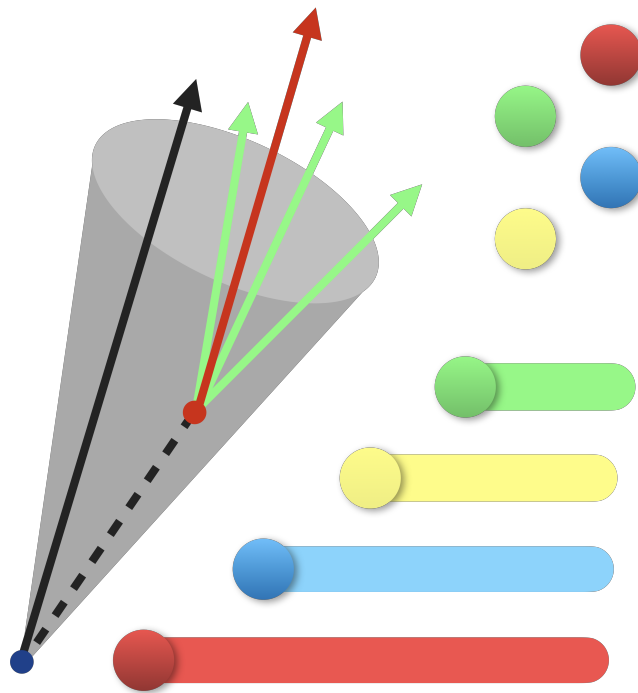
[†]These are the "loose", "medium" and "tight" WPs at 10 %, 1 % and 0.1 % mistagging rate, respectively [9].

ponds to coin flipping or random classification, an example for such a case is given by the red example in Fig. 5.1. Should AUC be less than 0.5, the binary classification is practically inverted (more positive examples are classified as negative, and vice versa, such that even a classification would perform better) [283, 284]. On the other hand, the separation between classes is also present for the $AUC < 0.5$ case, so flipping the axis restores the behavior.

The extension to the multi-class case is discussed when introducing the first jet tagging algorithm study in this thesis, in Chapter 6.

Part II

Development of Robust Jet Flavour Tagging Algorithms



6

Robustness Studies with Simulated Particle Jet Samples and the Loss Surface

“Success is about more than simple resilience. It’s about using that downward momentum to propel ourselves in the opposite direction. It’s about capitalizing on setbacks and adversity to become even happier, even more motivated, and even more successful. It’s not falling down, it’s falling up.”

– Shawn Achor

This chapter is based on Refs. [11, 16] and introduces key robustness concepts that play a role also for subsequent studies, first with the help of simulated samples only.

6.1 Introduction

As demonstrated in Chapters 3, 4 and 5, machine learning (ML) and especially deep learning (DL) [247, 279] are important tools for the experiments at the LHC at CERN. High-dimensional datasets are abundant in HEP, and their complexity and size is growing, especially in view of the HL-LHC era, pointing at even higher relevance of state-of-the-art ML techniques. Considerable performance gains over traditional methods are achieved for tasks such as pattern recognition or signal identification in background-dominated regions [286]. With DL, novel physics analyses are enabled, and existing analyses profit when transitioning from purely cut-based strategies to utilizing algorithms that capture correlations between observables by construction. With increasing complexity and performance of the NNs in use, for example those that leverage many low-level features for object identification, understanding the network’s decisions arises as a new research question. Studies associated with explainability [287] and safety [288] of artificial intelligence (AI) models with the aim of providing a more nuanced picture of the NNs in question can be translated to HEP as well. It has been shown that small modifications of inputs (i.e. applying so-called adversarial attacks that yield adversarial examples) can severely affect the performance of neural networks [12, 13]. In computer vision, even one-pixel attacks can adversely influence the NN prediction [289]. For HEP, where many identification algorithms rely on at times inaccurate simulation to train the classifiers, this is a worrying prospect [14]. Indeed, studying susceptibility to mismodelings and better understand these “intriguing properties of neural networks” [12] is a central objective of this thesis. The aforementioned effects can be attributed to reliance on linear operations in model building and a large number of input variables [12, 13]. While modern NN architectures also apply non-linear activation functions with which various classes of functions can be approximated [256], (piecewise-)linear activation functions such as ReLU make the networks vulner-

able to first-order adversarial attacks [12, 13].

As explained in Chapter 4, jet flavour tagging is a crucial technique for several analyses carried out with experiments at the CERN LHC. Many modern algorithms performing this task are multiclassifiers exploiting state-of-the-art ML methods with a large number of low-level features, making jet tagging a reasonable application for AI safety studies [14, 15, 288]. The benchmarked model to probe the performance / robustness trade-off is a DNN similar to the established DeepCSV [9] tagger, which has been mentioned in Section 4.2.4 and which has been investigated in Refs. [233, 290]. Systematic distortions aiming to deteriorate the performance of the DNN are created and the network's performance on such adversarial examples is studied. This is one realization of analyzing the possible impact of mismodelings associated with imperfect MC samples or detector effects, worse than randomly smearing inputs, as the attacks are specifically crafted for each individual model and feature set.

In contrast to artificially injected mismodelings which target worse-case scenarios, physics-induced mismodelings of input features are essentially independent of the model in question. These mismodelings can be associated with steps in the MC simulation chain, such as matrix element calculation with a fixed precision for the hard process, parton shower modeling (with a set of parameters steering fragmentation and hadronization), all up to fixed perturbation order, and finally the detector simulation. For the latter, detector misalignment or calorimeter calibration are possible sources of imperfect modeling which eventually translate to imperfect modeling of the input variables for the DNN in question [9, 14, 17, 220]. While mismodeling of features with small discriminating power may be acceptable, imperfections for those features which are of high relevance or highly ranked on the importance scale for the DNN predictive power can be detrimental when applying the algorithm not only on simulated samples, but also on recorded data [14, 17]. The choice of the MC generator has been shown to affect the NN performance [246]. Figure 4.5 shows performances of different b-tagging algorithms in simulation, after applying scale factors derived from data to simulation, and when also including a combined statistical and systematic uncertainty band. The DeepJet performance visualized with the ROC curve representing data does not reach that of simulated samples over the full efficiency-mistagging plane, even when taking the considered uncertainties into account. This effect is less pronounced for DeepCSV with an overall worse performance, where the mismatch is only evident for high efficiencies. For analyses with multi-jet final states such issues related to imperfect calibration can grow into a difficult-to-model experimental uncertainty. For this thesis, the underlying assumption is that a model with improved resilience towards adversarial attacks will also be less susceptible towards subtle mismodelings inherent in the typical HEP context, when performing standard validation methods with data. The approach does not claim to eliminate the existing mismodeling for the input features or correct them *a posteriori*, but the goal is to build a model that embeds the information in such a way that it generalizes better to a non-training domain [247, 291], that is, to detector data [14, 17]. Further, adversarial attacks as described above are usually understood as being “invisible” in the inputs, but very effective in deteriorating performance. As typical validation strategies in HEP use a fixed, oftentimes onedimensional histogram binning with which correlations are undetectable, subtle mismodelings may be hidden [14]. Two components are relevant when probing the model with simulation only, first, to estimate how tagging efficiency and misidentification rates are affected by artificial mismodelings, and second, to strengthen the model such that these mismodelings are less impactful on model performance [14]. Studies with

detector data are later on carried out in Chapter 7.

The selected mitigation technique is termed adversarial training [291–293] and will be used as a defense strategy against performance loss induced by systematically distorted samples by applying the adversarial attack during the training stage. Instead of training a classifier and adversary with a modified loss function [294], or performing domain adaptation via gradient reversal layers [295–298], the method applied in this thesis works on the input features directly. It also needs to be differentiated from data augmentation or uncertainty-aware learning [299]. For theory-induced or generator-dependent modeling effects, it is unlikely that adversarial methods are the best mitigation technique [223]. Another way to phrase this is that neither “nature” nor the “generator” know how to confuse the NN that will be used to identify jet flavour. Eventually, observed mismodeling in the discriminant shapes or when applying SFs is a convolution of several effects, the modeling of the inputs on the one hand, and the NN properties on the other hand. In the following sections, only the NN-related component can be probed, and if this translates positively to data/MC investigations is subject of Chapter 7. The results in this chapter need to be interpreted with caution: the network is instructed to better resist a certain class of adversarial attack, and thus, performance studies performed with this type of modeling effect are naturally biased in favor of the network’s capabilities. Second, for physics analysis, the network’s ability to withstand artificially introduced distortions is not of prime interest, but rather how well its performance generalizes between simulation and recorded data. The validity of the attack in terms of physics is enhanced by applying a constraint on the allowed impact of the attack. To achieve more solid claims of network resilience, the model is probed not only with the attack that is utilized during the augmented training strategy, but also with other types of mismodeling scenarios. A subsidiary study is conducted with the help of the underlying loss surface, which serves as a method to understand *why* the adversarial model turns out to be more robust than the nominal model and how this is related to geometrical properties.

6.2 Dataset and Input Features

The JET FLAVOR DATASET [8] accessible at the UCI Machine Learning in Physics Web portal under the link <http://mlphysics.ics.uci.edu/> is used for this study. As mentioned in the original publication [8], simulated samples are generated with MADGRAPH5 [86] and PYTHIA 6 [89] and further processed with DELPHES 3 [159] to model the ATLAS [116] detector response.

For jet clustering, the anti- k_T algorithm [188] is implemented with the FASTJET [189] package, with $R = 0.4$. The adaptive vertex fitting algorithm [166] is used to reconstruct SVs with the RAVE toolkit [300], interfaced with the detector simulation with code accessible under Ref. [301]. Jet flavour truth labeling is defined with the help of parton matching within a cone of $\Delta R < 0.5$. Three classes, light (up, down, strange quarks or gluons), charm, or bottom distributed as 48.7% : 12.0% : 39.3% of the total number of jets are contained in the dataset, with hierarchical precedence for the heavier flavour in case of multiple parton matches [8]. The partonic flavour definition is further unwrapped in Table B.1.

6.2.1 Input Features

Tables B.2 and B.3 list a detailed description of all input variables, which is based on Ref. [8] and summarized here.

A hierarchical organization is used for all input features. Tracks with their helix parameters and covariance matrix elements, as well as properties related to their associated vertex, are considered as low-level features. After sorting by impact parameter significance, the full dataset contains up to 33 tracks per jet, of which the first six are considered for this analysis. The sorting and further processing for ML purposes is handled with the awkward [302] package and implemented in Ref. [303].

Expert (high-level) features are constructed at the jet level, as a function of the low-level inputs via sums over all tracks or SVs, yielding for example a weighted sum of displacement significances, or kinematic features of the overall jet.

To facilitate further processing in ML models, missing or unavailable variables are replaced by convenient default values.

6.2.2 Preprocessing

A total number of 11,491,971 jets are available in the dataset and a random splitting into training (72%), validation (8%) and test (20%) sets is performed. Normalization of input features to obtain distributions with mean of 0 and standard deviation of 1 is performed, using the training set as a reference and excluding default values from the parameter derivation. To avoid any inference with manually imputed default values and the regular distribution, defaults are placed just below the minima of the primary feature distribution of interest, without sacrificing training convergence. With this preprocessing technique, fixed length input shapes are transferred to the first layer of the DNN and vanishing or exploding gradients due to large values are avoided [247, 248, 275].

To exclude a potential flavour dependence of the classifier related to the given kinematic distributions present in the dataset and mitigate the inherent class imbalance quoted above, sample weights are calculated in three dimensions: flavour, jet transverse momentum (p_T) and pseudorapidity (η) as mentioned in Ref. [9]. For this investigation, the target shape refers to the flavour-balanced average of the three initial 2D-distributions using a 2D grid of 50×50 bins, spanning ranges between (20, 900) GeV and $(-2.5, 2.5)$, respectively. The implementation in Ref. [303] offers multiple ways to use these weights during training, such as resampling, or loss reweighting per batch. The latter option is used for this study, modifying the loss per sample with a weight w_i according to Eq. (5.8).

6.3 Methods

6.3.1 Reference Classifier

A jet flavour tagging algorithm similar to those used at LHC experiments is used as an example, choosing a FCNN in analogy to DeepCSV [9] with five hidden layers of 100 nodes each. After flattening the six chosen tracks of the low-level category, 184 features and consequently 184 input nodes are defined. Further details on the architecture with methods introduced in

Sec. 5.1.2 include: dropout layers [250] with a 10% probability, ReLU activation for hidden layers and a softmax function for the outputs to arrive at three predicted probabilities, one for each possible label.

Focal loss [260] with a focusing parameter of $\gamma = 25$ is used as loss function, see Eq. (5.8) for the multi-class definition. This strategy has been proven useful to arrive at smooth classifier output distributions needed for calibration, in a setting with highly-imbalanced data. Supplementary material described in Sec. C.2 shows how reweighting with the nominal loss function without application of the focusing term may treat the class imbalance well, but that predictions are made with too high confidence placing scores almost surely at 0 or 1.

Training is performed on an NVIDIA Tesla V100 GPU at the CLAIX [304] computing cluster at RWTH Aachen University, using the PyTorch [255] library. The Adam optimizer (see Sec. 5.1.3.3) with PyTorch's [255] default settings is chosen to update the model parameters with a learning rate schedule that starts at 0.0001, as described in Eq. (5.12). A batch size of $2^{16} = 65,536$ is chosen, and as regularization technique, early stopping is applied. Model parameters are stored after every iteration through the full training set, facilitating later evaluation as a function of epoch.

6.3.2 Evaluation Metrics

The implemented tagger performs multi-class classification, however, in a physics analysis, the distinction is oftentimes made between two classes at a time. For this purpose, appropriate likelihood ratios of the bottom, charm and light output classes are constructed, following the pattern $X_{\text{vs}}Y$ for discriminating class X from Y via

$$X_{\text{vs}}Y = \frac{P(X)}{P(X) + P(Y)}. \quad (6.1)$$

In this definition, $P(X)$ and $P(Y)$ refer to the classifier's score for the respective flavours, such as bottom and light flavour in the case of $B_{\text{vs}}L$. This pattern will be revived in Chapter 7, where more than three classes are possible for advanced models.

ROC curves and AUC scores serve as performance metrics, as explained in Sec. 5.1.3.7. The efficiency and mistagging rates then refer to the $X_{\text{vs}}L$ scores defined above, with X acting as positive, and Y as negative outcome.

6.3.3 Adversarial Attacks

The Fast Gradient Sign Method (FGSM) [13, 247] is one way to generate adversarial inputs. Inputs are modified in a systematic way to increase the loss function in the direction of steepest increase around the raw inputs. The mathematical operator that facilitates retrieval of the "steepest increase" is the gradient of the loss function, here taken with respect to the inputs. For FGSM, only the sign of this vector is taken (possible values are therefore plus or minus one), which is then multiplied with a (small) limiting parameter ϵ controlling the desired severity of the impact. While such distortions not necessarily manifest in the input variable distributions (due to a small ϵ), shifting inputs by these modifications yields a systematic confusion of the network with, to first order in the ℓ_∞ norm, worst case disturbances.

Adversarial inputs x_{FGSM} are generated from raw inputs x_{raw} by computing

$$x_{\text{FGSM}} = x_{\text{raw}} + \epsilon \cdot \text{sgn}(\nabla_{x_{\text{raw}}} J(x_{\text{raw}}, y)), \quad (6.2)$$

where $\text{sgn}(\alpha)$ is the sign of α , the loss function is denoted as $J(x_{\text{raw}}, y)$, depending on the inputs (x_{raw}) and targets (y).

One interpretation of the FGSM attack is that this represents gradient ascent, i.e. the inversion of gradient descent along the loss function, in the input space [13, 291, 293]. It classifies as a white box attack [291] with full knowledge of network architecture and parameters.

Figure 6.1 is a visualization of the FGSM, choosing one input variable x_i only, for didactic reasons. To put this method to use in the application context, such distortions are applied in all input dimensions at the same time, and the limiting parameter ϵ is equal for all features.

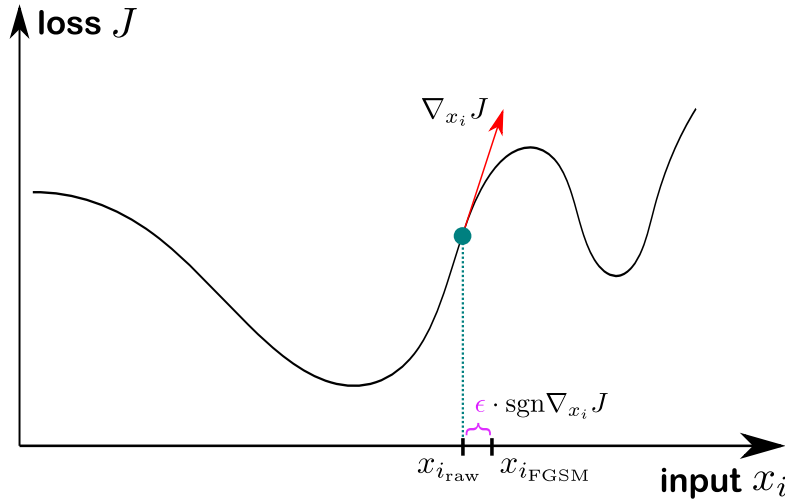


Figure 6.1: Generating adversarial inputs with the FGSM attack, simplified with one dimension only. The figure is taken from Ref. [233].

The reasoning for choosing a suitable norm of the gradient vector, in this case, the sign of the gradient, is based on the assumption that distortions shall stay in reasonable bounds, instead of getting arbitrarily large. Going only a small step in each direction is closer to what can be expected for possible uncaught or unnoticeable mismodelings or differences between data and simulation [12, 13, 13, 14].

With NNs that propagate feature attribution through activation functions, it is conceivable that small variations by ϵ for individual features add up to large overall impact on model performance. The susceptibility rises with the number of inputs, while a single feature may remain almost unaffected [13].

As FGSM is only a first-order attack, the global worst-case scenario may look different, depending on the actual properties of the loss surface [293]. Without iterative checks, given a fixed limiting parameter, FGSM could shift inputs into local minima or harmless regions of the loss function. Only for the direct vicinity of the original inputs, FGSM will maximally confuse the model, under the norm introduced above.

As the attack with a small floating point value ϵ shall be used for physics observables in a meaningful way, it is not applied to integer variables (e.g. the number of tracks) or defaulted values.

Additionally, large distortions that could be easily caught with validation strategies are excluded with the help of setting a limit of 25% with respect to the original value to clip the distortion. This yields a modified x_{FGSM} according to Eq. (6.3), with the original input value x , the transformed (preprocessed) value x' (inverting the normalization is denoted by $()^{-1}$), and the FGSM scaling factor ϵ .

$$x_{\text{FGSM}} = \left(x + \text{sgn}(\nabla_x J(x, y)) \cdot \min \left\{ \begin{aligned} &|(x' + \text{sgn}(\nabla_x J(x, y)) \cdot \epsilon)^{-1} - x|, \\ &|0.25 \cdot x| \end{aligned} \right\} \right)' \quad (6.3)$$

After application of the FGSM attack to low-level features, the high-level features are not re-computed, they are distorted according to the gradient of the loss function only. The FGSM attack does not fully take the feature correlations into account, besides choosing discrete distortions for the multidimensional gradient simultaneously. A related approach that replaces the discrete shift is subject of Sec. 7.3.2.

6.3.4 Adversarial Training

For this study, adversarial training serves as defense strategy. Perturbed inputs are injected already during the training phase [291], with algorithmic description shown in Fig. 6.2. A red line highlights the difference between nominal and adversarial training.

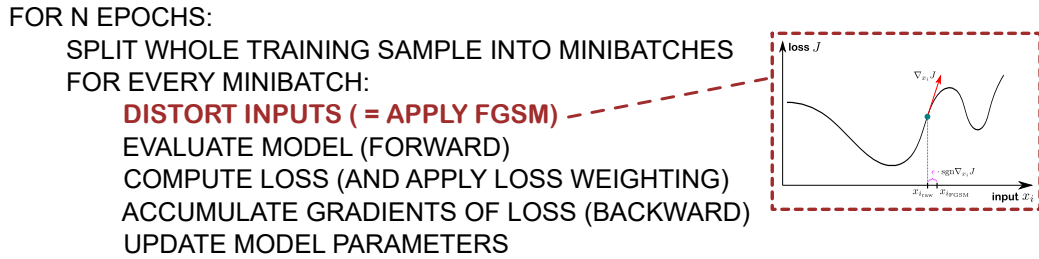


Figure 6.2: Adversarial training algorithm utilizing the FGSM attack in the training loop. Inputs are distorted prior to the forward and backward passes, modifying the standard training algorithm (black), based on Ref. [248] with an additional step [15]. The figure is taken from Ref. [233].

During adversarial training, the NN never sees the nominal inputs during the whole training step [291–293], as shown in Fig. 6.3 with the insertion of the red block prior to backpropagation. Applying the FGSM attack continuously to the training data for every intermediate state of the model after updating the model parameters is assumed to introduce a saddle point into the loss surface. This shall yield to a network that is less likely to learn the simulation-specific properties of the used sample, improving generalization capability [247, 291, 293]. Two optimizations are “competing”, the outer minimization problem handled by gradient descent, and the inner maximization via gradient ascent [293, 305].

It has been shown that susceptibility to first-order adversaries (FGSM-attacked samples) can be reduced with adversarial training [293], acting as a powerful regularization technique. It

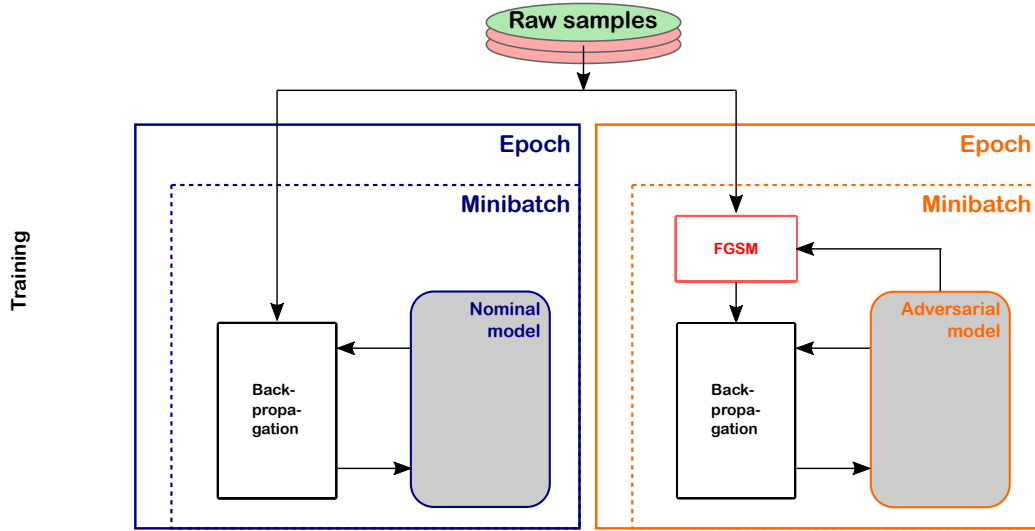


Figure 6.3: Comparison of the nominal and adversarial training against the FGSM attack. The figure is taken from Ref. [11].

has more systematic character than data augmentation (such as random input smearing), random deletion of connections (such as dropout and related techniques), or label smoothing by replacing certain labels with a probability to be wrong [247].

Effectivity of this mitigation technique can be attributed to the linearity of NN, which is also mentioned as a cause of NN susceptibility towards adversarial attacks. Adversarial training acts as a tool to adjust the loss surface to be locally constant around the input, which in turn downsizes the impact of perturbations evaluated with a high-dimensional linear function [247]. For a loss function that is almost flat in the vicinity of the raw inputs, slightly distorted inputs then cannot significantly increase the value of the loss function [306], making it a geometrical problem related to flattening the loss manifold [306–309]. This should improve the model robustness to those modifications applied on test inputs, with less decrease in performance than for the generic training. As the trade-off between performance and robustness suggests, it is possible that this comes at the price of maximally achievable performance on nominal inputs [247].

For the adversarial training stage, FGSM with $\epsilon = 0.01$ is chosen as the attack to defend against, the 25% impact limitation is not applied, falling back to Eq. (6.2) for training.

6.3.5 Inference

Inference is done in a two-step approach, as seen in Fig. 6.4. First, either original (raw / nominal) samples (green) or systematically distorted samples are acquired, with the same origin in the test set. As each training comes with its own respective loss surface that changes continuously during training, attacked samples need to be prepared for each model individually (represented by different colours blue and orange). Otherwise it could not be guaranteed that the respective attacked samples have deteriorating character for the respective models in question. Similarly, if different phases of the trainings are to be compared (i.e. after varying number of epochs), also the adversarial samples need to be crafted individually, each time.

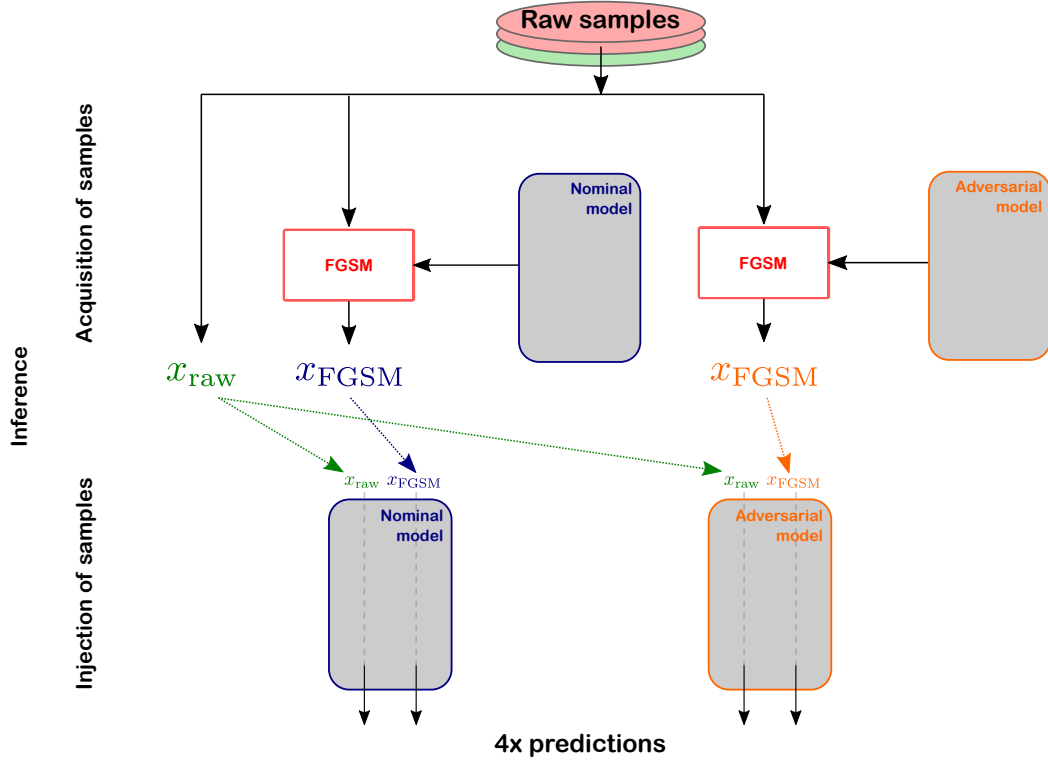


Figure 6.4: The inference process to perform a comparison of robustness of both training strategies. Section 6.4.2 describes the evaluation of the nominal training (green and blue paths) and Sect. 6.4.3 the corresponding studies for adversarial training, including all four combinations. The figure is taken from Ref. [11].

At convergence of both trainings, or in any configuration after a fixed number of epochs, there will be three sets of samples that can be injected into the different models: nominal samples (green, equal for both contenders), FGSM samples corresponding to the nominal training (blue), and FGSM samples that have been created for the adversarial training (orange).

6.4 Robustness to Mismodeling

6.4.1 Adversarial Attack

A verification of the distorted distributions is performed, investigating whether the produced disturbances would simulate the behaviour of systematic uncertainties remaining within an envelope expected by the typical data-to-simulation agreement. For four input variables, containing high- and low-level inputs, Fig. 6.5 shows the effect of the FGSM attack at two values of ϵ , compared to the nominal distribution. Even the largest value of $\epsilon = 0.05$ chosen for the following performance studies yields only marginal modifications of input shapes, staying within typical data-to-simulation agreements of the level of 10–20 % [9].

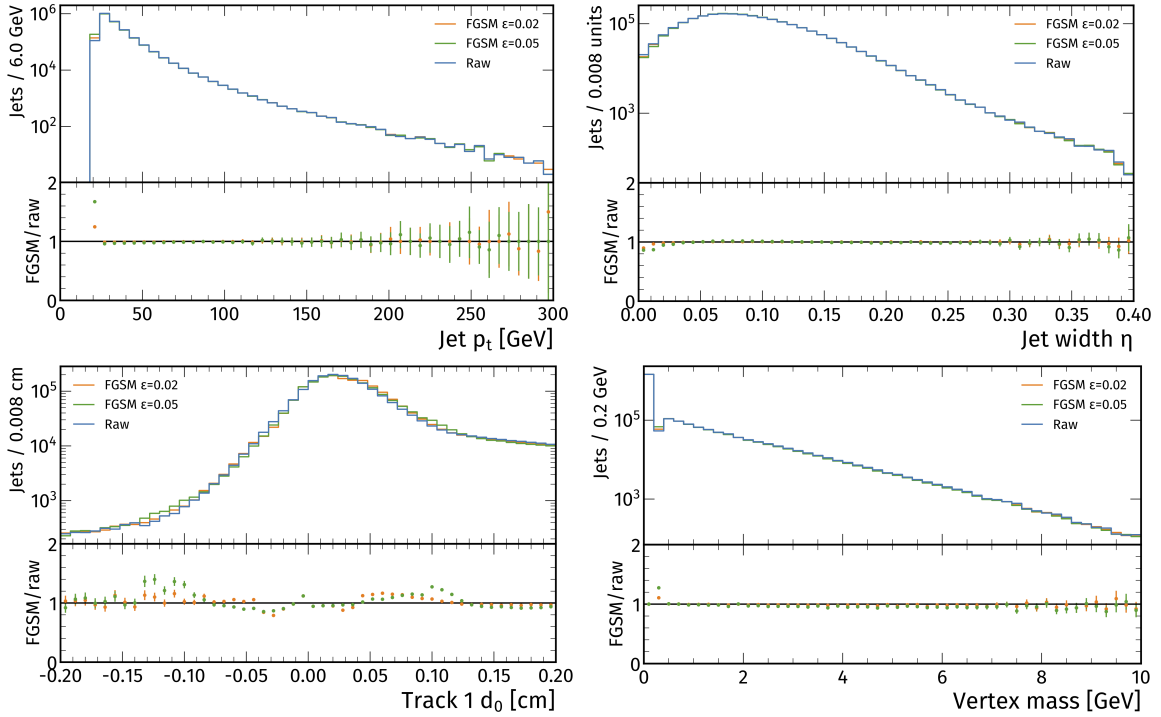


Figure 6.5: Distributions of raw and systematically distorted inputs, for a set of features containing high- and low-level information. The displayed range for the signed impact parameter (d_0) of the first track has been clipped to the most relevant central region, where distortions naturally appear enhanced. All figures and caption are taken from Ref. [11].

6.4.2 Vulnerability of the Nominal Training

First, susceptibility of the nominal model to FGSM attacks (mismodelings) of various magnitudes is established. Performance is probed with ROC curves for the BvsL (left) and CvsL (right) discriminants, Fig. 6.6 shows the resulting curves for nominal inputs and FGSM datasets generated with varying parameter ϵ^* . Best performance is achieved on undisturbed test samples with an AUC of 0.946, as expected, while increasing ϵ quickly leads to decreased performance. As shown earlier, $\epsilon = 0.05$ only causes barely visible differences in the input distributions, yet the corresponding AUC score is reduced to 0.883. At the 1 % mistag WP, a SF of 0.82 would be needed to explain a deterioration of signal efficiency from 73 to 60 %.

Another question to answer is how susceptibility relates to performance, given the ongoing hunt for better performing classifiers. For this, insights are sought by evaluating the performance of the classifier at various steps during the training on both the nominal and the perturbed datasets with a fixed $\epsilon = 0.05$. At each checkpoint, an AUC score is recorded. For selected epochs, ROC curves are produced, with a dependence shown in Fig. 6.7 for both discriminants. Longer training increases nominal performance, before it eventually saturates. Besides this unsurprising observation, with more epochs, vulnerability towards adversarial attacks becomes larger as well, following the exact opposite trend compared to nominal performance. The least performant configuration in this example, after only few epochs or iterations through the full training dataset, shows the highest robustness and minimizes the gap between dashed and solid lines.

*Corresponding output scores and and discriminant shapes are attached in Sec. C.3.

6.4.3 Improving Robustness through Adversarial Training

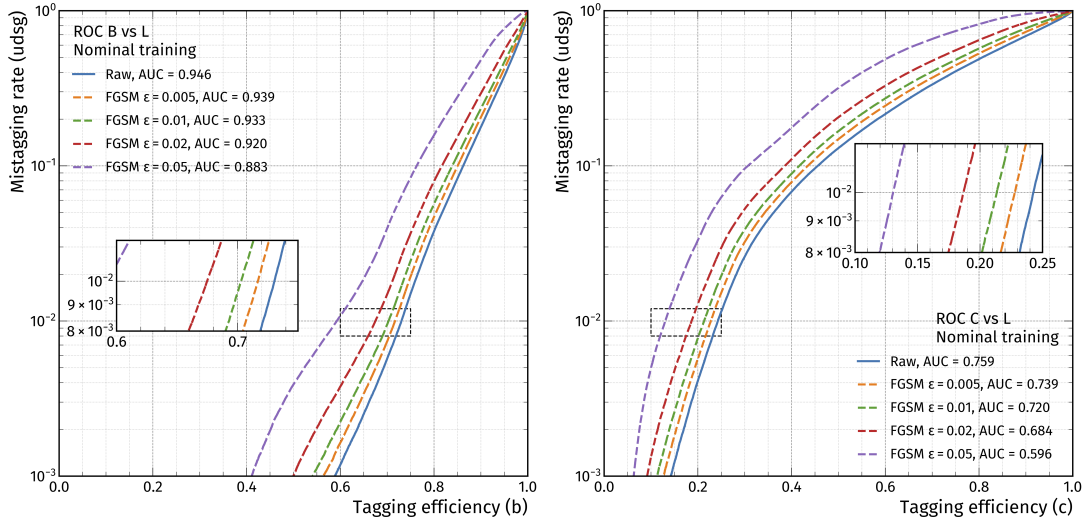


Figure 6.6: ROC curves for the BvsL (left) and CvsL discriminant (right), using the nominal training and applying FGSM attacks of different magnitudes. The model is evaluated when the training has reached peak performance. Both figures and caption are taken from Ref. [11].

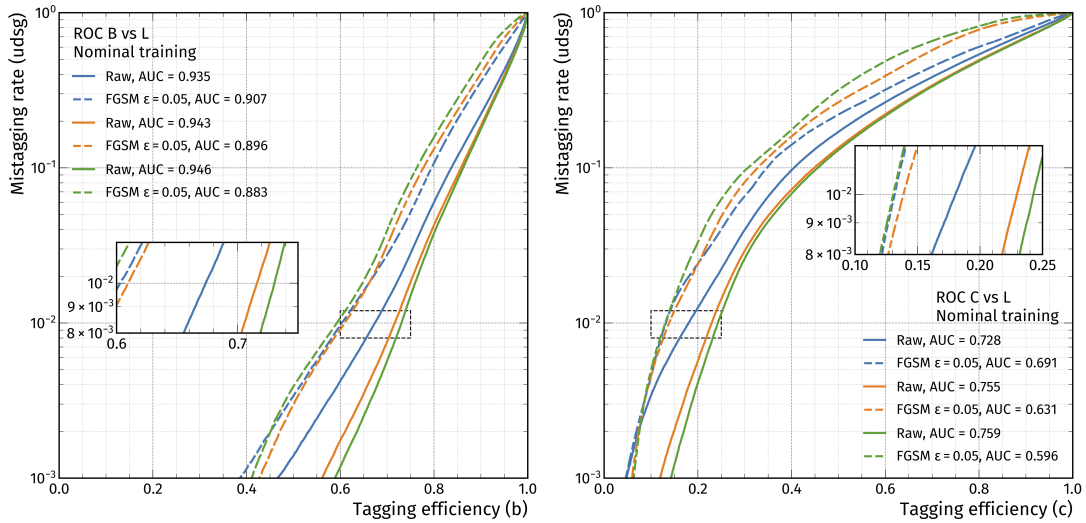


Figure 6.7: ROC curves for the BvsL (left) and CvsL discriminant (right), using the nominal training and applying FGSM attacks with $\epsilon = 0.05$ at various checkpoints of the training that each come with different nominal performance. Solid lines in different colors represent nominal performance gain with an increased number of epochs, dashed lines show corresponding performance on individually crafted FGSM samples for the particular checkpoints. Both figures and caption are taken from Ref. [11].

The studies described above are repeated with the adversarial model, choosing the same setup for the attacks when performing the inference for direct comparison of the nominal and adversarial training. FGSM samples are crafted individually per model, performance for BvsL and CvsL discrimination tasks are shown with ROC curves in Fig. 6.8.

Achieved AUC scores for BvsL are identical (0.946) and practically identical for CvsL (nominal: 0.759, adversarial training 0.757). When given systematically distorted samples, the

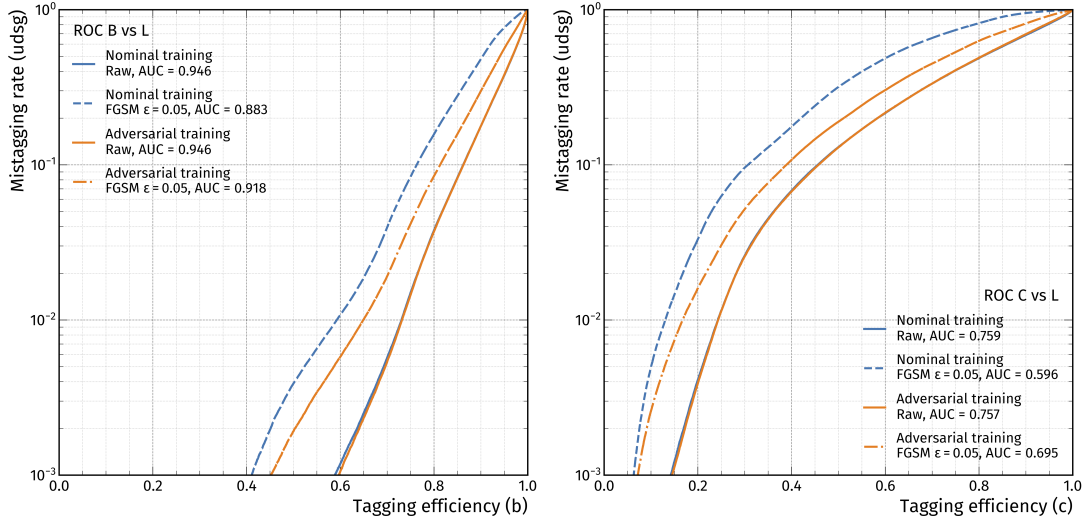


Figure 6.8: ROC curves for the BvsL (left) and CvsL (right) discriminants, comparing the nominal with adversarial training when applying the FGSM attack to both trainings individually. Nominal training is visualized in blue, adversarial training in orange, solid lines depict nominal performance, dashed lines show performance on distorted inputs (for nominal training), dashed-dotted lines represent the systematically distorted samples for adversarial training. Both figures and caption are taken from Ref. [11].

adversarial model maintains a high performance, as can be seen from the dashed-dotted lines corresponding to the colors mentioned above. For FGSM samples crafted for and injected to the adversarial training (orange dashed-dotted line), the ROC curve appears much closer to that showing nominal performance (solid line), compared to the nominal training (blue lines). This is best observed for the CvsL discriminant with nearly same nominal performance, where performance on distorted samples decreases by roughly 21% for nominal training, but only 8.2% for the adversarial training. This proves that it is possible to build a more robust jet flavour tagging algorithm that is simultaneously highly performant. No label leaking effect (see Ref. [310]) is observed, for the adversarial model, the performance on adversarial examples is not higher than performance on undisturbed data.

The susceptibility to mismodeling of the two classifiers as a function of performance is compared in Fig. 6.9, where FGSM samples have been generated individually for each model and checkpoint (denoting each epoch with a single point), scanning over different discrete stages of the training. At later stages of the training, performance improvements are only marginal, yielding a higher density of points for the high performance region, while at the beginning, gains are more pronounced. In an ideal case, the relation would be constant, showing no signs of decreasing robustness for increasing performance. The observed mapping however points at considerable deterioration (and thus higher susceptibility to mismodeling) of the nominal classifier. Adversarial training mitigates this effect to a large degree.

A recovery of some of the adversarial training robustness is observed, peaking at an AUC of around 0.938, before resistance worsens at higher performance. The intriguing trade-off between performance and robustness for the nominal training is evident. Due to high susceptibility, training to highest performance is not necessarily advisable. For adversarial training, at equally good performance on nominal samples, only a weak functional dependence between performance on first-order adversaries and the respective undisturbed performance

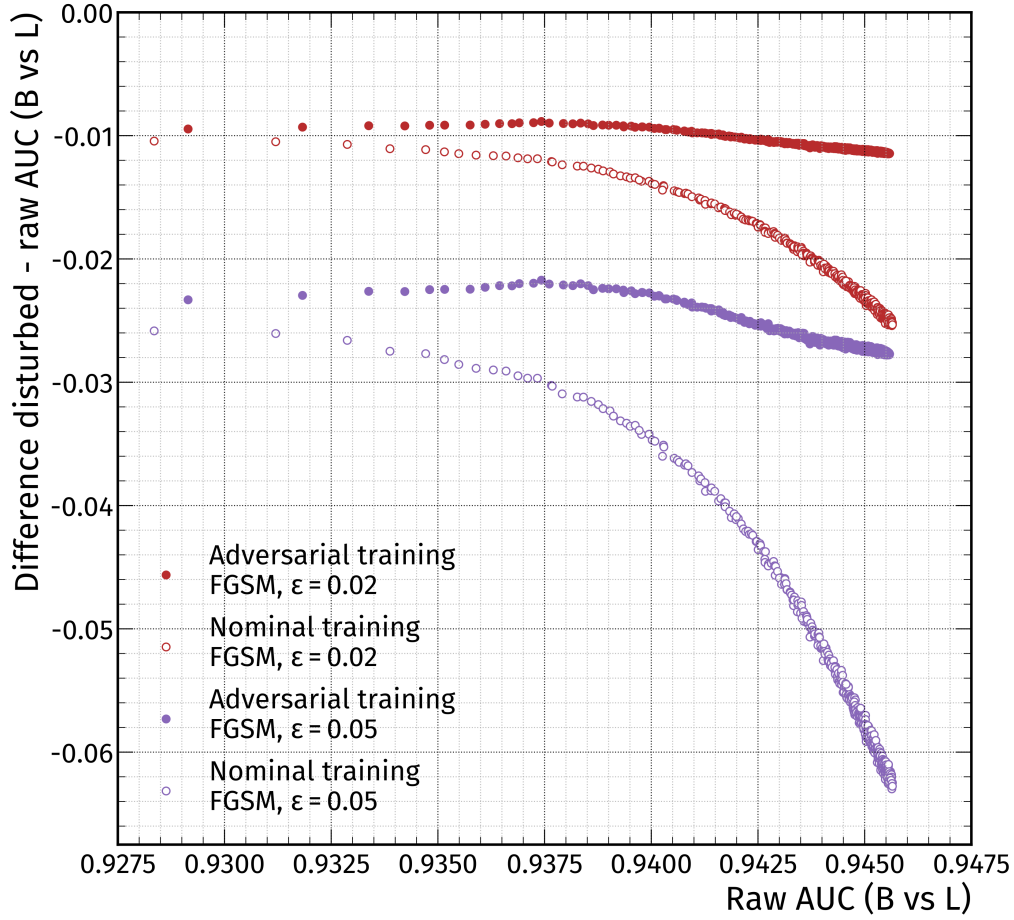


Figure 6.9: Relation between susceptibility and nominal performance for the nominal and adversarial training, tested on systematically distorted inputs with varying ϵ in different colors. The x axis shows nominal performance, measured with BvsL AUC, while the y axis shows the difference between disturbed and raw AUC. When there is a drop on the y axis while moving to higher nominal performance (x axis), this indicates higher susceptibility. The empty markers represent the nominal training, which becomes highly vulnerable with increasing nominal performance (with the drop always getting steeper), while the filled markers for adversarial training show a much flatter relation. The figure and caption are taken from Ref. [11].

is observed.

6.4.4 Probing Flavour Dependence of the Attack as a Proxy for Generalization Capability

Aiming to understand why the adversarial model is more robust than the nominal classifier, nominal and perturbed input distributions of a selected feature are investigated, split by flavour. A large distortion with ϵ of 0.1 generated via Eq. (6.2) is chosen intentionally for better visibility of geometric properties of the distorted samples, thus without the limitation described in Eq. (6.3). A feature offering discriminating power between heavy-flavour and light-flavoured jets, the signed impact parameter (d_0), is shown in Fig. 6.10. The former types of jets contain displaced tracks associated to a secondary vertex, which should naturally lead to more positive values for the d_0 variable. For the latter type, a roughly symmetric d_0 distri-

bution is expected, peaking at 0. There can be some skewness due to relatively long-lived, but light hadrons (K_s^0 or Λ) or contamination with tracks from heavy-flavour hadrons [9].

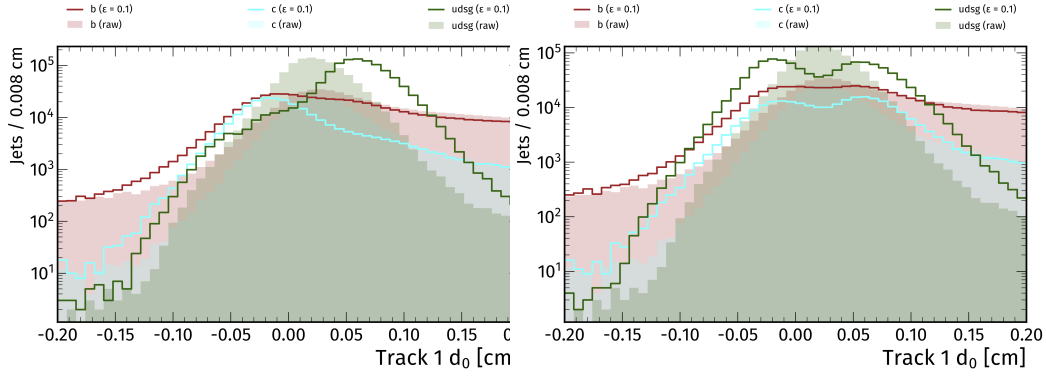


Figure 6.10: Signed transverse impact parameter distribution for the first track, split by flavour, before (filled histograms) and after (lines) applying the FGSM attack for the nominal (top) and adversarial (bottom) models, respectively. Clearly asymmetric shapes are produced when using the FGSM attack for the loss function assigned to the nominal training. Applying the FGSM attack based on an adversarial model shows suppressed flavour-dependency and relatively symmetric shapes. The attack uses the parameter $\epsilon = 0.1$, which is higher than the moderately chosen parameter of $\epsilon = 0.01$ during the modified training loop. Both figures and caption are taken from Ref. [11].

Light-flavour jets are shifted mostly into the positive region when applying the attack for nominal training, a region usually dominated by b jets; on the other hand, b jets are shifted to the negative region with low nominal abundance for that type of flavour. The distorted shapes for nominal training can be described as asymmetric and constitute an inversion of the expected physics as clearly preferred direction, whereas adversarial training shows more symmetric geometrical behaviour, becoming almost agnostic to the direction into which the FGSM attack shifts the inputs. The FGSM attack as applied for adversarial training results in shapes that could well be generated by “coin-flipping” the direction of the shifts, having trouble to decide for a clearly worse direction. This can be interpreted as a less susceptible model, even for noticeably large distortions, compared with nominal training.

Geometric properties of the (attacked) input distributions can conceivably be related to the geometry of the loss surface [306–309], which can be assumed to contribute to differences in robustness. Flatness of the loss surface in the vicinity of raw inputs, as demonstrated in Fig. 6.11, may influence symmetric or asymmetric shifts.

The minimum of the loss surface, into which the nominal training converges, is associated with the default distributions. For any flavour, there is a specific vector with fixed direction pointing away from that local minimum via the steepest increase in loss. On the other hand, adversarial training always “sees” (new) adversarial inputs, eventually yielding averaged out adjustment of the model’s parameters over further training epochs. Adversarial training always follows newly distorted inputs such that a locally constant loss manifold around the original inputs is a likely realization of the more complex saddle point problem including inner and outer optimization. It is conceivable that not the exact memorization of training data, but rather higher-order correlations play a role for high performance of adversarial training [291, 293, 306, 307]. This relates the flat loss surface around the nominal inputs to the

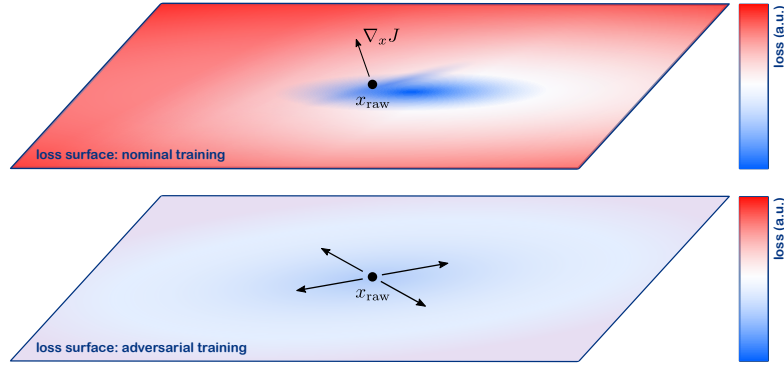


Figure 6.11: Illustration of the potential geometry of the loss surfaces for the nominal as well as the adversarial training. The figures and caption are taken from Refs. [11,233], inspired by Refs. [306–308].

absence of a preferred direction for first-order adversarial attacks crafted for the adversarial model. The seemingly rotational invariant behaviour fits with the observation that many vectors would fulfill the criterion of pointing in the direction of increasing loss, such that direction could also be chosen randomly.

As has been shown with the geometric properties of adversarial samples, a flat loss landscape for the adversarial model is highly probable, which explains higher robustness [306,307]. Adversarial training then generalizes better to on-manifold mismodelings of order ϵ , when compared with nominal training. This statement is only valid for cases where FGSM replicates mismodelings between data and simulation and can not be extended directly towards any type of modeling effect, given that robustness and generalization are not equivalent [291,293,311]. With the present simulated samples, further scenarios are investigated, while the inclusion of detector data (see Chapter 7) needs to be consulted to probe physics use cases.

6.5 Robustness in the Context of Other Mismodeling Scenarios

6.5.1 Smearing Inputs with a Gaussian Noise Term

The FGSM attack is targeted at increasing gradients of the loss function, however, there can be physical effects related to PS modeling or detector calibration which are completely independent of the loss surface. Such effects, unlike a “demon” [14] that always points in a preferred direction, are replicated with a smearing technique instead. This covers another class of disturbances, independent of the model under consideration, but which are more of statistical nature, i.e. non-targeted fluctuations. Slightly distorted inputs are generated randomly by shifting the values with a Gaussian noise term ξ [12,13,247,310]:

$$x_{\text{noise}} = x_{\text{raw}} + \xi, \quad \text{where} \quad (6.4)$$

$$\xi \sim \mathcal{N}(\mu, \sigma^2) \quad \text{with} \quad P(\xi) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{\xi-\mu}{\sigma}\right)^2}. \quad (6.5)$$

This smearing can directly be applied to the inputs, as these have been scaled to a standard deviation of one, with a mean of zero, as described in Sect. 6.2.2.

Figure 6.12 shows an illustration of how this distortion looks like for one arbitrary input x_i .

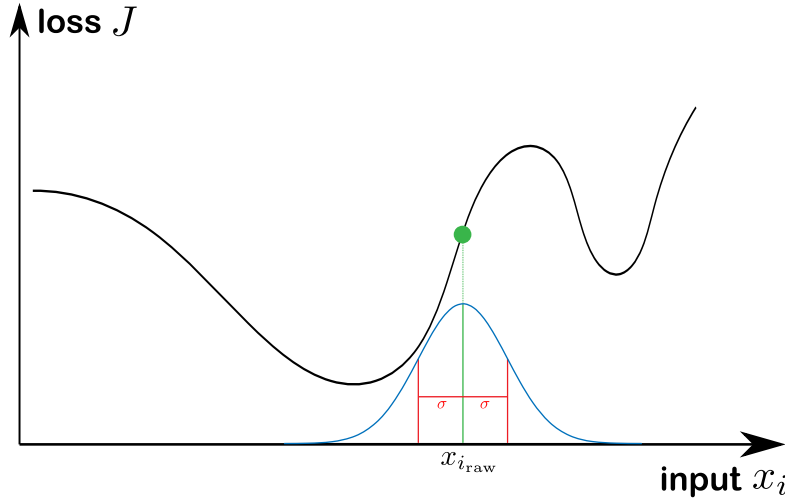


Figure 6.12: Visualization of the random shift of inputs by adding a Gaussian noise term. With the slight distortion based on the blue probability distribution, the formerly green raw datapoint is shifted and the corresponding loss modified. The change of the loss function with respect to the distorted inputs can go in either direction. The figure and caption is taken from Ref. [233], Gaussian distribution adapted from Ref. [285].

The magnitude of the distortion is given by $\sigma = 1$, leaving other parameters introduced along with the FGSM attack untouched; also the limitation by 25% of the value is applied and the random fluctuations are spread around $\mu = 0$. Also when tested on randomly smeared inputs

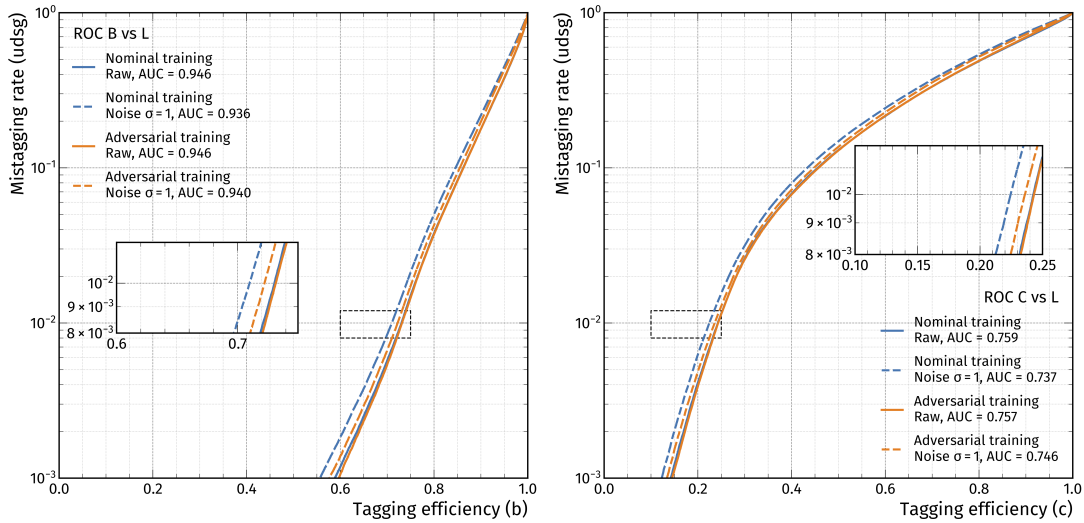


Figure 6.13: ROC curves for the BvsL (left) and CvsL (right) discriminants, comparing the nominal with adversarial training when smearing the inputs with a Gaussian noise term. Nominal training is visualized in blue, adversarial training in orange, solid lines depict nominal performance, dashed lines show performance on distorted inputs. Both figures and caption are taken from Ref. [11].

(see Fig. 6.13), the adversarial model performs better than the nominal model, difference in AUC to measure susceptibility is smaller by a factor of 2. Hence, also in this scenario replicating typical mismodelings found in the HEP context, the adversarial training is more robust.

6.5.2 Transferability of Adversarial Samples as a Black-Box Attack

Also when transferring adversarial samples created for one model to another, independent model, performance can be deteriorated, known as transferability of adversarial samples [13, 291, 293, 312]. Choosing models with same architecture ensures comparability, the only difference is the applied training strategy (nominal or adversarial) which yields a different set of model parameters. As an example, FGSM inputs crafted for nominal training can be injected to both models considered in this chapter, such that a black-box attack is applied to the adversarial model [293] (without taking the exact parameters of the adversarial model into account). This corresponds to creating adversarial samples as shown with the blue branch in Fig. 6.4, but giving these also to the adversarial (orange) model. The setup is similar to previous investigations, with $\epsilon = 0.05$ and the limitation introduced in Eq. (6.3). Also for this perturbation, the adversarial model is more robust, as revealed in Fig. 6.14.

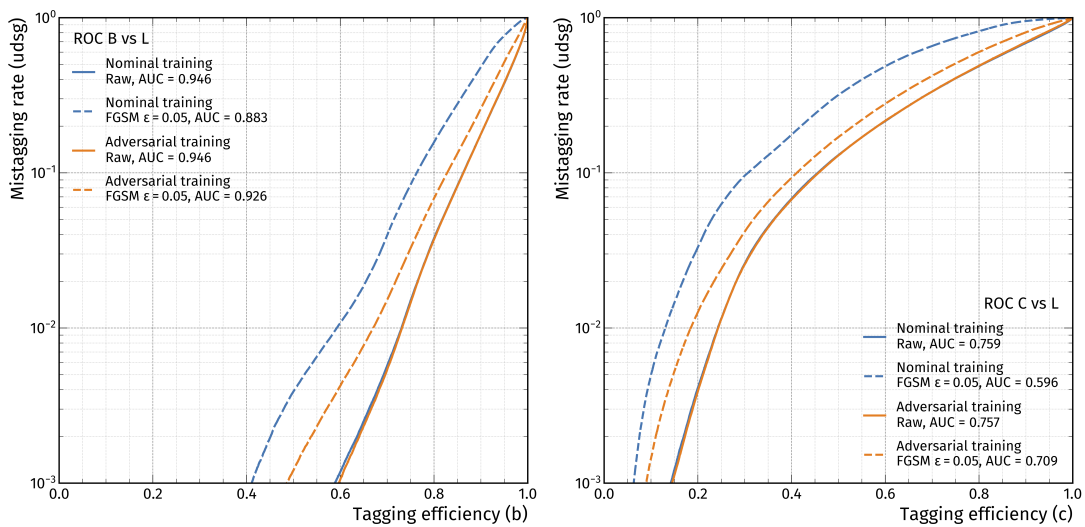


Figure 6.14: ROC curves for the BvsL (left) and CvsL (right) discriminants, comparing the nominal with adversarial training when applying the FGSM attack to the nominal training and injecting the obtained inputs to both models. Nominal training is visualized in blue, adversarial training in orange, solid lines depict nominal performance, dashed lines show performance on distorted inputs that were obtained with the help of the loss surface of the nominal model. Both figures and caption are taken from Ref. [11].

6.5.3 Shifting Inputs Systematically with Up/Down Variations

For this scenario, input features are shifted upwards (or downwards) without prior knowledge of the model parameters, by simultaneously adding (subtracting) small distortions to (from) the nominal values. The dataset does not contain systematic uncertainties, therefore, they are estimated from existing commissioning results that were published earlier by the CMS and ATLAS collaborations [9, 10, 313, 314]. The smallest baseline chosen as 0.05 is weighted by a factor s_i ranging from 1 to 5, taking the maximally observed data-to-simulation disagreement for input i into account:

$$x_{\text{sys},i} = x_{\text{raw},i} \pm 0.05 \cdot s_i, \quad \text{where } s_i \in \{1, \dots, 5\}. \quad (6.6)$$

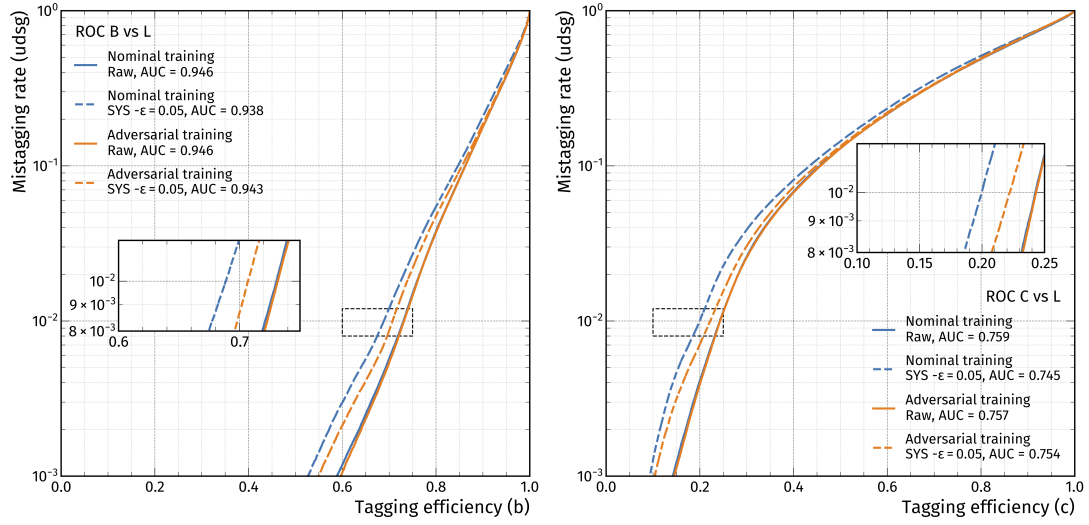


Figure 6.15: ROC curves for the BvsL (left) and CvsL (right) discriminants, comparing the nominal with adversarial training when shifting inputs systematically downwards. Nominal training is visualized in blue, adversarial training in orange, solid lines depict nominal performance, dashed lines show performance on distorted inputs. Both figures and caption are taken from Ref. [11].

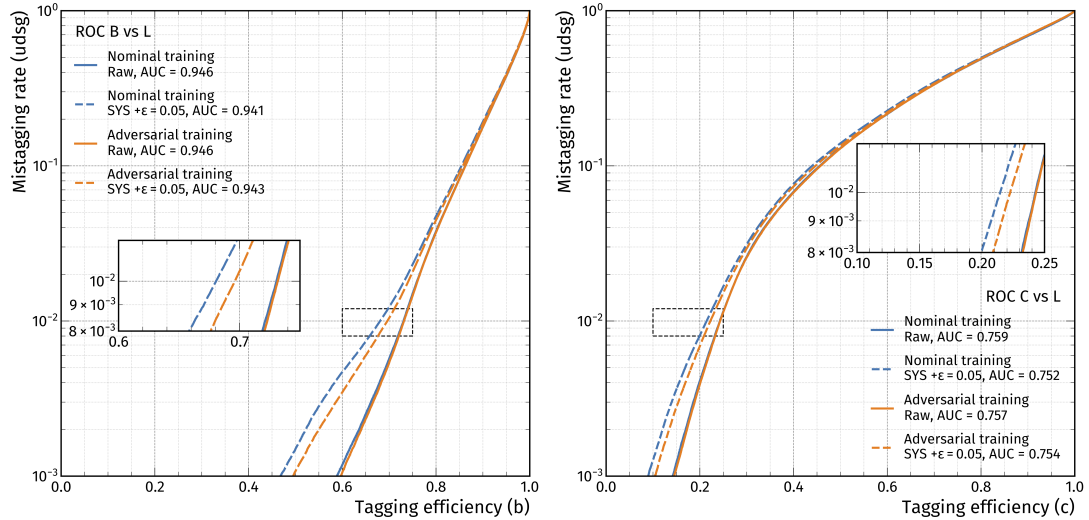


Figure 6.16: ROC curves for the BvsL (left) and CvsL (right) discriminants, comparing the nominal with adversarial training when shifting inputs systematically upwards. Nominal training is visualized in blue, adversarial training in orange, solid lines depict nominal performance, dashed lines show performance on distorted inputs. Both figures and caption are taken from Ref. [11].

The initial factor of 1 is incremented s_i -times in steps of 1 to account for the largest deviation in the data-to-simulation ratio, counting how many intervals of 0.1 fit between the observed ratio and perfect agreement (i.e. a ratio of 1). As this constitutes a limitation by construction, the additional 25% restriction is not necessary here. Not all features of this dataset have a direct counterpart for officially commissioned algorithms, such that reasonable intermediate

factors are assumed in Eq. (6.6).

Also in the case of simultaneous up- or downwards variations, higher resilience is achieved with the adversarial model (see Figs. 6.15 and 6.16). The up/down variations are not specifically crafted for each model, and do not have as large an impact as specific adversarial attacks. Correlations, as partially treated with the sign of the gradient of the loss function for FGSM, are not taken into account for this perturbation. Hence, the advantage of adversarial training over nominal training is not as pronounced in this scenario that does not model the worst-case. Yet again, for all discriminants and WPs considered, the adversarial training is more robust towards such systematical shifts of input features.

6.6 Loss Surface Study

The geometric properties of the underlying loss surface serve as a proxy for robustness or generalization capabilities of the NN in question. The approach followed for this study differs from classical loss surface investigations [308] that scan the loss as a function of model parameters (weights and bias terms). Instead, as the main difference between the nominal and adversarial training is whether the training samples are distorted or not, the input features serve as variable axes when probing the loss function dependencies.

6.6.1 Properties of Loss Manifolds for a Jet Tagging Algorithm Trained on Nominal or Adversarial Samples

Observations have shown that adversarial training behaves somewhat symmetrically when exposed to adversarial samples, with nominal training yielding asymmetrical shapes (see Fig. 6.10). With Fig. 6.11, a possible explanation can be gleaned with the help of the underlying loss surface. This hypothesis shall be investigated now with realistic scans of such surfaces for the two models under consideration.

The first visualization of that kind is presented in Fig. 6.17, constructed from a random jet drawn from a sample which has not been used for training or validation (i.e. from the test set). Two well-understood global jet features, transverse momentum and pseudorapidity, are uniformly binned into a grid of 500×500 variations around the central nominal value. Taking the full distribution as reference, the investigated range covers $\pm 0.5\sigma$ by performing the binning in the coordinates after standardization. The target is unchanged, and the associated loss function reevaluated for adversarial and nominal training on the resulting 250000 samples.

For adversarial training, keeping transverse momentum fixed and varying only pseudorapidity does not change the network prediction error or loss. Nominal training is strongly dependent on input modifications in any of the two directions spanned by the pseudorapidity and transverse momentum axes, but the overall network prediction error is lower than that for adversarial training. The difference related to susceptibility can be seen as manifestation of a flatter manifold for adversarial training, yielding the previously shown invariance with respect to distortions of specific features.

A clear direction for adversarial attacks would be found for nominal training, as can be understood from Fig. 6.18, but less so for adversarial training, the chosen direction is one of various choices, of which none are preferred essentially. Besides this finding, the shown sur-

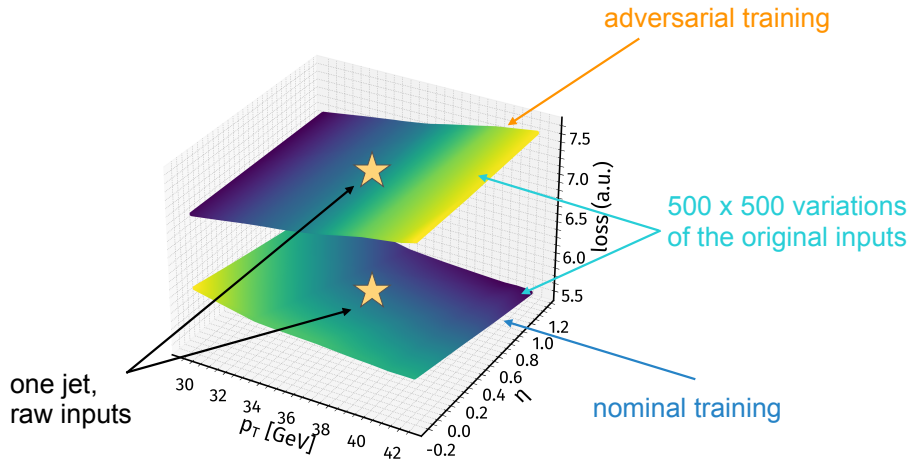


Figure 6.17: Different geometries of loss manifolds for nominal (bottom) and adversarial (top) training. The figure and caption is taken from Ref. [16].

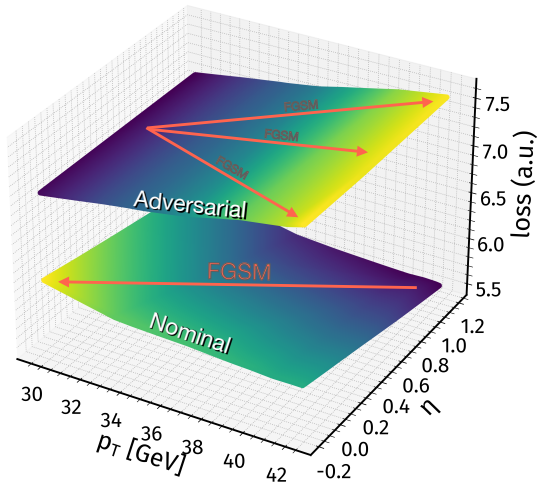


Figure 6.18: Possible directions of adversarial attacks for different models. Starting from kinematic quantities which yield small loss, multiple arrows can be found for an FGSM attack imposed for adversarial training, while only one such arrow is constructed for nominal training. The figure and caption is taken from Ref. [16].

faces can be modeled with planes oriented in the space spanned by the two axes, requiring only a linear model instead of more complex ones. This is in agreement with the observation that first-order attacks are a reasonable choice for such jet flavour tagging algorithms.

6.6.2 Discussion

The study of the loss surfaces associated with the two models introduced earlier helps to understand why the models behave the way they do when exposed to systematic mismodelings. Should experimental effects and limitations for precision and resolution of the input features be of linear kind or their properties be of Gaussian nature, adversarial training is likely to perform better as well, being trained on augmented simulation as a very specific regularization technique to transfer to the detector data domain. Besides this, the loss surface study also allows deriving further improvements of the adversarial training technique itself.

6.6.2.1 Using Properties of the Loss Manifold during Training

The results related to flatness of the loss manifold can be incorporated into the training. Viewing from the macro-scale shown above may not be sufficient, and calculations should take all features into account (thus preventing their visualization). With the many-dimensional loss surface, which is only limited by computing resources, also other variables with more discriminating power can be taken into account, not only those that are reweighted to common target distribution. Measuring flatness, or rather, measuring curvature of the surface [315] around the original inputs, can serve as a metric to study robustness of NN models. Curvature as a summary quantity is a suitable addition as a penalty term in the loss function during training. This can be extended for a mesh grid of points within an allowed $B_{\frac{\epsilon}{2}}$ -ball. By introducing another hyperparameter as a prefactor for this term, it can be controlled how much focus shall be put on flattening the loss manifold and how impactful this regularization shall be. During (adversarial) training, this modified loss function would then replace the standard CCE loss or focal loss, as studied previously in this chapter.

6.6.2.2 Building other Attacks which Preserve Directionality of the Gradient of the Loss Function

First-order attacks appear sufficient for scenarios investigated so far, but taking the sign of the gradient (FGSM) may be inefficient when the relative contributions of features, thus the exact direction of the gradient matters. Such individual contributions can be maintained quantitatively in the attack by moving to another p -norm of gradients, e.g. choosing $p = 2$ instead of the infinity-norm. A dedicated implementation of such an attack will be introduced in Sec. 7.3.2 as the so-called Normed Gradient Method. To keep disturbances small, the resulting distortion vector can be scaled by the inverse of the aforementioned norm, which allows comparisons across different jet samples. Neither direction nor magnitude are easily predictable for such an attack, which is in direct contrast to FGSM, where only $\pm\epsilon$ shifts are possible. Also Projected Gradient Descent (PGD) [293,310,316] only offers the required flexibility when introducing a norm different from that used in standard FGSM attacks, which serves as the building block of PGD for every iteration. The modified attack then includes correlations between features in quantitative manner, which was a shortcoming of the FGSM attack typically mentioned in the HEP context. Large distortions are not needed, because to introduce a performance deterioration for the given loss surface, a better direction is chosen, compared to what would be possible with FGSM, making them less noticeable with typical validation methods such as one-dimensional histograms.

6.7 Conclusion

The performance of a jet flavour tagging algorithm, exposed to systematically distorted inputs that have been generated with an adversarial attack, the Fast Gradient Sign Method, was investigated. The relation between model performance and robustness was shown and the trade-off between performance on unperturbed and on distorted test samples explored, investigating ROC curves and AUC scores for the BvsL and CvsL discriminants.

Earlier findings relating higher performance with higher susceptibility for nominal training

strategies are confirmed, with a deep neural network that replicates a typical jet tagging algorithm. A defense strategy to counter first-order adversarial attacks was applied by injecting adversarial samples already during the training stage of the classifier, not altering the network architecture. Compared to a previous study [233], the dataset properties were altered in size, feature dimension and event simulation technique. Also, a physics-motivated restriction was introduced for the adversarial attack, the resilience was probed with systematic uncorrelated up-/down shifts of features as well, and the loss surfaces associated with the two models were scanned for the first time.

No difference in performance between the nominal model and the new adversarial classifier was observed, while robustness towards adversarial attacks is enhanced by a large margin. When discriminating c from light jets, both trainings reached an AUC score of approximately 76%, and an FGSM attack with moderate impact on input distributions decreases the performance of the nominal training by 21%, and only by 8.2% for the adversarial training. With a study of nominal and distorted input distributions, geometric properties of the attack were related with geometric properties of the underlying loss surfaces for a nominal and an adversarially trained model. The loss manifold itself was studied with respect to input features, showing differences in flatness or curvature. This was further investigated as the cause of invariance to small distortions when probing the adversarial model. More advanced training strategies were derived that utilize this new type of information directly, of which one is especially targeting applications in HEP where correlations between variables matter and distortions of variables can not be arbitrarily large.

While higher robustness can be partially related to improved generalization capability, detector data needs to be utilized to explore this conjecture. Further, another interesting field for exploration is the application of this attack and defense strategy to more complex neural network structures, involving for example convolutional layers, or working with larger (or smaller) dimension in the feature space, which affects the effectivity of adversarial training. Also, more complex attacks such as one of those introduced in the discussion part can be probed in new contexts. With the approach followed for this work, only the model parameters and criterion are necessary to craft suitable adversarial samples, making it a comparatively generally applicable strategy. This constitutes the first successful application of adversarial training to build a robust jet flavour tagger suitable for usage at the LHC, resisting first-order adversarial attacks and a variety of other mismodeling scenarios.

Probing Adversarial Techniques with Detector Data: Novel Jet Tagging Algorithms for the CMS Experiment

“It doesn’t matter how beautiful your theory is, it doesn’t matter how smart you are. If it doesn’t agree with experiment, it’s wrong.”

– Richard P. Feynman

All studies presented so far in this thesis have been conducted with simulated samples only, stemming from fast detector simulation for which no comparisons with real recorded data are possible [11, 16]. Focusing on just the impact of artificially introduced input modifications in a limited setting does not yield conclusive proof that adversarial techniques could be truly beneficial in a realistic use case. Thus, many of the tests are repeated with simulated samples produced for the CMS experiment [19, 22]. These are then enriched with detector data obtained during the LHC Run 2, for which performance comparisons with standard algorithms are available. The availability of this bigger picture already during algorithm development stage, enabled by novel evaluation software, started a new endeavour, creating and deploying the first robust flavour tagging algorithm in the official reconstruction chain for CMS. This chapter presents a step-by-step description of different generations of algorithms and newly introduced adversarial attacks, defenses and performance metrics. Novel proposals based on the obtained results are featured as well, along with an overview of parallel studies being conducted. Before introducing the individual architectures and related developments, the existing and required new software tools are reviewed. The challenges and proposed solutions become clearer with an overview of the available software infrastructure.

7.1 Technical Aspects Enabling Robustness- and Data/MC-Studies with Custom Algorithms and Inputs

Usually, when developing new tagging algorithms to use them in an analysis, the approach is to look at performance in simulation with one data tier, and decide based on that, which model (at which point of the training) to include. Then, the official integration into the experiment software is performed, and later on, further figures of merit with the help of detector data with derived data tiers are derived. See Refs. [160, 195, 196, 317–325] for the respective public plugins, data tiers and frameworks for two typical LHC experiments, on which the software review is based. If there are no choices to be made, other than a selection based on, for example, the lowest validation loss for a given architecture, or the best AUC score for a

given classification task, this linear framework design with a clear split between training and performance evaluation is sufficient.

The goal when applying all attempts to improve algorithm robustness to typical NNs used at a LHC experiment however also involves a quickly available check of the impact on agreement between data and simulation. As every model trained in this context will eventually also be used for inference with collision data, and as there are inherent differences to be considered between nominal or adversarial models, the entire software pipeline needs to accommodate for that and enable direct studies of data/MC agreement while still finalizing the algorithm development / training process. If the utilized data/MC-framework required official integration (see the left part of Fig. 7.1), that choice which exact model to include would have already been made, leaving no room to test multiple available options or input variations. The fixed route in the old routine produces various data tiers, but at each point, with one selected set of NN scores or discriminants only. The solution for that is to modify the evaluation scripts such that they can be called by the ML developer while training the network in question. This new variant (right part of Fig. 7.1) therefore saves the integration and (re-)derivation of data tiers that are inherent in the established pipeline. With the help of this new full framework capable of producing a set of data/MC distributions in multiple phase spaces and from that interfacing to existing scale factor tools, the following chapters contain various examples of performance evaluation in data with models that are not part of the official reconstruction.

More details on the implementation can be found in the appendix, in Chapter D.

7.2 The First Robust DeepJet Algorithm

As a first example in the CMS context, the DeepJet algorithm [18] will be investigated. At the time of writing, this is a widely used algorithm for small-radius jets that significantly improved the performance with respect to its predecessor, DeepCSV, for Run 2 analyses (see also Sec. 4.2.4). It is therefore the logical next step to jump from purely feed-forward neural networks (just linear layers stacked on top of each other) as studied in the previous chapter to this architecture with its foundations in computer vision. This section, including Sec. 7.2.3, is largely based on results presented in Ref. [19].

7.2.1 The Nominal Algorithm and its Adversarially Trained Companion

The architecture used for AI safety studies with the DeepJet algorithm is identical to the latest version used for CMS Run 2 and for the first Run 3 re-training, to ensure comparability with the official configuration. Its schematic overview is presented in Fig. 7.2, with a detailed description of the input features and targets given in the appendix, Section B.2.2, especially in Tables B.7, B.8, B.9, B.10, B.11, and B.4 where the used physics processes and their application for training and validation are summarized as well.

DeepJet in its current state groups jet or jet constituent related input features into four distinct sets that are each processed individually, before their respective input processes are combined. Using up to 25 charged (yellow in Fig. 7.2) and neutral (blue) PF candidates each (i.e. CPFs and NPFs), and up to 4 SVs (green), DeepJet takes a large fraction of its inputs from low-level quantities that are only minimally pre-processed. CPFs are sorted by one of the criteria

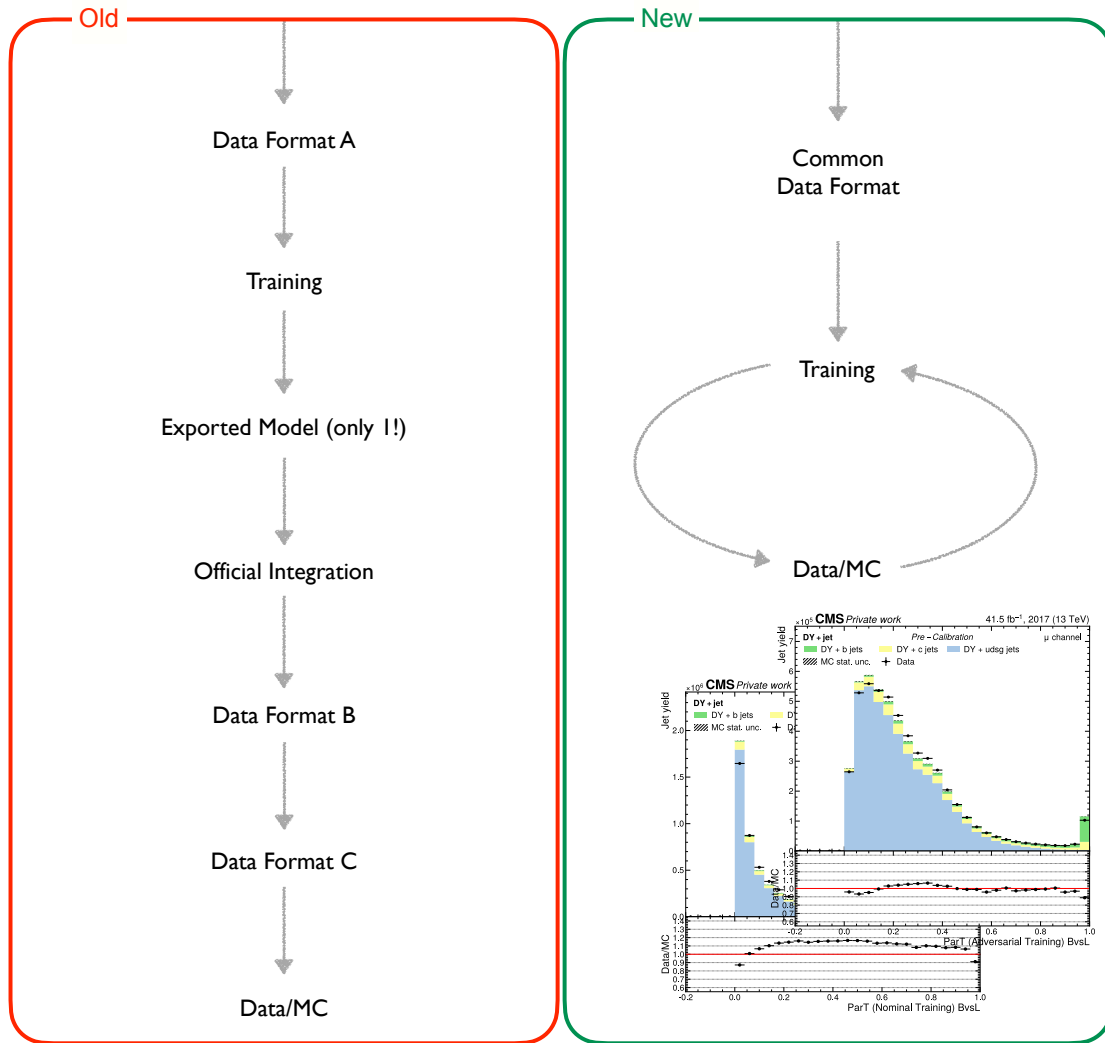


Figure 7.1: Comparison of existing (left) and new (right) pipelines incorporating NN training for jet tagging algorithms and evaluation with data. The former method involves multiple variants of data on a closed route that allows no algorithm tweaks based on validation (as all steps are done once, for a fixed model), and the latter is a continuous loop that can steer the training with information gleaned from data/MC studies.

SIP 2D significance, minimum pseudoangular distance to any SV, or transverse momentum of the candidate divided by the transverse momentum of the jet (decreasing priority of criteria), for NPFs, only the latter two criteria are taken into account, as NPFs, unlike CPFs, are not built from tracks for which SIPs can be computed. Sorting of SVs is done via flight distance significance in the x - y -plane (i.e. 2D flight distance significance). [18, 160] For each existing candidate, there may be 16 (6, 12) features for charged candidates (neutral candidates, SVs). Zero-padding is employed for jets that have less than these numbers of objects (constituents or SVs). Global variables (orange) are only fed through linear layers as inputs to a dense network, marking the high-level content of DeepJet. Features that could not be calculated

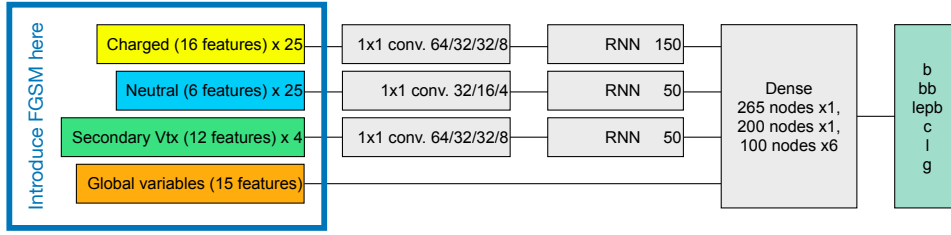


Figure 7.2: The DeepJet architecture in Run 2 configuration and schematic representation of FGSM input modifications. Previously shown in Refs. [18] (original) and [19] (this modified version).

with enough precision or where quality criteria were not met, but the constituent itself exists, receive special placeholders or defaults, mostly at -999 , -1 , or 0 .

A batch size of 4000 is used for nominal and adversarial training. The Ranger class of Ref. [326], implementing Lookahead [327] and RAdam [328] with initial learning rate of 0.005 is used to optimize the one-hot-encoded CCE loss function while performing backpropagation. The nominal training is run for 30 regular epochs, 30 % of those 30, i.e. 9 epochs are done with a learning rate decay following the relation

$$\text{new_lr} = \text{old_lr} \cdot 0.01^{\frac{1}{30}}, \quad (7.1)$$

where new_lr refers to the new learning rate, while old_lr was the previous epoch's learning rate. For adversarial training, the number of epochs was doubled, using the same approach for scheduling. This is in line with findings of Chapter 6, where adversarial training needs more epochs to show similar convergence behaviour as nominal training. The best checkpoints in terms of smallest validation loss are used for nominal and adversarial training respectively when carrying out performance evaluation. During adversarial training, the parameter controlling severity of distortions is fixed to $\epsilon = 0.01$.

For DeepJet training and inference, some inputs are pre-processed with linear shifts. These linear shifts give rise to features for which a different ordering between physical values and defaulted values is more natural or can better capture the property of a feature when separating different classes with ML algorithms. For the example of pseudoangular distances between neutral candidates and the jet, the physical distributions fall steeply towards higher magnitudes, while non-existing values are placed at 0. Thus, shifting such a feature into the negative region ensures that values with smaller likelihood are closer to the non-existing or defaulted values. Numerical infinities, including values with absolute value larger than 10^{32} , are replaced with placeholder values (usually 0) [196, 320].



How input normalization, defaults and individual distortion parameters are treated in different algorithms and frameworks

In DeepJet, all inputs are processed with a Batch Normalization [273] layer (with parameters $\text{eps} = 0.001$, $\text{momentum} = 0.6$), the same is true for all intermediate layers of the network. The setup is therefore different with respect to the custom pre-processing and

training pipeline employed for the JET FLAVOR DATASET of Chap. 6, where no Batch Normalization layers are used, but all normalization is statically done once before starting the training. DeepJet thus accepts non-standardized input features and handles them internally.

In all studies of robustness of flavour tagging algorithms for CMS, default values and integers are excluded from any manipulations, and suitable multiplicative factors have been derived for every feature to account for their different numerical values (their mean and standard deviation can vary by several orders of magnitude, and all shifts limited by ϵ should have comparable impact). This is necessary because DeepJet(Core) does not pre-compute the scaled quantities. Scaling while training just to derive ϵ for the adversarial attack would take long and is sensitive to the statistical properties of the currently selected batch. Instead, other techniques are employed for the CMS studies of adversarial robustness where the magnitudes of the attack are derived a priori for each feature, in line with the already employed pre-processing of the features. A simple approach is to calculate, for each feature, the minimum and maximum value from a representative sample, e.g. the training set, and compare the spanned interval with a reference interval. A suitable reference is $[-1, 1]$, with its length 2, just using a heuristic argument related to the Gaussian distribution with $\sigma = 1, \mu = 0$. An FGSM attack with general $\epsilon = 0.01$ would then be scaled by the ratio of feature-min-max-interval-width and reference, to ensure similar impact for each variable. In practice, taking a hypothesized transverse momentum spectrum from $p_T = 20 \text{ GeV}$ to 1000 GeV as an example, this feature's individual ϵ is scaled by $\frac{1000 \text{ GeV} - 20 \text{ GeV}}{2 \text{ GeV}} = 490$. However, the shape of each input distribution and potential outliers affecting this procedure are not taken into account. Removal of outliers can be achieved by selecting only a central quantile of each distribution, for example rejecting the lower and upper 0.5 %. This parameter itself is variable, better results may be achieved with other two-sided levels of 1 % for features with larger fractions of outliers. This is the setup that was followed for robustness studies of DeepJet. It shall be noted already here that an improved multiplication technique that will be used for the successor, RobustParticleTransformer (RobustParT), is also based on a quantile, but a stricter one (see Sec. 7.3). There, only the central 60 % are kept from which the standard deviation is computed, which is then taken as the multiplicative factor in front of each feature's custom ϵ . This is closer to what is done for the custom tagging pipeline of Chapter 6, where all features are standardized with the help of standard deviation and mean prior to any training or inference step. Especially when increasing the number of inputs, or the number of candidates, this does make a difference with respect to the simple approach and keeps the attack at reasonable levels for complex input structures. More details on RobustParT, for which this technique is introduced for the first time, are described in later sections. The author's explorative analysis and derivation of suitable parameters as explained above lay the groundwork for robust Run 3 tagging algorithm developments. Given its universality, standardization of ϵ parameters via quantiles for adversarial attacks can also be applied to any other algorithm that employs Batch Normalization, for any input variable with distinct values.

7.2.2 Performance of DeepJet on Simulated Samples with Nominal or Adversarial Training and Exposing the Models to Adversarial Attacks

As for the model investigated in Ref. [11], performance of the DeepJet algorithm using nominal or adversarial training will be studied on simulated samples first. With simulation, truth flavour obtained from the generator level can be accessed, which also allows crafting dedicated adversarial attacks as a tool to probe robustness of the respective models. In order to do this in a precise manner, the six “Class” attributes as defined in Table B.7 are used to tell the attack which prediction shall be disturbed in which direction, thus taking six loss surfaces into account. As for the first study carried out in Chapter 6, the distortion is limited to not introduce shifts exceeding a defined percentage of the nominal feature value. For this purpose, the configuration for DeepJet uses 20 %. Jets exceeding a transverse momentum of 20 GeV and within the tracker acceptance, here within $|\eta| < 2.5$, are selected. ROC curves for the different tasks of b- or c-tagging are attached in Figs. 7.3, 7.4, 7.5. Discriminants for DeepJet (and later, for RobustParT) are built from the six outputs by defining B as $P(b) + P(bb) + P(lepb)$, C as $P(c)$, and L as $P(uds) + P(g)$, such that BvsL, CvsB, and CvsL are computed as likelihood ratios of type XvsY in Eq. (6.1). Starting with separating b from light jets as displayed

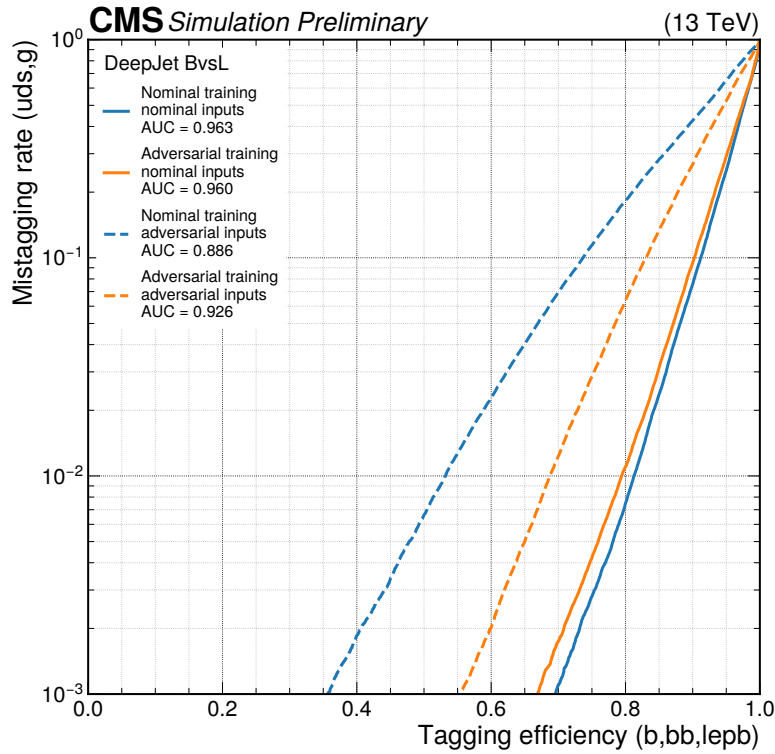


Figure 7.3: ROC curves for the DeepJet BvsL discriminant comparing the nominal (blue) with adversarial (orange) training with and without adversarial attack. All solid lines represent performance on nominal, thus the original samples, while dashed lines show the performance on systematically shifted inputs. Figure taken from Ref. [19].

in Fig. 7.3, it is evident how adversarial training does come close to the performance of a nominal training (in terms of overall AUC score, 0.960 and 0.963, respectively), but it does not quite reach the same efficiencies at fixed mistagging rates (for example, at 1% mistagging rate, efficiency for nominal training is 81%, while it only reaches 79% for adversarial training).

For charm-tagging, the pattern is similar and visible in Figs. 7.4 and 7.5, where the margin between nominal and adversarial training is slightly larger than for b-tagging. In terms of

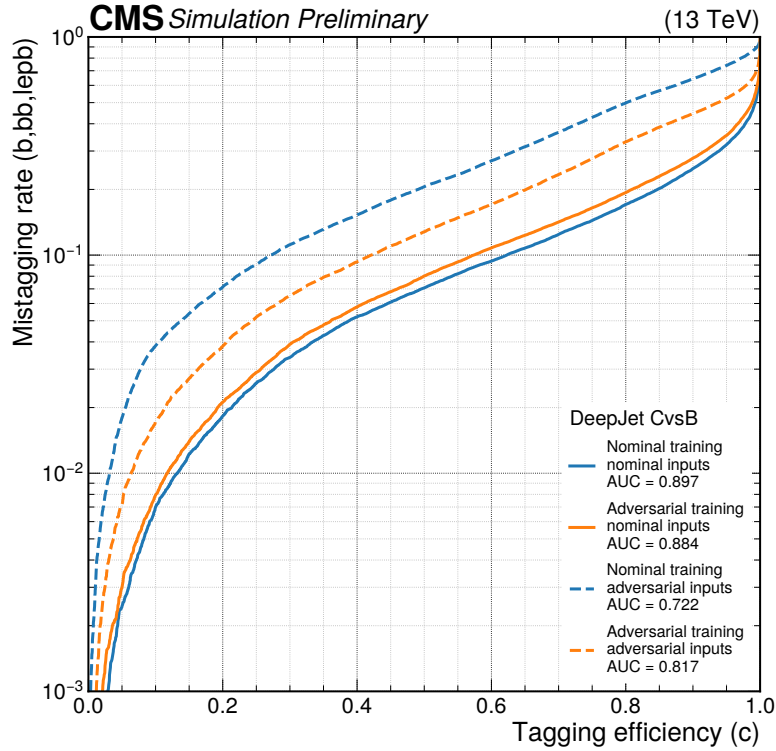


Figure 7.4: ROC curves for the DeepJet CvsB discriminant, comparing the nominal (blue) with adversarial (orange) training with and without adversarial attack. All solid lines represent performance on nominal, thus the original samples, while dashed lines show the performance on systematically shifted inputs. Figure taken from Ref. [19].

susceptibility towards adversarial attacks, the FGSM attack with $\epsilon = 0.01$ does have a drastic impact for both variants, but the adversarial training remains better in all cases. The most affected task is separation between c and light jets, where the attack does introduce such a large confusion for the nominal training that the resulting performance metrics resemble those of a random guess (with AUC close to 0.5). All results are qualitatively comparable to what has been shown for a significantly less complex network in Chapter 6. However, with about triple the number of inputs and more involved feature extraction techniques in a model with more trainable parameters, the adversarial attacks have a larger influence on the performance. This is what can be expected from overall adversarial impact on the NN activation being proportional to the number of inputs, as discussed in Ref. [13].

7.2.3 Performance Comparison of Nominal and Adversarial Training in Three Control Regions

Three regions are picked to replicate the evaluation carried out in Refs. [17, 240], but with the *robust* DeepJet algorithm, and using a slightly simplified set of physics processes to only target the muon channel [19] and focusing on the year 2017 only. The physics selections are essentially what was done for Refs. [17, 329], re-written and performed with the custom vectorized framework for arrays as introduced in Sec. D to include the ML inference and adversarial at-

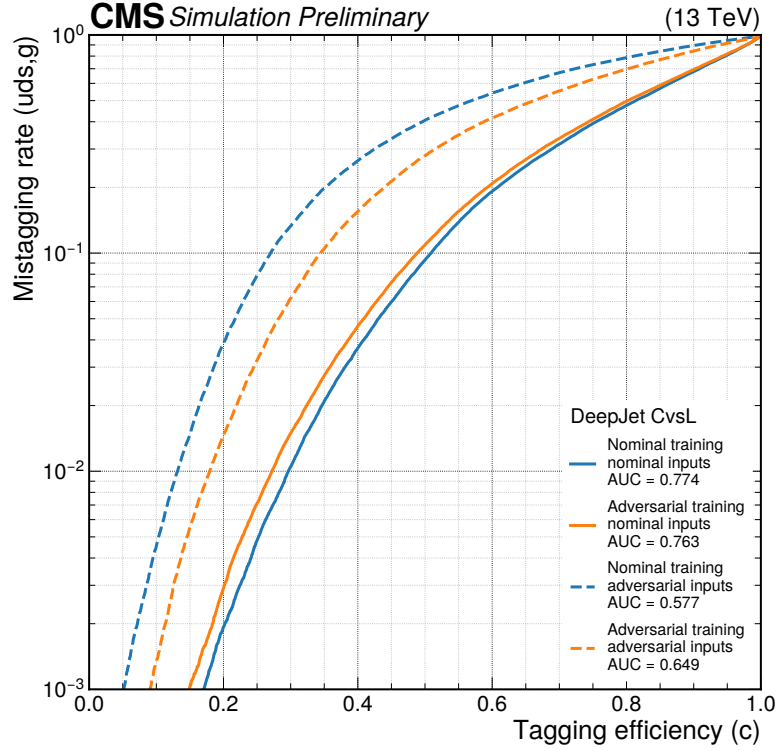


Figure 7.5: ROC curves for the DeepJet CvsL discriminant, comparing the nominal (blue) with adversarial (orange) training with and without adversarial attack. All solid lines represent performance on nominal, thus the original samples, while dashed lines show the performance on systematically shifted inputs. Figure taken from Ref. [19].

tacks / defenses on-the-fly, and then adapted to run on legacy samples similar to what is done for Refs. [239, 240]. For all regions, a general pre-selection is applied. These are quality cuts ensuring well-reconstructed physics objects are selected for subsequent use in the individual phase space regions. Jets need to have a transverse momentum of at least 20 GeV, lie within the tracker acceptance (which is performed by imposing $|\eta| < 2.5$ for the chosen year 2017), and pass tight criteria to reject leptons misidentified as jets [207, 330]. Soft muons are such muons that do not exceed a transverse momentum of 25 GeV and for which $|\eta| < 2.4$. Additionally, they are required to pass muon identification criteria and the variable governing lepton isolation [152, 331] must surpass 0.2 (0.5) for the light-flavour (W+c) selections, respectively. To compute this PF-based isolation, the momentum ratio of reconstructed objects within $\Delta R = 0.4$ around the iso-muon candidate with respect to the transverse momentum of the muon, is calculated. On the contrary, for isolated muons, the isolation needs to be less than 0.15, with same pseudorapidity and ID cuts, and $p_T > 30$ GeV (W+c / semileptonic $t\bar{t}$) or $p_T > 12$ GeV (DY+jets).

For the different selections, besides the object criteria listed below, also the trigger paths listed in Table B.6 together with the respective datasets must be passed, which are not listed below but always assumed implicitly. Only certified data from lumi blocks present in the lumimask JSON file are chosen for the analysis. The three phase space regions can be summarized as follows, based on [17, 240]:

- **light-flavour selection (DY+jets).** At least two isolated, oppositely charged muons (with $p_T > 12$ GeV) are required (and less than two electrons for this muon chan-

nel). After sorting muons by transverse momentum, the leading muon must also have $p_T > 20$ GeV. To select a sample enriching the Drell-Yan+jets contribution to obtain mostly light-flavoured jets, a constraint is placed on the reconstructed Z boson candidate from the oppositely charged muons: its mass is required to be within ± 10 GeV of the assumed Z boson mass of 91 GeV. At least one jet in the event must fulfill the basic jet selection criteria. The jet with highest transverse momentum is considered the “probe” jet, and is the jet for which the distributions will be shown.

- **charm-flavour selection (W+c).** The goal is to identify c jets based on the semileptonic decay of c hadrons which involves a soft and non-isolated muon, and an isolated lepton and a neutrino (missing energy / MET) from the W boson decay. There are two cases to consider regarding the charges of the two involved leptons (for the simplified tests in this thesis and in Ref. [19], only considering the muon channel): opposite signed (OS) charges, or same signed (SS). The targeted signal process (W+c) always yields OS events, the most important background which involves gluon splitting into $q\bar{q}$ however produces an equal amount of OS and SS events. Therefore, for this phase space, a so-called OS – SS subtraction is performed for any distribution of a variable to obtain a high purity of signal events containing c jets [9,17]. Further requirements are exactly one isolated muon (and no electrons), between (inclusively) one and three jets passing the basic quality criteria, at least one jet must contain a soft, non-isolated muon (matching done by pseudoangular distance, which is set to be less than the jet radius of 0.4). Should there be multiple jets satisfying this condition, jets are ordered by transverse momentum in descending order and the first one is chosen as muon / probe jet. A window of [80 GeV, 100 GeV] for the invariant mass of the two muons is excluded (veto Z bosons). Additionally, the following criteria are applied: $\Delta R_{\mu-\text{jet,iso}-\mu} > 0.5$, $M_T^W > 50$ GeV, $p_T^{\text{soft}-\mu} / p_T^{\mu-\text{jet}} < 0.4$, $\text{Jet_MuEF} + \text{Jet_nEMEF} < 0.7$.
- **bottom-flavour selection (Only semileptonic $t\bar{t}$, di-leptonic channel is omitted for this simplified approach).** This selection is very similar to the W+c phase space, however, the selection regarding number of jets is inverted (here: at least four jets passing the basic jet quality criteria) and the OS – SS subtraction is not done. An additional veto on events with an isolated muon with transverse momentum greater than 12 GeV is placed to avoid double counting of events that would pass the (here omitted) di-leptonic $t\bar{t}$ criteria. Another veto is placed on a Z boson candidate mass window of [85 GeV, 95 GeV], also, this invariant mass must not lie within [0 GeV, 12 GeV] to exclude resonances. The muon jet is utilized as probe.

7.2.3.1 Distributions of Multiple Discriminants

For visualizing purposes, it has been decided to split the stacked histograms in the following way: if a particular selection enriches flavour X , merge all processes that result in truth flavour jets different from X into one category, and keep all those processes separate that yield X -flavoured jets. This explains why for the light-enriched selection (shown in Figures 7.6, 7.7 and 7.8), only four entries exist in total (there is only one process considered in this simple setup for simulated samples), while for the charm- and bottom-enriched selections (included as Figures 7.9, 7.10, 7.11 and 7.12, 7.13, 7.14, respectively), more fine-grained categories can be found, as multiple simulated processes can yield the respective flavour with high abun-

dance (see Sec. B.2.1 and Ref. [17]). The different processes resulting in X -jets are then taken as different hues for one main colour. Each figure contains two subfigures, left columns show the respective distributions for the nominal training, while the right column is obtained from adversarial training. Wave patterns or large discrepancies present for nominal training are removed or at least reduced, most visibly for the light-flavour enriched selection, for three independent discriminant distributions, less visible for charm jets. As the changes between nominal and adversarial training are not equally large across selections and their interpretation requires quantitative figures of merit, the next section adds more metrics to support the qualitative statements, before performing the interpretation of results.

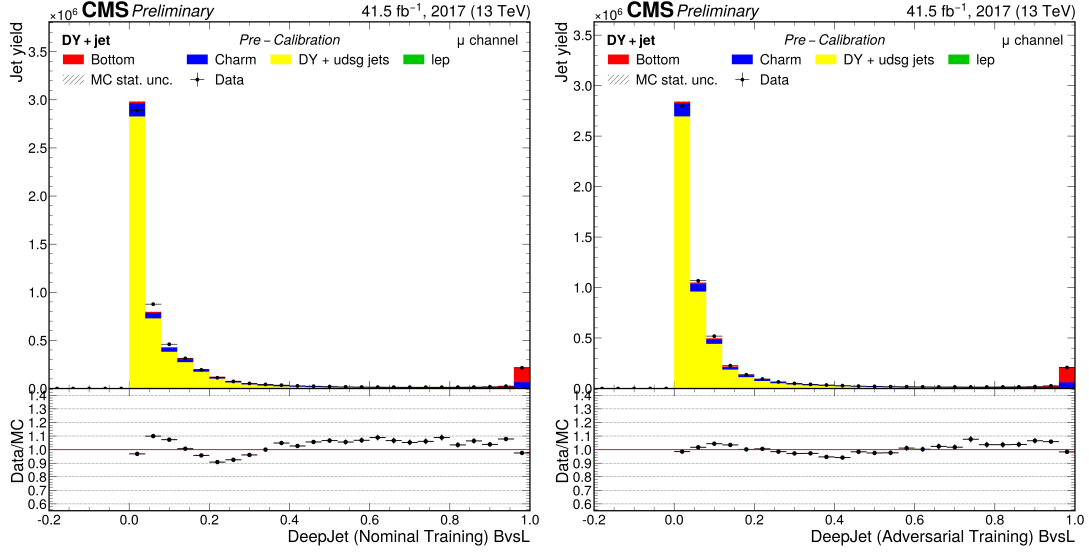


Figure 7.6: BvsL discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the light-flavour enriched selection. Figures taken from Ref. [19].

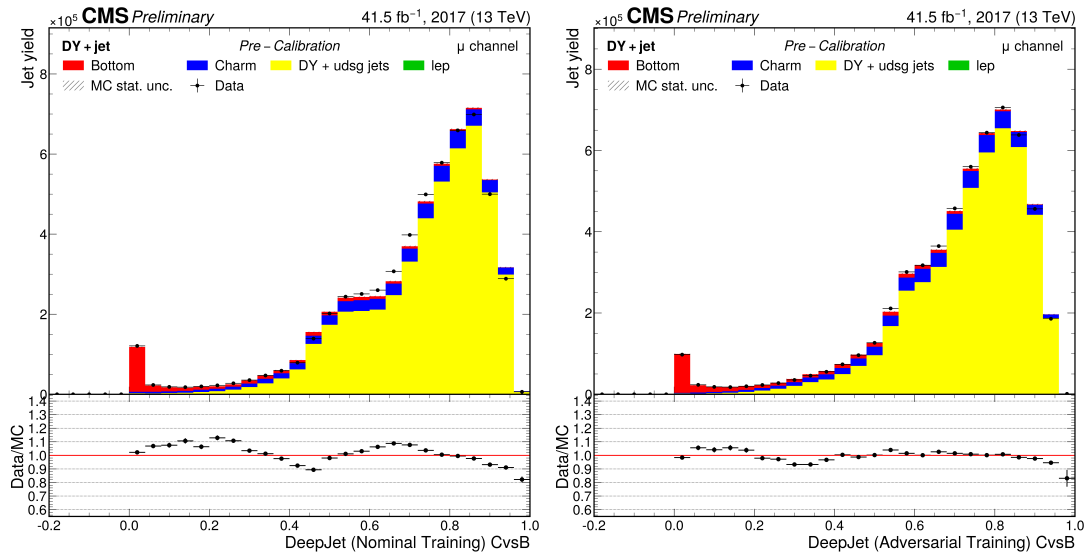


Figure 7.7: CvsB discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the light-flavour enriched selection. Figures taken from Ref. [19].

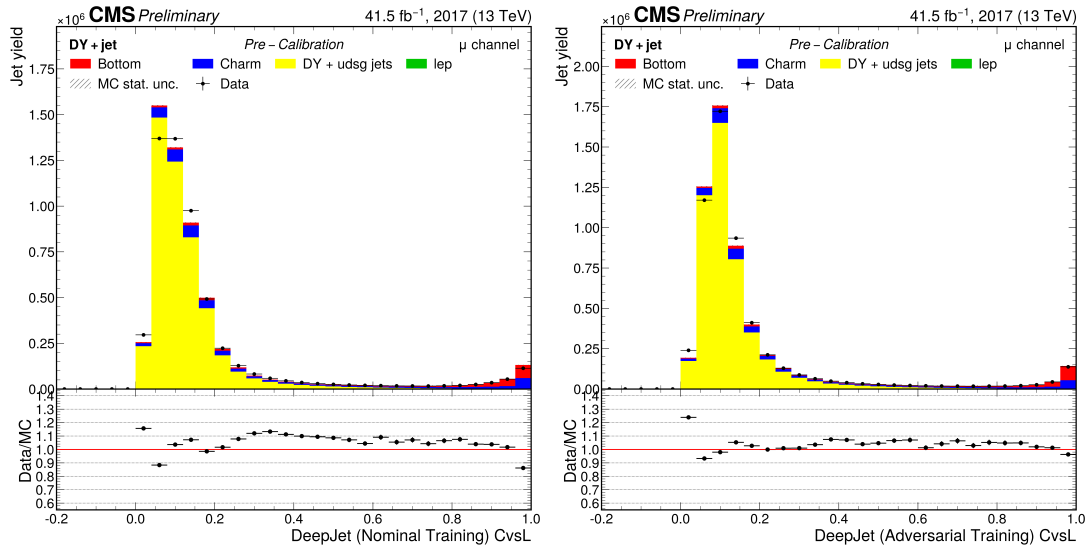


Figure 7.8: CvsL discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the light-flavour enriched selection. Figures taken from Ref. [19].

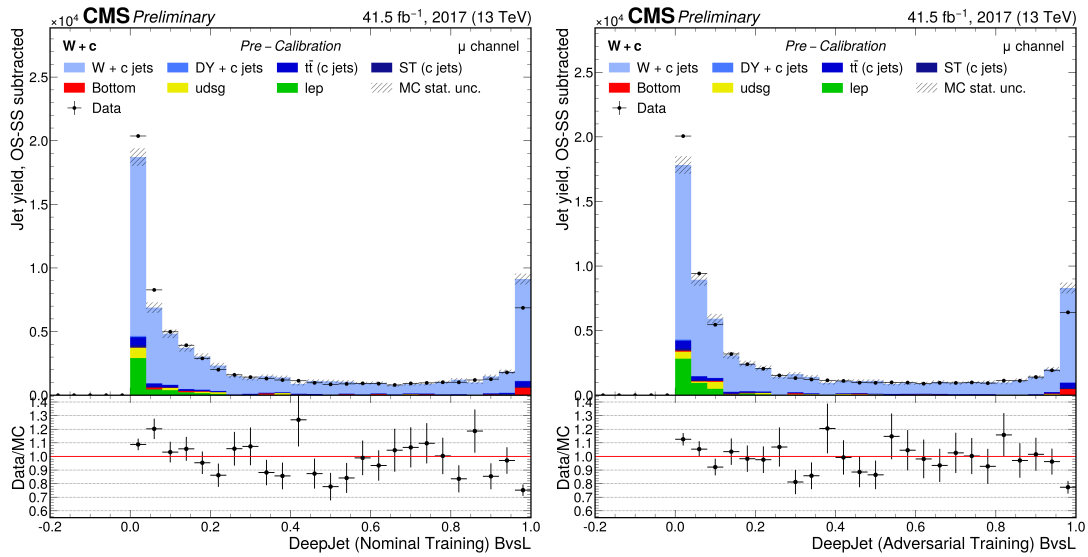


Figure 7.9: BvsL discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the charm-flavour enriched selection. Figures taken from Ref. [19].

7.2.3.2 Quantification of Agreement between Data and Simulation

Quantifying the agreement is performed with the Jensen-Shannon (JS) divergence [332], also further abbreviated as JSD. Compared to the Kullback-Leibler (KL) divergence [333] that makes up two components of the JS divergence, it is symmetric under exchange of the two compared distributions, and fulfills all remaining criteria of being a metric. The reference distribution against which the two studied distributions are compared is obtained from averaging these two distributions, bin-by-bin. The JSD is a distance measure between two distri-

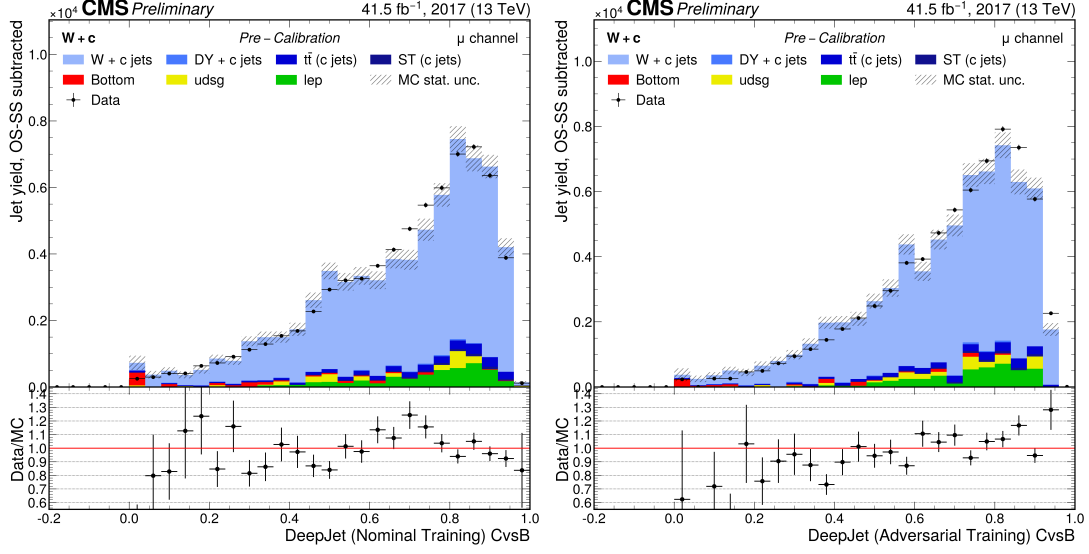


Figure 7.10: CvsB discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the charm-flavour enriched selection. Figures taken from Ref. [19].

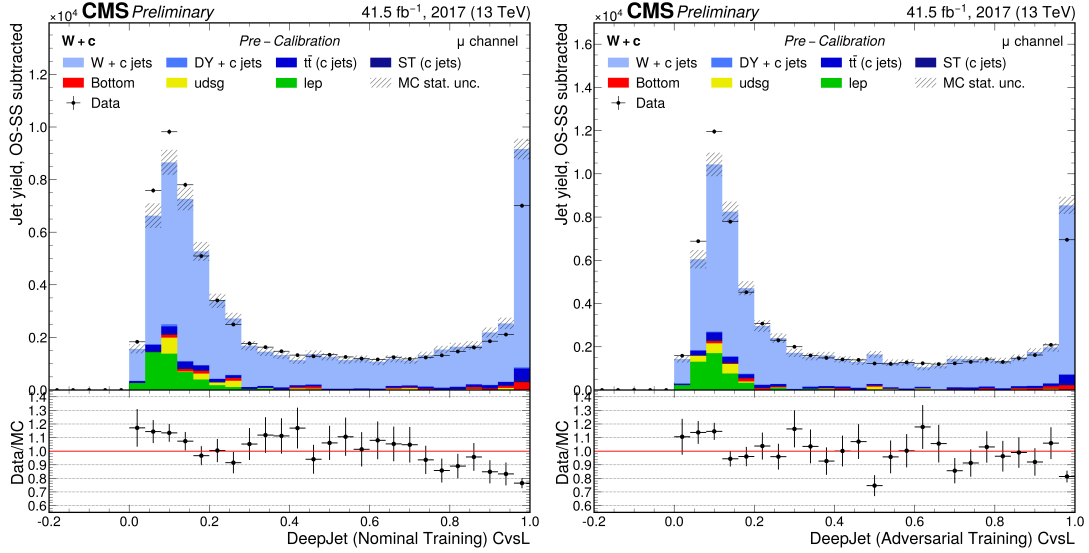


Figure 7.11: CvsL discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the charm-flavour enriched selection. Figures taken from Ref. [19].

butions P and Q , and defined as

$$\text{JSD}(P||Q) = \frac{1}{2}D(P||M) + \frac{1}{2}D(Q||M), \quad (7.2)$$

$$\text{where } M = \frac{1}{2}(P + Q), \quad (7.3)$$

$$\text{and } D(X||Y) = \sum_{\text{all bins } k_i} X(k_i) \cdot \log \frac{X(k_i)}{Y(k_i)}. \quad (7.4)$$

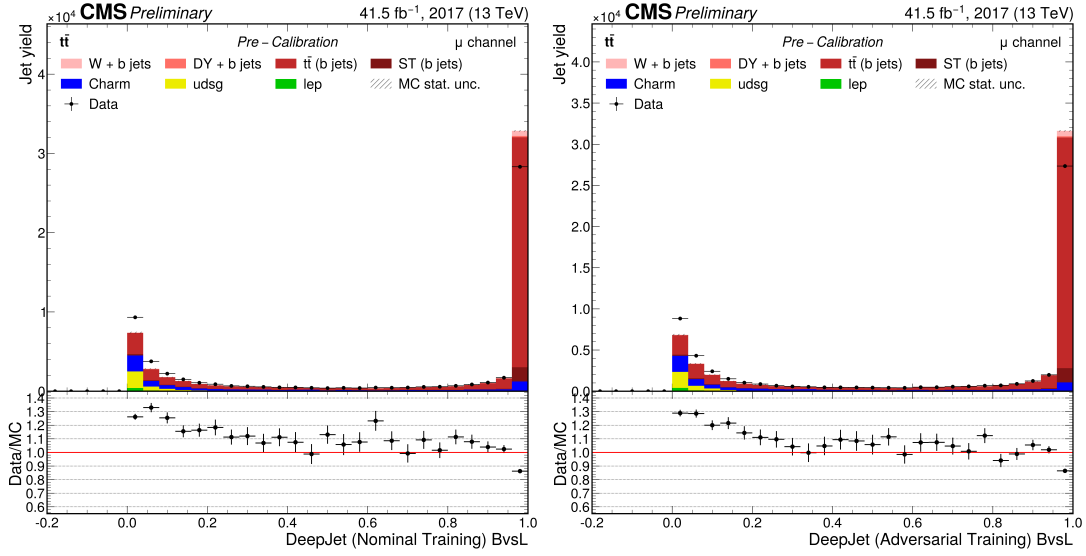


Figure 7.12: BvsL discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the bottom-flavour enriched selection. Figures taken from Ref. [19].

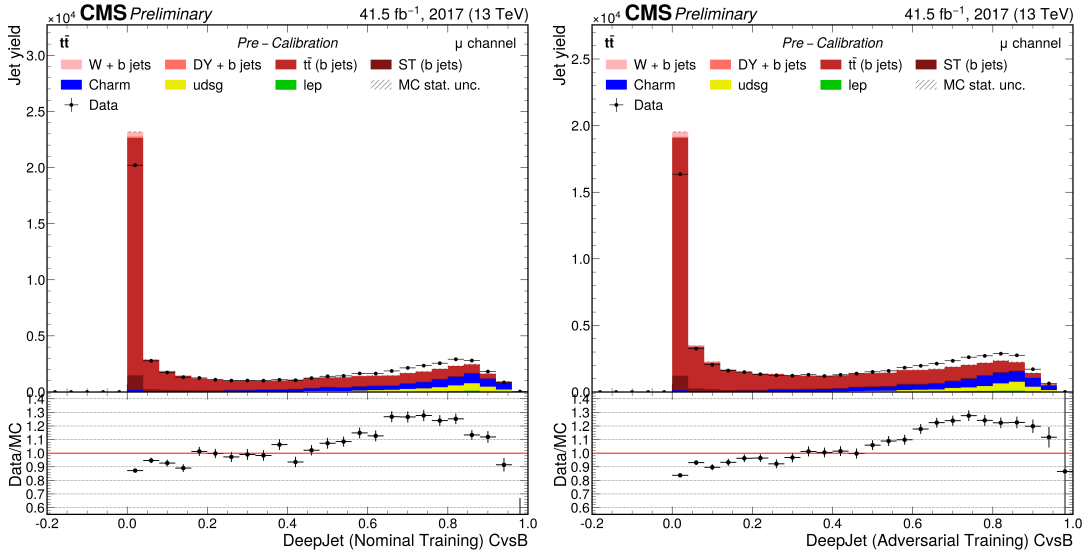


Figure 7.13: CvsB discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the bottom-flavour enriched selection. Figures taken from Ref. [19].

In Eq. (7.4), $D(X||Y)$ refers to the KL divergence introduced above. With this definition, it is evident that the best possible value for perfect agreement between the two distributions under investigation is 0. In such cases, both P and Q are equal to M , all bins match individually, thus every logarithm of the ratios and their sum evaluates to 0. For the purpose of comparing data to simulation (data/MC), the respective distributions are inserted into Eq. (7.2) after performing a histogram binning into 30 equal-width bins, covering the default value bin and thus the range between -0.2 and 1.0 inclusively. Due to the symmetry argument, one may either identify P with Data, Q with MC, or vice-versa. For bins where the resulting number of entries is 0 either for data or for the sum of all simulation, the bin is excluded from the cal-

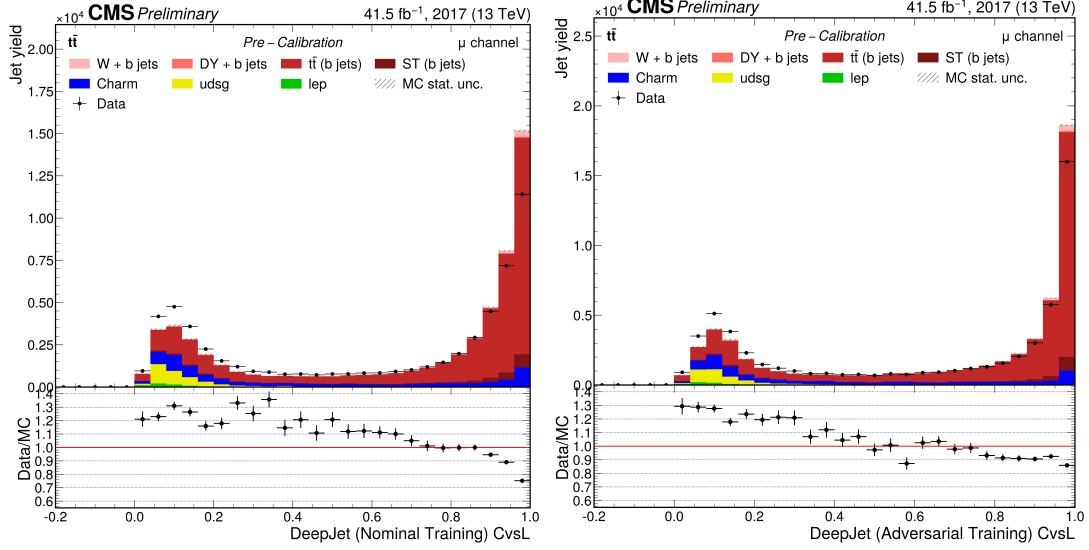


Figure 7.14: CvsL discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the bottom-flavour enriched selection. Figures taken from Ref. [19].

culuation of JSD. Similar, if due to OS–SS subtraction a resulting count evaluates to a negative value, then this bin is also not taken into account as the logarithm is only defined for positive arguments (the entry is effectively replaced with a 0 in the sum). Another statistic used to compare the two distributions P and Q is χ^2 , with its definition

$$\chi^2(P||Q) = \sum_{\text{all bins } k_i} \frac{(P(k_i) - Q(k_i))^2}{Q(k_i)}. \quad (7.5)$$

Here, symmetry is not maintained between P and Q , and the convention of choosing P as data and Q as simulation is followed for the remaining part of this analysis. Both the χ^2 as well as the JSD values are obtained after normalizing the respective distributions to ensure comparability across phase spaces that have different total numbers of jets passing the respective selections.

The process is repeated for nominal and adversarial training. In both cases, inference is performed for data and simulation with each of the models, but no adversarial attacks are applied. Thus, this study is a direct comparison of how the different trainings would perform when applied in a physics analysis, independent of artificially introduced input modifications. This is comparable to testing two different algorithms, like for example DeepCSV versus DeepJet. Resulting JSD values are listed in Table 7.1.

For light jets, adversarial training yields better agreement for the three discriminants investigated. For charm jets, a similar statement holds for two of the three discriminants. The exception for CvsB can be understood as a result of the introduction of a positive slope for the data/MC ratios. For bottom jets, adversarial training yields better agreement for two of the three discriminants investigated. A similar conclusion can be obtained when switching from JSD to χ^2 as a measure to compare the two distributions, examples are given in the appendix, Sec. E.

Table 7.1: Agreement between data and simulation, expressed through the Jensen-Shannon divergence, in three control regions (light-, charm- and bottom-enriched samples) and for three discriminants each (BvsL, CvsB, CvsL). Reproduced from Table 1 in Ref. [19].

JS divergence (a.u.) DeepJet 2017 (13 TeV)	udsg jets			c jets			b jets		
	BvsL	CvsB	CvsL	BvsL	CvsB	CvsL	BvsL	CvsB	CvsL
Nominal training	0.000358	0.000353	0.000947	0.002632	0.002350	0.002263	0.003506	0.002528	0.004820
Adversarial training	0.000063	0.000058	0.000466	0.001887	0.003074	0.001766	0.003329	0.003005	0.002924

7.2.3.3 Scale Factors

To derive charm-tagging scale factors using the same strategy as outlined in Ref. [17], a finer binning (with 60 bins) is used for the calculation. The corresponding results for JSD and χ^2 are given in Sec. E as well.

First, the histograms are normalized such that the calibration only concerns the shape, but not the event yields. For the simplified SF derivation, a merging of channels is not applicable, as there is only one channel (muon channel) in the first place. One of the two axes, CvsB or CvsL, must be chosen as a “fixed” axis for which all bins are kept at the same width. In this case, CvsB is chosen and sliced into ten equal width bins covering the range $[0, 1]$, which yields ten individual CvsL distributions. Although the method is suitable to correct also the so-called “-1 bin”, the custom algorithms investigated for this thesis never yield such default output scores: first of all because unlike DeepCSV, DeepJet and later in this thesis also RobustParticleTransformer are run in any case, independent of selection criteria related to tracks, second, because numerical zeros in denominators do not occur due to safety measures during calculation and processing of network logits.

In this case, with all obtained distributions, an adaptive binning technique for the variable CvsL axis and fixed CvsB axis is employed. This needs to account for two complementary goals: providing high granularity and small statistical uncertainties for the bins [17]. For the subsequent iterative fit, the goal is to minimize a global χ^2 metric that describes the agreement between data and simulation for all three flavours. This optimization, as described in Ref. [17], takes the previously obtained binning scheme and iteratively derives and applies SFs for the three jet flavours / phase space regions consecutively until a termination criterion for the global χ^2 , being the sum of the individual selection’s χ^2_{sel} , is reached. For each of the fixed bins, the three sets of jets are compared for their purity, and beginning with the purest one, SFs to adjust the individual variable bins are derived, however never changing the starting value of 1 by more than a configured step size per iteration (for DeepJet and RobustParticleTransformer, 0.005). With every obtained set of SFs per CvsL-bin in a CvsB-slice, the respective next flavour (after ordering by purity) can then already make use of previously obtained SF arrays (current estimates of SFs are applied for consecutive derivations, and so on). This three-flavour / three-step approach is iteratively applied for every fixed bin, and each time, the following quantity is derived and target of minimization [17]:

$$\chi^2_{\text{sel}} = \sum_{i=1}^n \frac{(N_{\text{sim},i} - N_{\text{data},i})^2}{\sigma_{\text{sim},i}^2 + \sigma_{\text{data},i}^2}, \quad (7.6)$$

with statistical uncertainties in the i -th bin in simulation ($\sigma_{\text{sim},i}$) and data ($\sigma_{\text{data},i}$), a total

number of n bins, the i -th jet count in simulation ($N_{\text{sim},i}$, the scale factor weighted sum of individual flavour contributions per bin) and data ($N_{\text{data},i}$) [17]. Similar to a neural network training, if the figure of merit for optimization does not improve for a defined number of iterations (here: 50), the algorithm stops, and the iteration yielding the lowest global χ^2 is taken as the best multi-dimensional estimate for the scale factor arrays. For bins where the purity is low or the statistical uncertainties are too large, the SF is fixed at 1 ± 1 [17,240]. When deriving the SF maps, only statistical uncertainties are considered so far.

Just like for the individual distributions, also the SFs reveal that the light-flavour enriched region is affected most when switching from nominal to adversarial training. Especially in the region with small uncertainties in the last row of Figs. 7.15 and 7.16, the adversarial training shows a significantly larger number of bins with scale factors close to 1, and that region is more spread out as well. Without applying the calibration itself, the adversarial training has a “calibrative” effect for certain parts of the phase space.

7.2.3.4 Discussion and Next Steps

First of all, it can be concluded that in seven out of nine cases for the 30 bins case, there is a significant improvement for the agreement between the two domains, data and MC, measured with JS divergence. This is remarkable, because adversarial training and adversarial attacks are not necessarily replicating the actual mismodeling between simulated samples and data. Nothing guarantees that performing a defense strategy against an artificial input disturbance is related to physics via detector effects, imperfect calibration, or simulation that is performed only up to a certain order and can not predict the processes with infinite precision. The control region for which the adversarial training improves the situation most noticeably, is the light-flavour enriched selection. This results in an almost perfect match between the two distributions for all discriminants. How this could be related to input (mis-)modeling shall be studied in the following section as a follow-up.

When interpreting the cases for which adversarial training deteriorates the level of agreement (measured with JSD), there is more to consider than just looking at this scalar quantity alone. It is not the only relevant or sufficient instrument to measure performance and agreement for such algorithms, because one should also investigate the distributions themselves. In fact, taking CvsB for the charm-enriched selection as an example, for adversarial training, a higher fraction of jets in data than in simulation would be tagged as charm jets when cutting at, for example, 0.8 (note the positive slope in the ratios). For nominal training, the opposite is the case, taking a fixed cutoff value that would serve as “tag” or “not-tagged” metric in an analysis, more charm jets in simulation would become true positives, less so for data. This reasoning is allowed here because all these selections have been investigated in terms of purity, and the charm jet selection is particularly pure in catching true charm jets, and not contaminated with large fractions of other flavours [17]. Just like these tagger-independent selections are used to derive scale factors, one may make use of them when discussing performance of nominal or adversarial training. Thus, a similar flavour-composition as found in simulation is assumed in data. Using this finding, the agreement for CvsB (measured with JS divergence) does not improve for adversarial training, because this training generalizes better to data than performing on simulated samples (the opposite is true for nominal training, which, as usual, shows higher performance in simulation than in data for charm tagging). Therefore, the

reason for disagreement is bad generalization for nominal training, and better generalization for adversarial training. Another way to interpret this is that adversarial training is likely not optimal in terms of charm tagging performance in simulation, but therefore the generalization gap to this new domain (data) shrinks so much that the adversarial training can already be described as an "over-correction" for the charm selection (there is a positive slope, at high CvsB values, data exceeds simulation).

In general, different behaviour for the three different phase spaces investigated is observed. Mismodeling between data and simulation is not necessarily identical for different flavours, using individual hyperparameters ϵ_i (or better, scanning over a set of hyperparameters) per flavour i while training or applying domain adaptation could improve the situation, especially for heavy-flavour jets. Compared to light jets, for charm jets a smaller parameter ϵ_c could be chosen during training, as it is possible that different flavours need different severity of the impact or different strength of regularization. Doing so, over-correction can be avoided by introducing a feedback-loop between training and data/MC validation. This is possible with the new software framework introduced earlier, which incorporates such validation or commissioning studies into the usual task of training neural networks.

Given that there is a tradeoff between performance and robustness, a clear goal needs to be defined, to which level the pre-calibration agreement shall be improved (one possible goal could be to achieve data/MC close to 1 as shown for the light-flavour in the study above). This should be verified with uncertainty bands for pre- and post-calibration. One also needs to define how much nominal performance (i.e. MC-performance) is acceptable to sacrifice (in optimal cases, negligible). To recover performance lost within the context of improved robustness, techniques like focal loss may be applied to give higher weight to hard-to-classify samples [11, 260].

7.2.4 Do Adversarial Attacks Mimic or Reduce Input Mismodeling?

7.2.4.1 Introduction and Software

The first ever study that evaluates the impact of adversarial attacks on input modeling with direct comparison to data is conducted. In the previous section, the focus was on agreement between detector data and simulated samples for tagger scores or discriminants, thus, at the output level only. One of the most commonly asked questions, "How are adversarial attacks related to data/MC agreement of tagger input features?" shall be answered. This investigation has been started with a framework that carries out multiple tasks, like performing commissioning and studying performance from a machine learning perspective. Not requiring the integration into the official reconstruction software by combining the steps into one makes such studies feasible. Thus, this opens new ways to learn about how neural network tagging algorithms translate between input and output modeling and which impact novel techniques, in this case adversarial methods, have on these properties.

7.2.4.2 Expectation and Goals of this Study

A priori, there is no reason why applying an adversarial attack, i.e. an artificial modification without physics knowledge, should yield noticeable improvements in input modeling. With this first validation of different variants of MC samples against detector data, inputs are inves-

tigated in the same way as outputs were studied in previous sections. A weighting technique derived from the results of this section will be discussed in Sec. 7.4, resulting in a proposal for a novel data/MC-aware tagging algorithm.

7.2.4.3 Selection and Setup

The light-flavour enriched selection is chosen as an example, for which the relevant enhanced samples with all necessary features are available. In Table 7.1, this selection appeared as the one most positively affected (in terms of agreement expressed through JS divergence) by the introduction of adversarial training. The same configuration and analysis cuts are used for three individual setups of simulated samples, while detector data remains unchanged for all cross-checks. Technically, the analysis runs only once per event (it is independent of the input features), but three variants of tracked branches are stored. The first refers to the central clean input values, the second takes the nominal model as a reference and applies the FGSM attack on top of the clean samples, the third and last variant does a similar attack, but derived for the adversarial model. These attacked inputs have been obtained during a dedicated upstream inference step and become spectator branches that are part of the tuples used during the event selection. For variables that are both part of the selection (e.g. if they describe the global kinematic properties of a jet) and at the same time part of the input features for the model, then only the nominal, original values are used when applying the selection cuts. It shall be emphasized that the selected jets or selected events are exactly same across all compared setups, and that the selection is completely blind with respect to the potentially attacked input features. The variable studied in detail is the pseudoangular distance between the first neutral particle flow candidate and the jet axis. It is a suitable candidate to investigate the impact of small differences that can play a role when distinguishing light-quark jets from gluons. While most studies in this thesis focus on heavy-flavour identification, this introduces variety. During pre-processing, this feature is engineered by a linear shift of -0.6 to span only the interval between -0.6 to -0.2 (excluding defaulted values, not shown). Re-translating back from shifted into physical values, the distribution covers values between 0 and 0.4, matching the jet radius of $R = 0.4$. Such shifts are typical variants of feature engineering, when default values (0) exist and values with lower likelihood for physical jets should be placed closer to such defaults, than values with high likelihood. This introduces a reasonable ordering of numerical values that is injected into the neural network [160, 196, 321].

7.2.4.4 Results and Discussion

The *input* modeling improves when applying the FGSM to simulated samples, for which an example is shown in Fig. 7.17. At numerical values larger than -0.3 (in terms of physics quantities, larger than 0.3), it is typically not well modeled in a phase space region where the contribution of light jet flavour dominates, as can be seen in the first row of Fig. 7.17. This variable is sensitive to the width of the jet, or shape differences between jets initiated by gluons (wider) or light quarks (more narrow), see also Secs. 2.1.2.1 and 4.2.2. From Fig. 7.18, the different behaviour between nominal and adversarial training is especially evident in the light-flavour dominated region. Exactly in the region with “missing” contributions by simulation, the adversarial attack is a suitable mitigation technique. Both variants with applied adversarial attacks on top of the nominal or the adversarial model improve the modeling,

however, for adversarial training, the agreement is significantly better in the aforementioned region. This is a property which can explain why the output scores or discriminant shapes improve a lot from nominal to adversarial training (for which an attack works well in terms of input modeling). But the reason causing this match between input modeling and adversarial attack for an admittedly arbitrary neural network remains unknown. One may assume that the feature extraction is sensitive to input features which carry more discrimination power in simulation than they should when looking at actual detector data, and that exactly those features with over-estimated separation power due to imperfect modeling are what drives the network's decision. This matches results found in robustness and interpretability studies for DeepCSV [233, 290]. All attacks studied in this thesis are untargeted, i.e. they change a jet of a given flavour in such a way that it looks less like a jet of the given flavour, but do not prioritize any other "wrong" flavour. Light jets are likely to be distorted such that they appear more like heavy-flavour jets, and vice versa (see also Sec. 6.4.4). For input properties for which the choice of MC generator is such that mismodeling is of this previously described type (at least to first order), the attacked inputs then automatically correct for this modeling effect. There happen to be several inputs mismodeled in such a way, when judging from the available data/MC results. A goal for future research could be to identify a suitable matching between well-modeled input features and features that are highly ranked to find good variables with which to perform the training and defense strategy. If less badly modeled features are included in an ablated model, then the necessity to utilize attacks with high impact can be reduced accordingly. It can be concluded that the observed mismodeling in discriminant outputs is, as expected, a result of two mechanisms: input modeling (same for all models using these inputs), and the "intriguing" properties [12] of the network itself. Both require individual treatment, with adversarial training, some fraction of the input modeling can be incorporated, and some fraction of the network behaviour can be corrected as well. How this strategy can be improved (not having to rely on the defense mechanism alone) is subject of Sec. 7.4.

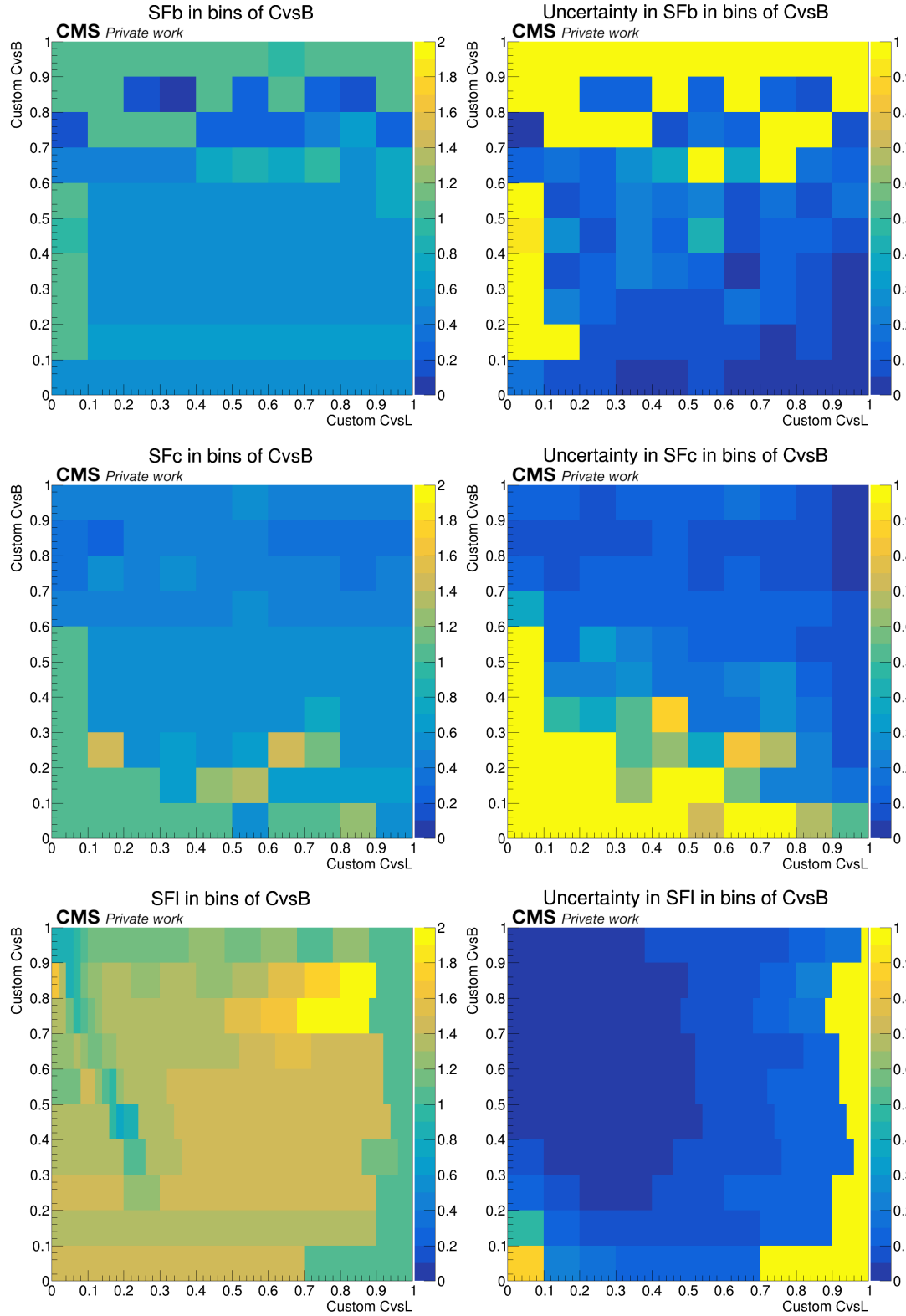


Figure 7.15: Scale factors for nominal training and associated statistical uncertainties, for three different flavours (b, c, light) from top to bottom.

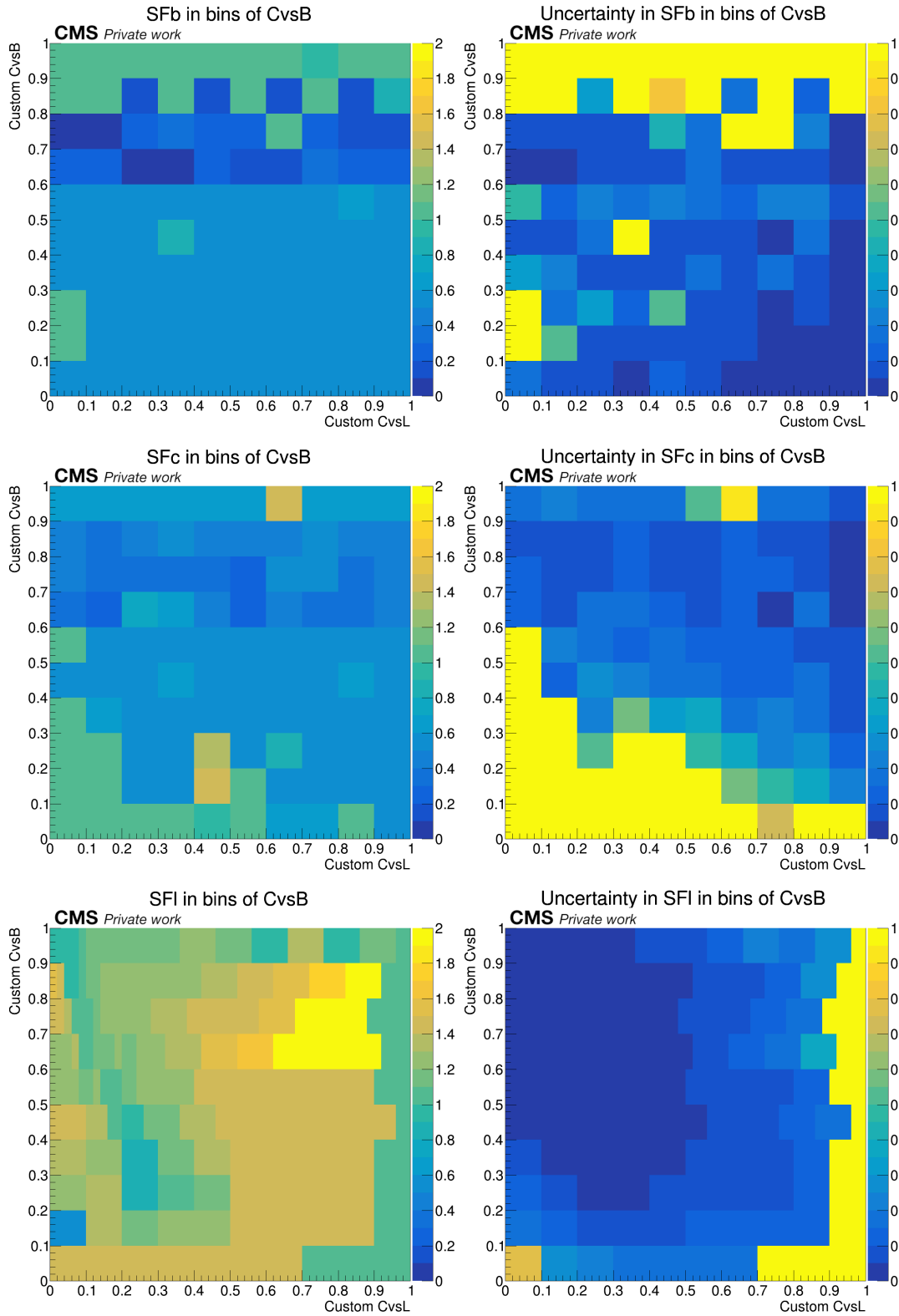


Figure 7.16: Scale factors for adversarial training and associated statistical uncertainties, for three different flavours (b, c, light) from top to bottom.

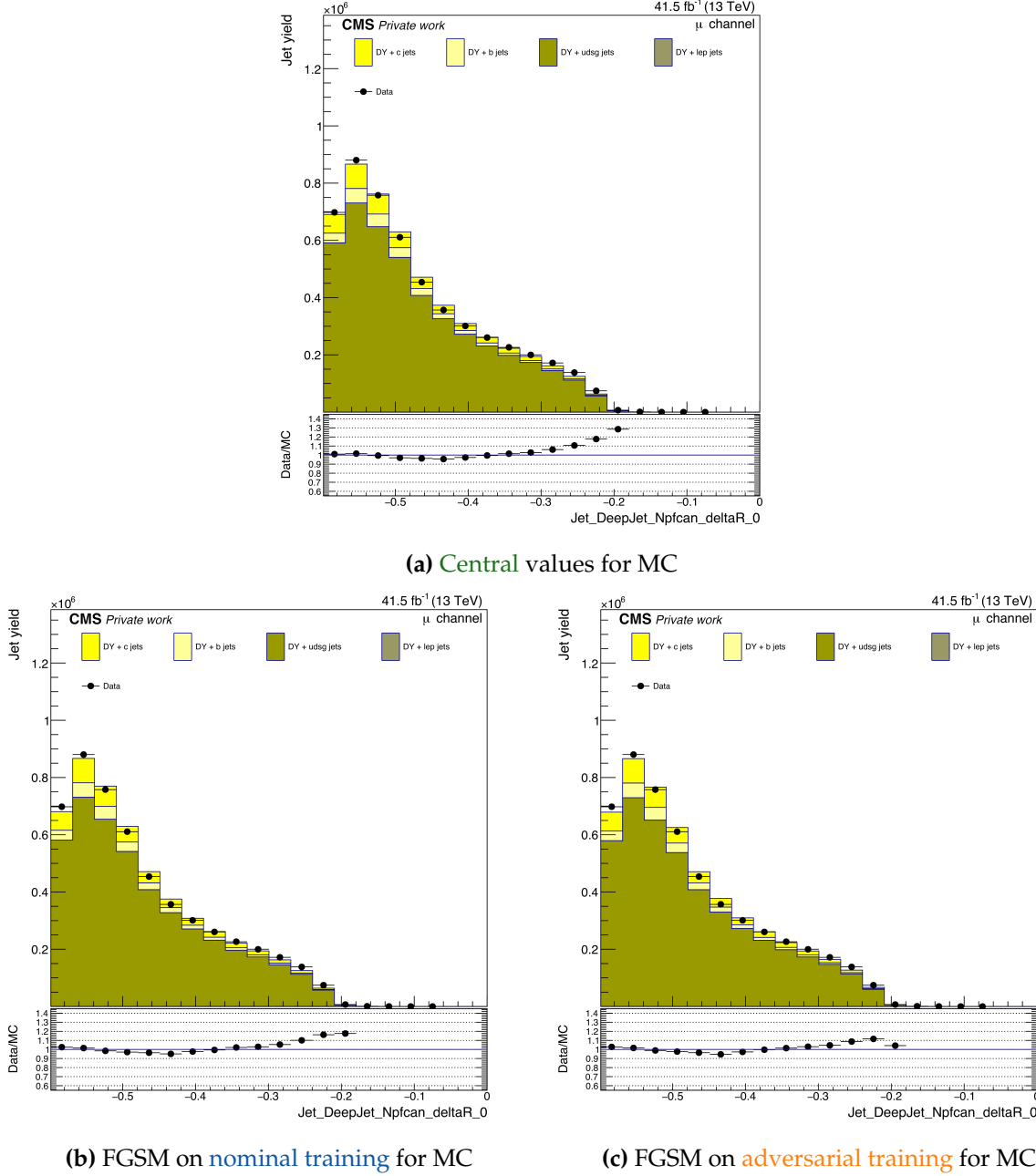


Figure 7.17: Pseudoangular distance between the first neutral particle flow candidate and the jet axis for simulated samples and data. Clean MC samples (first row) are compared with systematically attacked samples (second row), when drawing FGSM inputs for the nominal (left) or adversarial training (right).

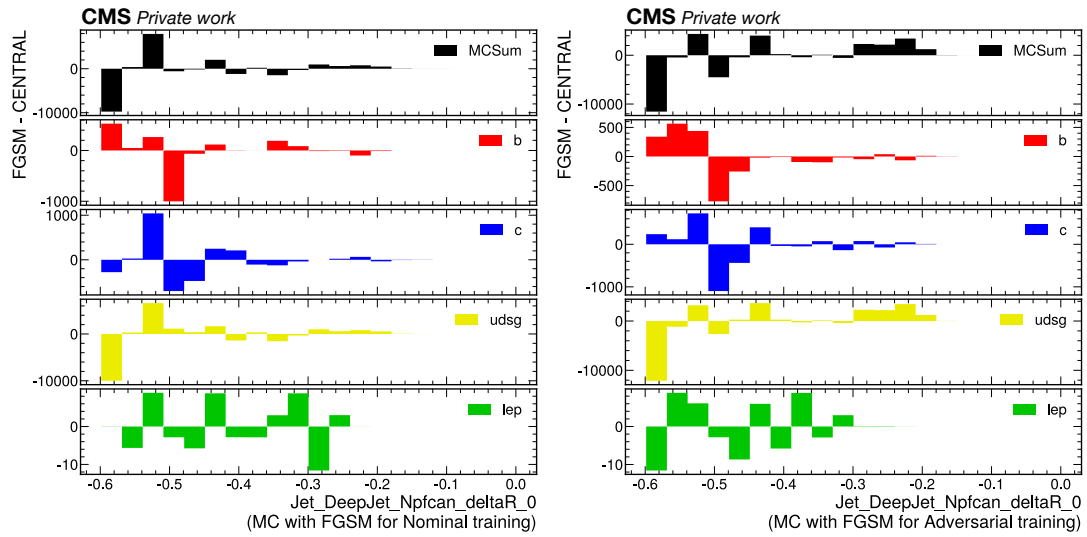


Figure 7.18: Changes for the pseudoangular distance between the first neutral particle flow candidate and the jet axis as introduced by the FGSM attack, split by flavour, comparing nominal and adversarial training on the left and right, respectively.

7.3 A New Era: Development and Integration of a Robust Transformer Network: RobustParticleTransformer

Transformer models [334] have revolutionized various areas of ML applications [335, 336], such as natural language processing [337] or computer vision [338]. While this new architecture has brought performance improvements for a variety of tasks, it is also a natural next step to investigate safety and robustness of this new tool [336, 339]. In 2022, a new NN architecture has been proposed for jet flavour tagging, the so-called Particle Transformer (ParT) [20], which constitutes a suitable proving ground for novel adversarial attack and defense techniques.

7.3.1 A Transformer Architecture for AK4 Jets

The ParT network that, similar to ParticleNet [241], uses a large number of input features with modern feature extraction techniques, was first tested on large-radius jets ($R = 0.8$) in the original publication [20], and is applied to small-radius jets ($R = 0.4$) in CMS, as first documented in Ref. [21]. Therefore, a general introduction to the model architecture is given first, followed by the specifics for CMS. The core part of this section will then focus on the novel strategies and the author's work that preceded the first implementation in CMSSW, in particular, the newly developed adversarial method first applied to ParT which culminates in the state-of-the-art RobustParT algorithm [22]. This latter aspect, the merging of a new attack / defense mechanism with the available highly performant model, is the result of combining the advancements of the architecture on the one hand with insights on robustness and physics applications on the other hand. To the author's knowledge, this is the first identification algorithm in CMS for which such studies are done and their results known before integration of the model in the standard algorithm sequence for the collaboration*. This is possible, as both lines of development, the algorithm itself [20, 21], and the robustness studies as well (cf. Ref. [11, 19]), are developed in an inter-operable, modular way that enabled merging in PyTorch. Everything is steered with transparent configuration scripts and bookkeeping is done to track every network training in its development phase. RobustParTAK4 is set to perform the same classification task as DeepJet, hence the same number and definition of classes (b, bb, lepb, c, uds, g).

The network on which the new adversarial methods are applied shall be described first with a brief summary. For more details, the reader is referred to the original publication [20] and the (initially non-robust) adoption in CMS [21], on which the descriptions below are based.

In general, the architecture is comprised of two types of blocks, of which there can be one or multiple realizations each, the Particle Attention Block structure, and the Class Attention Block structure. For these blocks, two types of inputs are injected: particles and interactions (pairwise features). For the CMS Run 2 and Run 3 models that are studied in this thesis, these inputs (C regular particle features for N particles, and C' features that contract two particles each to yield $N \times N \times C'$ matrix entries) are described in Table B.12 in the upper and lower part, respectively. An overview of the ParT architecture as first presented in Ref. [20] is shown in Fig. 7.19.

Before the particle (interaction) features are injected into the respective blocks, they are projected to the $N \times d - (N \times N \times d' - \text{dimensional})$ space with two MLPs that yield the $\mathbf{x}^0$ (U, also

*See also <https://github.com/cms-sw/cmssw/pull/41275>

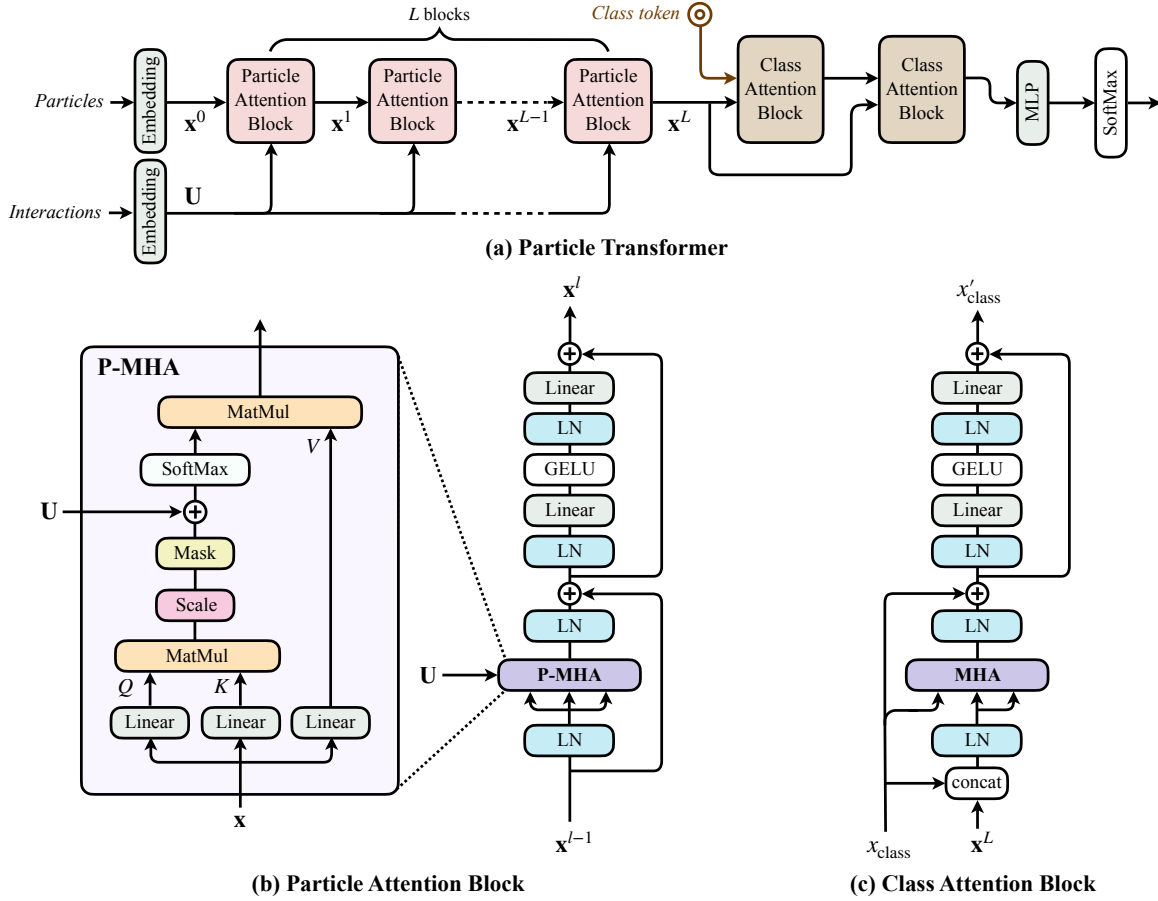


Figure 7.19: General overview of the ParT architecture (a), the Particle Attention Block (b) and the Class Attention Block (c). Figures (a)-(c) taken from Ref. [20].

called interaction matrix / bias) embeddings. There are no additional positional encodings, as the particles are treated as permutation-invariant [20]. The interaction features used for the AK4 implementation vary slightly between the Run 2 and Run 3 setup (see Table B.12), but are in any case representations derived from the particles' four momenta. The MLP for particle features consists of 1D-convolutional layers (kernel size 1) of dimensions (128, 512, 128), the one for interactions is similar, but with dimensions (64, 64, 64, 8) (see also Ref. [21]). Both use GELU activation and perform batch normalization between layers, both as described in Secs. 5.1.2 and 5.1.3.4, respectively. Looking at the individual blocks of (Robust)ParT, the Particle Attention Block first employs layer normalization (LN) [340], before performing multiple matrix multiplications to compute the so-called particle multi-head attention [20, 21, 334]. This is a variation of multiple masked, scaled dot-product attention layers running in parallel, which takes the interaction bias into account [20, 21, 334]:

$$\mathbf{P} - \text{MHA}(Q, K, V) = \text{SoftMax} \left(\frac{QK^T}{\sqrt{d_k}} + \mathbf{U} \right) V, \quad (7.7)$$

with Query Q , Key K and Value V [334], the linear projections of the particle embedding $\mathbf{x}^l$ in the l -th Particle Attention Block, and the scaling factor $\frac{1}{\sqrt{d_k}}$ (d_k being the dimension of queries and keys, d_v the dimension of values). As mentioned above, the matrix multiplication requires appropriate dimensions, which explains why one of the components (K^T) is transposed. For

the features, the overall dimension is 128 throughout the model. After the P-MHA stage, a two-layer MLP with GELU activation and LN follows. In the case of RobustParTAK4, there are $L = 3$ such Particle Attention Blocks, and each of them uses eight attention layers in parallel (i.e. there are $h = 8$ heads). Skip connections are introduced around the P-MHA and MLP component of the individual blocks. For the Particle Attention Block as shown in Fig. 7.19 (b), dropout is fixed to 0.1.

One Class Attention Block is employed (see Fig. 7.19 (c), and Ref. [341]). The MHA part of this block, unlike P-MHA for the Particle Attention Block, does not use the interaction matrix as bias term. The MHA component does however include a global class token x_{class} (as in CaiT [341]) from all particles, obtained from the encoded particle information and pairs of Lorentz vectors that have already been used for the Particle Attention Blocks. No dropout is used for the Class Attention Block.

The last step is to transform the obtained Class Attention outputs (i.e. the last class token), followed by a single-layer perceptron stage, to output logits of the NN. This can be converted to probability scores for each of the (in our case, six) classes, using the SoftMax function. When not doing inference, this last conversion is optional.



Logistics of dataset preparation and computing environment

The lxplus7 batch system is accessed using an HTCondor job with GPU requirements^a to use an Nvidia A100 card for training. For comparability with other trainings, the DeepJetCore environment is loaded into an Apptainer (formerly called Singularity) [342, 343] container image, with a customized DeepJet package that has been extended for ParT studies. A pre-installed PyTorch version with cuda support (1.10.2+cu111) is added to the PYTHONPATH environment variable. The setup is therefore very similar to the official DeepJet training pipeline introduced earlier. The DeepNTuples package that is used to create the initial .root files for training and validation has been extended to include new features with respect to those used for DeepJet, otherwise, the description of the primary datasets used to train DeepJet also holds for the ParT training, i.e. employing jet clustering with the anti- k_T algorithm using $R = 0.4$. The merging and reweighting strategy across QCD multijet and top quark pair production samples remains the same, the 2D resampling targeting kinematic distributions resembling the b jet distribution yields roughly 140 million jets for further usage.

^aSee also <https://batchdocs.web.cern.ch/gpu/index.html>.

A split between training and validation sets with 85 % and 15 % each is performed to support continuous evaluation during the training loop while leaving a significant fraction of jets to learn from. There are more similarities with DeepJet, i.e. choosing CCE loss for the classification task, saving each checkpoint after finishing a full epoch, using the epoch with best validation loss as the one chosen for performance measurements and integration.

The latest hyperparameter selection includes a batch size of 512, initial learning rate of 10^{-3} which is linearly decayed to 10^{-5} after 70 % of the training schedule (usually 15 – 20 epochs), and optimization of the model parameters with outer Lookahead algorithm [327] using $(k, \alpha) = (6, 0.5)$, and inner RAdam optimizer [328] with parameters $(\beta_1, \beta_2, \epsilon) = (0.95, 0.999, 10^{-5})$.

7.3.2 The Normed Gradient Method

The previously explored attack and defense mechanism (FGSM) is augmented for the purpose of this latest transformer architecture: systematic shifts of features are controlled by their overall 2-norm [310, 344], limiting the total impact while preserving individual contributions of input variables and their correlations with respect to the loss surface. This is done to partially mitigate the effect of rapidly increasing adversarial impact with larger numbers of inputs. The equation to generate adversarial inputs using the Normed Gradient Method (NGM) then reads [310]

$$\vec{x}_{\text{NGM}} = \vec{x}_{\text{raw}} + \underbrace{\epsilon \cdot \frac{\nabla_{\vec{x}_{\text{raw}}} J(\vec{x}_{\text{raw}}, y)}{\|\nabla_{\vec{x}_{\text{raw}}} J(\vec{x}_{\text{raw}}, y)\|_2}}_{\Delta_{\text{NGM}}}, \quad (7.8)$$

with the gradient of the loss function $\nabla_{\vec{x}_{\text{raw}}} J(\vec{x}_{\text{raw}}, y)$ evaluated at the original (raw) input vector $\vec{x}_{\text{raw}}$ of a given jet with truth flavour y . A restriction based on the individual input features (for example, the 25 % limitation introduced in Chapter 6), is then not strictly necessary anymore, but also implemented as optional parameter.

Unlike the FGSM attack, NGM does not take the sign of the gradient, but the gradient is normed to unit length, which is then added as distortion while maintaining directionality. Compared to standard one-step attacks [291, 310], NGM targets each type of constituent individually, because CPFs, NPFs, SVs, and their pairwise features are injected separately to the NN. Additionally, NGM introduces a fallback-option for constituents with missing values or special default values (performing multiple operations in parallel with PyTorch tensors on a GPU), and checks for integer features that shall be kept unmodified as a physics-motivated constraint.

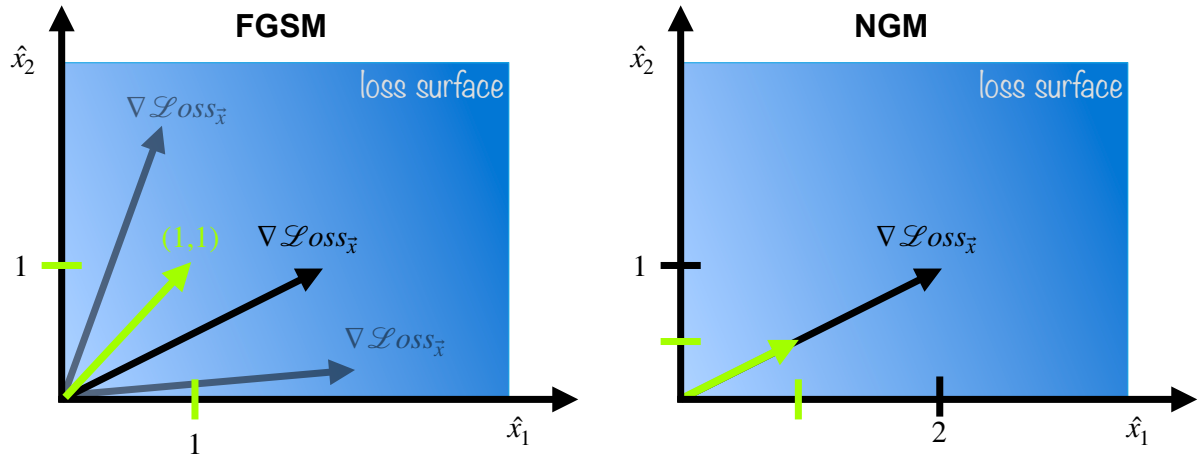


Figure 7.20: Two-dimensional comparison of the FGSM and NGM attacks, which either reflect only the sign of the gradient, no matter the exact direction or magnitude of the gradient vector (left), or take the direction and relative impact of features into account (right). Figures not drawn to scale.

In Fig. 7.20, the two methods are compared visually. The left subfigure shows how the FGSM attack treats three different example gradient vectors in the exact same way, as only the sign matters per feature. The number of inputs directly affects the total impact. On the right, the

NGM attack follows a different principle, the gradient vector is maintained regarding direction, but it is scaled on that axis to always end on the unit circle. For each jet, NGM produces an impact vector that will eventually be scaled to length ϵ , independent of the number of inputs.

FGSM treats all features separately without taking their correlations into account in quantitative way (only qualitatively). For any model with N features allowed to be distorted by such an attack (thus excluding integer features), there are 2^N possible directions for the attack, which is already a large, but definitely finite number. The intuition is that for each variable, a shift can be positive or negative, and FGSM does not correlate these features. For the two-variable example of Fig. 7.20, only the quadrant determines the direction of the attack, and the magnitude is given as a contraction of the parameter ϵ and the number of features. A fixed impact per feature is kept for every jet with the same signs of the components of the gradient vector (either $+1$, or -1 – nothing else). For all three vectors (which are all three effectively treated as $(+1, +1)$), the distortion Δ_{FGSM} is

$$\Delta_{\text{FGSM}} = \epsilon \cdot ((+1) \cdot \hat{x}_1 + (+1) \cdot \hat{x}_2), \quad (7.9)$$

with $\hat{x}_i$ being the unit vector related to the i -th input feature dimension. This yields a total impact or length of the distortion vector $L_{\Delta_{\text{FGSM}}}$, when measured with the 2-norm, of

$$L_{\Delta_{\text{FGSM}}} = \epsilon \cdot \sqrt{N}, \quad (7.10)$$

with N once more denoting the number of distorted features.

A similar calculation can be done with NGM, which instead does take the correlations across features into account, the exact direction of the gradient vector matters and yields $\mathbb{R}^N$ possible directions. For the example vector that is shown in Fig. 7.20 (right), the distortion must be normed to length $\sqrt{5}$ via the Pythagorean theorem (another way to phrase the 2-norm in the 2D example). In total, the distortion is given as

$$\Delta_{\text{NGM}} = \epsilon \cdot \left(\left(+\frac{2}{\sqrt{5}} \right) \cdot \hat{x}_1 + \left(+\frac{1}{\sqrt{5}} \right) \cdot \hat{x}_2 \right), \quad (7.11)$$

and because this vector (excluding the parameter ϵ in front of it) is already normed, the total impact simplifies to a fixed

$$L_{\Delta_{\text{NGM}}} = \epsilon. \quad (7.12)$$

This works in similar manner for more dimensions, but this example is chosen as one of those that can be drawn.

7.3.2.1 Expected Physics Applicability of Different Attacks

By definition, particular values of input features which are distorted with the FGSM attack will always be shifted by a fixed magnitude that has been defined a priori. Thus, FGSM shifts entire distributions either in positive or negative direction, the splitting into left and right is primarily dominated by a) the model under attack b) the flavour of the jet for which this feature is to be distorted (see Fig. 6.10 for an extreme example). Adversarial training tends to

react in a more symmetric manner, while nominal training has been observed to show highly asymmetric distorted shapes which reveal inversion of the expected physics [11]. With the NGM attack, no more sharp edges and predictable shifts are expected, independent of the model, as the magnitude can vary on a per-jet basis, taking into account the interdependence of features and their relevance for gradient ascent. This leads to spread out shifts for which a pattern can not easily be deduced, and therefore less visible manipulations, both when a human is inspecting such distributions, or when performing a statistical test for similarity.

An alternative, that is often mentioned together with FGSM, is Projected Gradient Descent (PGD) [15,293,310]. The idea is to iteratively apply one-step attacks, and eventually clip the resulting summed distortion with some limiting parameter and norm. It is an effective method that has been found to lead to higher robustness [293], but multiple iterations of FGSM are applied that essentially perform several ℓ_∞ -measured manipulations. The variants that were considered for jet tagging so far use the sign of the gradient for inner shifts within the PGD loop, and thus generate not two, but a multiple of two peak-structures in the distributions. If inspected as precisely as FGSM and NGM, these structures become easily identifiable as artificial or non-physical distortions. Only when introducing a jet-dependent clipping mechanism (like the 2-norm utilized in the author's NGM), PGD can be considered valid for physics applications. Otherwise, the problem of generating ℓ_∞ -bounded distortions showing up as slightly shifted copies of the original distribution, as in FGSM, is only amplified, while also increasing the runtime by the number of PGD-iterations. Unwanted behaviour resulting from iterative application of FGSM is illustrated in Fig. 7.21. Without introducing for example the 2-norm (as in NGM) somewhere in the algorithm, PGD-modified inputs are expected to accumulate at or around specific points on a grid, when starting the iteration with clean samples or even with smeared samples, respectively.

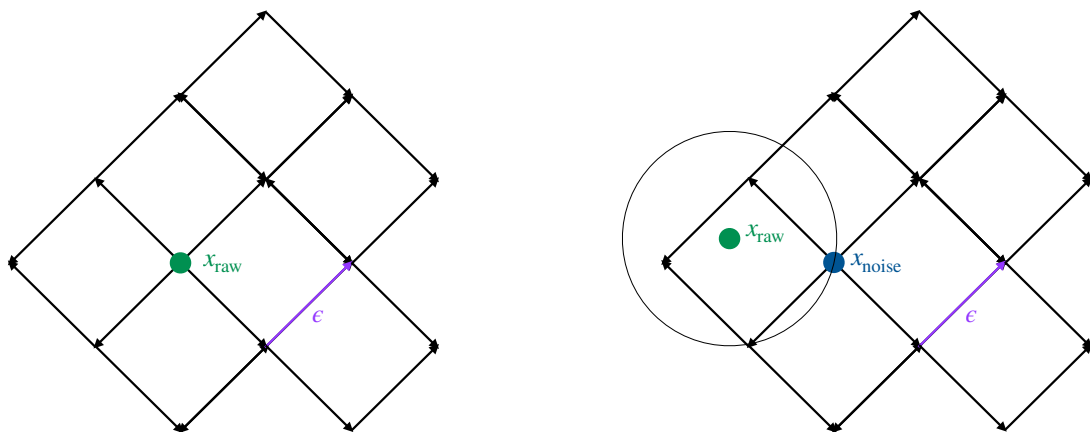


Figure 7.21: How a possible input modification with a simple PGD implementation looks like, when utilizing the ℓ_∞ norm (known from the FGSM attack) only, without (left) and with random initialization (right).

Another constraint is to produce “invisible” distortions, or distortions that are hidden well enough to not get caught with typical validation techniques. Besides that, the premise is to find an attack that can also act as a “defense” against physical mismodelings, or rather the discrepancies between simulation and data. The effectiveness of NGM in that regard as a

member of the class of first-order attacks is studied in the following.

7.3.3 Setup for Adversarial Training

For adversarial training, NGM is employed as defense mechanism throughout the entire training phase in every epoch, the general ϵ parameter is fixed to 0.01.

The first training on Run 2 samples performs the NGM attack in every batch (the classical approach followed for every network studied so far). However, to improve throughput during training and push the model more towards higher performance while maintaining a reasonable level of robustness, the NGM attack is only used in every second batch for a retraining (still on Run 2 samples, later also for Run 3). Per epoch, half of the samples are clean, the other half consists of perturbed samples, similar to Ref. [310]. Also, there have been two different configurations making use either of the overall length of feature intervals with respect to a reference (only for the first training on Run 2 samples), or standard deviations of certain quantiles of intervals as pre-factor for every feature (Run 2 retraining and Run 3 training, with recomputed factors to accommodate for the switch from CHS to PUPPI jets, see Section 4.1.1 and references therein). This distinction between different, input-dependent ϵ -factors has been discussed in the last blue box of Sec. 7.2.1.

7.3.4 Tests on Simulated Run 2 Samples

As a first investigation, the (Robust)ParTAK4 algorithm is trained on Run 2 samples, and different configurations of inputs for inference are probed to study performance and robustness. A list of utilized samples for this study on simulation is given in Table B.4. In this case, an independent sample of hadronic top quark pair events is used and a selection of jet transverse momentum greater than 30 GeV is applied.

7.3.4.1 Performance / Robustness Study of (Robust)ParTAK4 with ROC curves

The ROC curves for the first training against NGM, with the first variant of input-dependent ϵ -factors, are given in Figs. 7.22, 7.24 and 7.26. For the retraining, the ROC curves are attached just after the ones for the first training, in Figs. 7.23, 7.25 and 7.27. To compare not only the robustness against the same attack with which the training has been augmented, also FGSM-attacked samples are injected into the nominal and NGM-augmented trainings. Between the “old” and “new” trainings and attacks, it is noticeable how the impact of the second variant of attacks is smaller for the nominal training. This confirms that the “new” ϵ -parameters are less likely to overestimate typical input mismodelings and are better suited to incorporate the individual features’ markedness, which otherwise is biased towards inputs with long tails[†]. Further studies will proceed with this updated training.

For the b-tagging task, this new training strategy is shown in Fig. 7.23 and it is revealed how both the nominal and adversarial training reach the same performance in terms of AUC score

[†]Note how this conclusion has been drawn from the impact on the nominal (blue), not the adversarial models (orange). This is because for the adversarial ones, two things have changed: using the attack only in every second batch, and using the new ϵ -values. For the nominal trainings, only the expected small variations from different network initializations are visible, which makes them a better proxy to study the impact of different ϵ -setup choices.

(0.978), and that the adversarial training has about twice better resilience against NGM or FGSM attacks. A similar observation can be made for c-tagging, the gains are most pronounced for high efficiencies and high mistagging rates.

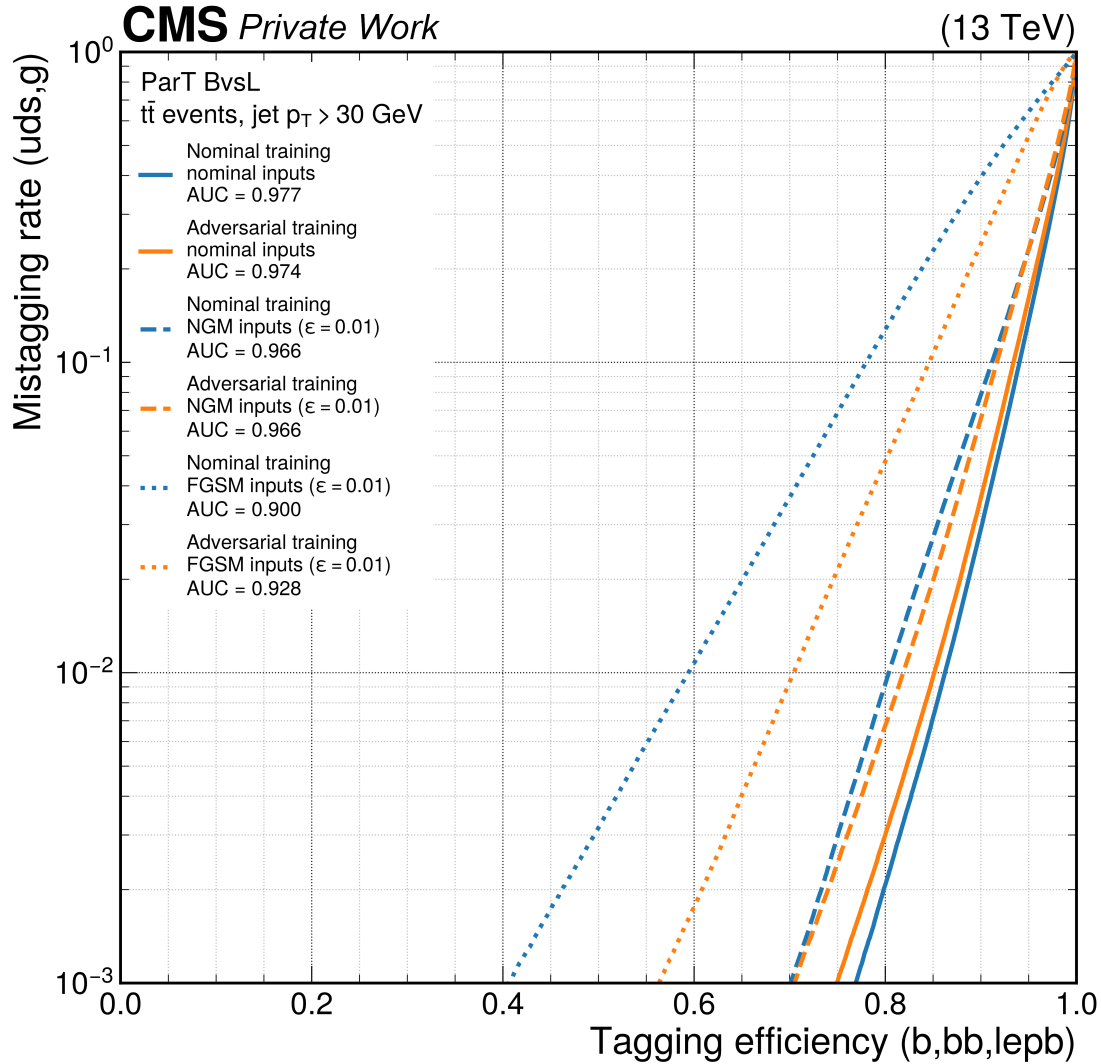


Figure 7.22: Performance of the first Run 2 (Robust)ParTAK4 algorithm for the BvsL classification task, comparing nominal (blue) and adversarial (orange) training on nominal inputs (solid lines), NGM inputs (dashed lines), and FGSM inputs (dotted lines).

7.3.4.2 A Correlation Study for Input Distributions and Tagger Discriminants: Probing Validity of Adversarial Attacks

In a quest for physically meaningful distortions and corresponding defenses, different strategies are validated one-dimensionally with histograms and multi-dimensionally to probe correlations.

A set of features (with discriminating power for different discrimination tasks) is chosen to study the impact of different attacks. Signed impact parameters are taken as a proxy to investigate observables separating heavy- from light-flavoured jets, pseudoangular distances that discriminate (light) quark from gluon jets are probed as well. Figure 7.28 shows the exemplary

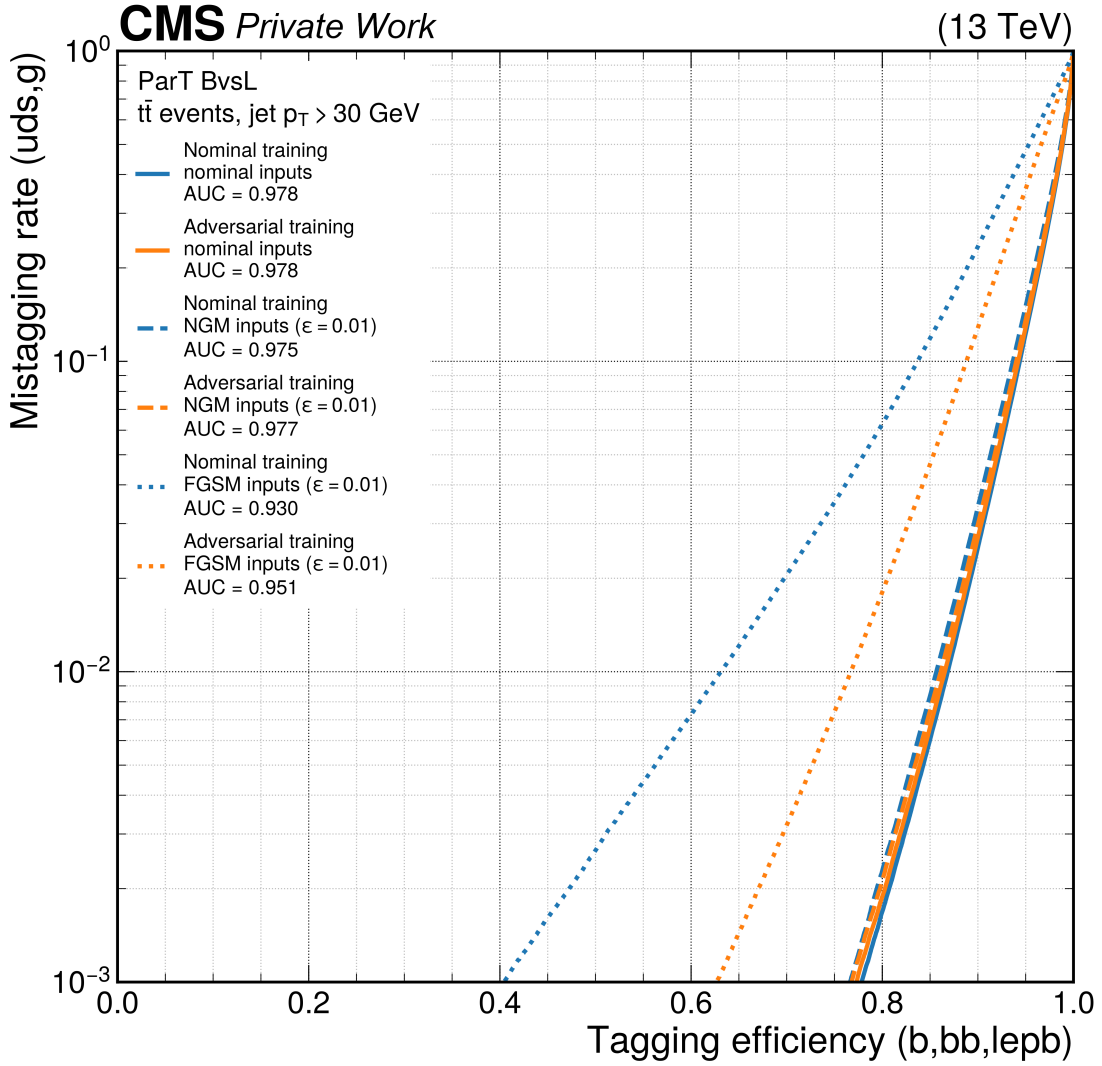


Figure 7.23: Performance of the retrained Run 2 (Robust)ParTAK4 algorithm for the BvsL classification task, comparing nominal (blue) and adversarial (orange) training on nominal inputs (solid lines), NGM inputs (dashed lines), and FGSM inputs (dotted lines).

nominal and distorted inputs, when performing an FGSM attack (on the left), or choosing the NGM attack (right). As expected, a feature that is sensitive to light quark or gluon contributions in angular distributions (see Sec. 4.2.2) reveals differences for these two classes between FGSM- and NGM-attacked samples. For example, the bins showing highest likelihoods in the nominal distribution for light-quark initiated jets still do for the overlaid NGM sample, while FGSM introduces a noticeable distortion, also for gluon jets shifted into the small angular distance region that should be dominated by the uds-category. As these distributions are normed to unit area, different behaviour can be observed, also for the SIP value for the second CPF in a jet[‡]: FGSM places many of the heavy-flavour jets close to zero, which leads to a clear peak structure dominated by one bin, while NGM produces practically no noticeable distortions in this distribution. Moreover, the tails of the distribution at positive values (heavy-flavour dominated) are not visibly affected.

In the 2D-scatter plot in Fig. 7.29, nominal inputs are taken as x axis coordinates, and distorted

[‡]The second track is chosen to achieve higher purity to study heavy-flavour contributions [220].

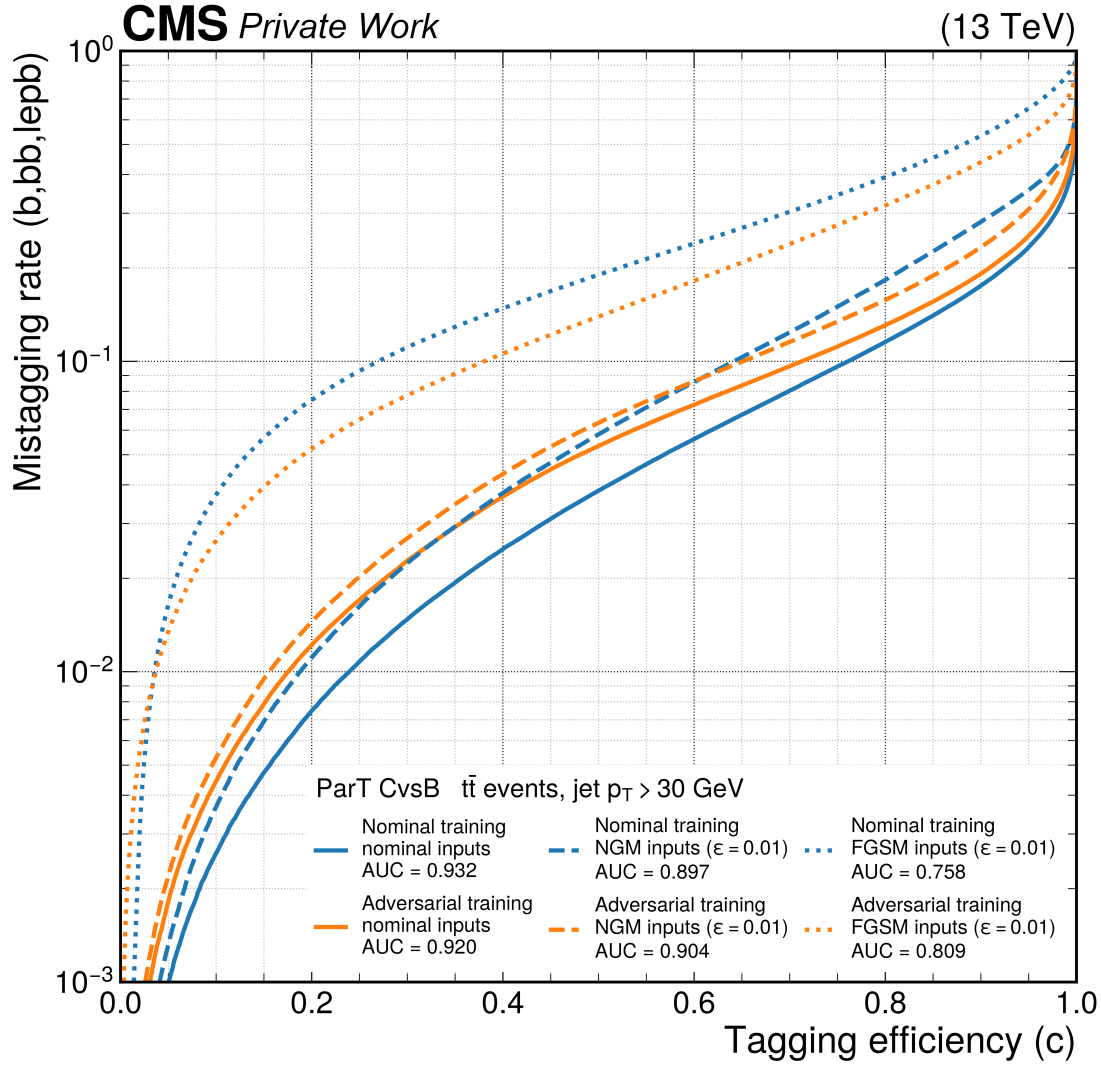


Figure 7.24: Performance of the first Run 2 (Robust)ParTAK4 algorithm for the CvsB classification task, comparing nominal (blue) and adversarial (orange) training on nominal inputs (solid lines), NGM inputs (dashed lines), and FGSM inputs (dotted lines).

inputs are shown on the y axis. For FGSM, inputs clearly only populate fixed and disconnected areas which are subject to the previously chosen magnitude of the distortion. Thus, the patterns reveal the linear nature of the attack (two straight lines), only a small fraction of jets is left unchanged, either as precaution to not shift any defaulted values (e.g. at 0) or if the gradient with respect to this feature for a particular jet vanishes. With the NGM attack, shifts are limited by the same overall magnitude, but the total shift is distributed among all features, therefore no sharp straight lines are observed. Instead, the distorted samples are scattered around their original values, without particular prominently populated areas besides the default value or jets with vanishing gradient in feature-dimension. Also for output scores (see Fig. 7.30), comparisons between nominal and NGM-attacked samples reveal the smoothening effect of this novel attack. Each panel corresponds to one output node or class, and thus, a primary colour in the flavour-split sample that should naturally appear at high output scores (upper right corner). One example, the second panel for the bb -category shows that the attack moves these jets downwards in this 2D representation, towards smaller output

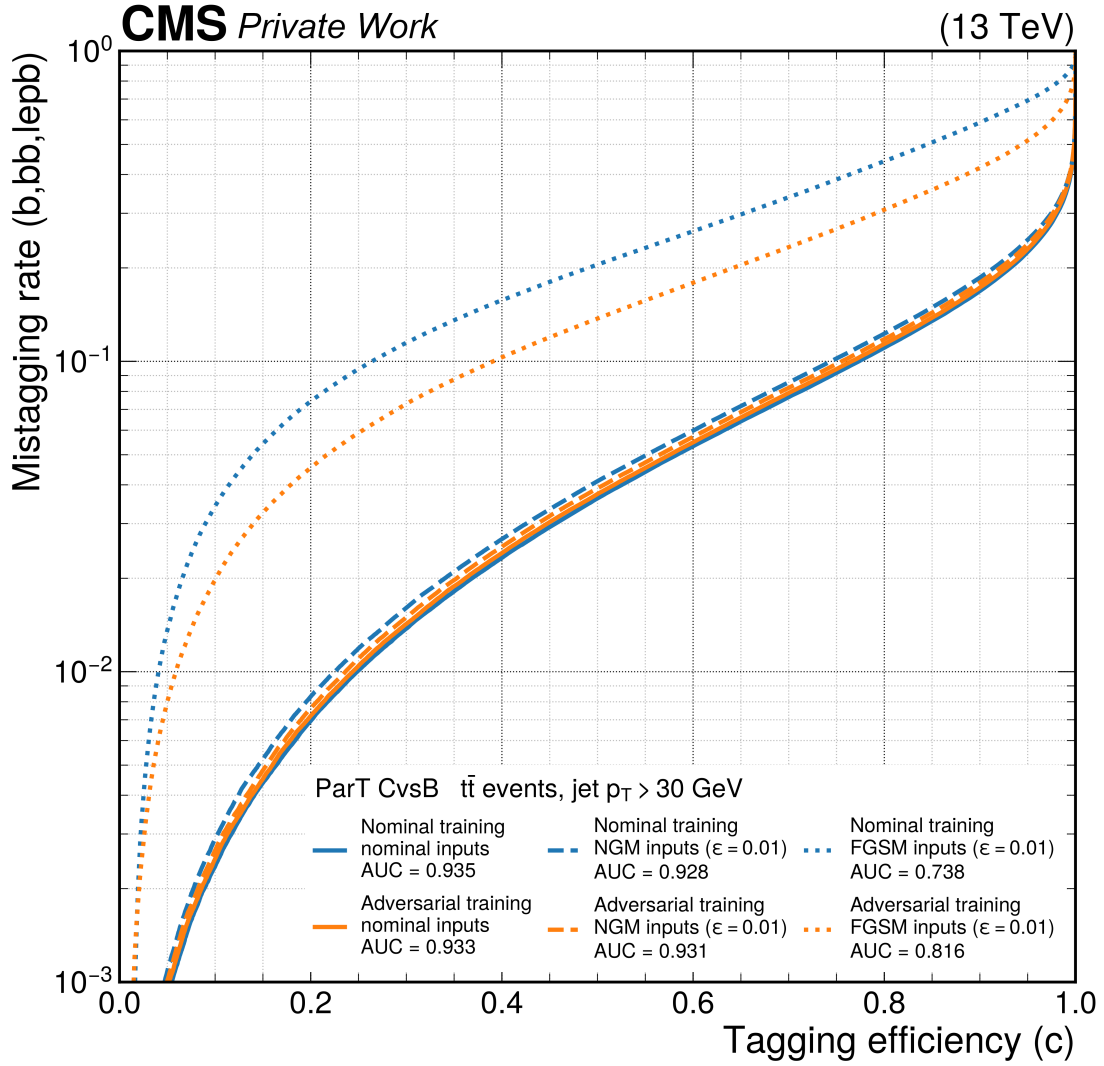


Figure 7.25: Performance of the retrained Run 2 (Robust)ParTAK4 algorithm for the CvsB classification task, comparing nominal (blue) and adversarial (orange) training on nominal inputs (solid lines), NGM inputs (dashed lines), and FGSM inputs (dotted lines).

scores (i.e. away from well-performing classification). On the other hand, many jets of negative classes (anything not bb) move upwards, over the diagonal through this panel, yielding further misclassification. Similar observations are possible with the other classes and panels, which underlines the adversarial behaviour of NGM samples.

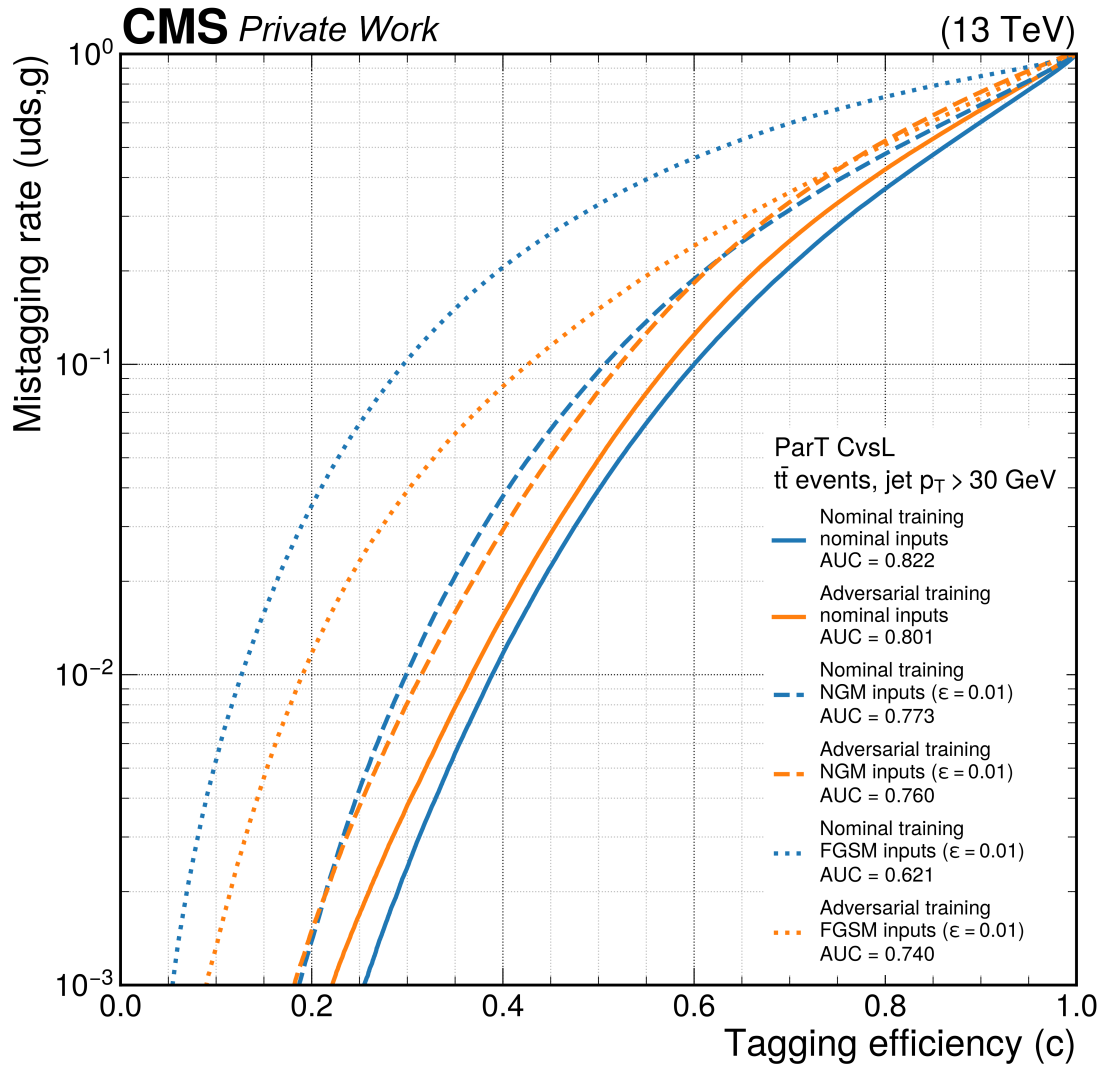


Figure 7.26: Performance of the first Run 2 (Robust)ParTAK4 algorithm for the CvsL classification task, comparing nominal (blue) and adversarial (orange) training on nominal inputs (solid lines), NGM inputs (dashed lines), and FGSM inputs (dotted lines).

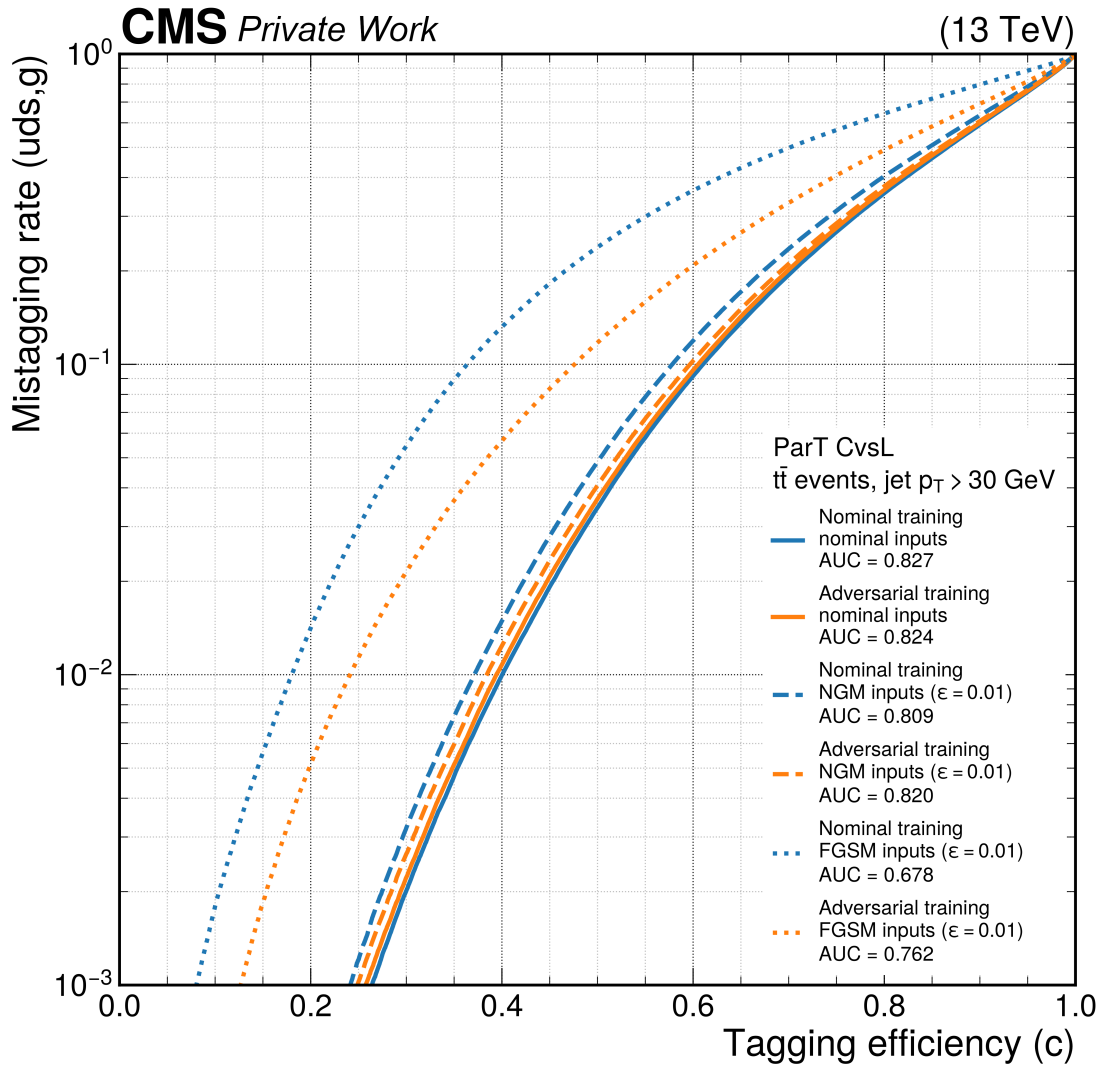


Figure 7.27: Performance of the retrained Run 2 (Robust)ParTAK4 algorithm for the CvsL classification task, comparing nominal (blue) and adversarial (orange) training on nominal inputs (solid lines), NGM inputs (dashed lines), and FGSM inputs (dotted lines).

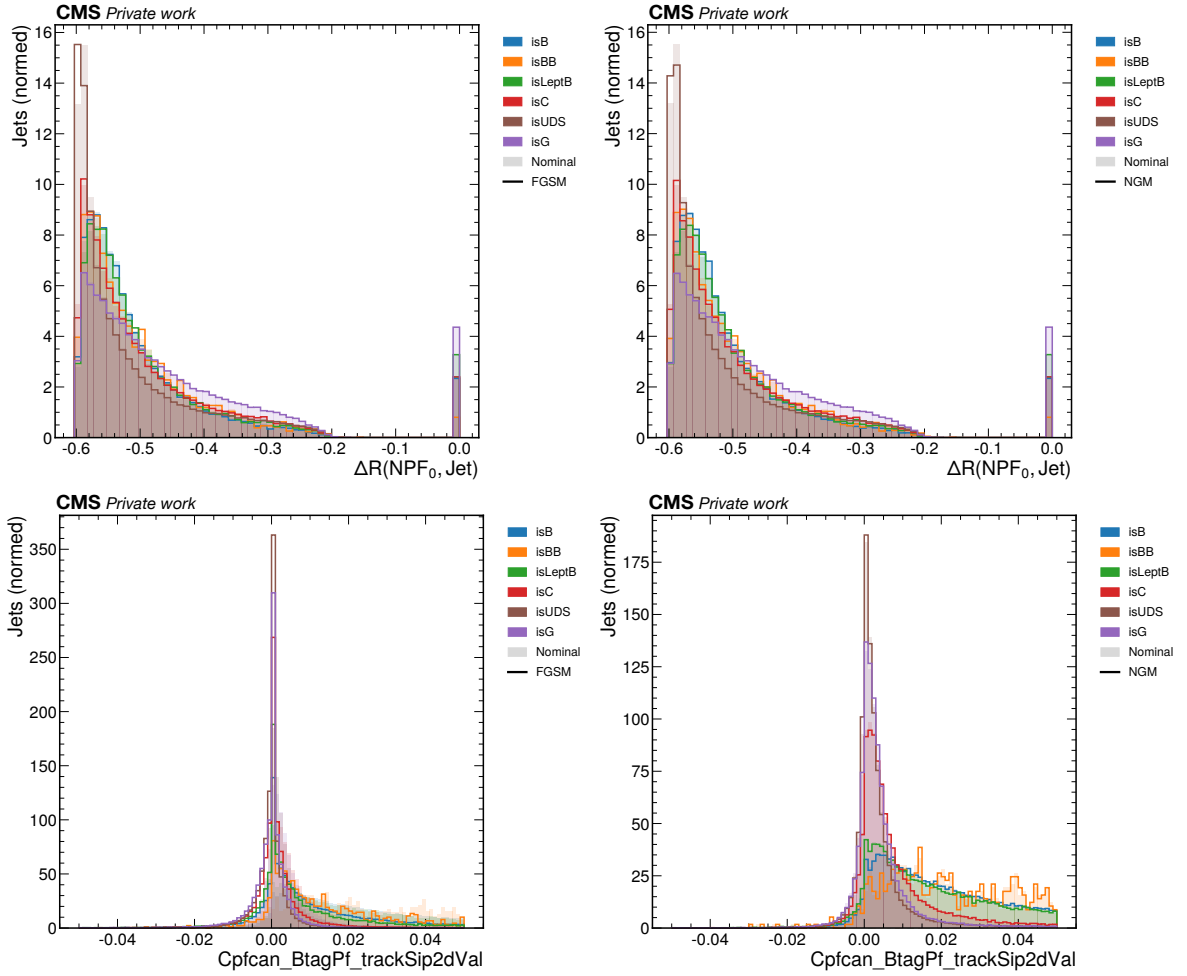


Figure 7.28: Distributions of pseudoangular distance of the first NPF candidate with respect to the jet (top) and 2D SIP value for the second track (bottom), comparing nominal inputs with systematically distorted inputs. FGSM attacks are overlaid in the left column, NGM-attacked inputs are shown on the right. 2D SIP values given in units of cm.

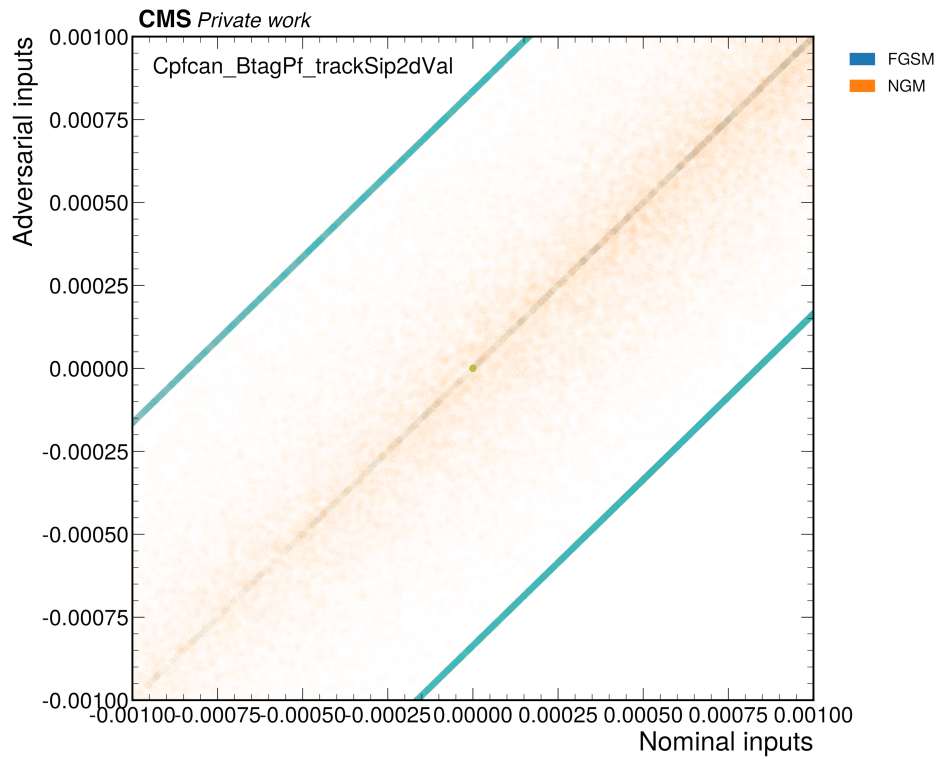


Figure 7.29: Correlation between nominal (x axis) and attacked (y axis) inputs using FGSM (blue) or NGM (orange), for the second charged PF candidate (track) SIP value in two dimensions, all values given in units of cm.

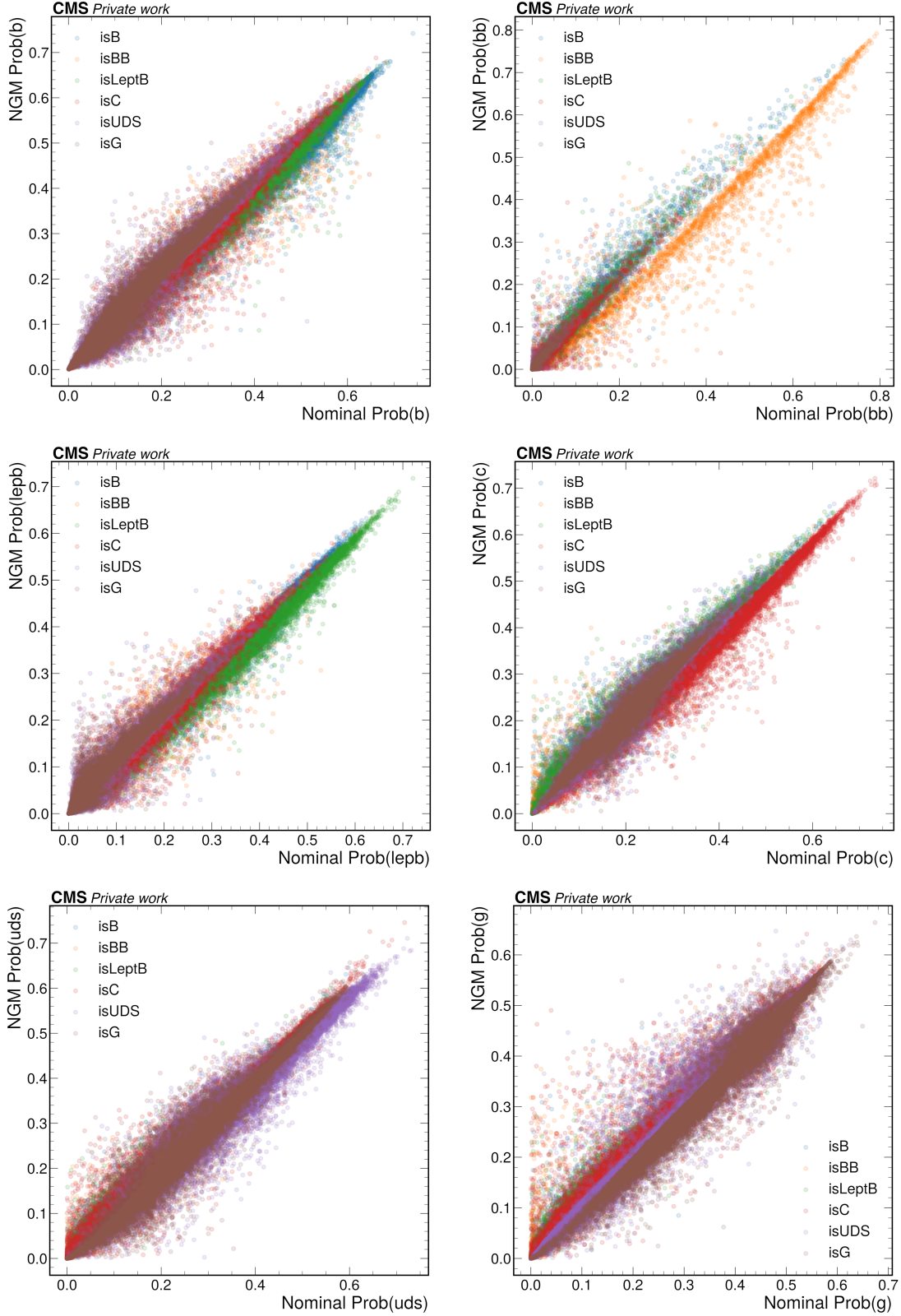


Figure 7.30: Impact of NGM attack on output scores of a nominal ParticleTransformer training, split into six ground truth flavours for b (bb) scores on the left (right) column in the first row, leptonic b (c) jets on the left (right) in the second row, uds (g) jets on the left (right) in the third row.

7.3.5 Validation with LHC Run 2 Data

These are the first and currently only data-to-simulation results with a recent neural network architecture, Particle Transformer, adapted to jet flavour tagging, comparing nominal with adversarial strategies. The same setup as for DeepJet [18] is used to evaluate the nominal and the adversarial transformer model on AK4 jets, using the light-flavour enriched selection as an example. To improve clarity, the "lep jets" category has been merged into the usdg category, as it has been found to be negligible in this selection (see also Figs. 7.6, 7.7, 7.8). For (Robust)ParTAK4, the respective results are shown in Figs. 7.31, 7.32, 7.33. Compared with DeepJet, the behaviour is qualitatively similar, adversarial training acts as a calibration technique without running the actual calibration method. In previous paragraphs, the nominal performance of the nominal and adversarial training has been compared and found to be nearly same, the discriminant distributions reveal how the two models achieve this with slightly different shapes, most noticeable for CvsB. For nominal training, simulation exceeds data in regions that should be dominated with light-flavoured jets in this light-jet selection. Again, as with DeepJet, this shows that nominal training does not generalize well to data, or overperforms on simulation (see for example the low BvsL region in Fig. 7.31, and similar for low CvsL scores in Fig. 7.33). Adversarial training, on the other hand, achieves ratios compatible with 1, with better agreement prior to any calibration. The discrepancy for nominal training is more pronounced for ParT than for DeepJet, and consequently, the effect of adversarial training is more pronounced as well in RobustParT. Future studies may repeat the

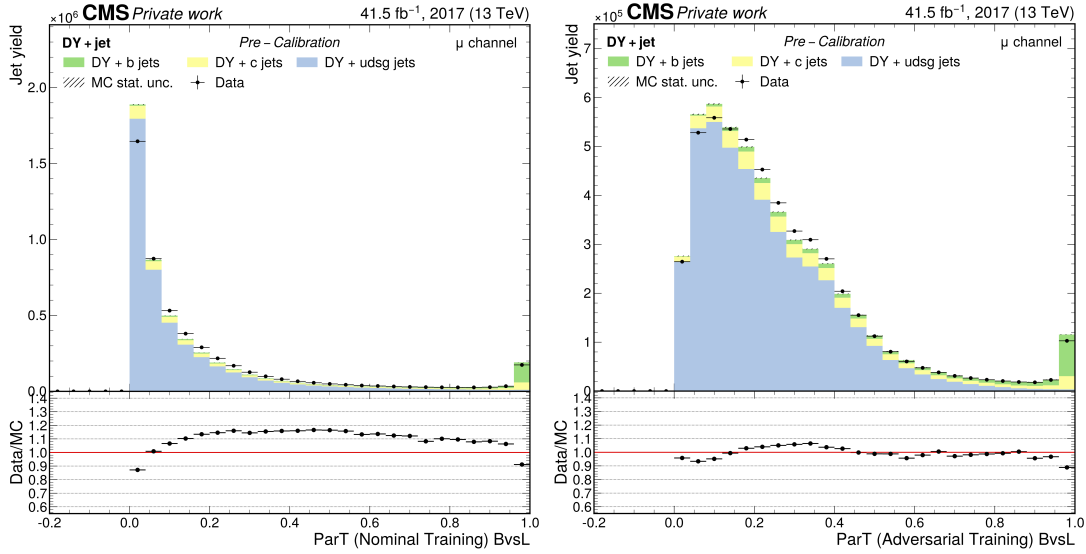


Figure 7.31: BvsL discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the light-flavour enriched selection.

same validation also in other phase space regions once all legacy samples are processed with the relevant RobustParT input features. The production of custom PFNano (user) samples and official campaigns focuses on Run 3 at the time of writing.

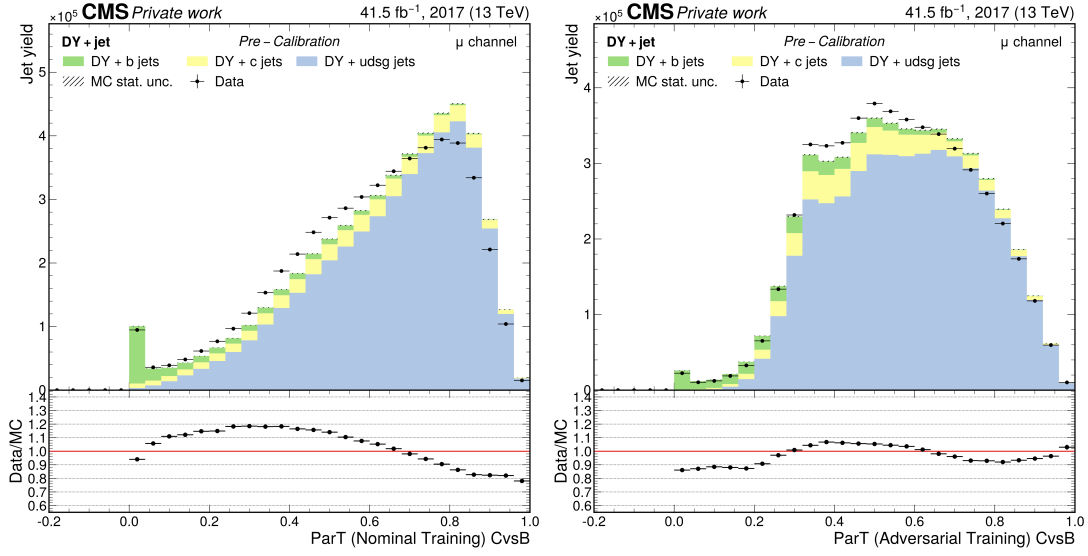


Figure 7.32: CvsB discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the light-flavour enriched selection.

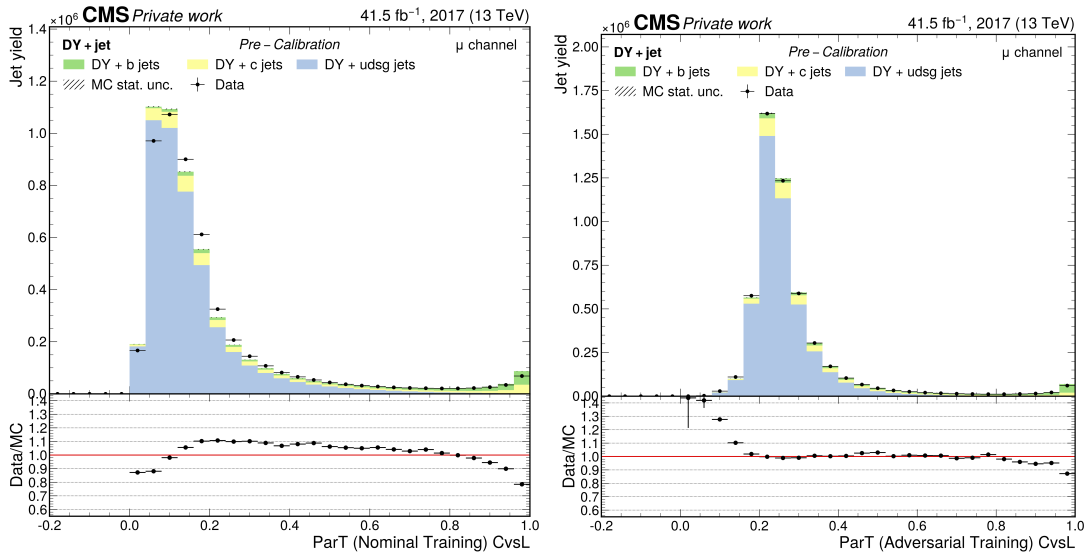


Figure 7.33: CvsL discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the light-flavour enriched selection.

7.3.6 Implementation for Run 3

7.3.6.1 Overview of Reconstruction Changes between Run 2 and Run 3

Along with several upgrades in terms of hardware, newly installed subdetector systems and improvements in computing infrastructures [139], Long Shutdown 2 also brought new algorithms for object reconstruction in Run 3, for example the change of the default AK4 jet collection from CHS to PUPPI jets, as discussed in Section 4.1.1. The jet collection from which training samples are taken should match the one that is used in the official reconstruction and the following physics analyses. Second, input variables describing momenta of hadronic constituents will need to be handled differently, multiplied with a dedicated weighting factor from the PUPPI algorithm. For this purpose, derived files for Run 3 studies are obtained from the list of datasets mentioned in Table B.5, which use PUPPI with the v15 tune. Otherwise, the data preparation steps are similar to what has been described for the initial Run 2 configuration. More detailed descriptions of the technical aspects related to changes between Run 2 and Run 3 RobustParT are available in Sec F.3.

7.3.6.2 Classification Performance Comparison across Algorithms and Impact of Different Adversarial Attacks

A first test is conducted, where the nominal training is compared against its adversarial counterpart, just using the original inputs without any distortion. This refers to a performance test with nominal samples for the two different trainings, otherwise all parameters are same. For this purpose, events matching the semi-leptonic top quark pair topology are chosen (see Tab. B.5), with the same selection criteria for jets as employed for Ref. [21] (using lower threshold for jet transverse momentum of 30 GeV). From Fig. 7.34 it is clear that the two trainings reach similar performance after convergence. The respective ROC curves overlap almost perfectly when separating either b from light jets, or b from c jets. A similar validation with QCD jet samples has been conducted (not shown here), only jets exceeding a transverse momentum of 300 GeV are selected. On these samples, nominal (adversarial) training achieves BvsL AUC scores of 0.9641 (0.9631) [22]. For BvsC, the AUC scores are 0.9080 (0.9058), respectively [22]. The result shows no qualitative differences compared to the top quark pair events. It is thus possible to use the robust transformer model for small-radius jet flavour tagging without significant loss of performance. When performing the same classification tasks with different generations of flavour tagging algorithms available for Run 3, their performance metrics can be studied choosing identical conditions and selection criteria. This is done in Fig. 7.35. Shown are ROC curves for DeepJet (black), ParticleNet (using the model trained for Run 2 (green) and a re-trained model for Run 3 samples (blue)), and RobustParT (red), all for small-radius jets with the PUPPI pileup-reduction method in version 15. In this order, ROC curves for BvsC separation appear closer to the lower right for the majority of the investigated efficiency/mistagging rate plane, meaning higher performance for the latest developments. Considering discrimination between b and light jets, RobustParT performs best for the full phase space, ParticleNet performance in its latest configuration deteriorates for this sample. This deviation involving charm jet misclassification as b jets for ParticleNet (Run 3) may be optimized with hyperparameter tuning, but is out of scope for this thesis. In summary, the robust transformer network training outperforms all other currently available algorithms

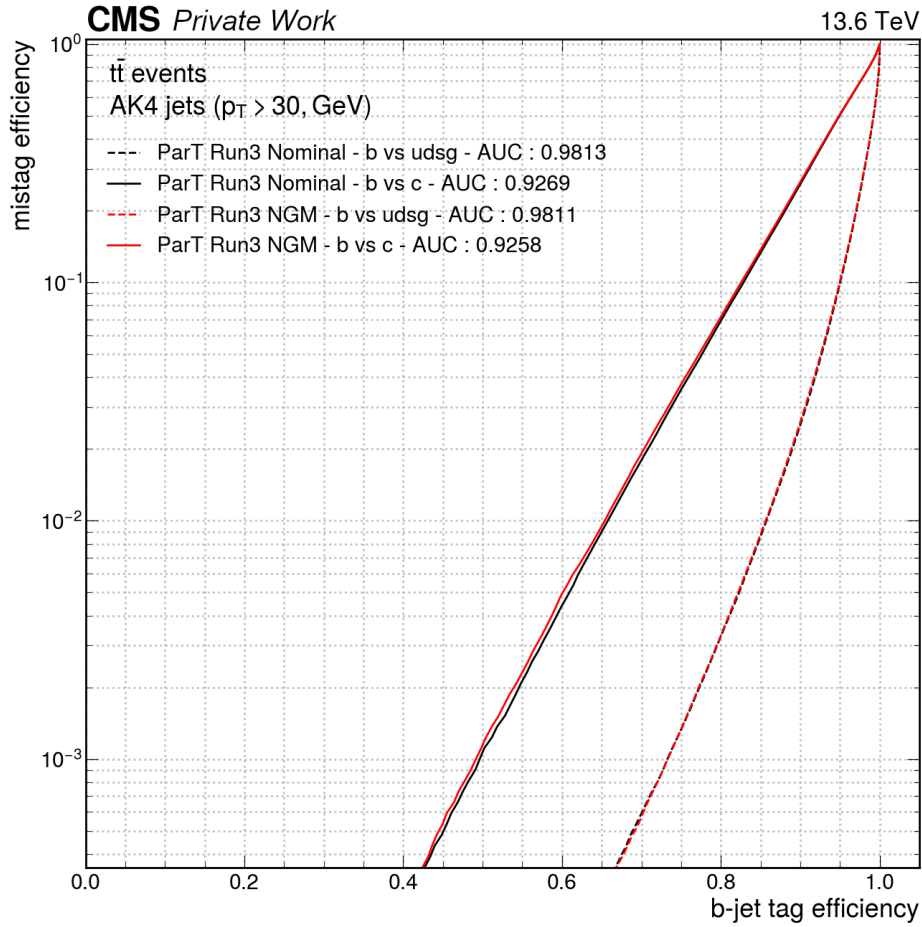


Figure 7.34: Performance impact of adversarial training against NGM for the CMS Run 3 (Robust)ParticleTransformer algorithm applied to AK4 jets in semi-leptonic $t\bar{t}$ samples. Figure obtained in collaboration with Alexandre de Moor, VUB.

in the Run 3 setup for AK 4 PUPPI jets for the BvsL task, and all but PNet (Run 3) in the very high efficiency region for the BvsC task. Nominal and adversarial training are studied with respect to susceptibility towards adversarial attacks. The adversarial training is performed in a defense mode against the NGM attack, thus, against a reasonable input modification in terms of physics properties. The application of FGSM represents a more severe attack compared to NGM and is expected to confuse the networks more drastically. This is confirmed with Fig. 7.36: using just BvsL (all dashed lines) as an example and comparing same colours, NGM attacks result in higher residual performance than FGSM attacks (FGSM is more aggressive, although the parameter for distortion is smaller). The pattern is repeated for BvsC (solid lines). Second, although the adversarial model (in red and blue) is only safeguarded against NGM, it does also withstand FGSM attacks better than the respective nominal model. Thus, NGM is an effective defense strategy for multiple first-order attacks in the context of jet flavour tagging, those representing physical manipulations, and those representing artificial worst-case scenarios.

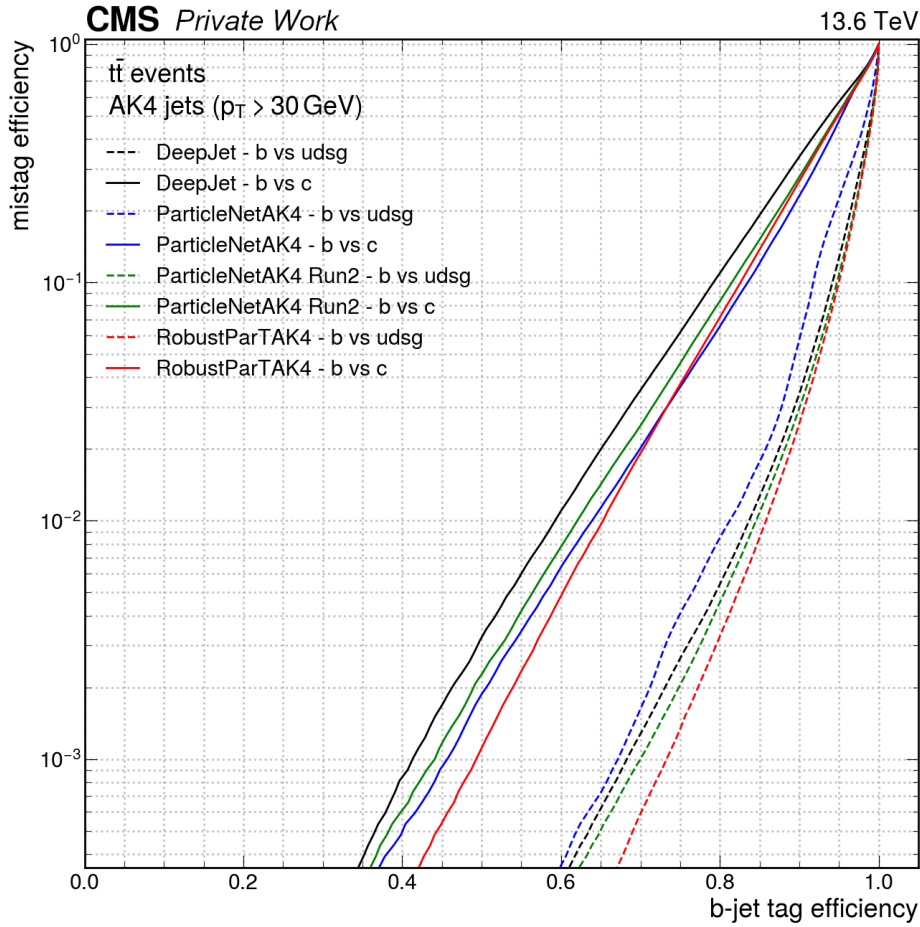


Figure 7.35: Performance of different b-tagging algorithms evaluated with CMS Run 3 simulation of AK4 jets from top quark pairs decaying semi-leptonically. Figure obtained in collaboration with Alexandre de Moor, VUB.

7.3.6.3 RobustParT and Beyond, a Summary

Previously discussed properties of RobustParT, like state-of-the-art performance and robustness, led to its inclusion for Run 3 reconstruction. It is now one of the standard algorithms provided for analyses utilizing flavour tagging of small-radius jets, it is the first robust neural network supported by CMS and available for MiniAOD and NanoAOD. The knowledge of data-to-simulation agreement in advance, prior to integration of either of the alternatives, is a useful piece of information that enhances the simulation-based metrics. This underlines the importance of a full framework which makes such a comparison possible, as the one further developed for this thesis, which was explained in Sec. D. Besides providing scale factors for the now integrated model, it is advisable to obtain the same also for the alternative training to gain valuable information. This should be the focus for the future, and should yield the full picture for Run 3 models of (Robust)ParT, ParticleNet, DeepJet and beyond. On the other hand, as this is the first and currently only robust identification algorithm integrated in CMS, other physics object identification algorithms could follow and profit from similar gains on

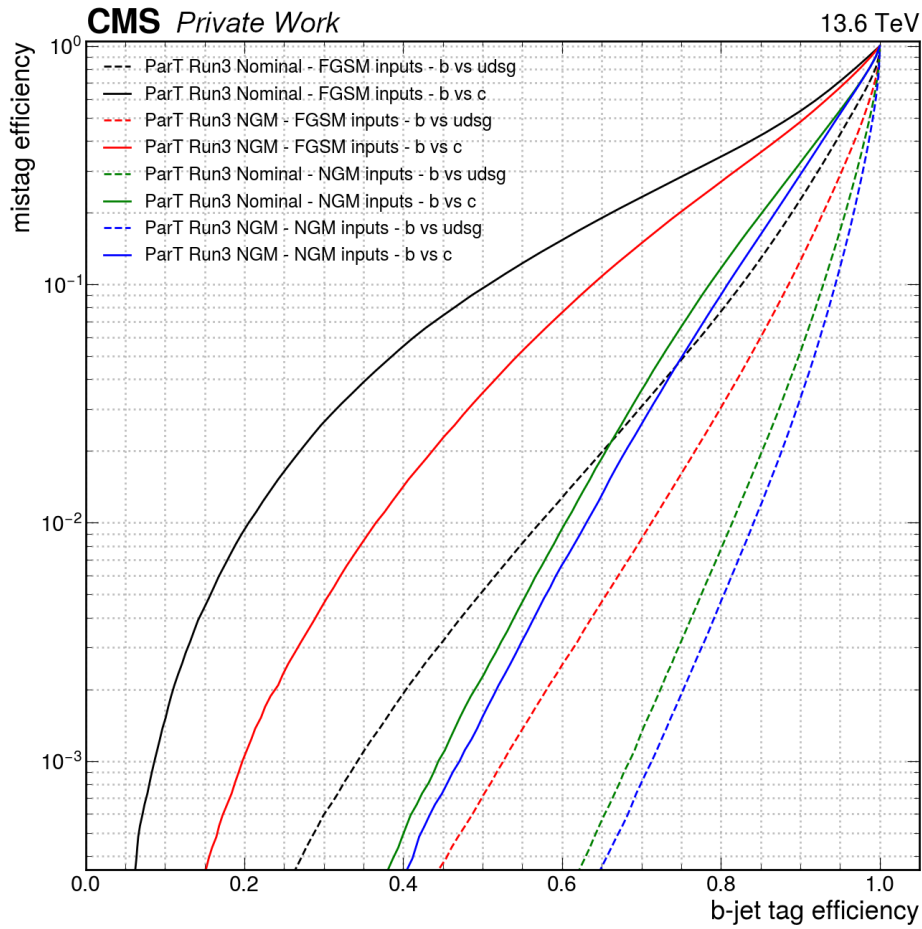


Figure 7.36: Impact of FGSM or NGM attacks on nominal training or adversarial training for (Robust)ParticleTransformer applied on AK4 jets using the Run 3 setup with simulated top quark pair events. Figure obtained in collaboration with Alexandre de Moor, VUB.

the performance/robustness metric. This includes for example large radius jet flavour identification, tau and top tagging or advances in the direction of full event identification.

7.4 Related Approaches

Besides adversarial training, the main technique studied in this thesis, there are also alternative techniques and supplementary projects proposed in the context of improving robustness and data/MC modeling of jet tagging algorithms.

7.4.1 Ablation Studies and Interpretability

In a parallel project [290], feature ablation techniques [345] have been employed to investigate how this affects the model performance and robustness. For the mentioned thesis that was supervised by the author, the software framework described in Sec. D is used to perform

multiple comparisons of DeepCSV-like models that use the full set of features, or only subsets of the full set[§]. This ablation technique is one deciding factor for the dependence between data/MC-JSD and ROC-AUC and could be repeated with the DeepJet and RobustParTAK4 algorithms as well, as the improved evaluation pipeline allows such studies for up to three different setups per analysis. As expected from Ref. [13], more features will result in a more susceptible network. The expectation is confirmed for jet tagging in this thesis with more complex models taking more features, and in Ref. [290], keeping the model fixed and varying the number of inputs. This may yield only tiny improvements in performance that can well be sacrificed [290].

7.4.2 Alternative Sample Distortions for Training, Taking Input Modeling into Account

Given that the new framework incorporates, besides output score and discriminant modeling, also input modeling in three control regions for the data tier that can be used for training, the input modeling itself can be used as additional information during training. In particular, every 1D histogram per flavour is accompanied by data/MC ratios. If, instead of artificial adversarial attacks that are derived for a specific model, the actual input modeling shall be corrected (individually per variable), a novel approach can be derived, visualized in Fig. 7.37. The ratios are understood as function values at the x axis grid points (knots) defined by the center of each input feature bin (step 1.), these points can be connected with a spline function f of degree n (step 2.), and the resulting function can be differentiated with respect to the feature axis (step 3.). To simplify the application, the derivative can be discretized per bin by taking the mean of the spline function (of degree $n - 1$) in the given interval $[a, b]$:

$$\bar{f} = \frac{1}{b-a} \int_a^b f(x) dx. \quad (7.13)$$

This results, for each feature and each flavour, in as many corrective weights as there are bins, which can directly be applied to augment feature distortions in the “correct” direction (step 4.). If the derivative does not exceed a pre-defined threshold (that could in the future be related to the uncertainty bands from commissioning results), one may assume a Gaussian noise term being efficient enough as distortion. For a jet with its nominal feature value falling into such a bin with positive slope / derivative surpassing the threshold value, the distortion would then take the mean derivative for that bin as a multiplicative factor in front of a previously fixed $\epsilon_{\text{modeling}}$.

The advantages of this method are that the lookup of corrective shifts related to input modeling can be done once, in a decoupled way before starting the training, and that this multidimensional distortion is independent of the model under consideration. Thus, the model training time will not be significantly different from a nominal training, as the DataLoader is already equipped with an additional column (e.g. to correct for flavour, pseudorapidity and

[§]The choice of features is aided by feature rankings related to the model under investigation. These can be either performed using the individual feature AUC scores (as prototyped with or without adversarial attacks in https://github.com/AnnikaStein/AI_Safety_2021/blob/master/new_march_21/auc_ranking_attack_combine.ipynb) or more advanced methods like Integrated Gradients (IG) [346], as followed for Ref. [290]. For more details on this method and applications in NN interpretability studies, also in HEP, the reader is referred to Refs. [287, 347–351].

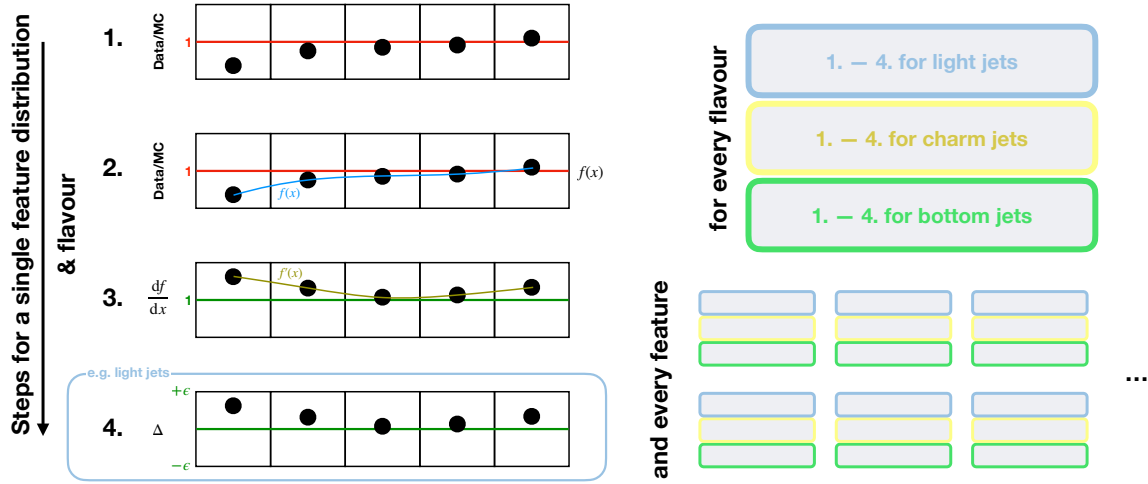


Figure 7.37: Step-by-step derivation of input shifts related to observed data/MC ratios per input feature and flavour.

transverse momentum), it is just a more complex additional correction that has been derived in advance. Even if this per-input correction requires keeping the computed distortions in memory until the model parameters are updated, the order of magnitude of resource usage will be comparable to that of first-order attacks (where gradients of the loss function with respect to the inputs also need to be kept in memory). To further speedup this technique, one may decide to only correct a subset of inputs which are found to have high discriminating power (independent of the model) or which are affected most by adversarial attacks (for a given model).

7.4.3 Flavour-Dependent Adversarial Training Strategy to Improve Data-MC Agreement in Three Control Regions

It has been shown that different control regions exhibit different mismodeling, both in terms of overall strength and appearance of slopes in ratios between data and simulation [19]. Therefore, using one ϵ value for all truth flavours is not optimal. While one can not guarantee that the introduced distortions will result in moving simulated inputs exactly in the direction that is needed to improve agreement with data, it is possible to change the severity of the attack according to the observed summary quantities when comparing the two domains (e.g. JS divergence), as first mentioned in the interpretation of Ref. [19]. One possible approach is to study the observed values for nominal training and from that decide for suitable starting values of ϵ_{udsg} , ϵ_{charm} and ϵ_{bottom} , i.e. the magnitude of the attack can be chosen independently per truth flavour. The exemplary instructions for a general flavour-dependent one-step attack with distortion vectors $\vec{D}_{1\text{-Step-Attack,flavour}}$ (obtained from some p -norm of the gradient of the loss function with respect to the inputs) are then:

$$\vec{x}_{1\text{-Step-Attack,udsg}} = \vec{x}_{\text{udsg}} + \epsilon_{\text{udsg}} \cdot \vec{D}_{1\text{-Step-Attack,udsg}}, \quad (7.14)$$

$$\vec{x}_{1\text{-Step-Attack,charm}} = \vec{x}_{\text{charm}} + \epsilon_{\text{charm}} \cdot \vec{D}_{1\text{-Step-Attack,charm}}, \quad (7.15)$$

$$\vec{x}_{1\text{-Step-Attack,bottom}} = \vec{x}_{\text{bottom}} + \epsilon_{\text{bottom}} \cdot \vec{D}_{1\text{-Step-Attack,bottom}}. \quad (7.16)$$

In the case of DeepJet, light-flavour is already tuned well, for charm jets, the parameters could be reduced to mitigate over-correction, and the b jet region (thus, the jets which are most easily identified in terms of ROC-AUC scores) needs larger distortions as performance in simulation is the highest. The one-size-fits-all concept instead is easy to implement, as it requires no

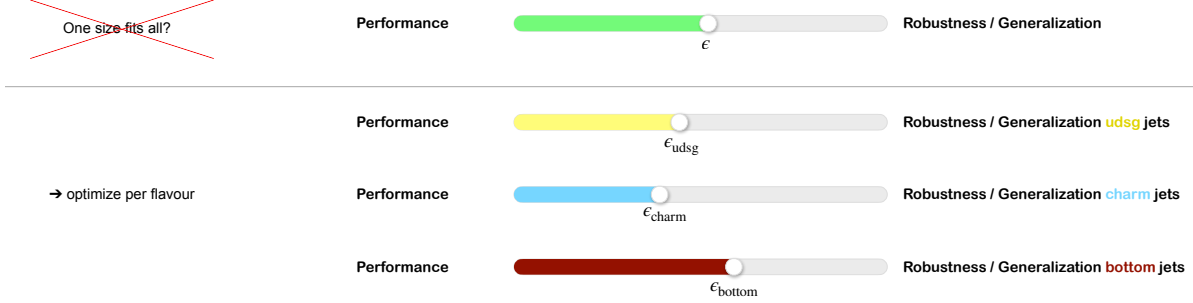


Figure 7.38: Schematic illustration of how different choices for ϵ_{udsg} , ϵ_{charm} and ϵ_{bottom} affect performance and robustness for different flavours.

knowledge of the input modeling in data. For example, when developing algorithms for a new run of the LHC during preceeding shutdown periods, there are no data available from which one could obtain suitable information per phase space. One either has to assume the behaviour will be same with respect to results of earlier runs, or stick to the same ϵ for all flavours. The latter has been done in the startup phase for 2023 data-taking at CMS, as not all relevant results with Run 3 conditions / scale factors had been obtained from 2022 data before deciding on the model for 2023 and beyond. Therefore, the official implementation still runs on same-size ϵ for all flavours.

7.4.4 Gradient-Reversal for Domain Adaptation

For related identification tasks such as tau tagging [175], there are techniques making use of gradient-reversal layers that incorporate knowledge of data-to-simulation agreement. This domain adaptation mechanism could be employed with the new framework introduced in this chapter, at least after every epoch, if not after every batch – as all the necessary information in the three sufficient phase space regions is available. One of the numeric quantities to describe the agreement between the training and inference domains (e.g. KL divergence, JS divergence, or χ^2) could be chosen to feed the loss function with a penalty term, introducing more hyper-parameters that control the relative importance of the three phase spaces, with the default being that all three regions contribute equally. Similar to the flavour-dependent attack, the weights per flavour are tunable and may be either guided by statistical properties (for example, if there is a noticeable difference between the number of events in each region), or by already known scale factor results of previous iterations with a similar / nominal model. This would act as a semi-automated calibration while training and could be an interesting approach for future studies, as it combines multiple strategies into one (flavour-dependence and physically meaningful adaptation of the identification algorithm).

Part III

Open Data Studies

Preparation of an Experiment-Independent Software Framework for Open Data

“This is where the world is going: direct access from anywhere to any type of data, whether it’s a small piece of data or a small answer but a long algorithm to create that answer. The user doesn’t care about this.”

– Hasso Plattner

8.1 Introduction and Problem Statement

Many of the studies presented so far are either based on simplified, fast simulation only (Chapter 6), or very experiment-specific and not accessible or reproducible for a broader scientific audience (Chapter 7). Thus, the goal is to provide the necessary infrastructure to perform such studies not only with full simulation, but also with detector data, accessible to the general public. At the CERN Open Data Portal (ODP, <http://opendata.cern.ch/>), many datasets from LHC experiments and others are available free of charge and under the terms of the CC0 waiver, which makes them perfectly suited for this purpose that also involves redistribution for the general ML community. Using the available data as is can be considered as a first step in this direction. However, there is no dataset so far that both includes the inputs and targets as needed to reproduce any of the small-radius jet tagging studies in this thesis, and all existing datasets, from which one can derive the relevant features, require expert knowledge of the experiment software, in this case, of CMSSW [160].

Such difficulties are not limited to one experiment or use case. While several individual working groups may use their own respective software tools to first of all train their reconstruction algorithms, or investigate their robustness, it is beneficial to synchronize such efforts to overall improve the efficiency and profit from each partner’s experience. The caveats are the completely different data structures and complexity of dataset size, dimensionality, available computing infrastructures or machine learning frameworks in use. As such, this is a heterogeneous environment where employing one wrapper for multiple use cases marks a challenge on its own. There are experiments for which the data is already present in tabular format suitable for ML purposes (data that is stored in one of the common .csv or .hdf5 formats needs no further preparation to be used with PyTorch, for example [248]). When looking at the data from multipurpose detectors like CMS [7] or ATLAS [116], the main way to store and distribute samples is to use ROOT (.root files) [352]. This is related to the fundamental difference of HEP data with respect to purely tabular data or multichannel images as used for computer vision. In HEP, one oftentimes has to deal with inhomogeneous data for which a rectangular

representation is not suitable. There can be multiple objects per event, but their number is not fixed, and individual physics objects may differ in the number of non-defaulted properties or cross-references. There is plenty of data (their handling in the future will pose new questions, see for example Ref. [139, 156]), and zero-padding all of them to produce rectangular arrays is not feasible. The aim of this framework and data preparation for AI safety studies therefore involves a tradeoff between efficient storing and accessing of data, without too much reliance on the experiment-specific implementation of this to make the datasets “ML-friendly”. To do this, a wrapper is proposed which works with variable-sized inputs.

8.1.1 A Wrapper for Open Data AI Safety Studies

For efficient AI safety studies that can be applied to different data structures, new infrastructure needs to be created. Figure 8.1 summarizes a first implementation of that kind. The core

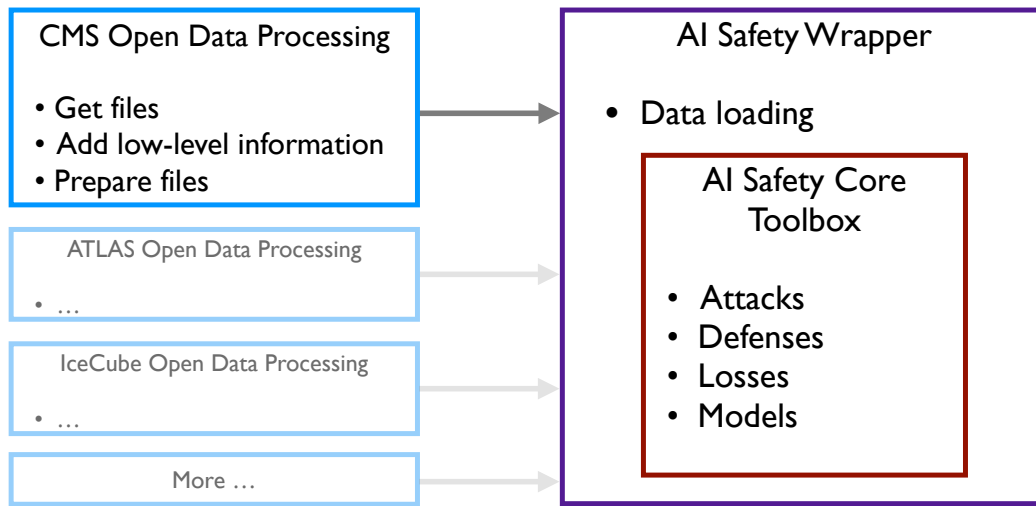


Figure 8.1: Proposed structure for the Open Data “AI Safety Wrapper” that reads data from various datasets and applies a set of common attacks or defenses in an enclosing framework.

idea is to split the elements that are different across experiments from those that are same. In the case of the AI safety project, this refers to individual teams preparing their respective data in such a way that the universal wrapper understands this data for further use. To show that this works as expected, both steps (data preparation and testing with experiment-independent tools) are described in the next sections.

8.1.2 Data Preparation

The data preparation step consists of two stages, one working directly with experiment software to add new information or read existing information, and a second part to convert the obtained datasets into a format suitable for the AI safety wrapper discussed above. Both is contained in a single package that will be introduced incrementally in the following sections.

8.1.2.1 Working with Low-Level Information in CMS Open Data (Experiment-Specific Part)

This section describes the implementation of custom inputs and targets for CMS Open Data, focusing on simulation / data related to the year 2015*. It is one possible realization of the blue box in Fig. 8.1. While inspired from the officially supported tool PFNano within the CMS collaboration [196], the PFNano package can not be used with the open MiniAOD data [317] from 2015 as is, because many of the accessed modules or algorithms were not included for that year. For example, some of the tagging algorithms that could be used for comparison were not invented yet and the datastructure misses many of the features that, in a best case scenario, could have been accessed and copied otherwise. Obtaining the necessary inputs is not a pure “Analysis” task. The implementation therefore requires two central modifications:

- opt-out of any producers aiming to access modern algorithms (building on work from Sneha Vireshwar Dixit[†] and Nick Smith[‡] of not using the respective functions by introducing a 2015 Open Data era), as they are not available and would yield errors at runtime,
- and add a custom producer that calculates the interesting features from first principles instead. As this is related to writing or checking-out a new C++ plugin, available at <https://github.com/ErUM-AISafety/OpenNano> and denoted OpenNano, the software needs to be compiled before usage. The relevant steps to be performed by the user are summarized in Listing 8.1.

```
source /cvmfs/cms.cern.ch/cmsset_default.sh
cmsrel CMSSW_10_6_30
cd CMSSW_10_6_30/src/
cmsenv
git cms-init
git cms-merge-topic 39040
git clone -b master git@github.com:ErUM-AISafety/OpenNano.git PhysicsTools/OpenNano
scram b -j 18
```

Listing 8.1: Setting up OpenNano on top of a Run 2 CMSSW version for sites mounting /cvmfs.

There is caveat related to building and using this tool standalone outside of officially supported computing sites for CMS, and a fix involves adding a local site configuration .xml file for the CMSSW environment. Examples are documented in the README file of OpenNano.

8.1.2.1.1 Accessing Open Data The CERN ODP itself acts both as database with which users can explore the individual datasets, and as a tool that offers filelists or files for direct download. Utilization slightly differs from the CMS Data Aggregation System (DAS) that employs NoSQL / MongoDB [353] to query the datasets and which is integrated into the experiment software, the ODP instead recommends using the `opendata-client` [354] tool which shall help with automatization. In principle, this tool offers the relevant functionality and can be installed with additional bindings for XRootD [355], however, the author’s tests of the client were limited by slow or temporarily unavailable database access through the `opendata-client`. Additionally, in order to obtain individual files associated with a certain process, the user

*See also <https://opendata.cern.ch/docs/cms-getting-started-2015> and <https://twiki.cern.ch/twiki/bin/view/CMSPublic/WorkBookMiniAOD2015>.

[†]See <https://github.com/cms-sw/cmssw/pull/39040>.

[‡]See <https://github.com/cms-jet/PFNano/pull/46>.

needs to search for the record ID on the web. The author therefore introduces a workaround with which one can perform all data exploration standalone on a remote computing cluster from the command line. The idea behind this is that all open data considered for this study is kept at the CERN /eos storage, therefore, remote filepaths will match a certain pattern. Many filepaths can be read at once and stored conveniently in a text file, one file per line, one example for semi-leptonic top quark pair events is given in Listing 8.2.

```
xrd fs eospublic.cern.ch ls /eos/opendata/cms/mc/RunIIFall15MiniAODv2/
TTToSemiLeptonic_13TeV-powheg/MINIAODSIM/
PU25nsData2015v1_76X_mcRun2_asymptotic_v12_ext1-v1/00000 > samples_MiniAOD/ttsemi.txt
```

Listing 8.2: Exemplary command to list and store multiple remote filepaths matching a pattern related to a specific physics process on the CERN /eos storage via XRootD [355].

8.1.2.1.2 Custom Producer At the core of the OpenNano package, there is a new plugin (inside OpenNano/plugins/CustomTaggerTableProducer.cc) in charge of modifying an existing table related to small-radius jets. Together with a header file related to truth jet flavour, this produces branches as required to perform small-radius jet tagging, without accessing the (non-existing) TagInfoCollection known from filling the inputs of new ML algorithms. Instead, all secondary vertices and PF candidates are accessed via the respective tokens and then associated with the jets by employing constituent loops within the jet loop of every event. To match the constituents to the jet in question, a criterion on the squared pseudoangular distance is placed to be less than the square of the jet radius ($0.4 \cdot 0.4$). The square is used for numerical precision reasons instead of employing square roots in the calculation. Up to 25 charged PF candidates, up to 25 neutral PF candidates, and up to 5 SVs are taken into account. Sorting of constituents follows the same principle as described for DeepJet in Sec. 7.2.1. Iterating through the jet constituents, two-dimensional vectors are filled for every input variable in question, employing standard measures to prevent injecting numerical infinities. For integer features related to counts of constituents / vertices, the datastructure is limited to one-dimensional vectors instead. The basic setup covers a large fraction of input variables related to DeepJet (see Tables B.8, B.9, B.10, B.11), except for global variables matching the pattern TagVarCSV (as used in DeepCSV), and using up to five instead of four SVs (closer to the Run 2 RobustParT algorithm).

8.1.2.1.3 Preprocessed or Raw Features The official datasets used to train or evaluate CMS tagging algorithms usually already have some preprocessing applied, such as replacements of defaulted values, or shifts of entire distributions to help with model convergence. To improve clarity and ensure comparisons of the given features are possible without already imposing unphysical modifications (such as shifting pseudoangular distances into negative values), the user can opt-out of using preprocessed quantities only and instead keep all features as they appear from first principles. In all officially supported versions of a similar framework, PFNano, this is not possible, as the accessed inputs are exactly how they enter to the respective networks, thus, in their preprocessed form. Internally, this is handled by a boolean flag. The implementation within the C++ producer involves additional lines, compared to standard tools used for the collaboration, see Listing 8.3. There are then multiple ways to switch to these non-clipped variables, either “hard-coding” the boolean flag in the C++ portion, or calling the producer with alternative arguments from the Python part of the package.

```

// a private member variable of the so-called
// JetConstituentTableProducerCustomTagger class:
private:
const bool add_CustomTagger_noclip_;

// for the constructor:
add_CustomTagger_noclip_(iConfig.getParameter<bool>("add_CustomTagger_noclip")),

// example for an individual feature to be computed from first principles
// without any preprocessing:
if (add_CustomTagger_noclip_) {
    sv_deltaR_nSV_noclip[i_sv_in_jet][i_jet] = std::fabs(reco::deltaR(*sv, jet_dir));
    // similar for all other variables
}

// putting noclip features into the Jet table, with hint that this is not
// preprocessed / clipped (unlike typical CMS jet tagging features):
for (unsigned int p = 0; p < n_sv_; p++) {
    auto s = std::to_string(p);
    if (add_CustomTagger_noclip_) {
        input_name = "CustomTagger_sv_deltaR_noclip_" + s;
        description = "pseudoangular distance between jet axis and the " + s + ". SV
direction (no clip)";
        customTable->addColumn<float>(input_name, sv_deltaR_nSV_noclip[p],
description, nanoaod::FlatTable::FloatColumn, 10);
        // similar for all other variables
    }
}

// necessary entry for fillDescriptions with default value (false)
// -> don't include the noclip features unless requested
desc.add<bool>("add_CustomTagger_noclip", false);

```

Listing 8.3: Implementation of additional physics-motivated, non-clipped features within the OpenNano/plugins/CustomTaggerTableProducer.cc module for OpenNano.

8.1.2.1.4 Configuration for the End-User Exemplary for a configuration snippet that calls the respective functions for a custom tagging algorithm for simulated samples, OpenNano introduces the following function to be applied on top of regular NanoAOD production from MiniAOD:

```

def Opennano_customizeMC_allPF_add_CustomTagger_and_Truth(process):
    addPFCands(process, True, True) # same in PFNano
    add_CustomTagger(process, True, keepInputs=['CustomTagger'], storeAK4Truth="yes")
    process.NANOADSIMoutput.fakeNameForCrab = cms.untracked.bool(True) # same in PFNano
    return process

```

Listing 8.4: Function definition in OpenNano to add new features and targets for custom algorithms. Whereas the first part addPFCands adds PF candidate information for all candidates, also those not associated with certain jets, and fakeNameForCrab is only relevant when running the tool with the CMS Remote Analysis Builder (CRAB) [7, 156, 356] facility of CMS, the main addition is the modification of the Jet table with inputs and fine-grained truth labels for a custom tagging algorithm with add_CustomTagger.

Such a function as defined in Listing 8.4 can be requested by users by mentioning it explicitly with a command line option for the cmsDriver command of CMSSW. The created configuration file steers the event loop and which tasks are performed with the MiniAOD input. In the case of OpenNano, this is already enough to perform local processing of individual files and receive a

.root file as output containing the newly computed branches. This is a “flat” file, a collection of columns within tables, not addressing the object-oriented CMS Event Data Model (EDM) anymore. With now established tools in the HEP community, such a file could be opened in experiment-independent ways, for example with the `pyroot` [357] or `uproot` [358] packages for Python. Expecting that also other users outside the HEP community will use the open low-level information datasets at some point, these data are converted into other portable formats suitable for ML purposes.

8.1.2.2 Converting the Data for Machine Learning Purposes and Batch Production

Three datastructures are implemented, into which the .root files can be converted: awkward-arrays saved as .parquet files, pandas-DataFrames saved as .pkl files, and NumPy-arrays saved as .npy files. Python scripts performing the conversion are provided with `OpenNano`, in different variants (`process_opennano{variant}.py`). The user can decide to convert the full event information, thus all information associated with any physics object, HLT paths and further metadata, or stick to the small-radius jets only. Of course, with a modification of the conversion script, the user can also decide for any other than the two provided default options by listing the required tables individually. Using jet-level information only is suitable for tasks just dealing with simulation, like training and basic simulation-performance evaluations, keeping other information as well is the method of choice when running physics analysis or performing data/MC comparisons with the custom algorithms (employing selections based on other physics objects, for example using muons, electrons, MET). As the individual tables (when using more than just jets) all require individual indexing, in that case, there are multiple pandas-DataFrames created, one for each table / physics object. With .parquet files for awkward-arrays, no such external splitting is necessary, all information per event is contained in a single file. Processing individual files is possible locally, as long as the packages for the desired datastructures are installed (the README contains instructions on how to obtain the relevant data science frameworks, for example using a package manager such as `conda` [359]).

For batch processing of large amounts of data, local production quickly reaches its limitations. Therefore, `OpenNano` also includes the relevant batch scripts to start a whole new pipeline without user interference, for example at a HPC cluster such as the RWTH Compute Cluster CLAIX [304] that employs the SLURM workload manager. The provided new conversion scripts can run independent of CRAB [356], and without allocating resources from the Worldwide LHC Computing Grid (WLCG) [360], thus independent of authentication via a Grid certificate. This maximizes portability and also lets independent researchers who only have access to their university’s computing center process large numbers of such files automatically. The `OpenNano` batch processing also works with multiple central processing units (CPUs) running per job, using multiple streams in parallel for CMSSW.

The conversion happens directly after creating each `OpenNano` ROOT file, never saving the large .root file on disk.

8.1.2.2.1 Comparison of Data Tiers and Data Formats: Practicability, Flexibility, Resource Usage When investigating the different means of storing the prepared files containing low-level information, there are multiple metrics to consider. On the one hand, the different

formats can be compared for their resource usage in terms of disk space. This aspect is of particular relevance for HEP applications, such as jet flavour tagging, because the machine learning algorithms are usually trained with hundreds of millions of samples that each have high dimensionality (hundreds of input features). Another crucial point, again of importance for the aforementioned field, are datasets with missing features or arrays with variable length. Nothing guarantees that each event holds the same number of jets, and there is no law that makes sure each and every jet contains exactly same numbers of charged or neutral candidates, or fitted vertices. Using dataformats or datatypes that fill non-existing values with placeholders are therefore sensitive to this particularity. One popular structure in scientific python applications are NumPy arrays, however, they are by construction rectangular and would block NaN-values with placeholders in memory or on disk, once saved. While less popular in the overall scientific community, the usage of awkward-arrays is now an established "pythonic" tool in HEP and core to whole analyses relying on columnar operations. For an exemplary file containing 632 columns (inputs or targets) and 714 jets, the three options result in different file sizes as shown in Table 8.1, and only awkward and pandas are suitable to maintain the sub-indexing of jets per event.

Table 8.1: Different storage options considered for OpenNano and size of resulting files containing small-radius jet information.

Storage option	Size [MB]
NumPy / .npy	3.5
pandas / .pkl	1.8
awkward / .parquet	0.79

8.2 Testing, Deploying, and a Note on the Educational Potential

The new datastructure associated with converted CMS Open Data parquet files has been tested by Taylor C. Briggs (exchange student, Columbia University) within the DAAD Rise Germany program 2023 in a two month long project, which the author supervised. Therefore, it is possible to assess what information and documentation is necessary to get people started with Open Data for jet flavour tagging, and conclusions on the educational potential can be drawn.

8.2.1 Preliminaries

First of all, as both the AI safety / ML aspects as well as jet flavour tagging were new concepts for the student, introductory material similar to the one for the CMS Data Analysis School[§] was provided, and existing code and strategies from Ref. [11] were discussed. The curriculum for the tutorial includes many of the aspects discussed in the "Foundations and Methods" part of this thesis, for example, different types of algorithms and their performance evaluation, understanding why certain input features behave the way they do (especially when splitting by flavour), and an introduction to how NNs learn from the given data. After a couple of

[§]Internal link to the author's presentation and material: <https://indico.cern.ch/event/1257234/timetable/#39-short-exercise-3b-flavour-t>.

weeks, the focus was then on applying the concepts to the new dataset, as now hosted on kaggle.

8.2.2 Creating the First Loss Surface Plot with CMS Open Data for Jet Flavour Tagging

8.2.2.1 Learning Objective

The goal of the DAAD project was to create a visualization of the loss surface similar to what has been done for Fig. 6.17, however, using CMS Open Data (i.e. the full CMS experiment simulation) in place of the Delphes simulation of the ATLAS detector. To do this, all preprocessing steps similar to Ref. [11] need to be done in order to repurpose the existing code in the new wrapper for this completely new dataset. The focus is on arriving at an interoperable script or notebook that reads this new data and, with modifications as tiny as possible, introduce a first measure for interpretability studies in this context. If even this complex task works, then the training loop and data loading of Ref. [11] will be applicable as well and will not require new tools.

8.2.2.2 Methods and Individual Competences Practiced while Working on this Task

Many individual skills are practiced, for example working with columnar data (awkward-arrays), understanding the difference between inputs, targets, model predictions, and spectator branches that are not used anywhere (e.g. `Jet_CustomTagger_jetIdx`, the jet index within an event), defining and using a NN (in PyTorch), preprocessing data to run inference or training with a NN, defining and using loss functions (in PyTorch), loading batches of data and plotting multidimensional graphics (using awkward-arrays, PyTorch tensors, and converting into NumPy-arrays for plotting with `matplotlib`, and picking appropriate axis ranges and scales, like logarithmic axes).

The model used for this first study on the newly prepared dataset is similar to that of Ref. [11], a FCNN with five hidden layers of 100 neurons each, where now 626 inputs are mapped to three possible output classes. ReLU activation is used for all hidden layers, the last layer uses Softmax activation, dropout is applied for hidden layers with a probability of 0.1 to zero-out individual neurons. Targets are obtained from PyTorch `LongTensor` objects created with the `hadronFlavour` column, mapping the latter from (5, 4, 0) to (1, 2, 3). As plotting the loss surface only works when injecting the scaled inputs in the same way the network would usually see them, preprocessing (e.g. standardization) needs to be applied when performing inference with the different inputs over which the loss is scanned. To later on show the behaviour in physically meaningful manner with the standard scales and units, the transformation is inverted before plotting.

8.2.2.3 Results

First of all, an intermediate result is application of the available preprocessing code from Ref. [11] to this dataset. Here, the flavour-split distributions have been standardized for further usage, exemplary features are presented in Fig. 8.2.

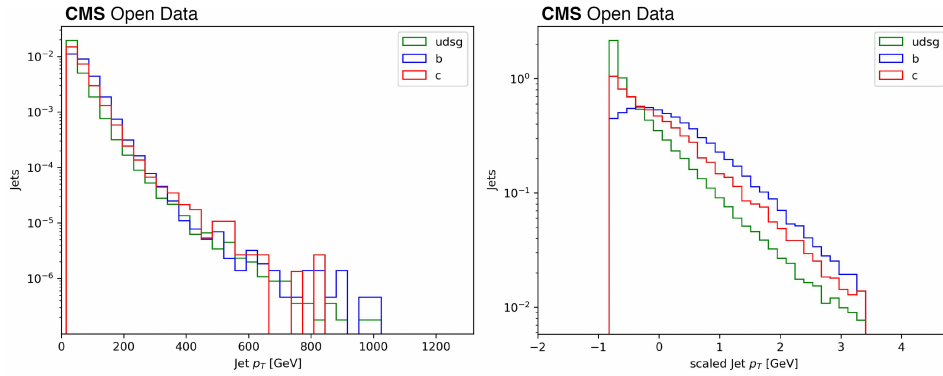


Figure 8.2: Distributions of jet transverse momentum before pre-processing (left) and after pre-processing (right). Figure obtained in collaboration with Taylor C. Briggs, Columbia U.

An outstanding question from Ref. [16] was how the loss surface looks like for untrained models, after having related the flatness of said surface to model robustness and performance. Figure 8.3 shows how the loss surface for an untrained jet flavour tagging algorithm can look like when picking jet transverse momentum and pseudorapidity, after random parameter initialization. The scale for the loss axis is arbitrary, and shall not be compared with that for

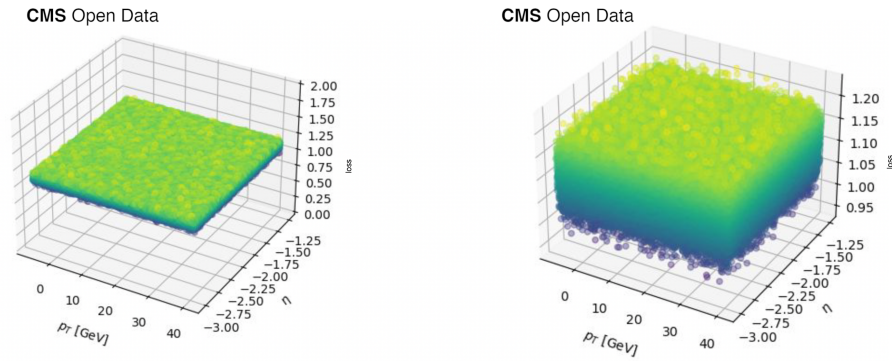


Figure 8.3: Loss surfaces as a function of jet transverse momentum and jet eta for the Open Data model and samples, when looking at the macroscopic scale (left) or focusing on microscopic fluctuations (right). Figure obtained in collaboration with Taylor C. Briggs, Columbia U.

other models, but the distinguishing properties can be read off nonetheless. While Fig. 6.17 shows a clear tendency of obliqueness with some slant angle for well-converged models, the new model studied in this section has a macroscopically flat geometry, with microscopic fluctuations. This can be related to starting out with an unbiased network that has not seen any jets during training yet. In a sense, this constitutes a closure test for the loss surface study, which is now available for a “zeroth” epoch, or after hundreds of epochs of training.

8.2.2.4 Applicability in Educational Contexts

Given the limited amount of time to walk through the introductory material and subsequently apply the new concepts to a new setup, the time-to-insight during this project was very short. In fact, the exchange student was the first to ever look at the given dataset. Having a certain,

achievable goal in mind [361] and a target result to compare against (i.e. the loss surface of a fully trained network) proved to be useful. The introduction did not include detailed instructions on how the experiment-specific software works, but instead, as the dataset can be used in standalone way, the student can directly jump into the project and work with high-level programming languages such as Python. As some of the existing AI safety code has to be adapted to this new dataset, the simplified data accessing does not compromise any physics learning objectives: the student still needs to understand how the features are to be interpreted to arrive at the solution, it is just the experiment-software overhead that is factored out of the learning process.

8.2.3 Deployment on kaggle

Providing datasets on kaggle requires creating an account and setting up an introductory page for the data, along with its metadata like sources, citation, license. When deploying datasets or code to work with these data on kaggle, the data preparation step is already finished. To offer ways to reproduce the dataset, all code is stored in a dedicated repository as well (see Sec. 8.1.2.1), but walking through this CMS-specific code is not mandatory to use the dataset for ML purposes. Thus, the user does not automatically know exactly what has happened to arrive at the present key-value pairs per jet. A basic documentation of what this dataset contains, how many samples are available, which physics process is used as example, and where to find explanations or definitions of the respective columns (eventually pointing to this thesis), is added as description of the dataset: <https://www.kaggle.com/datasets/annikastein/jet-tagging-opendata-mockup>. All added information yields a perfect usability score (10.0 out of 10.0) for the author's dataset on kaggle, which measures completeness, credibility, and compatibility. With kaggle as a general ML-platform, much of the physics knowledge is factored out, leaving a typical tabular-format-like dataset suitable for standard classification algorithms / supervised learning, as long as the event-level variables are flattened to jet-level. One file (245783 jets obtained from 33000 semi-leptonically decaying top quark pair events) is currently made available that contains the above-mentioned datastructure for low-level information related to jet flavour tagging, along with dedicated truth branches. The original entry of the MiniAOD file is specified at the CERN ODP <https://opendata.cern.ch/record/19966>[¶].

Users can either download the file directly, or work within kaggle by opening the sample notebook the author provides, obtained in collaboration with Taylor C. Briggs. On kaggle, there is a pre-installed Python 3 environment^{||} which already comes with most of the requirements, missing packages can be installed within the container image, in this case, `awkward-2.4.6` with its dependencies using the inline command `!pip install`. Just a few seconds later, the user can walk through the step-by-step instructions related to reading the example parquet file, getting jet-level information (flattening from event-level) and plotting the variables to get an overview of the dataset. At the end of the notebook, the user has acquired the necessary prerequisites to use the data in own projects. As shown above, such open datasets, once pre-

[¶]Quick access to the parent dataset is possible via `/eos/opendata/cms/mc/RunIIFall15MiniAODv2/TTToSemiLeptonic_13TeV-powheg/MINIAODSIM/PU25nsData2015v1_76X_mcRun2_asymptotic_v12_ext1-v1`, for sites mounting CERN EOS.

^{||}The corresponding Docker image is hosted at <https://github.com/kaggle/docker-python>, where CPU as well as GPU variants are offered. Some of the relevant packages are for example `pandas`, `matplotlib`, `PyTorch` (with CUDA [362] support).

pared with the process outlined in the kaggle notebook, can seamlessly be interfaced with existing code bases such as the one for Ref. [11], which wraps the individual experiment's data with the interoperable framework to study robustness or interpretability. On kaggle, both the dataset as well as the related notebook can be updated at any time, which is foreseen to happen few times per year.

8.2.4 Comparison with Other CERN / CMS Open Data Frameworks

The first difference between the new framework and existing tools is the deployment on kaggle that ensures availability for the entire ML community, without reliance on CERN- or CMS-specific software that would need to be understood by the user. Second, besides the domain knowledge factor, the barrier-to-entry is lower by focusing purely on pythonic tools that can be shipped as a single conda environment, or, even simpler, without requiring an individual setup and utilizing the kaggle notebook feature instead.

Considering the physics processes, if the user prefers to reproduce samples individually, at least every MiniAOD sample of the 2015 era at the ODP can be accessed (or later legacy samples from Run 2), and thus, there is no limitation to the author's choices of a certain topology. Regarding the dataset content, branches that are needed for an analysis or studies of small-radius jet tagging algorithms as close to the official configuration as possible, are enabled with OpenNano. This includes the derived quantities and object matching and selection criteria that are only added at MiniAOD level via cleaning and skimming tasks, which are not available for open datasets built from AOD instead (or only via a detour including several MiniAOD producers). One core implementation difference is working with so-called EDProducer classes, instead of EDAnalyzer classes as in the Physics Objects Extractor Tool (POET) [363]. The paradigm followed for OpenNano is to build more information than there is, and attach it to the existing tables, instead of reading and mapping to an output branch. The jet collection itself is fundamentally different compared to the most advanced branch of POET (for Run 1 2011 samples in AOD format), in that OpenNano works on typical CHS or PUPPI jets, with payloads for jet corrections by applying Physics Analysis Toolkit (PAT) functions^{**}, with now popular choice of $R = 0.4$ for clustering, while the established tool focuses on AK5 PFjets. In that sense, OpenNano is closer to the CMS Open Data HiggsToBBNTupleProducerTool [364] (which itself is derived from DeepNTuples [320] for AK8 tagging), both go to the low-level content for PAT jets and read MiniAOD from Run 2 as input. However, the DeepNtupleizerAK8 [364] is also not meant to include more than flat information per (fat) jet, the additional capacity of OpenNano regarding any other object collection also known from NanoAOD is not given for other available open data tools. Also, OpenNano offers the necessary flexibility to produce samples either locally or with a computing cluster, without relying on CERN computing sites (e.g. it adds the necessary steps before users can execute a CMSSW command such as the first one starting with cmsrel in Ref. [364]), and without relying on virtual machines. For those who do not want to produce and convert samples containing the low-level jet information, there are relevant instructions to work with reprocessed data in completely independent way^{††}, with

^{**}See also <https://twiki.cern.ch/twiki/bin/view/CMSPublic/SWGuidePATTools>.

^{††}For completeness, a conversion script that moves the .root files into the portable ML format HDF5 is also attached in Ref. [364] at https://github.com/cms-opendata-analyses/HiggsToBBNTupleProducerTool/blob/opendata_80X/NtupleAK8/scripts/convert-uproot-opendata.py, however, via the pandas package that needed more resources than awkward, see Table 8.1, with fixed placeholders for non-existing values.

kaggle as the hosting platform.

Part IV

Conclusion

Summary and Discussion

In this thesis, novel algorithms for jet flavour tagging have been introduced, implementing a set of adversarial attack and defense strategies to improve model robustness with respect to artificial mismodeling, and with respect to modeling differences between simulated samples and data recorded with the CMS detector at the CERN LHC.

First, the Fast Gradient Sign Method that is a well established first-order attack has been probed extensively with a fully-connected deep neural network for simulated samples. It has been found that a model trained against FGSM attacks withstands injected distortions better than a nominal model. The artificially introduced modelings for testing classifier robustness, constructed to be nearly invisible in distributions by imposing a physics-motivated constraint, were: FGSM-attacked samples, randomly distorted samples with noise drawn from a Gaussian distribution, and systematically shifted samples introducing uncorrelated up- or down-variations. The adversarial training, while reaching same performance as nominal training, has been found to outperform the nominal training on all types of mismodelings. Besides truly artificial scenarios, this constitutes better evaluation metrics for adversarial training also when probing systematic shifts closer to reality in the HEP experiment context, modeling systematic uncertainties. Besides that, the relation between performance and robustness has been studied by scanning both evaluation metrics at various points during training. The nominal training was found to be increasingly impacted by adversarial attacks, the better its performance. For the same test, adversarial training was found to achieve nearly constant robustness metrics for the entire performance range covered. This showed that there is a clear trade-off between model performance and robustness for a nominal model, while the adversarial model mitigates this effect to a large extent. With purposefully large distortions, the flavour-dependence of attacks imposed for a nominal and adversarial training was studied, which revealed inherent differences considering symmetry of the distorted distributions. This was later connected to differences in loss surface geometry by probing the two manifolds for nominal and adversarial training with a fine grid varying two input features, yielding the potential directions of the gradient vector. Two methods were derived for further studies: incorporating the curvature of the loss surface during training, or utilizing the direction of the gradient vector instead of taking its sign only, the latter was later implemented in the context of the CMS data studies. This first part of the thesis has been published in Refs. [11, 16].

Second, adversarial techniques were probed for the DeepJet algorithm for the first time, with realistic simulated samples and recorded data of the CMS experiment, focusing on legacy samples of the year 2017 during Run 2. A novel software framework incorporating training and evaluation in one was developed to learn more about modeling effects already during training of classifiers, taking investigations with detector data into account. The FGSM attack and defense strategies were modified to suit the convolutional neural network by de-

riving individual weights per input feature, taking statistical properties of their distributions into account. The pattern that had been found earlier, was confirmed also for DeepJet: with nominal and adversarial training reaching nearly same performance, but with a significantly better robustness of the adversarial training. As DeepJet has considerably more inputs than the FCNN classifier studied first, a finding known from computer vision was confirmed: the model susceptibility rises with the number of inputs. Using comparisons between simulation and data in three control regions with a binning scheme applied in for official scale factor derivation, and for three independent discriminants targeting b- and c-tagging, adversarial training has been found to yield better agreement in seven out of nine cases. It yields exceptionally large improvements for light-flavoured jets, with data/MC ratios compatible with 1 before application of any calibration method. For those summary quantities not improving with adversarial training, the distributions themselves were used to explain how adversarial training generalizes much better than nominal training, and this was in turn used to propose a novel flavour-dependent attack that shall target each flavour with a suitable impact. These DeepJet studies summarized so far have been documented in Ref. [19]. Further, scale factors for the light-flavour enriched region improved considerably for adversarial training, mostly in regions with high abundance of jets, with marginal improvements of scale factors for b- and c-jets. Aiming to explain why the adversarial training, admittedly using artificial distortions not connected to physics, outperforms nominal training in data/MC comparisons, selected input distributions were compared with and without applied attacks. Noticing how nominal training has a memorizing effect of the samples it was presented (clean simulation), adversarial training and corresponding adversarial attacks were found to mimic input mis-modeling to some extent. Based on this, an alternative adversarial input modeling strategy was developed as a suitable algorithm to take domain shift between data and simulation into account.

As a third research question, the recently proposed ParticleTransformer architecture was studied in terms of adversarial robustness, and all technical work to implement the first robust transformer network in the CMS software was performed. As the so-called “RobustParT” algorithm was set to incorporate more inputs than DeepJet and a complex architecture featuring multi-head attention blocks and pairwise features, the defense strategy against FGSM was replaced. Taking the proposed novel attack regarding directions of gradient vectors and correlations of features into account, the Normed Gradient Method (NGM) was introduced as a suitable tool to achieve performance, robustness, and physically meaningful, invisible and unpredictable distortions. Improvements were observed in terms of adversarial robustness, and performance on detector data, focusing on the light-flavour enriched selection. Moving from Run 2 to Run 3 samples, technical contributions were made regarding change of jet collection related to changing from CHS to PUPPI as the pileup mitigation technique. RobustParT was found to not only outperform its direct predecessor, DeepJet, for small-radius jet flavour tagging, but also a comparatively complex ParticleNet tagger, both in terms of efficiency/mistagging rates, and in terms of throughput in the experiment software. Variable-sized inputs also in the C++ part of the implementation, feature ablation and the inherent properties of transformer networks to be computationally efficient contributed to this improvement. These collaborative developments and official integration were summarized for the collaboration in an internal note describing the architecture and adversarial attack [22].

The final contribution was the development of an experiment-independent framework to en-

able interpretability and robustness studies with CERN Open Data. Besides technical contributions involving extended producers for low-level feature calculation for small-radius jet tagging, a case study on its educational applicability and relevance was performed with the help of an exchange student. The existing software first introduced for the FCNN mentioned in the beginning was found to work equally well with a completely different type of dataset, now reading and evaluating CMS Open Data. Increasing its reach, the dataset was converted to an ML-friendly format and deployed on the ML platform kaggle. With purely pythonic tools, interested researchers can now utilize the data without any overhead related to domain-specific software such as CMSSW.

This thesis constitutes an in-depth study of novel adversarial techniques for modern deep learning algorithms for small-radius jet tagging, as used in the CMS experiment, with simulation and data. With RobustParT, the first adversarially robust algorithm was integrated into experimental HEP software. The study of data to simulation agreement to modify the training accordingly and derive suitable parameters for adversarial attacks and defenses, before the algorithm was officially integrated, was a first of its kind in CMS. Usually, jet flavour tagging algorithms are integrated in a linear fashion and any commissioning or scale factor derivations are performed with one, previously fixed model only, months after the training. Getting such results within hours and modifying the training paradigm accordingly was possible due to several technical contributions targeting usability of existing data tiers and making use of high-performance and high-throughput computing facilities.

In the future, the newly developed NGM method can be extended for other object identification tasks, as it is universally applicable to any physics object for which low-level information is used in ML classifiers. For the chosen application, current state-of-the-art performance has now been achieved with improved pre-calibration agreement between data and simulation for all flavours when averaging over the three considered discriminants. This will be a valuable property for any analyses utilizing this model. As Run 3 proceeds, RobustParT will be one of the standard algorithms of the collaboration. The deployment of reprocessed open data enables performing all previous studies for a broader audience, independent of any experiment software, when reading in the samples from the hosting platform kaggle. This is a crucial step forward towards reproducibility of ML-assisted object identification results, usually only accessible for a small group developing the respective algorithm in experimental context. This complete set of inputs and target branches for small-radius jet tagging will hopefully contribute to more exchange between computer scientists looking for other applications of novel adversarial methods, and physicists aiming to improve their classifier robustness and generalization capabilities with state-of-the-art techniques. The example featuring a short-termed project performed by an exchange student is a motivating illustration that the provided materials and documentation are sufficiently educative and concise.

Appendix



Further Details for Mathematical Construction of the SM

These representations are all taken from Ref. [26] and complement the concise notation in Section 2.1.

A.1 Generators of Lie Algebras

Strong Interaction (SU(3)) The Gell-Mann matrices representative of the gluon colour octet are:

$$\begin{aligned}\lambda_1 &= \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} & \lambda_2 &= \begin{pmatrix} 0 & -i & 0 \\ i & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} & \lambda_3 &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{pmatrix} & \lambda_4 &= \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} \\ \lambda_5 &= \begin{pmatrix} 0 & 0 & -i \\ 0 & 0 & 0 \\ i & 0 & 0 \end{pmatrix} & \lambda_6 &= \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} & \lambda_7 &= \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -i \\ 0 & i & 0 \end{pmatrix} & \lambda_8 &= \frac{1}{\sqrt{3}} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -2 \end{pmatrix}.\end{aligned}$$

Weak Interaction (SU(2)) The Pauli matrices are:

$$\sigma_1 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad \sigma_2 = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \quad \sigma_3 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

A.2 Dirac-Pauli Representation of the γ -matrices

$$\begin{aligned}\gamma^0 &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} & \gamma^1 &= \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & 0 \end{pmatrix} & \gamma^2 &= \begin{pmatrix} 0 & 0 & 0 & -i \\ 0 & 0 & i & 0 \\ 0 & i & 0 & 0 \\ -i & 0 & 0 & 0 \end{pmatrix} \\ \gamma^3 &= \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \\ -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}, \text{ and } \gamma^5 = i\gamma^0\gamma^1\gamma^2\gamma^3 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}\end{aligned}$$

Samples, Configuration, Target Definitions and Input Features

B.1 Simulated Delphes Dataset

B.1.1 Targets

Internally, a partonic definition of flavour is used to obtain the truth jet flavour from generator level particles. Matching is done against a cone of radius $R = 0.5$. Table B.1 lists the different classes with descriptions related to the constituents after hadronization, mainly for comparability with the other samples studied in this thesis. The classification is strictly hierarchical, starting from b, over c, to light jets.

Table B.1: Assigning truth flavour classes in `delphes-rave` to small-radius jets, ordered hierarchically.

Class	Code	Description
b	5	b jets, with one or multiple b hadrons (this does include contributions from gluon splitting into $b\bar{b}$)
c	4	c jet, independent of the number of c hadrons, contains no b hadron (this does include contributions from gluon splitting into $c\bar{c}$)
light	0	light quark- or gluon-initiated jet, contains neither b nor c hadrons

B.1.2 Input Variables

See Tables B.2, B.3.

Table B.2: Expert / high-level features for the neural network. The first two features represent high-level jet, the next four high-level track, and the remaining ones high-level vertex variables. Adapted from Refs. [8,301], previously shown in Ref. [11].

Short name	Description
Jet p_T	Transverse momentum of the jet with respect to the beam line
Jet η	jet pseudorapidity
Track 2 (3) d_0 (z_0) significance	Magnitude of impact parameter significance of the second (third) track, transverse to the (along the) beam line, after ranking them by $ d_0 $ significance
N tracks over d_0 threshold	Number of tracks with transverse impact parameter significance over 1.8
Jet Prob	Light jet probability (see Ref. [227]); product of likelihoods over all tracks to have come from a light quark jet
Jet width η (ϕ)	Width of the jet in η (ϕ) coordinates, obtained from all tracks in the jet via $\left(\frac{\sum_i p_{T,i} \Delta \eta_i^2}{\sum_i p_{T,i}} \right)^{1/2} \quad \left(\text{or} \left(\frac{\sum_i p_{T,i} \Delta \phi_i^2}{\sum_i p_{T,i}} \right)^{1/2} \right) \quad \text{with respect to the jet axis}$
Vertex significance	Weighted sum over displacement significances for all secondary vertices in the jet $\frac{\sum_i d_i / \sigma_i^2}{\sqrt{\sum_i 1 / \sigma_i^2}}$
N secondary vertices	Number of reconstructed secondary vertices in the jet
N secondary vertex tracks	Number of tracks associated to a secondary vertex, summed over all secondary vertices in the jet
Vertex ΔR	Sum of angular separation between the secondary vertices and the jet, weighted by the number of tracks at the SV $\frac{1}{\sum_i \text{SV}_{i_n\text{Tracks}}} \sum_i \text{SV}_{i_n\text{Tracks}} \cdot \Delta R(\text{Jet}, \text{SV}_i)$
Vertex mass	Decay chain mass, i.e. sum over all secondary vertex masses in the jet, under the pion mass hypothesis for reconstructed particles
Vertex energy fraction	Summed fractions of total track energy in the jet associated to secondary vertices

Table B.3: Low-level and intermediate-level features for the neural network. The first seven features represent low-level tracking, the remaining ones intermediate-level vertexing variables. These quantities are used for up to six tracks, ranked by signed transverse impact parameter significance. Vertex information is stored on a per-track level by utilizing a $N \rightarrow 1$ mapping. Adapted from Refs. [8,301], previously shown in Ref. [11].

Short name	Description
Track d_0	Impact parameter of the track, transverse to the beam line
Track z_0	Impact parameter of the track, along the beam line
Track ϕ	Azimuthal angle with respect to the beam axis
Track θ	Polar angle with respect to the beam axis
Track Q/p	Charge over momentum
Covariance between helix parameters	All 15 independent entries of the symmetric 5×5 covariance matrix between helix parameters of the track
Track weight	Probability for a track to be associated with the primary vertex
Vertex mass	Invariant mass of constituents used in secondary vertex fit
Vertex displacement	Displacement of secondary vertex in transverse direction with respect to the interaction point
Vertex displacement significance	Secondary vertex displacement divided by uncertainty of that displacement
N tracks	Number of tracks associated to the secondary vertex
Vertex-Jet $\Delta\eta$	Angular separation in η between jet axis and secondary vertex
Vertex-Jet $\Delta\phi$	Angular separation in ϕ between jet axis and secondary vertex
Vertex energy fraction	Fraction of jet energy carried by tracks associated to the secondary vertex

B.2 CMS Datasets

B.2.1 Sample Lists for Run 2 and Run 3 Studies

In this thesis, both Run 2 (Table B.4) as well as Run 3 (Table B.5) simulation is used to study jet flavour tagging algorithms. Additionally, with 2017 legacy samples, the performance in data (Table B.6) is investigated with datasets for which the HLT triggered on a single muon, or two muons.

B.2.2 DeepJet

B.2.2.1 Targets

Six distinct classes are used as targets when using DeepJet as CMS AK4 jet flavour tagging algorithm, for both Run 2 and Run3, namely *b* (bottom), *bb* (at least two b hadrons), *lepb* (leptonic bottom), *c* (charm), *uds* (light quark), *g* (gluon). In a sequence obtained from generator-level information via particle counting and distance calculations with respect to the jet, each jet is denoted with one of those six truth classes. There are minor contributions with (fake) jets having undefined flavour or misidentified leptons labeled as if they were actual jets, or contributions from pile-up. Such contributions are not used for training in the official setup in the DeepJet (Core) environment served with DeepNTuples. For attacks carried out within the custom full development framework, such "jets" need to be assigned to any of the six classes in order to allow the adversarial methods to run, which aim at local degradation of loss associated with one of the six possibilities. In such cases, the most logical class that comes to mind is light flavour (especially due to pile-up). With proper jet selection criteria in place for physics analysis or calibration, their contribution is eliminated to a large extent later on, making this re-assignment a harmless operation, considering the full picture. Since 2022, fine-grained truth level information is accessible also in PFNano, when the author of this thesis added the `FlavSplit` branch, first for versions suitable for Run 2 (CMSSW 106X), later also for Run 3 (versions 12 and 13). The custom new branch replaces individual flags per label stored as additional columns. This was done based on heuristics that encoding each precise label as a boolean referencing 1 byte each would be more costly than having this single integer column. Also, accessing only one branch instead of combinations of many branches can be more convenient for user code, requiring fewer operations. The definition is given in Table B.7. How the truth level distributions look like before and after application of suitable selection criteria to separate "fake" from "good" jets is shown in Fig. B.1. These diagrams can be read as multiple cutflow-diagrams (one colour represents a single new criterion for selection), split into the different truth flavours. Possible selection criteria, their combination and impact on the truth level distribution are studied, at first just considering the generator level information (available only for simulation) like `partonFlavour` and `hadronFlavour`. Then, common jet quality criteria are applied ("good"), followed by cleaning the sample from isolated muons* ("`isoMuon` cleaned"). In detail, the cleaned jets need to be reconstructed within the tracker acceptance (absolute value of jet pseudorapidity below 2.5), with $\text{jetID} \geq 5$, and either a transverse momentum above 50 GeV or $\text{puID} \geq 1$, pseudoangular distances between jets and isolated muons are at least 0.5, and the sum of muon energy fraction and neutral electromag-

*In this thesis, these are muons with $p_T > 30$ GeV, $|\eta| < 2.4$, $\text{tightID} > 0.5$, $\text{pFRelIso04_all} \leq 0.15$.

netic energy fraction are below 0.7, as per internal recommendations for Run 2 analyses corresponding to the dataset production campaign. The cuts are tested for top quark pairs in fully hadronic topology and for the Drell-Yan process with two leptons and additional jets, thus an example of a “V+jets” process. The introduction of criteria obtained from the *reconstructed* jets removes the majority of jets lacking the quality requirements and helps understanding where the contributions of irregular samples come from.

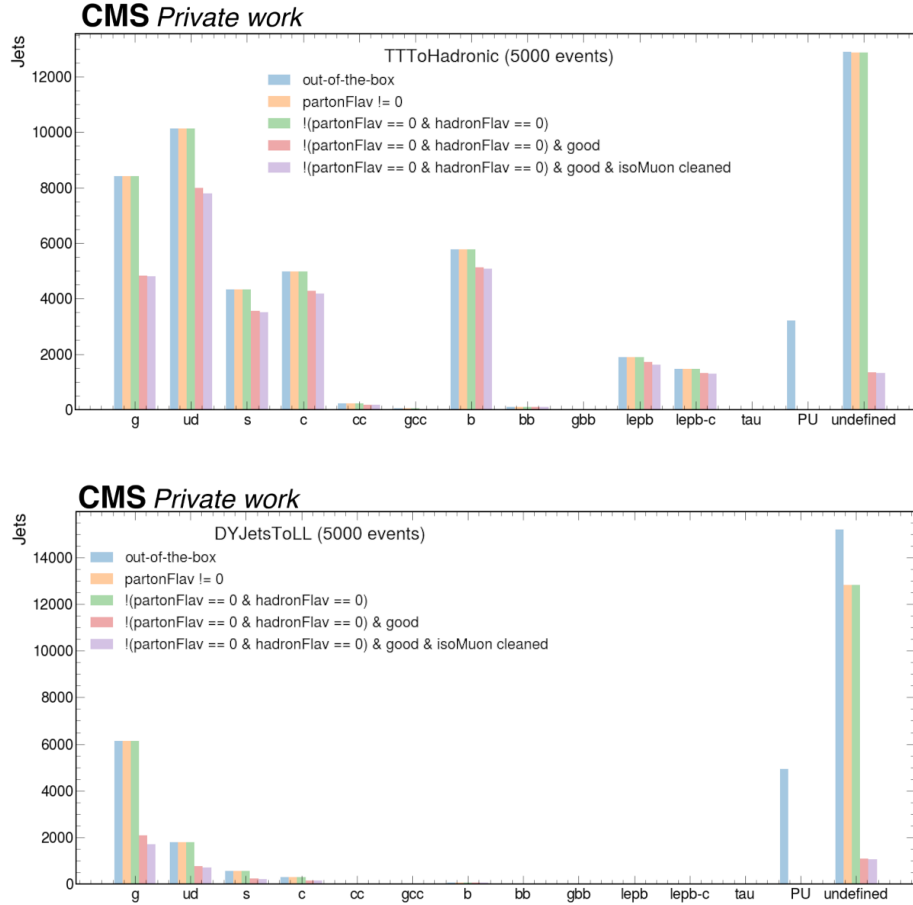


Figure B.1: Simulated PFNano samples for hadronic $t\bar{t}$ and Drell-Yan (into two leptons, with additional jets) processes, split into jet flavour and applying different selection criteria related to quality.

B.2.2.2 Input Variables

For DeepJet input variables, see Tables [B.8](#), [B.9](#), [B.10](#), [B.11](#).

B.2.3 RobustParticleTransformer

B.2.3.1 Targets

For AK4 tagging, RobustParticleTransformer uses the same target class definition as DeepJet.

B.2.3.2 Input Variables

For RobustParticleTransformer input variables, see Table [B.12](#). Most of the inputs of DeepJet (main exception: global features) are used, with few additions and minor differences between the Run 2 and Run 3 configuration.

Table B.4: List of simulated samples entering jet flavour tagging algorithm development and performance measurements, with usage specified in additional columns. Selections are denoted with DY, Wc, TT for light flavour, charm, and bottom jet phase spaces.

Name	Usage
<i>Simulation (DeepNTuples / PFNano samples)</i>	
/TTToHadronic_TuneCP5_13TeV-powheg-pythia8/RunIISummer20UL18MiniAOD-106X_upgrade2018_realistic_v11_L1v1-v2	Run 2 Training
/TTToSemileptonic_TuneCP5_13TeV-powheg-pythia8/RunIISummer20UL18MiniAOD-106X_upgrade2018_realistic_v11_L1v1-v2	Run 2 Training
/QCD_Pt_15to30_TuneCP5_13TeV_pythia8/RunIISummer20UL18MiniAOD-106X_upgrade2018_realistic_v11_L1v1-v2	Run 2 Training
/QCD_Pt_30to50_TuneCP5_13TeV_pythia8/RunIISummer20UL18MiniAOD-106X_upgrade2018_realistic_v11_L1v1-v2	Run 2 Training
/QCD_Pt_50to80_TuneCP5_13TeV_pythia8/RunIISummer20UL18MiniAOD-106X_upgrade2018_realistic_v11_L1v1-v2	Run 2 Training
/QCD_Pt_80to120_TuneCP5_13TeV_pythia8/RunIISummer20UL18MiniAOD-106X_upgrade2018_realistic_v11_L1v1-v2	Run 2 Training
/QCD_Pt_120to170_TuneCP5_13TeV_pythia8/RunIISummer20UL18MiniAOD-106X_upgrade2018_realistic_v11_L1v1-v2	Run 2 Training
/QCD_Pt_170to300_TuneCP5_13TeV_pythia8/RunIISummer20UL18MiniAOD-106X_upgrade2018_realistic_v11_L1v1-v2	Run 2 Training
/QCD_Pt_300to470_TuneCP5_13TeV_pythia8/RunIISummer20UL18MiniAOD-106X_upgrade2018_realistic_v11_L1v1-v2	Run 2 Training
/QCD_Pt_470to600_TuneCP5_13TeV_pythia8/RunIISummer20UL18MiniAOD-106X_upgrade2018_realistic_v11_L1v1-v2	Run 2 Training
/QCD_Pt_600to800_TuneCP5_13TeV_pythia8/RunIISummer20UL18MiniAOD-106X_upgrade2018_realistic_v11_L1v1-v2	Run 2 Training
/QCD_Pt_800to1000_TuneCP5_13TeV_pythia8/RunIISummer20UL18MiniAOD-106X_upgrade2018_realistic_v11_L1v1-v2	Run 2 Training
/QCD_Pt_1000to1400_TuneCP5_13TeV_pythia8/RunIISummer20UL18MiniAOD-106X_upgrade2018_realistic_v11_L1v1-v2	Run 2 Training
/QCD_Pt_1400to1800_TuneCP5_13TeV_pythia8/RunIISummer20UL18MiniAOD-106X_upgrade2018_realistic_v11_L1v1-v2	Run 2 Training
/QCD_Pt_1800to2400_TuneCP5_13TeV_pythia8/RunIISummer20UL18MiniAOD-106X_upgrade2018_realistic_v11_L1v1-v2	Run 2 Training
/QCD_Pt_2400to3200_TuneCP5_13TeV_pythia8/RunIISummer20UL18MiniAOD-106X_upgrade2018_realistic_v11_L1v1-v2	Run 2 Training
/DYJetsToLL_M-50_TuneCP5_13TeV-madgraphMLM-pythia8/RunIISummer19UL17MiniAOD-106X_mc2017_realistic_v6-v2	DY,Wc,TT (DeepJet)
/DYJetsToLL_M-50_TuneCP5_13TeV-madgraphMLM-pythia8/RunIISummer20UL17MiniAODv2-106X_mc2017_realistic_v9-v2	DY (RobustParT)
/ST_s-channel_4f_leptonDecays_TuneCP5_13TeV-amcatnlo-pythia8/RunIISummer20UL17MiniAOD-106X_mc2017_realistic_v6-v2	Wc,TT (DeepJet)
/ST_t-channel_antitop_4f_InclusiveDecays_TuneCP5_13TeV-powheg-madspin-pythia8/RunIISummer20UL17MiniAOD-106X_mc2017_realistic_v6-v1	Wc,TT (DeepJet)
/ST_t-channel_top_4f_InclusiveDecays_TuneCP5_13TeV-powheg-madspin-pythia8/RunIISummer20UL17MiniAOD-106X_mc2017_realistic_v6-v1	Wc,TT (DeepJet)
/ST_tW_antitop_5f_inclusiveDecays_TuneCP5_13TeV-powheg-pythia8/RunIISummer20UL17MiniAOD-106X_mc2017_realistic_v6-v1	Wc,TT (DeepJet)
/ST_tW_top_5f_inclusiveDecays_TuneCP5_13TeV-powheg-pythia8/RunIISummer20UL17MiniAOD-106X_mc2017_realistic_v6-v1	Wc,TT (DeepJet)
/TTTo2L2Nu_TuneCP5_13TeV-powheg-pythia8/RunIISummer20UL17MiniAOD-106X_mc2017_realistic_v6-v2	Wc,TT (DeepJet)
/TTToHadronic_TuneCP5_13TeV-powheg-pythia8/RunIISummer20UL17MiniAOD-106X_mc2017_realistic_v6-v2	Wc,TT (DeepJet)
/TTToSemileptonic_TuneCP5_13TeV-powheg-pythia8/RunIISummer20UL17MiniAOD-106X_mc2017_realistic_v6-v2	Wc,TT (DeepJet)
/WJetsToLNu_TuneCP5_13TeV-madgraphMLM-pythia8/RunIISummer20UL17MiniAOD-106X_mc2017_realistic_v6-v1	Wc,TT (DeepJet)

Table B.5: List of simulated samples for RobustParticleTransformer development for Run 3, always choosing the default `puppiJet` collection, for these samples with PUPPIv15.

Name
<i>Simulation (DeepNTuples samples only)</i>
/TTto2J1L1Nu_CP5_13p6TeV_powheg-pythia8/Run3Winter22MiniAOD-122X_mcRun3_2021_realistic_v9-v2
/QCD_Pt_50to80_TuneCP5_13p6TeV_pythia8/Run3Winter22MiniAOD-122X_mcRun3_2021_realistic_v9-v2
/QCD_Pt_80to120_TuneCP5_13p6TeV_pythia8/Run3Winter22MiniAOD-122X_mcRun3_2021_realistic_v9-v2
/QCD_Pt_120to170_TuneCP5_13p6TeV_pythia8/Run3Winter22MiniAOD-122X_mcRun3_2021_realistic_v9-v2
/QCD_Pt_170to300_TuneCP5_13p6TeV_pythia8/Run3Winter22MiniAOD-122X_mcRun3_2021_realistic_v9-v2
/QCD_Pt_300to470_TuneCP5_13p6TeV_pythia8/Run3Winter22MiniAOD-122X_mcRun3_2021_realistic_v9-v2
/QCD_Pt_470to600_TuneCP5_13p6TeV_pythia8/Run3Winter22MiniAOD-122X_mcRun3_2021_realistic_v9-v2
/QCD_Pt_600to800_TuneCP5_13p6TeV_pythia8/Run3Winter22MiniAOD-122X_mcRun3_2021_realistic_v9-v2
/QCD_Pt_800to1000_TuneCP5_13p6TeV_pythia8/Run3Winter22MiniAOD-122X_mcRun3_2021_realistic_v9-v2
/QCD_Pt_1000to1400_TuneCP5_13p6TeV_pythia8/Run3Winter22MiniAOD-122X_mcRun3_2021_realistic_v9-v2
/QCD_Pt_1400to1800_TuneCP5_13p6TeV_pythia8/Run3Winter22MiniAOD-122X_mcRun3_2021_realistic_v9-v2
/QCD_Pt_1800to2400_TuneCP5_13p6TeV_pythia8/Run3Winter22MiniAOD-122X_mcRun3_2021_realistic_v9-v2
/QCD_Pt_2400to3200_TuneCP5_13p6TeV_pythia8/Run3Winter22MiniAOD-122X_mcRun3_2021_realistic_v9-v2
/QCD_Pt_3200toInf_TuneCP5_13p6TeV_pythia8/Run3Winter22MiniAOD-122X_mcRun3_2021_realistic_v9-v2

Table B.6: List of recorded detector data, grouped by trigger paths (triggering two muons, or a single muon). These samples enter different selections to compare with simulation, denoted with DY, Wc, TT for light flavour, charm, and bottom jet phase spaces.

Name	HLT trigger path	Usage
<i>Data (PFNano samples only)</i>		
/DoubleMuon/Run2017B-09Aug2019_UL2017-v1	HLT_Mu17_TrkIsoVVL_Mu8_TrkIsoVVL_DZ_Mass8	DY (DeepJet)
/DoubleMuon/Run2017C-09Aug2019_UL2017-v1	HLT_Mu17_TrkIsoVVL_Mu8_TrkIsoVVL_DZ_Mass8	DY (DeepJet)
/DoubleMuon/Run2017D-09Aug2019_UL2017-v1	HLT_Mu17_TrkIsoVVL_Mu8_TrkIsoVVL_DZ_Mass8	DY (DeepJet)
/DoubleMuon/Run2017E-09Aug2019_UL2017-v1	HLT_Mu17_TrkIsoVVL_Mu8_TrkIsoVVL_DZ_Mass8	DY (DeepJet)
/DoubleMuon/Run2017F-09Aug2019_UL2017-v1	HLT_Mu17_TrkIsoVVL_Mu8_TrkIsoVVL_DZ_Mass8	DY (DeepJet)
/DoubleMuon/Run2017B-UL2017_MiniAODv2-v1	HLT_Mu17_TrkIsoVVL_Mu8_TrkIsoVVL_DZ_Mass8	DY (RobustParT)
/DoubleMuon/Run2017C-UL2017_MiniAODv2-v1	HLT_Mu17_TrkIsoVVL_Mu8_TrkIsoVVL_DZ_Mass8	DY (RobustParT)
/DoubleMuon/Run2017D-UL2017_MiniAODv2-v1	HLT_Mu17_TrkIsoVVL_Mu8_TrkIsoVVL_DZ_Mass8	DY (RobustParT)
/DoubleMuon/Run2017E-UL2017_MiniAODv2-v1	HLT_Mu17_TrkIsoVVL_Mu8_TrkIsoVVL_DZ_Mass8	DY (RobustParT)
/DoubleMuon/Run2017F-UL2017_MiniAODv2-v1	HLT_Mu17_TrkIsoVVL_Mu8_TrkIsoVVL_DZ_Mass8	DY (RobustParT)
/SingleMuon/Run2017B-09Aug2019_UL2017-v1	HLT_IsoMu27	Wc,TT (DeepJet)
/SingleMuon/Run2017C-09Aug2019_UL2017-v1	HLT_IsoMu27	Wc,TT (DeepJet)
/SingleMuon/Run2017D-09Aug2019_UL2017-v1	HLT_IsoMu27	Wc,TT (DeepJet)
/SingleMuon/Run2017E-09Aug2019_UL2017-v1	HLT_IsoMu27	Wc,TT (DeepJet)

Table B.7: Assigning truth flavour classes in PFNano to small-radius jets in CMS: the definition of the `Jet_FlavSplit` branch. The “Code” column represents the arbitrary choice of the author to introduce a single branch only instead of one-hot-encoding.

Class	Precise label	Code	Description
b	isB	500	<code>hadronFlavour = 5</code> , <code>nBHadrons = 1</code> , contains no neutrinos from a b or c hadron decay inside jet cone
bb	isBB	510	<code>hadronFlavour = 5</code> , <code>nBHadrons > 1</code> , no b hadron from gluon splitting inside jet cone
bb	isGBB	511	<code>hadronFlavour = 5</code> , <code>nBHadrons > 1</code> , at least one b hadron from gluon splitting inside jet cone
lepb	isLeptonicB	520	<code>hadronFlavour = 5</code> , <code>nBHadrons = 1</code> , contains at least one neutrino from a b hadron decay inside jet cone
lepb	isLeptonicB_C	521	<code>hadronFlavour = 5</code> , <code>nBHadrons = 1</code> , contains at least one neutrino from a c hadron decay inside jet cone
c	isC	400	<code>hadronFlavour = 4</code> , <code>nCHadrons = 1</code>
c	isCC	410	<code>hadronFlavour = 4</code> , <code>nCHadrons > 1</code> , no c hadron from gluon splitting inside jet cone
c	isGCC	411	<code>hadronFlavour = 4</code> , <code>nCHadrons > 1</code> , at least one c hadron from gluon splitting inside jet cone
uds	isS	2	<code>hadronFlavour</code> $\notin \{4, 5\}$, no hadronic tau, no mismatch between <code>hadronFlavour</code> and <code>parton</code> or hadron content in jet, <code>partonFlavour = 3</code>
uds	isUD	1	<code>hadronFlavour</code> $\notin \{4, 5\}$, no hadronic tau, no mismatch between <code>hadronFlavour</code> and <code>parton</code> or hadron content in jet, <code>partonFlavour</code> $\in \{1, 2\}$
g	isG	0	<code>hadronFlavour</code> $\notin \{4, 5\}$, no hadronic tau, no mismatch between <code>hadronFlavour</code> and <code>parton</code> or hadron content in jet, <code>partonFlavour = 21</code>
	isTAU	600	Not a HF jet based on <code>hadronFlavour</code> , but there is a hadronically decaying tau object contained in the jet (= no tau daughter is a lepton, and there are no non-neutrino daughters outside the jet cone). There exist variants of the flavour definition where a check for taus does not happen. The author recommends exchange between the involved parties, i.e. POGs related to jets, flavour tagging and tau identification, to ensure consistency. Several tools to streamline processes involving very similar objects exist already, but need to be enabled.
	isPU	999	No matched <code>genJet</code> associated with the reconstructed jet, with <code>partonFlavour = 0</code>
	isUndefined	1000	None of the above, leaving the following cases: • Similar as <code>isPU</code>, but <code>partonFlavour</code> $\neq 0$ • Similar as <code>uds/g</code>, but <code>partonFlavour</code> $\notin \{1, 2, 3, 21\}$ • Mismatch between <code>hadronFlavour</code> and <code>parton</code> or hadron content in jet (<code>hadronFlavour</code> $\notin \{4, 5\}$, but $\text{partonFlavour} \in \{4, 5\}$ or <code>nBHadrons > 0</code> or <code>nCHadrons > 0</code>), other implemented combination: <code>hadronFlavour = 5</code>, <code>nBHadrons = 0</code>

Table B.8: Global variables entering DeepJet.

Short name	Description
Jet p_T	Transverse momentum of the jet with respect to the beam line
Jet η	jet pseudorapidity
nCpfcan	Number of charged particle flow candidates inside jet
nNpfcan	Number of neutral particle flow candidates inside jet
nSV	Number of secondary vertices inside jet
nPV	Number of primary vertices associated with event
TagVarCSV_trackSumJetEtRatio	Transverse energy of the total summed four-momentum of the tracks, divided by the transverse energy of the jet ^a
TagVarCSV_trackSumJetDeltaR	Angular distance between the summed four-momentum of all tracks in the jet and the jet axis
TagVarCSV_vertexCategory	Integer specifying type of secondary vertex (or vertices) in the jet, if they exist, considering output of SV reconstruction algorithm: 0 (NoVertex), 1 (PseudoVertex), 2 (RecoVertex)
TagVarCSV_trackSip2dValAboveCharm	2D SIP of the first track in the jet that lifts the combined invariant mass above the assumed charm quark mass threshold of 1.5 GeV
TagVarCSV_trackSip3dValAboveCharm	3D SIP of the first track in the jet that lifts the combined invariant mass above the assumed charm quark mass threshold of 1.5 GeV
TagVarCSV_trackSip2dSigAboveCharm	2D SIP significance of the first track in the jet that lifts the combined invariant mass above the assumed charm quark mass threshold of 1.5 GeV
TagVarCSV_trackSip3dSigAboveCharm	3D SIP significance of the first track in the jet that lifts the combined invariant mass above the assumed charm quark mass threshold of 1.5 GeV
TagVarCSV_jetNSelectedTracks	Number of tracks in the jet passing pre-defined quality criteria for selection
TagVarCSV_jetNTracksEtaRel	Number of tracks in the jet for which the relative pseudorapidity with respect to the jet can be computed with sufficient precision

^aThe definition in the code (since 2014) is listed here (“transverse energy”), not what is mentioned in Ref. [18] or further internal documentation (“jet energy”). It compares quantities of similar nature for better discrimination power. See also <https://github.com/cms-sw/cmssw/blob/master/RecoBTag/SecondaryVertex/interface/CombinedSVComputer.h>, last accessed October 13, 2023.

Table B.9: Charged PF candidate variables entering DeepJet.

Short name	Description
Cpfcan_BtagPf_trackEtaRel	Pseudorapidity of track, relative to the jet axis
Cpfcan_BtagPf_trackPtRel	Transverse momentum of track, relative to the jet axis
Cpfcan_BtagPf_trackPPar	Dot product of jet momentum and track momentum
Cpfcan_BtagPf_trackDeltaR	Pseudoangular distance between track and jet axis
Cpfcan_BtagPf_trackPParRatio	Dot product of jet momentum and track momentum, divided by magnitude of track momentum
Cpfcan_BtagPf_trackSip2dVal	2D SIP of the track
Cpfcan_BtagPf_trackSip2dSig	2D SIP significance of the track
Cpfcan_BtagPf_trackSip3dVal	3D SIP of the track
Cpfcan_BtagPf_trackSip3dSig	3D SIP significance of the track
Cpfcan_BtagPf_trackJetDistVal	Minimum distance between track and jet, i.e. at their point of closest approach
Cpfcan_ptrel	Transverse momentum of the CPF candidate ^a , divided by jet transverse momentum, shifted by -1
Cpfcan_drminsv	Pseudoangular distance between CPF candidate and the closest SV, shifted by -0.4
Cpfcan_VTX_ass	Association quality of CPF candidate with PV (denoting whether it is used in the fit under loose (6) or tight conditions (7), not used in fit but compatible in z-direction (5), not compatible with PV but with the closest jet from PV (4), not associated with PV but compatible with beam spot (0), or (1), for which none of the criteria are true, so the vertex closest in z-dir. is associated)
Cpfcan_puppiw	PUPPI weight of the candidate
Cpfcan_chi2	χ^2 value from track fit, normalized w.r.t. nDOF if track details are available, otherwise default to -1 ^b
Cpfcan_quality	If track details are available, integer flag describing quality of fitted track via selection criteria considering χ^2 , compatibility with beam spot, closest HLT vertex and resulting in either high purity or high efficiency; otherwise default to 1

^aFor Run 3, this is multiplied by a constituent weight from the PUPPI algorithm^bUsually, χ^2 values are floating point numbers (and during RECO, the datatype is indeed double). However, for packed candidates entering the tagging algorithm (PAT process), the datatype is unsigned integer, which means it is also treated as such by adversarial attacks and therefore left unchanged by FGSM, NGM and other methods.

Table B.10: Neutral PF candidate variables entering DeepJet.

Short name	Description
Npfcan_ptrel	Transverse momentum of the NPF candidate ^a , divided by jet transverse momentum, shifted by -1
Npfcan_deltaR	Pseudoangular distance between NPF candidate and the jet axis, shifted by -0.6
Npfcan_isGamma	Boolean flag representing photon (1) or not a photon (0) via the pdgId of the NPF candidate
Npfcan_HadFrac	Fraction of NPF cand. energy deposited in hadronic calorimeter
Npfcan_drminsv	Pseudoangular distance between CPF cand. and closest SV, shifted by -0.4
Npfcan_puppiw	PUPPI weight of the CPF candidate

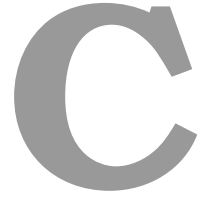
^aFor Run 3, this is multiplied by a constituent weight from the PUPPI algorithm

Table B.11: Secondary vertex variables entering DeepJet.

Short name	Description
SV_pt	Transverse momentum of SV
SV_deltaR	Pseudoangular distance between SV and jet axis, shifted by -0.5
SV_mass	SV mass
SV_ntracks	Number of tracks associated with SV ("numberOfDaughters")
SV_chi2	χ^2 associated with SV fit
SV_normchi2	χ^2 associated with SV fit, divided by nDOF
SV_dxy	2D flight distance of SV
SV_dxysig	2D flight distance significance of SV
SV_d3d	3D flight distance of SV
SV_d3dsig	3D flight distance significance of SV
SV_cothetasvpv	Cosine of angle between SV flight direction (thus a distance vector w.r.t. PV) and SV momentum vector, that is: dot product between the respective vectors, divided by product of their magnitudes
SV_enratio	Energy fraction of SV w.r.t. jet energy

Table B.12: Variables entering RobustParticleTransformer in Run 2 or Run 3 configuration, compared with DeepJet. Unused variables are shown with a hyphen (–), used features are marked with a checkmark (✓). The maximum numbers of considered constituents are given in brackets.

Short name	Description	Run 2	Run 3
<i>Regular input features</i>	<i>(properties of individual constituents)</i>		
DeepJet global variables	Purely high-level jet-related input features	–	–
DeepJet Cpfcan variables	16 charged PF candidate input features	✓(25)	✓(25)
Cpfcan_distminsv	Minimum distance between CPF cand. and closest SV	✓(25)	–
DeepJet Npfcan variables	6 neutral PF candidate input features	✓(25)	✓(25)
Npfcan_phirel	Absolute value of difference in azimuthal angle (ϕ) between the NPF cand. and the jet	✓(25)	✓(25)
Npfcan_etarel	Absolute difference between pseudorapidities of the NPF cand. and the jet	✓(25)	✓(25)
DeepJet SV variables	12 secondary vertex input features	✓(5)	✓(4)
SV_phirel	Absolute value of difference in azimuthal angle (ϕ) between the SV and the jet, shifted by -0.5	✓(5)	✓(4)
SV_etarel	Absolute difference between pseudorapidities of the SV and the jet, shifted by -0.5	✓(5)	✓(4)
<i>Four vectors</i>	<i>(for constituent interaction terms / pairs)</i>		
Cpfcan_pt	Transverse momentum of the Cpfcan	✓(5)	–
Cpfcan_px	Momentum in x -direction of the Cpfcan	–	✓(4)
Cpfcan_py	Momentum in y -direction of the Cpfcan	–	✓(4)
Cpfcan_pz	Momentum in z -direction of the Cpfcan	–	✓(4)
Cpfcan_eta	Pseudorapidity of the Cpfcan	✓(5)	–
Cpfcan_phi	Azimuthal angle of the Cpfcan	✓(5)	–
Cpfcan_e	Energy of the Cpfcan	✓(5)	✓(4)
Npfcan_pt	Transverse momentum of the Npfcan	✓(5)	–
Npfcan_px	Momentum in x -direction of the Npfcan	–	✓(4)
Npfcan_py	Momentum in y -direction of the Npfcan	–	✓(4)
Npfcan_pz	Momentum in z -direction of the Npfcan	–	✓(4)
Npfcan_eta	Pseudorapidity of the Npfcan	✓(5)	–
Npfcan_phi	Azimuthal angle of the Npfcan	✓(5)	–
Npfcan_e	Energy of the Npfcan	✓(5)	✓(4)
SV_pt	Transverse momentum of the SV	✓(5)	–
SV_px	Momentum in x -direction of the SV	–	✓(4)
SV_py	Momentum in y -direction of the SV	–	✓(4)
SV_pz	Momentum in z -direction of the SV	–	✓(4)
SV_eta	Pseudorapidity of the SV	✓(5)	–
SV_phi	Azimuthal angle of the SV	✓(5)	–
SV_e	Energy of the SV	✓(5)	✓(4)



Delphes Simulation-Only Cross-Checks

C.1 Training Progress and Convergence with Training and Validation Loss

The nominal and adversarial trainings need different numbers of epochs to show convergence, which is shown in Fig. C.1. Different hues of blue represent nominal training, hues

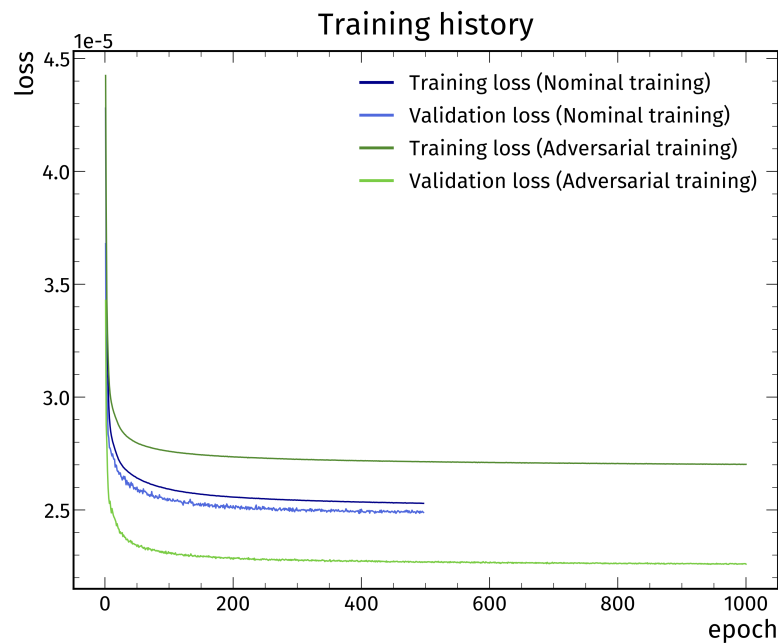


Figure C.1: Training and validation loss of nominal Delphes training and adversarial Delphes training.

of green represent adversarial training. Darker hues are the training loss, lighter variants show validation loss. At epoch numbers where nominal training does already not improve anymore in terms of performance for the validation set, there is still a slope for adversarial training. It has been trained twice as long in terms of iterations through the full training set. The difference between training and validation loss is larger for adversarial training (showing better generalization to unseen samples, because validation loss is smaller than training loss). Curves for adversarial training enclose both curves for nominal training. Thus, although the dataset is different in size and feature dimensionality compared to initial studies with a DeepCSV-like training conducted in a previous thesis [233], this pattern is common to both of them.

C.2 Regular Training with Cross Entropy Loss versus Focal Loss

For classification tasks carried out with the chosen architecture, usually cross entropy loss suits well. There is however a large class imbalance in the training set. A nominal training just using a reweighting technique to compensate this numerical difference may still be affected by these properties, such that the network assigns classes with high confidence just to improve on the loss minimization criterion. Resulting output distributions may then show artefacts where wide intervals are populated with very few samples only, and the majority of samples is "pushed" towards either the 0 or 1 end (previously observed already in Ref. [233]). This becomes a problem for tagger calibration that needs adequate numbers of jets to derive scale factors with sufficient precision. Figures C.2 and C.3 show how focal loss improves on this matter with respect to regular cross entropy loss for the different discriminants (BvsL, BvsC, CvsB, CvsL). Similar, although usually not calibrated on their own, the original output score distributions are given for focal loss (Fig. C.4) and cross entropy loss (Fig. C.5) as well. This is not detrimental to overall classification metrics, as exemplified in Figures C.6 and C.7. In those confusion matrices, percentages of correctly classified jets change to a small extent, with charm jets affected most negatively (7 %), while the overall change is positive (1 %). The ROC curves representing nominal training without application of focal loss for BvsL and CvsL are given in Figs. C.8 and C.9, respectively, also including the dashed lines for performance on distorted samples (FGSM with $\epsilon = 0.05$). Considering the clearly visible kink at specific efficiency values, definition of tight working points similar to CMS would become impossible with such a tagging algorithm and can be attributed to the shapes described with Figs. C.2 and C.3.

C.3 Output Scores and Discriminant Shapes for Nominal and Distorted Samples

Figures C.10 and C.11 show tagger output scores for nominal and adversarial training, respectively, with nominal samples or systematically distorted samples. For both, FGSM attacks are created using $\epsilon = 0.05$ and limitation of 25%, as introduced in Eq. (6.3). Corresponding discriminant shapes are shown in Figs. C.12 and C.13.

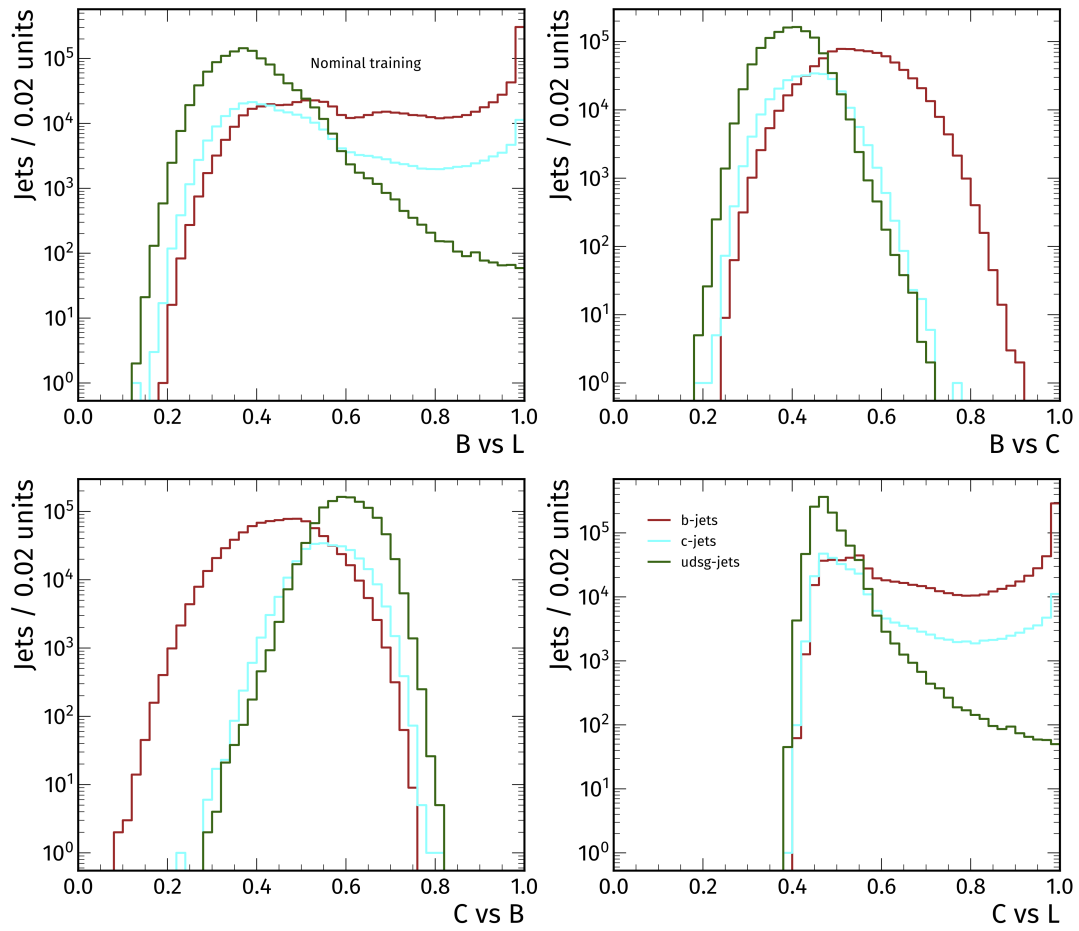


Figure C.2: Discriminant distributions for nominal Delphes training with focal loss.

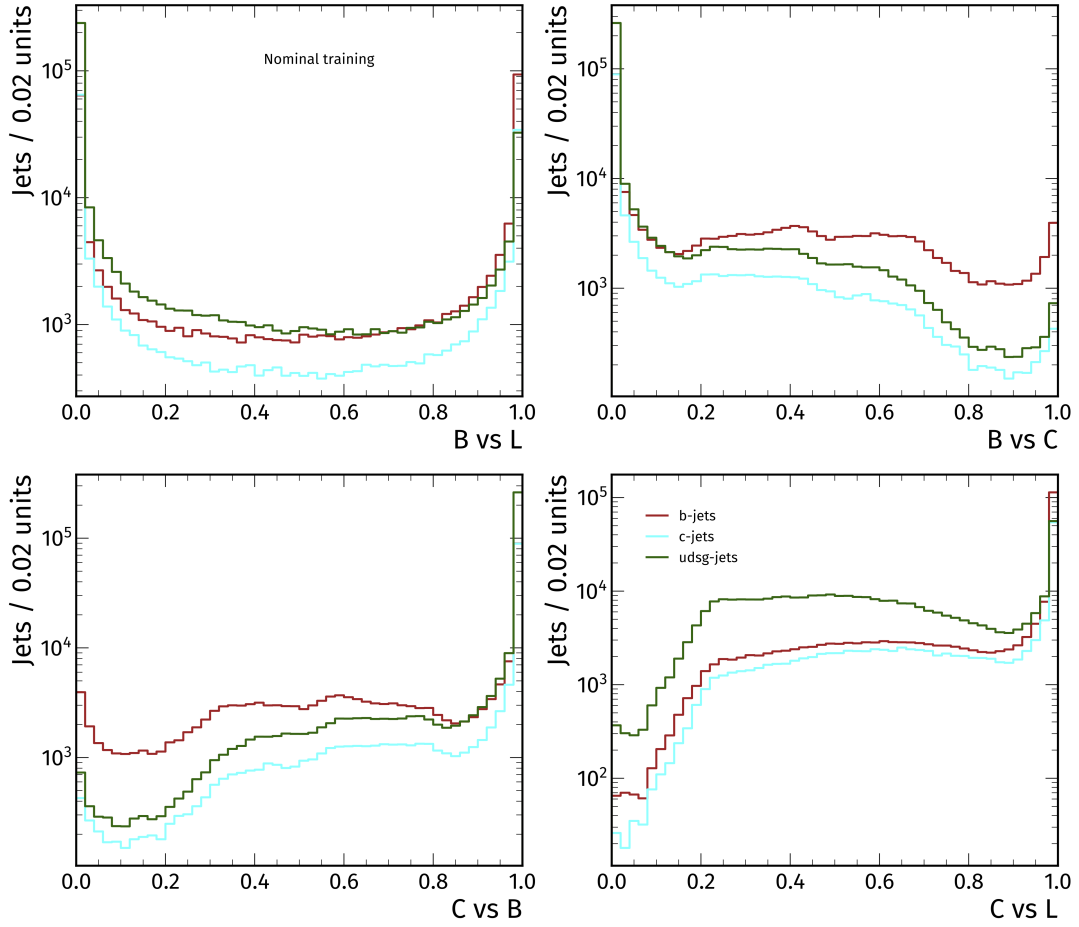


Figure C.3: Discriminant distributions for nominal Delphes training without focal loss.

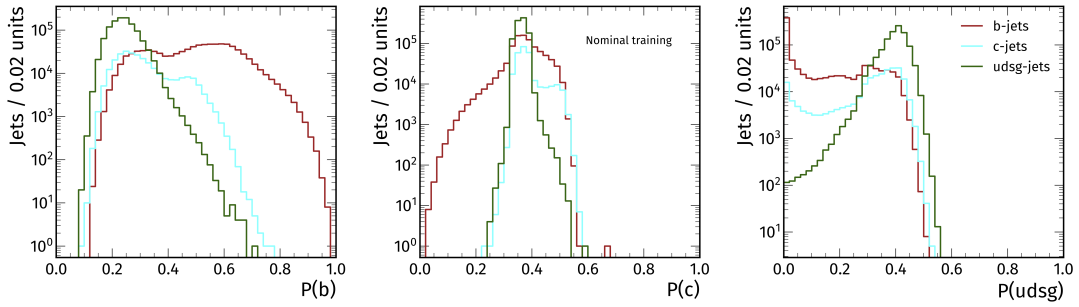


Figure C.4: Output score distributions for nominal Delphes training with focal loss.

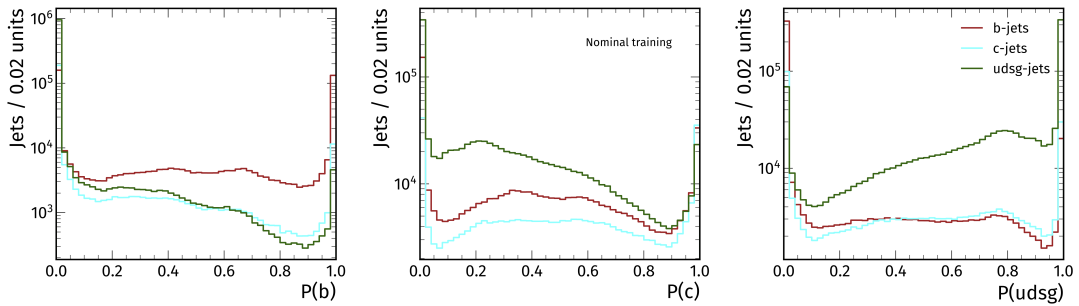


Figure C.5: Output score distributions for nominal Delphes training without focal loss.

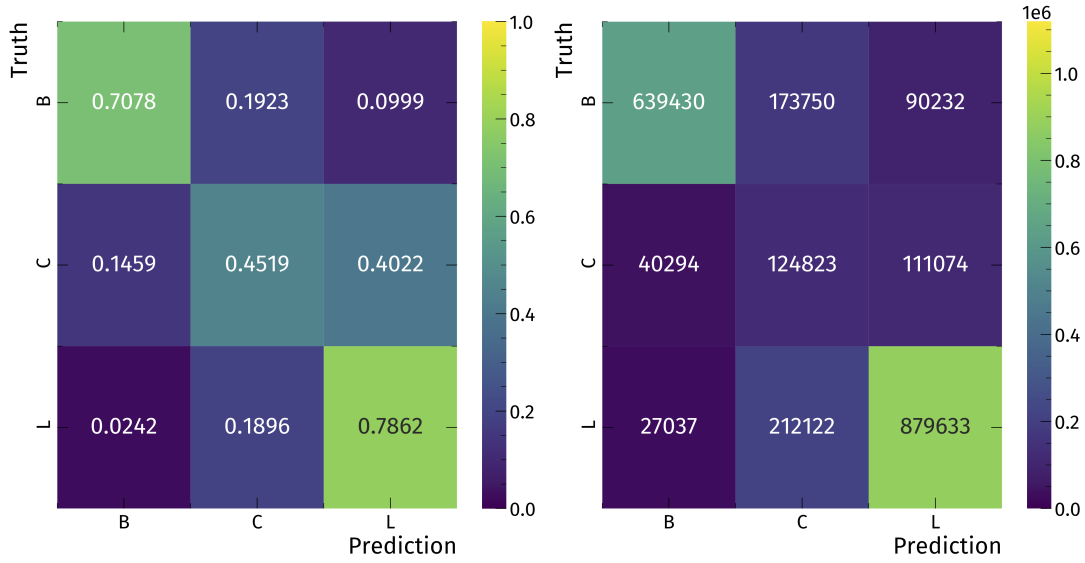


Figure C.6: Confusion matrices for nominal Delphes training using focal loss with focusing parameter $\gamma = 25$ on top of the kinematic and flavour reweighting technique. The horizontal axis represents the network's predicted class based on the maximum output score over the three last layer nodes, the vertical axis splits the samples by ground-truth labels. Normed values, i.e. percentages are shown on the left, absolute numbers on the right.

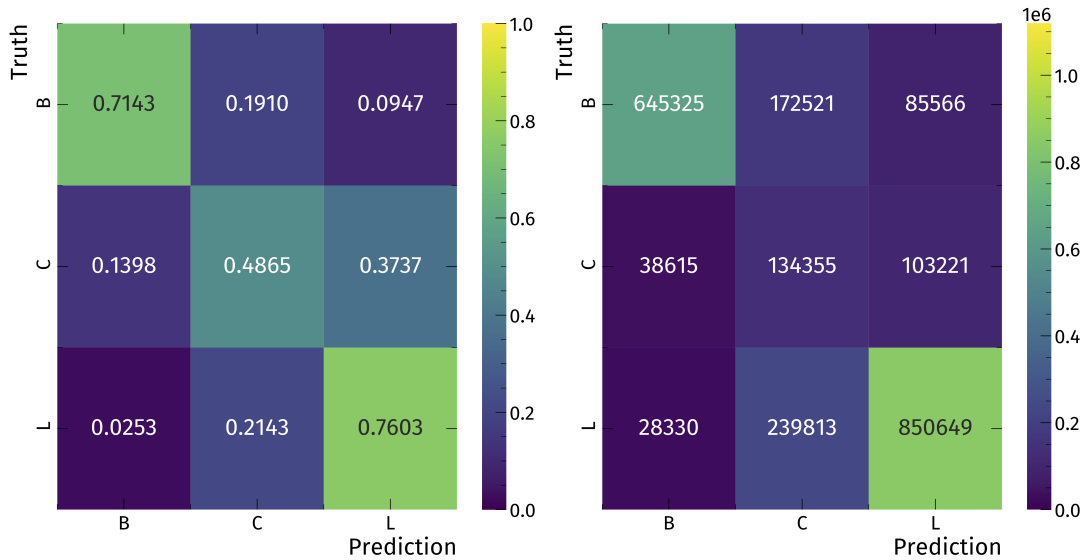


Figure C.7: Confusion matrices for nominal Delphes training using only the kinematic and flavour reweighting technique. The horizontal axis represents the network's predicted class based on the maximum output score over the three last layer nodes, the vertical axis splits the samples by ground-truth labels. Normed values, i.e. percentages are shown on the left, absolute numbers on the right.

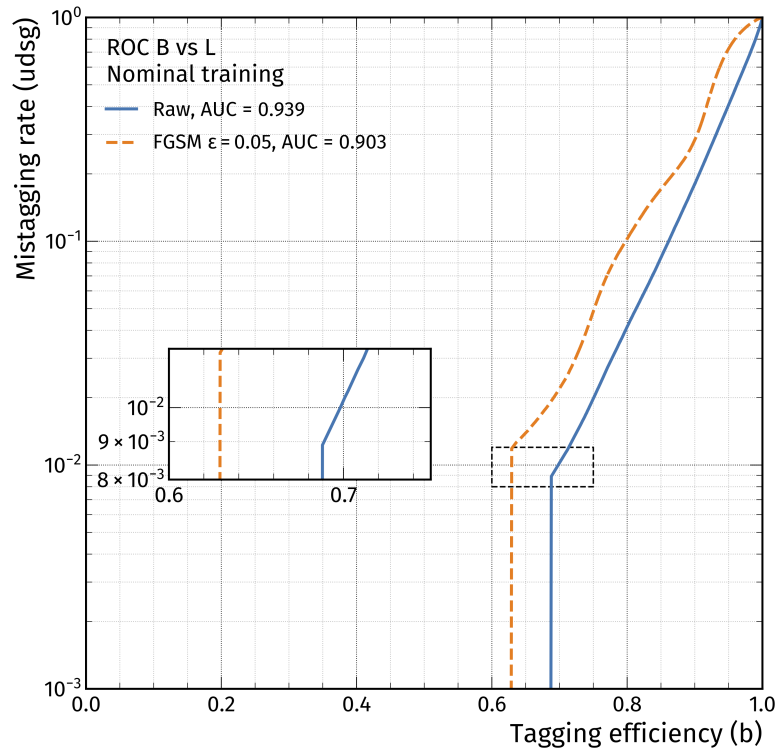


Figure C.8: BvsL ROC curve for nominal Delphes training without focal loss, showing nominal and systematically distorted inputs.

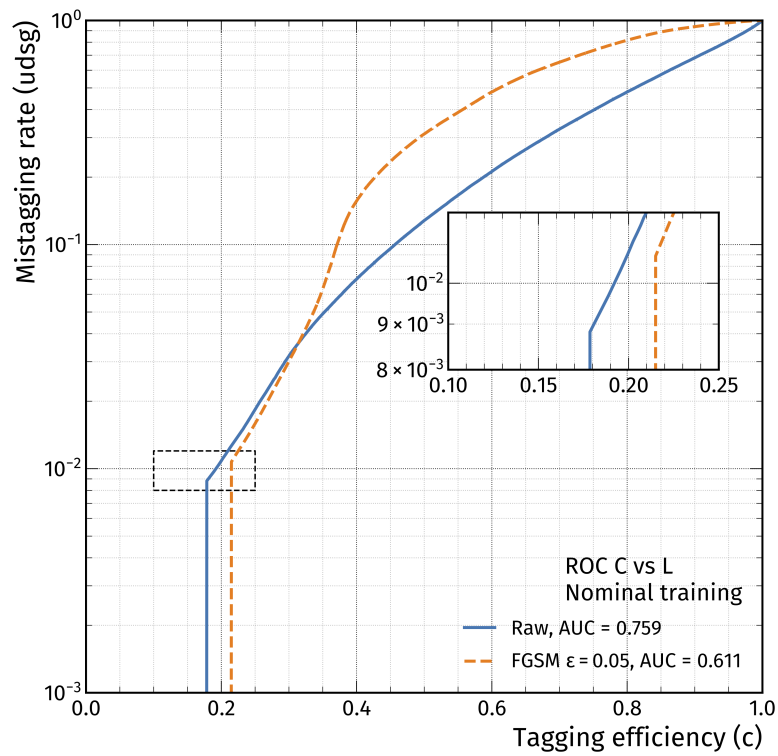


Figure C.9: CvsL ROC curve for nominal Delphes training without focal loss, showing nominal and systematically distorted inputs.

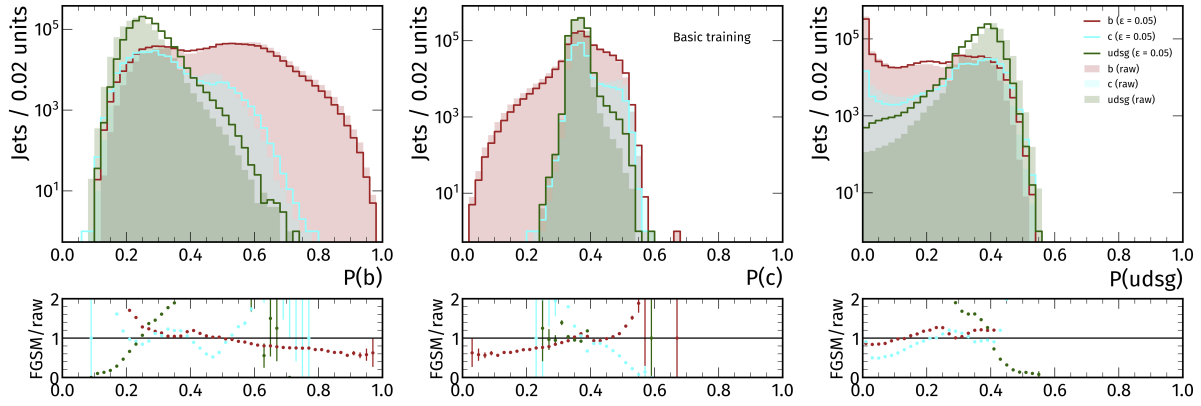


Figure C.10: Output scores for nominal training, exposed to nominal or systematically distorted inputs (with FGSM at $\epsilon = 0.05$, and applied restriction of 25%).

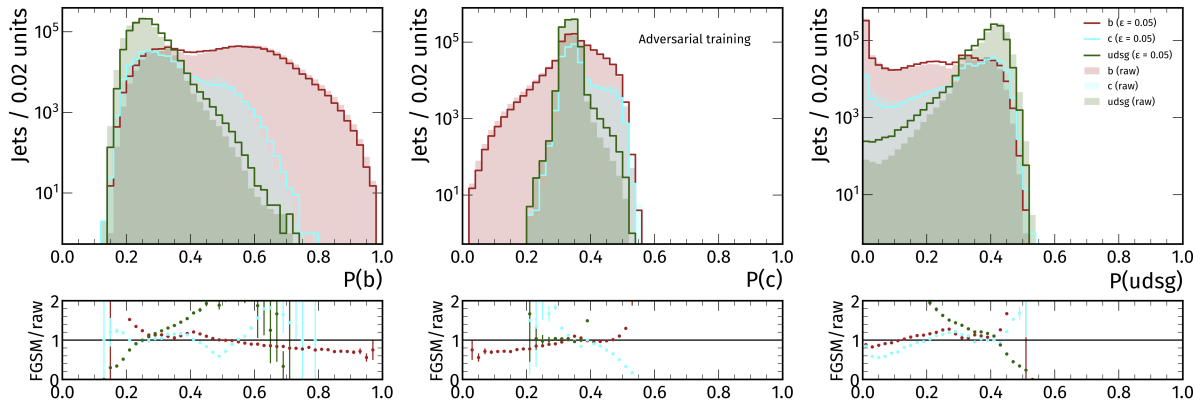


Figure C.11: Output scores for adversarial training, exposed to nominal or systematically distorted inputs (with FGSM at $\epsilon = 0.05$, and applied restriction of 25%).

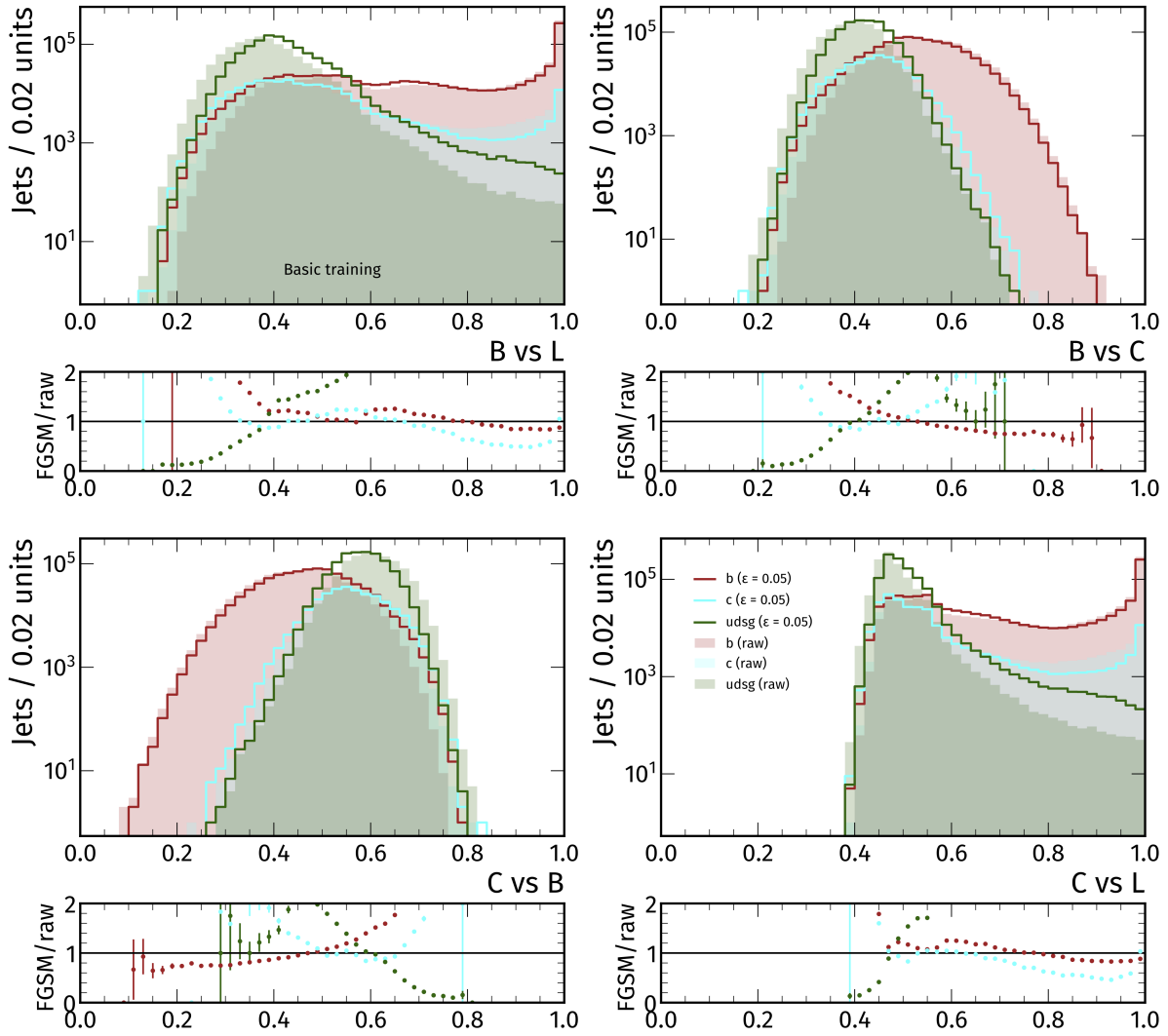


Figure C.12: Discriminant shapes for nominal training, exposed to nominal or systematically distorted inputs (with FGSM at $\epsilon = 0.05$, and applied restriction of 25%).

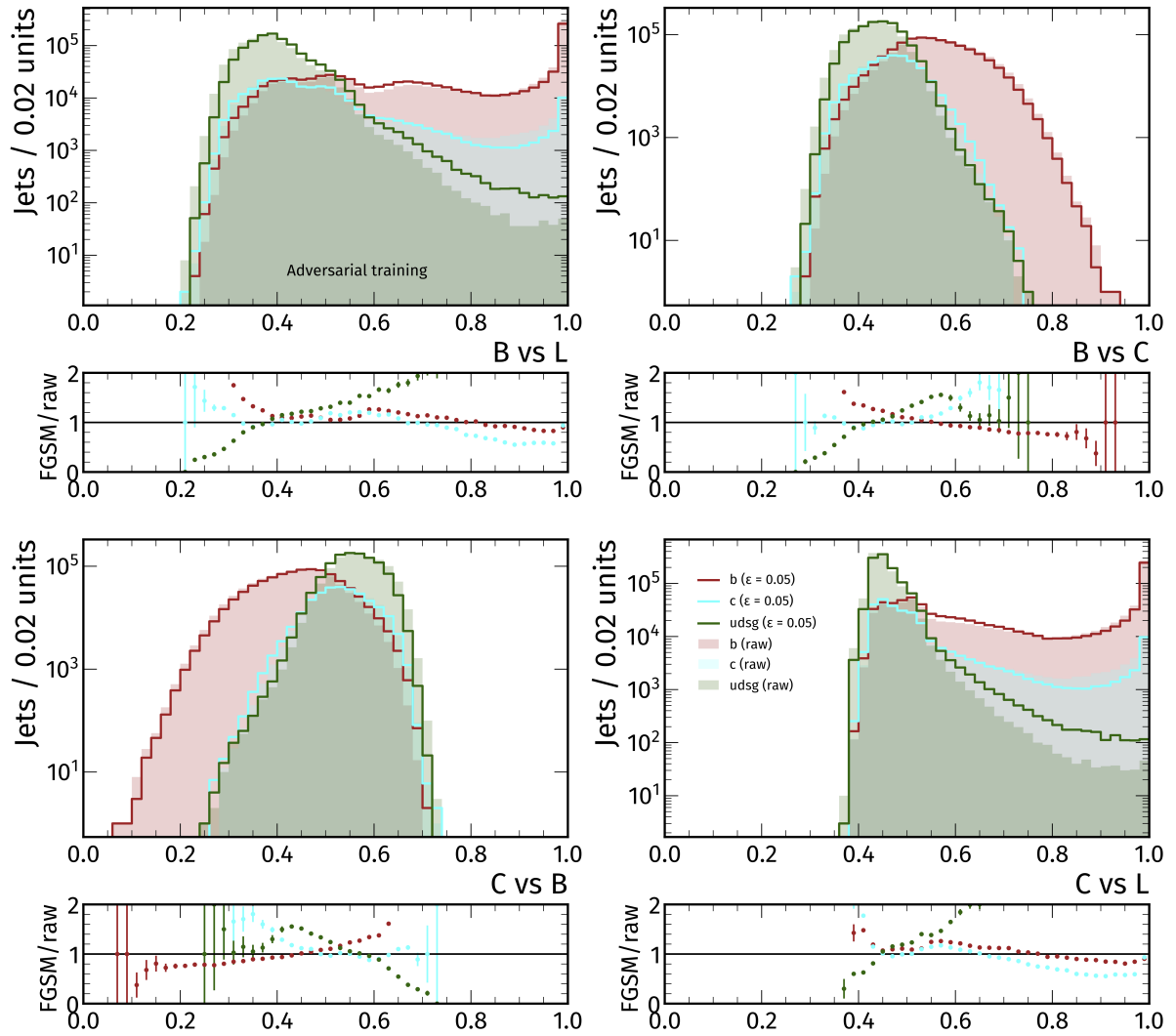


Figure C.13: Discriminant shapes for adversarial training, exposed to nominal or systematically distorted inputs (with FGSM at $\epsilon = 0.05$, and applied restriction of 25%).



A Modern Framework for Jet Tagger Development: From Training to First Data-to-Simulation Agreement Results in One Go

As the amount of data to be processed and reduced to meaningful quantities increases, especially now during the Run 3 data-taking period of the Large Hadron Collider (LHC) physics program [5, 6, 110] at higher energies and with more pile-up, efficient processing tools become increasingly important. Not only the amount of individual samples to be processed, but also the depth in information carried along with each event, or each physics object, forces researchers to adopt new approaches in storage, input/output-handling, feature extraction and data analysis techniques [137, 139, 156].

D.1 Existing Jet Tagging Software Tools of the LHC Community and Required Features for Robustness and Generalization Studies with Detector Data

Several workflows executed in typical reconstruction or identification algorithm development stages and performance evaluation scenarios as well as physics analysis involve production of samples with specialized data tiers and data formats [160, 195, 196, 317–325]. Usually, each step along the chain requires its own individual set of inputs. Such framework design has the disadvantage that the same information is stored in (slightly) different, yet incompatible formats, each requiring disk space. Also, every realization of the same underlying data needs to be processed, thus requiring computing resources, i.e. computing time in core-hours as well as the “actual time” that passes, to run over the parent data set. Jet tagging developments are split into a plethora of steps, using different data structures. Per campaign, only one variant of an individual algorithm is available, which until the beginning of this project only included nominal trainings. Available software frameworks are not built to study adversarial robustness, they are instead optimized per analysis software tag that only runs with one particular choice of model / checkpoint. Once such a fixed model is exported and included, it stays there, usually for multiple major versions of the experiment software, e.g. CMSSW [160] or athena [322]. The time that passes between finishing the model training and receiving the first results to study agreement between data and simulation can easily span multiple months as a typical cycle through model integration and commissioning, as the changes undergo a thorough review of pull or merge requests by multiple teams in a collaborative manner [160, 322].

D.1.1 Data Tiers and Use Cases

Training data needs to be created, it needs to be transformed to a format readable by the established training framework (choosing DeepJet(Core) [321] as an example for simplicity), which can only perform training with samples in its proprietary djcdc container structure. Counting through all steps that each require their respective datastructures, reviewing public standard software built on top of CMSSW [160]:

- MiniAOD as a parent dataset for subsequent steps and for some physics analyses [317],
- NanoAOD for several python-based physics analyses and for few commissioning studies with a reduced set of algorithm outputs [195],
- DeepNTuples derived from MiniAOD, containing jet tagging input features already prepared on jet level, used for training [320],
- djcdc-containers, also used for training (because DeepJetCore [321] can only read those files),
- PFNano for novel commissioning studies, containing also jet tagging input features [196],
- post-processed-Nano for scale factor derivation containing (besides tagger outputs) also a reduced set of inputs (mainly kinematic quantities) and systematic shifts to obtain systematic uncertainties for tagger efficiency or mistagging rates [318,319].

All these variants of the same events need to be processed and stored, and yet, to get the full picture in terms of adversarial robustness, the existing tools are not enough. More details per stage are given below.

D.1.2 Training Stage

Assume a new algorithm shall be developed in the context of one of the LHC experiments, for example CMS. Low-level information relevant for jet flavour tagging exists at the MiniAOD-level, stored with object collections as ROOT [352] files requiring experiment-specific software such as CMSSW [160]. However, neural networks trained with popular machine learning frameworks in the Python-ecosystem [255,276,365] read in arrays, tabular data or dataframes, typically obtained from common packages like NumPy [366], pandas [367], and are made available as tensors that allow gradient operations. This explains the existence of ntuples, which are yet again only a very specific variant of the data targeted at this particular algorithm training (again in ROOT format), but flattened to the level of jets [320]. The framework accompanying the DeepJet [18] tagger, DeepJetCore [321], which is used for a plethora of CMS analyses (non-exhaustive list: [199,200,234–237]), and publicly available [321], introduces various more steps on the way to a final tagger training, including the following:

- deriving weights for resampling, while resampling aims at effectively reproducing distributions of jet transverse momentum (p_T) and pseudorapidity (η) seen in one chosen flavour by removing samples until identically distributed kinematic quantities are achieved across flavours,
- storing resulting data in a format (.djcdc) only readable by the very framework that produced them.

At this point, the underlying information already exists in three variants, parent (MiniAOD), derived ntuples, and derived custom data structure. It should be noted that the last item is

tagger-input specific, adding another feature would not be possible on-the-fly, instead, the derivation chain would need to be repeated for ntuples or any other data tier after MiniAOD. The whole framework is embedded into container images via Apptainer [342,343], and at the time of writing this manuscript, the supported PyTorch versions in the available images are limited to older major versions [255] (i.e. < 2.0).

D.1.3 Integration into Official Reconstruction Pipeline

Once training is performed, performance studies are carried out to decide which model should make it into production. At this stage, the workflow only considers simulation, more precisely, the simulated (statistically independent) sample for cross-validation. Typical studies include derivation of ROC curves, AUC scores, differential efficiencies and mistagging rates. Based on this, highly-performing models are picked, converted into production-ready format (in case of CMS, via Open Neural Network Exchange (ONNX) [368]), and with each new architecture and / or additional input features, the experiment-specific code (mainly written in C++) needs to be updated for this new reconstruction algorithm. After all this is done, re-processing can happen to evaluate the newly introduced algorithm. Investigating the most recent modifications in the publicly available code base, this can very well take weeks or months and is strongly tied to existing schedules and deadlines induced by LHC or detector operations [160]. If all steps are done up to the level of the last element in the chain, e.g. MiniAOD / NanoAOD, sample production can start.

D.1.4 Studying Tagger Performance with Data

Assuming samples are produced (days or weeks after previous step), a pipeline to investigate data/simulation agreement and eventually derive scale factors (SF) applies selections enriching different jet flavours in pre-defined control regions [9,17,319,369]. Histograms are filled and statements on the level of agreement are made, taking uncertainties into account. All of this is usually developed independently for various derivation methods, some requiring particular branches not accessible in centrally produced samples. More information is added into custom ntuples, tools for “post-processed” samples are developed [17,318,319]. A large number of (sub-)frameworks exist, achieving similar goals with (sometimes only slightly) different data-processing methods. In summary, training and performance evaluation with data are decoupled steps, as the SF derivation does not perform ML inference itself. Adding to the challenge, the existing framework to be enhanced by the new one depends on PyROOT [357] imported as part of experiment-specific software, and its building-blocks are event loops with physics object loops nested therein. Parts of the existing implementation are written in Python 2. Lately, there are developments for scale factor measurements and commissioning with a common tool for Ref. [369], but there is still reliance on tuples that can be read by the “BTagAnalyzer” [319], as not all variants or phase spaces are migrated yet. The goal of the new framework is to eliminate such intermediate steps during algorithm development and performance evaluation, and to learn more about the neural network when exposed to detector data early on in the process.

D.2 Merging Multiple Tools into One with Fewer Required Data Structures

Duplication of data sets in various data formats for each purpose can be avoided, if the frameworks used for the full chain from feature engineering, neural network training, neural network evaluation, and final extraction of data-to-simulation agreement all use the same data structure. A new tool to perform various steps in jet tagger development in an efficient and comprehensive way is presented. One common data structure can be used for training, as well as for performance evaluation in data. The introduction of this new framework reduces the amount of data to be stored while accomplishing the same tasks, and shortens waiting times between algorithm development and first data-to-simulation comparisons becoming available, because integration into CMSSW [160] (as the prerequisite of the reviewed tools) can be skipped. The new framework allows investigating different variants, like different training paradigms, and their impact on data/simulation agreement, without producing any new large files on disk, and within the same run of the analysis framework. Besides being more time- and storage-efficient and thus enabling the results being available directly after finishing neural network training, this is the first framework capable of studying how adversarial techniques affect data/simulation agreement for tagger algorithm outputs as well as inputs. This software starts a paradigm shift: the novel approach facilitates algorithm development and performance evaluation in one single step so that data/MC agreement can be used as additional metric to steer further network training. Many learnings or proposed techniques from early studies of robustness only involving simulation can now be fully taken into account with the help of detector data. The “traditional” route was followed by the author of this thesis as well to include a novel algorithm in the official reconstruction in CMS, such that all evaluation and commissioning pipelines can be run for one fixed model as well. However, the following section explains the novel approach which was developed for the purpose of more advanced studies in this thesis. A direct comparison is given in Fig. 7.1, showing the standard development chain on the left, and the new development cycle on the right.

Considered Alternative

At the time of testing (in late 2022), *coffea*-processors [370] in conjunction with advanced ML frameworks were not optimized to produce and also store output arrays (with which adversarial methods, and e.g. feature correlations can be studied), but rather histograms. The deciding factors for the studies carried out in this thesis are continued memory efficiency, the necessity to checkpoint intermediate steps per chunk, *parsl* [371]- or *dask* [372]-induced overheads compared to pure file-based processing, and ability to process multiple variants of the same ML model in parallel. Therefore, the author decided to only implement purely analysis-related pipelines in *coffea* that do not perform resource-intensive ML steps with models that are about four times as large as DeepJet. This is an area where map-reducing to histograms profits from the above-mentioned processor structure. Indeed, for another project, the first awkward / *coffea* implementation of the Z boson to two leptons channel of the resolved topology of the vector boson associated Higgs boson decay to charm quarks analysis (modeled after Ref. [199]), the new framework presented above works as well. Using the *coffea*-processor approach is also sufficient and what has been done with a smaller number of branches by the author of this thesis in collaboration with Valentyn Vaulin, compared to the low-level studies that are the core of this thesis. See also Chap. G for a follow-up on low-level

information in the context of ML for physics analyses.

D.2.1 Implementation of a Custom Evaluation Pipeline for Algorithms in Development

The framework comprises two aspects, the data structure, as well as the usage by ML and data analysis tools. First, the data structure shall be described, using CMS parent data sets as a reference.

As previously mentioned, a suitable starting point is MiniAOD which is read by a set of plugins summarized under the term PFNano [196]. A significant portion of the resulting data structure is available since several years, but only in late 2021, the necessary components have been added* to enable usage with modern AK4 jet tagging algorithms such as DeepJet [18] and beyond, which until then were only part of other tools such as DeepNTuples [320]. The resulting ROOT files are readable by a variety of subsequent tools, including uproot [358], which can then be interfaced with other libraries ensuring interoperability (e.g. awkward-arrays [302, 373, 374], pandas-dataframes [367]). The term already alludes to two important components: being an extension of NanoAOD containing more details on Particle Flow (PF) [147] candidates and SVs. Thus, with customized functions either calculating features from first principles or accessing features already available as branches in MiniAOD, it is possible to enrich the content of commonly used NanoAOD files with missing features to perform tagger training and evaluation.

Compared to the approaches described in Sec. D.1, this constitutes a format suitable not only for tagger training (with a format only readable by the specific toolbox), but also for a variety of tasks typically conducted during physics analyses, when performing algorithm validation, or studying data/simulation agreement. Such a novel paradigm has a large impact on efficiency: files that would anyway need to be produced for commissioning or scale factor derivation can be repurposed for training as well. There are multiple consequences, for example, an entire production stage is eliminated, without duplication of input features in multiple data tiers, saving time and storage, and new evaluation techniques become possible. The author has implemented the necessary variables in said plugins, along with new variables that have not been part of existing algorithms, rendering the previously used ntuple-producer redundant.

Recent branch additions to PFNano mentioned above are already used in a purely map-reduce manner (*cf.* coffea [370]) for evaluation purposes where histograms are sufficient [369].

The remaining part of the full chain is the training and data/MC-evaluation procedure on top of PFNano. This has first been used in Ref. [233] in a simplified version to study a DeepCSV replica. For follow-up studies in Refs. [11, 19, 22, 290], the framework was significantly updated to cope with higher demands such as working with more complex networks or running different variants of a network in parallel.

The modules for preparation, training and evaluation use uproot [358], awkward [302], NumPy [366], scipy [375], scikit-learn [276], PyTorch [255]. They are designed to run in an interoperable way, just requiring a conda-environment [359], no other dependencies, no containers, no compilation, and cover the functionality provided in Sec. D.1.2 in a fully pythonic way, using arrays

*See the corresponding modifications here: <https://github.com/cms-jet/PFNano/pull/39>.

with columnar approach in all steps. It has been tested with recent python versions, allowing upgrading the PyTorch framework to version 2.0 and beyond. All drivers to interact with NVIDIA GPUs available on a variety of clusters can be added into the conda environment, compatibility with the “module” system (for example at CLAIX [304]) when enabling CUDA as well as cudnn has been verified [362].

A quick overview on how the different steps are implemented shall be given.

The preparations-step performs all tasks that need to be finished before any training can be started. To facilitate this chain of operations, for most python-modules an accompanying shell-script is provided that can be executed by computing clusters, utilizing SLURM (<https://slurm.schedmd.com/>, e.g. implemented as in Ref. [304]) or HTCondor (<https://htcondor.org/>, e.g. implemented as in Ref. [376]). With the help of aforementioned packages, selections on the samples are placed to remove entries with kinematic properties outside the considered tracker acceptance. Based on the number of generator-level particles of a certain kind, jets are labeled with an additional column (see Sec. B.2.2.1), input distributions are scanned for outliers. Defaulted or unphysical values are replaced with configurable placeholders. A two-dimensional, configurable binning is implemented (equal-width or customized to ensure all bins contain samples) to derive, for each flavour considered, a suitable weighting factor. The target is again configurable, by default, the average across flavours is used, flat distributions or reference distributions (the latter being what CMS is using for AK4 jet flavour tagging [321]) for a given flavour are implemented as alternatives, similar to Ref. [303]. The weights can be configured to either correct only imbalance on two kinematic quantities (p_T, η), only the class imbalance, or both (default). Similar to Ref. [11], sample splitting into training, validation and test sets is conducted, and standardization of distributions, with the training set acting as reference, is applied (CMS standard tools instead use batch normalization [321]). With results of previous steps in the chain, defaulted values are not considered when deriving the scaling parameters. In the training modules, the model is first defined independently outside all other wrapper tools and can thus be imported also elsewhere in later steps. A custom dataloader is shipped with the framework that ensures quick tensor shuffling and replaces an inner loop with fast, vectorized operations [263, 264], loss functions are stored as classes and as simple python functions covering multiple use cases and a custom reweighter class that takes previously created weights into account is added as well [265]. Compared to the DeepJetCore package [321], this is done with fully-pythonic tools, and does not require specific modules written in the C programming language. The core of all resampling or loss-reweighting techniques for the given use case is looking up flavour, transverse momentum and pseudorapidity and then drawing from a distribution, or performing this task only once and using individual sample weights. The latter has been found to be more efficient, which is why this lookup is performed in advance. Its results yield a new weight column that is appended to the inputs, ensuring tight connection during shuffling. All of these components are accessed by the wrapper module, which carries the training and validation loops, loads the data and stores, after each full iteration through all available samples, the model parameters with additional metadata like criterion and optimizer state (denoted epoch checkpoint). Several standard metrics are available in the evaluation-step, inputs and their correlations can be visualized (properties of each input variable like units, and several helpers to better display relevant intervals assist here), loss curves (training / validation) are available, output scores of the algorithm in question as well as discriminants, ROC curves and AUC complete this set. Finally, the attacks-module

covers what has been introduced to investigate tagger robustness [11, 19] and comes into play in Secs. 7.2 and 7.3.

As the title of this section claims, the framework would not be complete if it only replaced previous tagger *training* tools. Therefore, the new approach of tagger evaluation with PyTorch running on-the-fly on top of PFNano for inference shall be outlined as well.

For each PFNano file to be processed for further analysis, a job is submitted via customized HTCondor submission shell-scripts. They are configurable by keywords triggering which models, after which epoch, and with which setup or hyper-parameters, shall be evaluated besides the common algorithms already provided officially. Currently, the framework is designed to perform the analysis with up to three different networks at once, as this has been found to be a reasonable trade-off considering queueing times at typical computing clusters and available resources (e.g. at NAF-DESY). As mentioned previously, when interfacing with existing experiment-specific code, an HTCondor job must be able to work both with the established conda-environment, as well as with the PyROOT infrastructure. Therefore, the conda-environment is compressed and can be un-packed and (de-)activated at different stages during execution of the shell-script. For this purpose, the tool `conda-pack` [377] that creates a tar-ball is used. As shipping large environments with HTCondor jobs is limited to few GB, the prepared environment is instead stored on a network drive accessible for each condor job. Therefore, only the individual job parameters need to be shipped. The inference-modules ensure that during every job, the parameters (like model architecture) are known to all executables (bookkeeping). As with the training part, also here the PFNano files are converted into PyTorch tensors for evaluating the networks. This is done batch-wise, where the batches are customizable to maximize resource usage while taking available memory into account. Default settings are provided to profit from opportunistic resource allocation. For example, for the Run 2 RobustParT algorithm (see Sec. 7.3), an analysis / inference batch-size of 2000 jets proved to be useful for standard jobs going into the so-called “short-queue” at NAF-DESY [376]. It should be noted that predicting is again a vectorized task after flattening from event- to jet-level involving only the outer loop over batches and NumPy-arrays are used as output data structure. If one decides to interface this step with analysis-frameworks making use of event-loops, the flat NumPy-arrays are missing the relevant mapping between jet and event. Thus, reshaping with the awkward-package [302] is done to mediate this conflict. For each event, the framework notices all jets that have been selected, as well as the jets that did not pass the selection, ensuring the indexing is not affected. Finally, resulting arrays are histogrammed after the selections have been applied. Any input and any output of the different architectures studied, is added, and with the existence of the `attacks`-module, all those comparisons can be done not only with the original samples, but also with distorted inputs. This opens a whole new way to study features, their correlations, and how (adversarial) attacks can be understood in the context of data/MC agreement for inputs and outputs. Because the selections are independent of the individual models or the attacks applied to the inputs, and the framework can process and store full arrays on top of storing histograms, the runtime of the analysis and histogramming module is not affected by the choice of model / checkpoint / attack. The analysis-part treats such tagger scores as “spectators” only.

D.3 Conclusion and Recommendations

Indepent of how the combined training- and data/MC-pipeline is implemented (with the above being one such example), it is advisable to use such a custom framework early on in development, as it does not slow down the integration as a whole but yields valuable information in the decision process. For the ML practioners involved in jet tagger training, this adds an additional piece of work, but yields performance evaluation in data with multiple variants in parallel. This would not be possible with fixing one model / one checkpoint for integration and validation in data only. With more than one set of model parameters or input variations, one can assess the model's generalization capability and resilience to mismodeling, because the custom pipeline allows re-runnnng the algorithm on-the-fly while probing data performance.



Supplementary studies with DeepJet

E.1 Alternative Quantification Metrics for Data/MC Agreement

E.1.1 Discriminant Distributions with Fine-Grained Binning

The main results with the DeepJet algorithm have been obtained using the setup that is also chosen in previous CMS results [17], including a coarse binning of 30 equal-width bins. The values for scalar measures are directly affected by choosing a different binning, and especially for studies on adversarial robustness with small modifications it is advisable to get a fine-grained picture. Thus, the same study has been repeated with 60 instead of 30 bins. Figures E.1, E.2, and E.3 show agreement between data and simulation for the same phase space regions and discriminant choices as described previously in Section 7.2.3, but with twice the number of bins, thus half bin width. Overall, the conclusions are similar, but the argument reaches further in terms of granularity: also on such smaller scales, adversarial training improves the agreement in the majority of the cases, quantitative descriptions follow.

E.1.2 Jensen-Shannon Divergence and χ^2 Values with 30 or 60 Bins

The JSD metric describing agreement between data and simulation is introduced also for the distributions with 60 bins. Results are displayed in Fig. E.4 right next to those for 30 bins, to allow easy comparison. The resulting values are comparable (qualitatively and quantitatively for the udsg and b selection), but for CvsB in the charm jet enriched selection, adversarial and nominal training flip their ordering. A similar pattern is visible when looking at the respective χ^2 values in Fig. E.5. First of all, the charm selection is largely affected by the OS – SS subtraction and the sample size is small compared to the other two selections (size of available datasets containing relevant input features). Thus, statistical fluctuations become more prominent, especially for a discriminant that is by definition separating b from c jets (with, compared to other classification tasks, very similar properties). In the current state, only limited conclusions should be drawn for this specific combination of discriminant and selection, as the binning setup has more influence on the behaviour than the choice of the training strategy. In general, switching from coarse to fine binning, adversarial training gains in the number of improved selection / discriminant pairs, but there is no major change in relative differences in JSD or χ^2 .

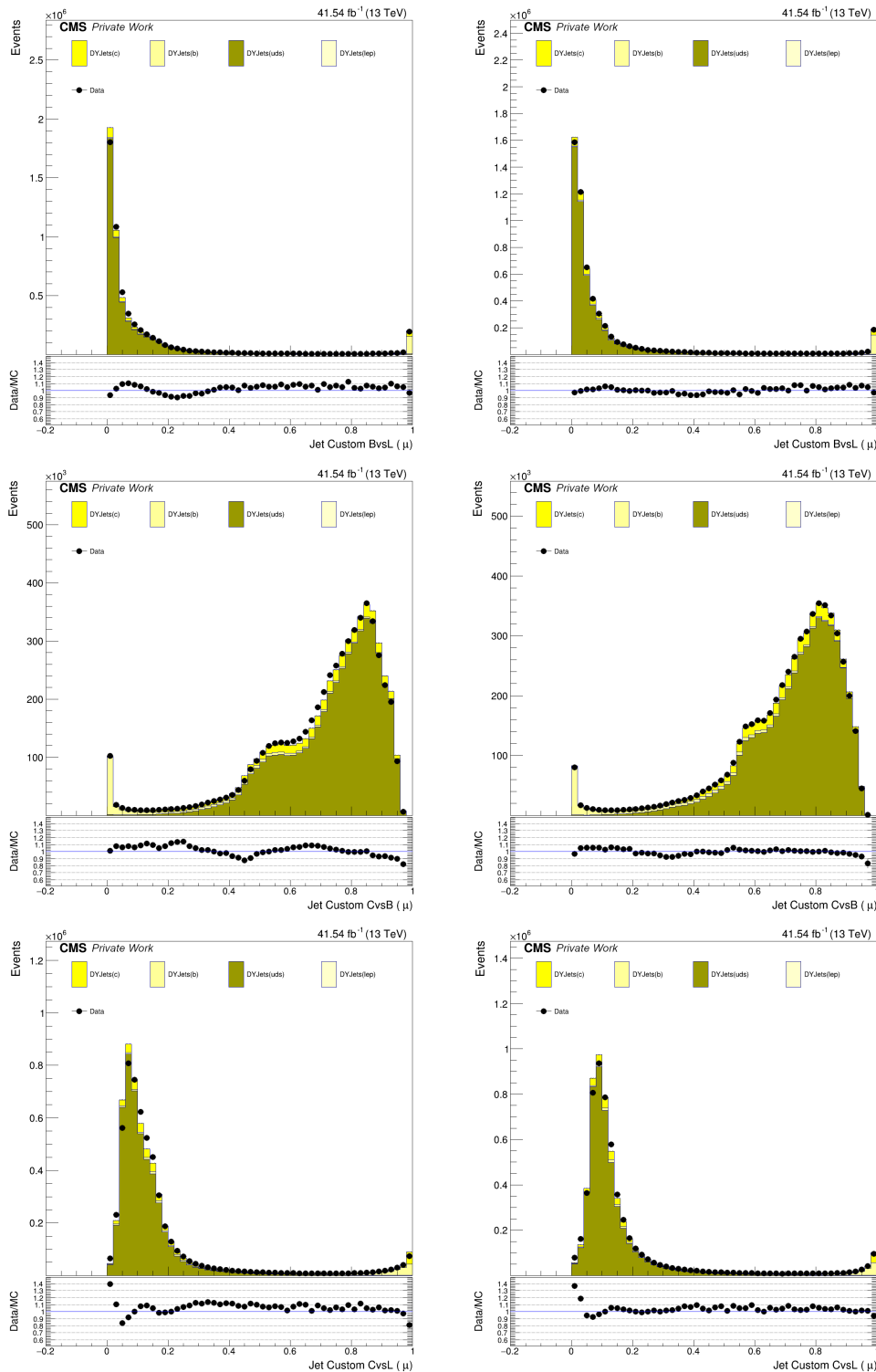


Figure E.1: Comparison of nominal (adversarial) training on the left (right), in the light-flavour enriched selection with 60 bins for histograms of the three discriminants BvsL, CvsB, CvsL from top to bottom.

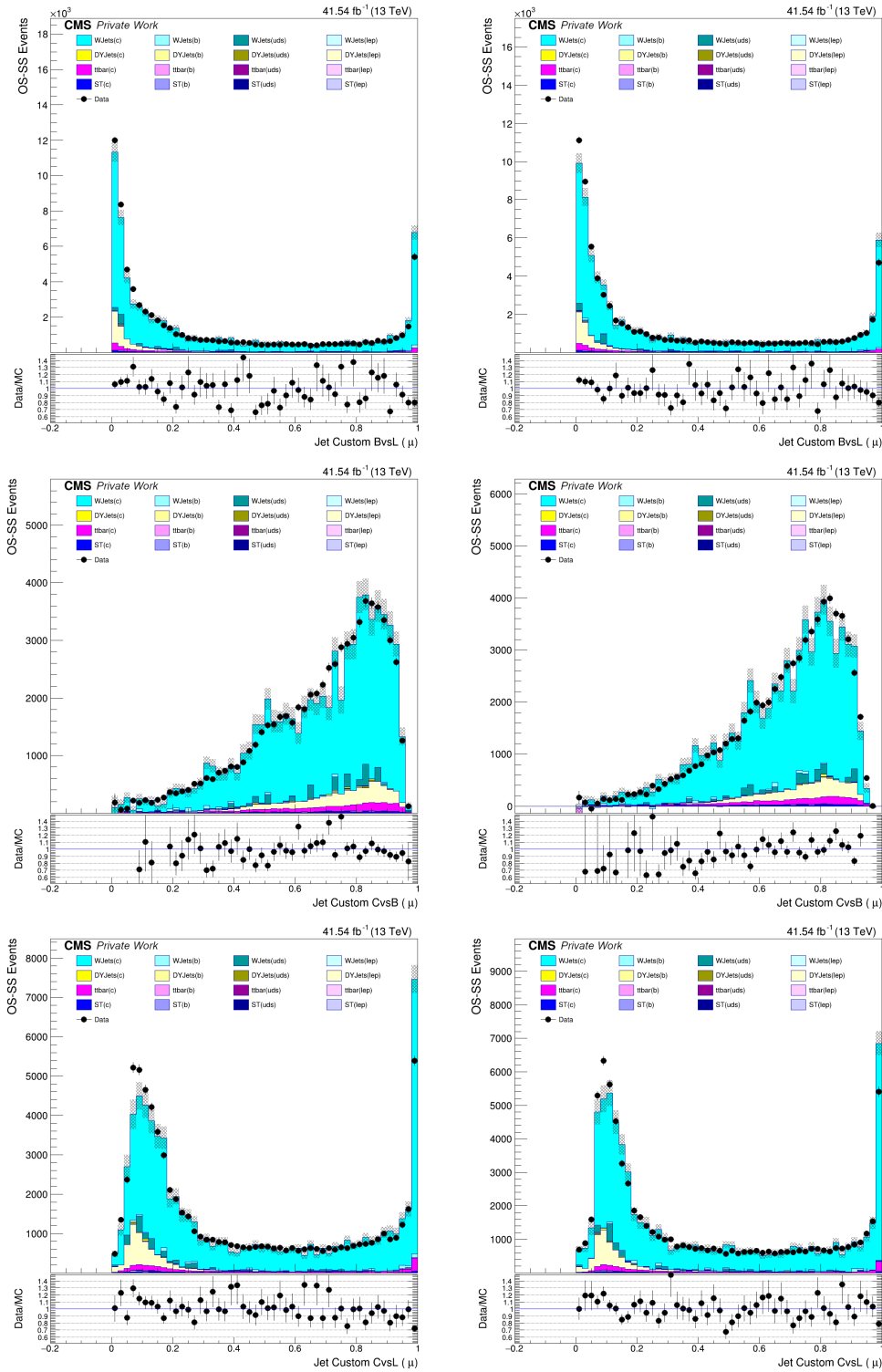


Figure E.2: Comparison of nominal (adversarial) training on the left (right), in the c-flavour enriched selection with 60 bins for histograms of the three discriminants BvsL, CvsB, CvsL from top to bottom.

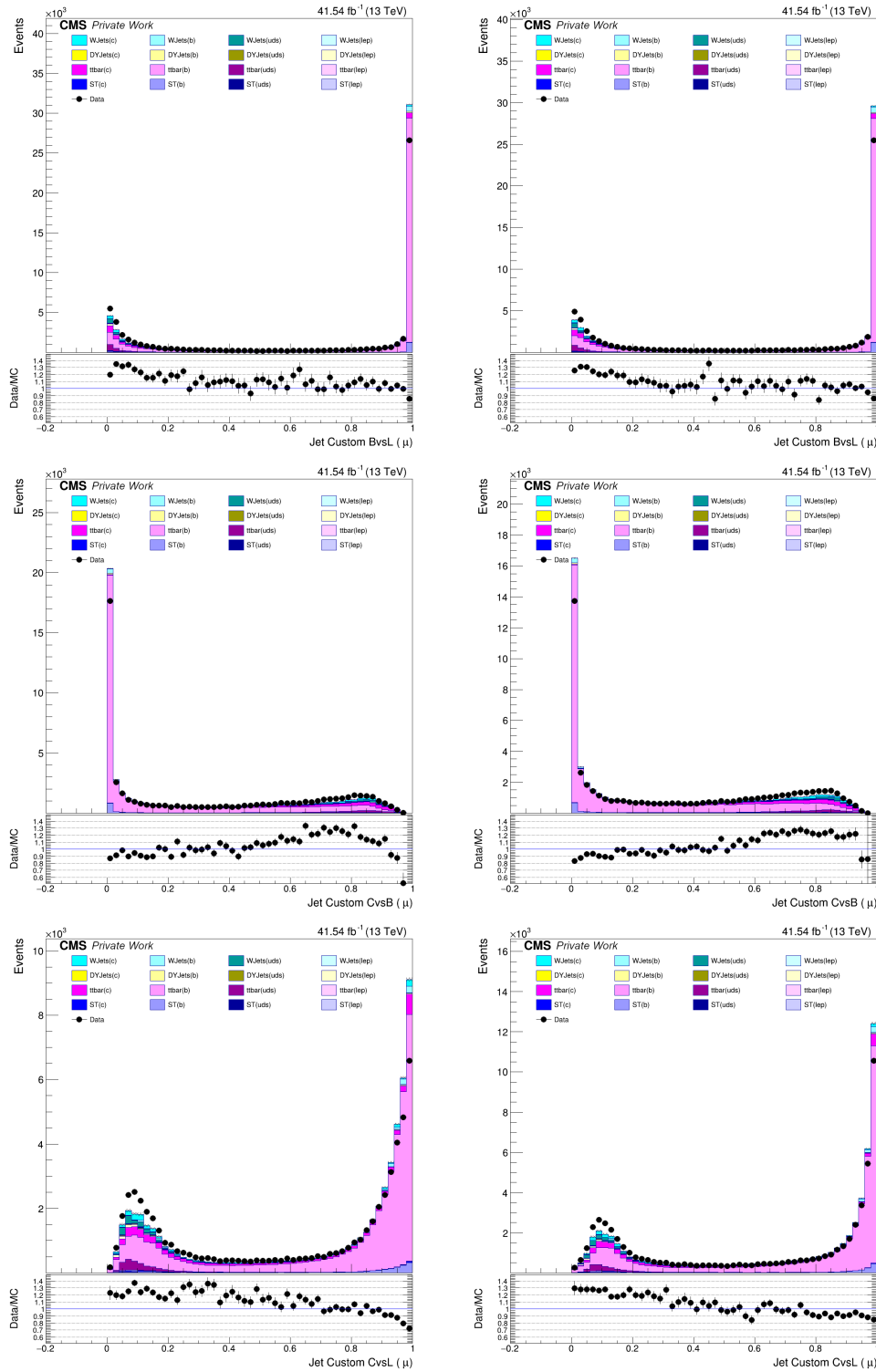


Figure E.3: Comparison of nominal (adversarial) training on the left (right), in the b-flavour enriched selection with 60 bins for histograms of the three discriminants BvsL, CvsB, CvsL from top to bottom.

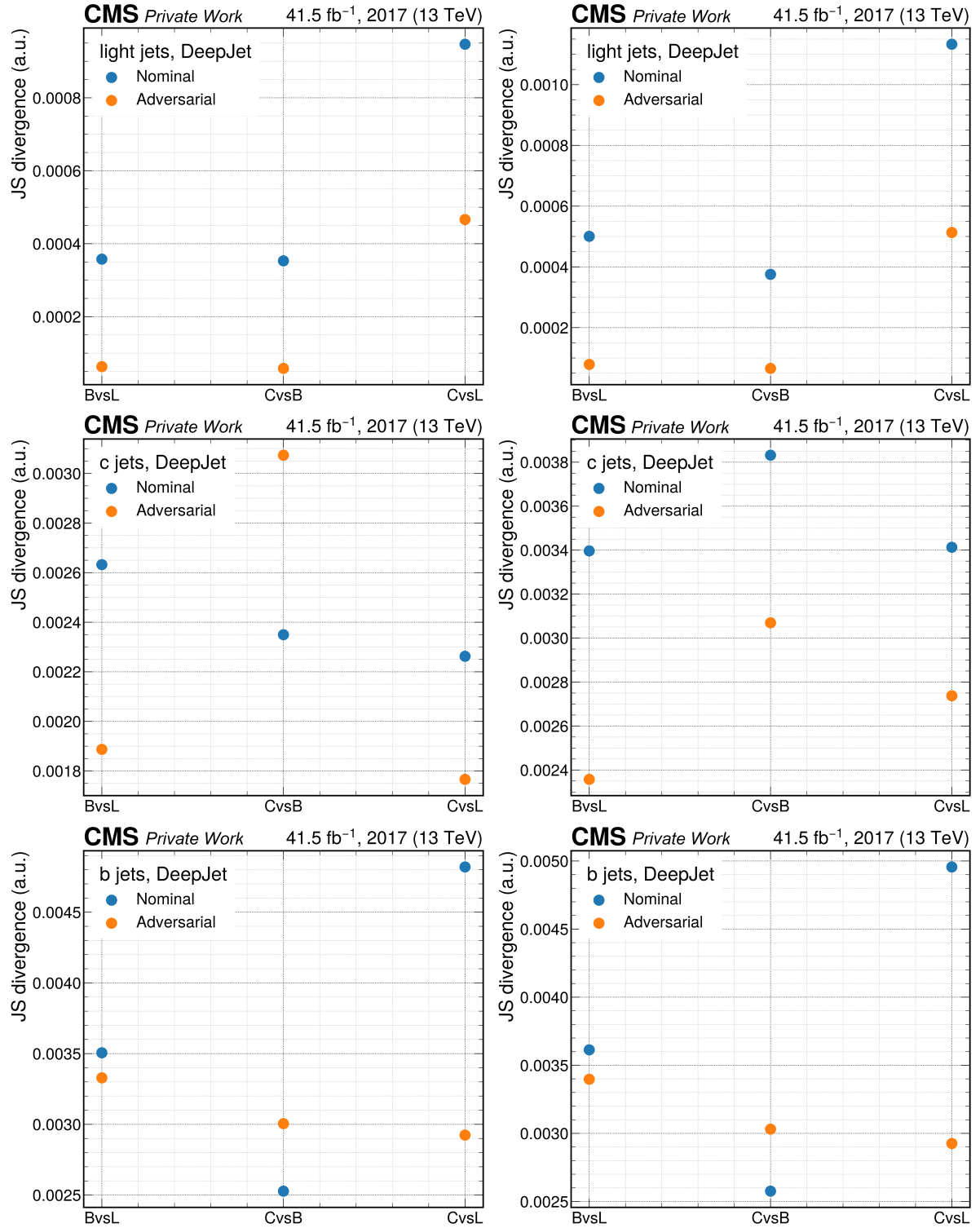


Figure E.4: JS divergence scores for three different selections (udsg, c, b jets from top to bottom) and two different binning setups, with 30 bins (60 bins) on the left (right).

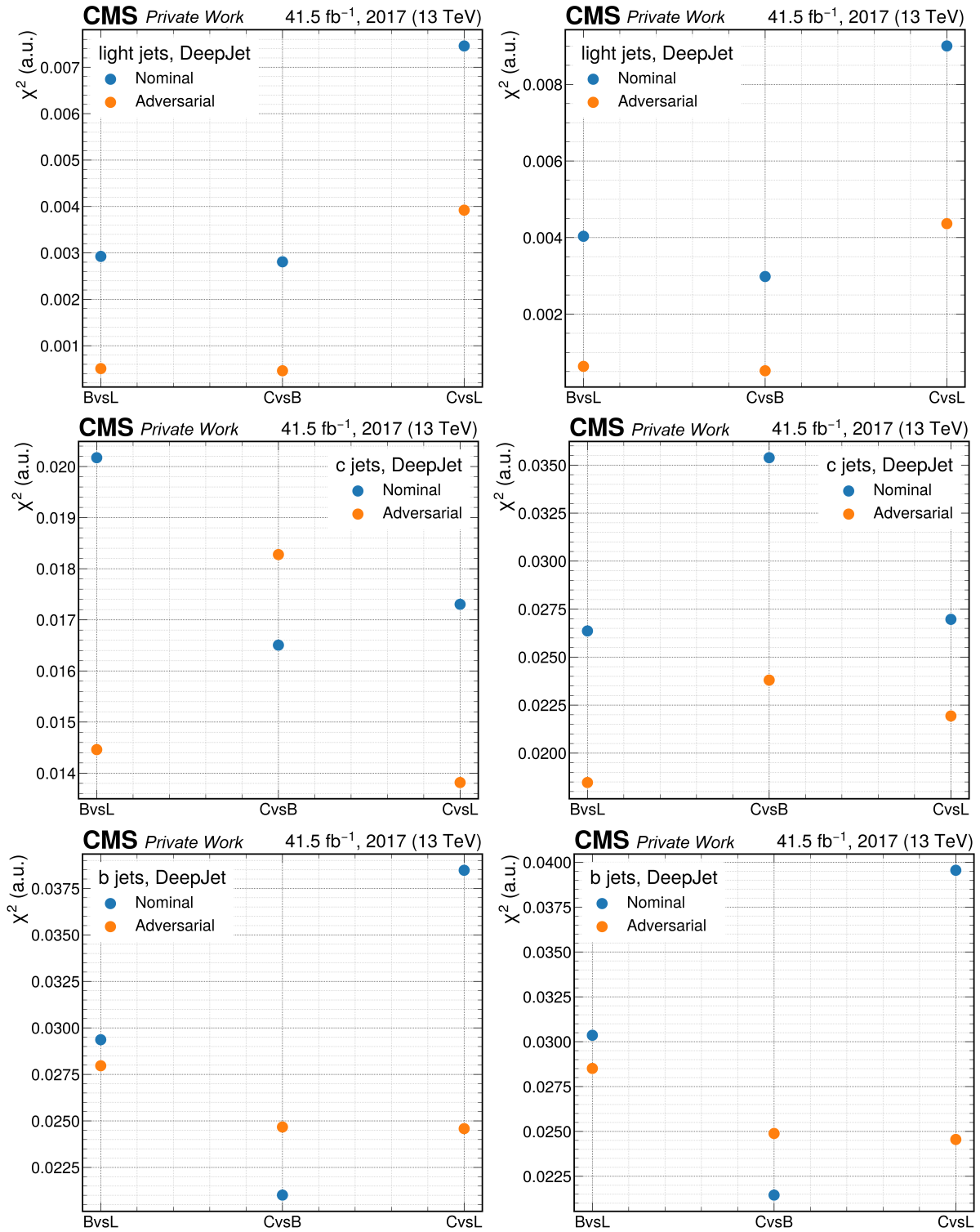
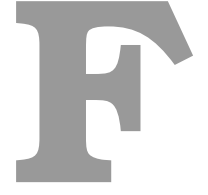


Figure E.5: χ^2 values for three different selections (udsg, c, b jets from top to bottom) and two different binning setups, with 30 bins (60 bins) on the left (right).



Supplementary Studies with RobustParticleTransformer

F.1 Correlations Between Nominal and Distorted Samples (Flavour-Dependent)

While Fig. 7.29 shows an overlay of FGSM- and NGM-attacked samples for the same set of nominal inputs, the following two figures separate the attacks and then split the correlations into the six different flavours. In particular, in Fig. F.1 for FGSM, as there are contributions of all flavours both for positive and negative shifts, it becomes impossible to spot specific clusters of jets of a given flavour. It is however clear that the majority of jets is indeed modified (by $\pm\epsilon$), while only a small fraction is unchanged (diagonal line for defaults or vanishing gradients). In Fig. F.2, there is again no particular pattern for specific flavours, the NGM is

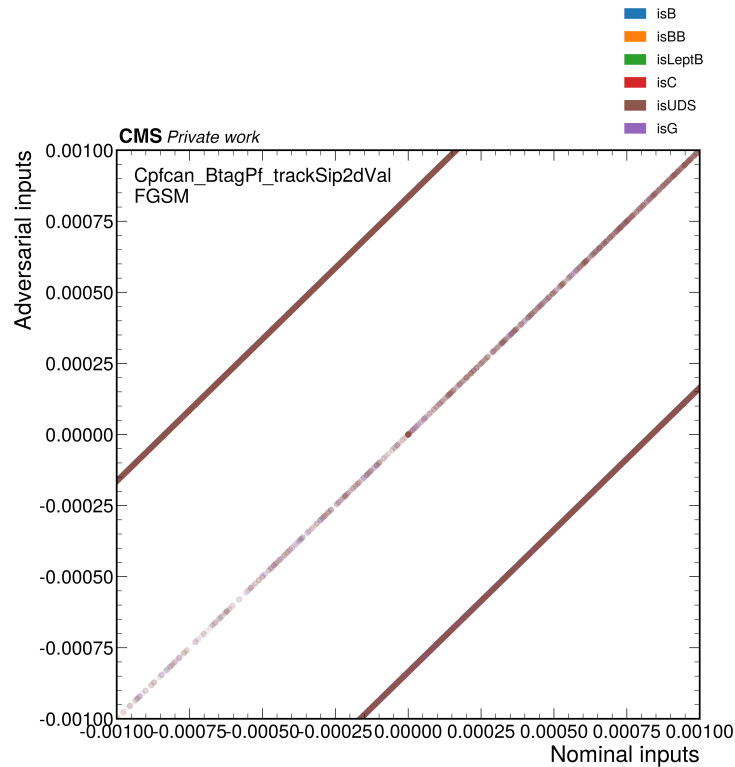


Figure F.1: Correlation between nominal (x axis) and FGSM-attacked (y axis) inputs, split by flavour (colours), for the second charged PF candidate (track) SIP value in two dimensions, all values given in units of cm.

one example of attacks that produce “unpredictable” shifts that are different for every jet. The

dimensionality of the gradient vector and infinite numbers of distortions are evident. Such distortions, although bound with an overall limiting parameter $\pm\epsilon$, have varying magnitude and direction for each sample and do not *look* different than a random noise term.

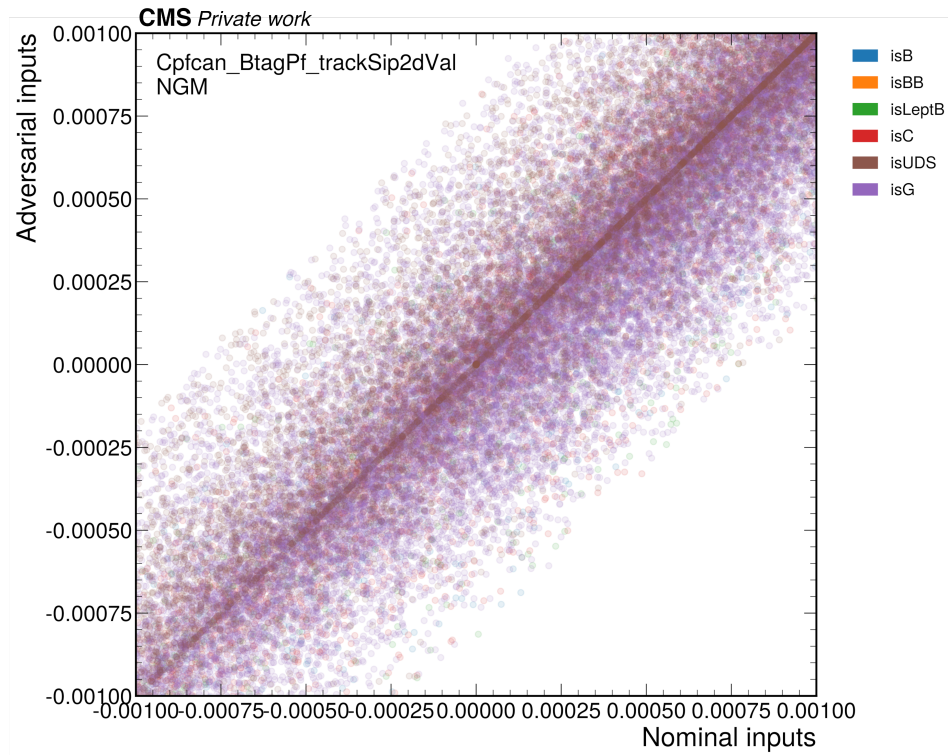


Figure F.2: Correlation between nominal (x axis) and NGM-attacked (y axis) inputs, split by flavour (colours), for the second charged PF candidate (track) SIP value in two dimensions, all values given in units of cm.

F.2 Alternative Training Scenario

As adversarial training imposes additional complexity during training, an alternative setup has been tested. An initial nominal training phase is followed by few epochs of adversarial training, to study if one can reduce the training time in total but yet achieve a robust and highly performant model. Additional hyper-parameters for tuning are the chosen loss function and the epoch after which to switch from one configuration to another. The individual variables are treated with the first version of taking parameters scaled with the help of interval length covered with the feature distribution. One example scenario is given in Fig. F.3, showing resulting training and validation loss and the milestone epochs where some parameters are modified. For all epochs, the batch size is fixed at 512. The nominal training is conducted for 10 epochs, followed by 5 adversarial training epochs defending against the NGM attack. While nominal training shows continuous improvement of training and validation loss, with the beginning of adversarial training, there is a sudden increase in validation loss, while training loss improves quickly. Such behaviour, where the generalization gap becomes worse for adversarial training, known as catastrophic overfitting [378,379], is partially mitigated in subsequent epochs with a switch to focal loss with $\gamma = 6$. From the fifteenth epoch on, the gap

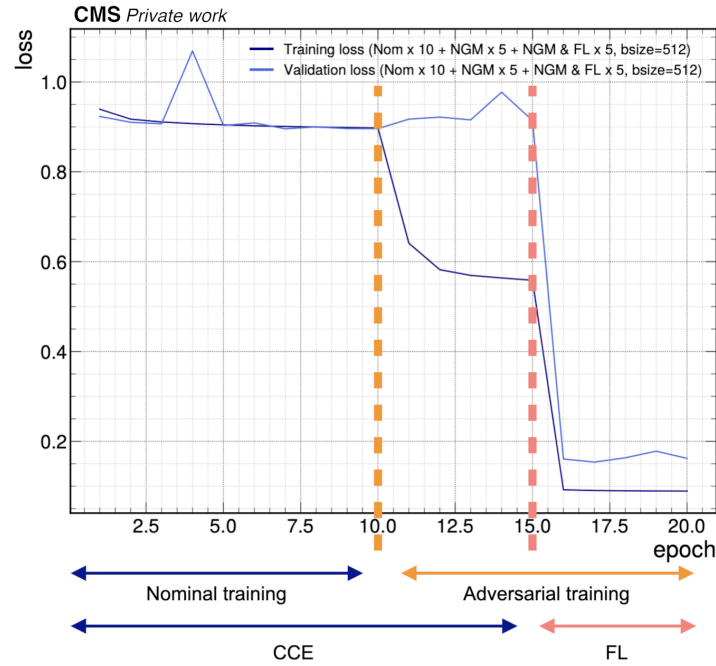


Figure F.3: Training history for an alternative training scenario of RobustParticleTransformer where adversarial training and focal loss are introduced to fine-tune the initial nominal training period with cross entropy loss.

between training and validation loss does not grow anymore.

For the last recorded epoch, the performance has been cross-checked with the test set introduced in the main part of the thesis, the corresponding ROC curves are given in Figs. F.4, F.5, F.6. Compared to the full-range adversarial training, the pre-training places the model more towards robustness on the performance – robustness scale.

This test shows that starting with a nominal warm-up phase may not be useful, if the subsequent adversarial training phase is very sensitive to overfitting and does not converge to similar performance as its plain nominal training counterpart. However, this is only the first time a fine-tuning of a nominal training is performed with adversarial training for this RobustParT network, and it can not be excluded that a better choice of parameters exists with which one can achieve performance, robustness and overall speed-up of the training all at once.

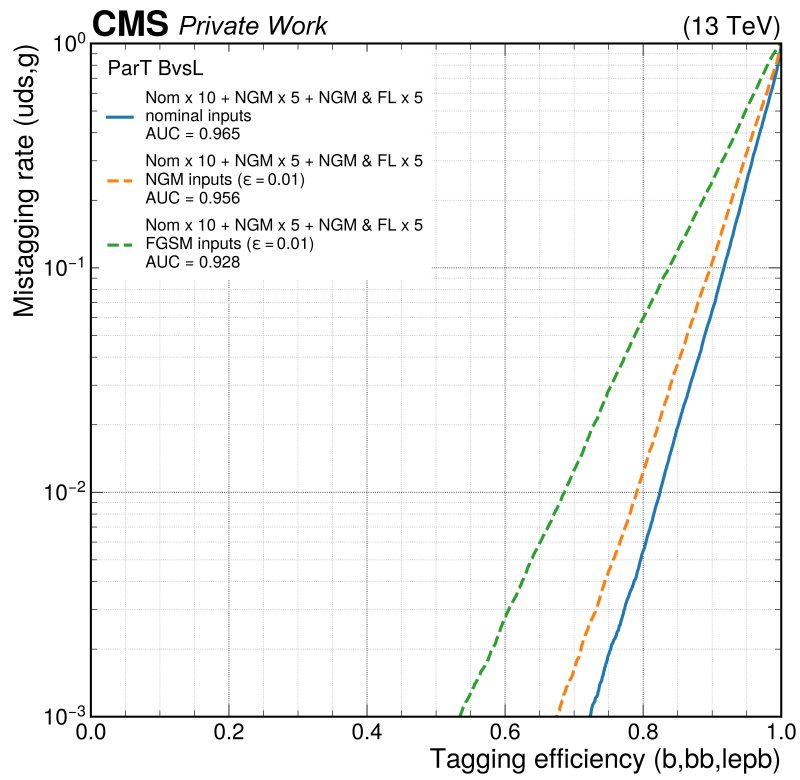


Figure F.4: BvsL performance for a pre-trained ParT NN with subsequent adversarial training phase and moving to focal loss, evaluated on nominal samples (blue), NGM-attacked samples (orange), and FGSM-attacked samples (green) with parameter $\epsilon = 0.01$ for attacks.

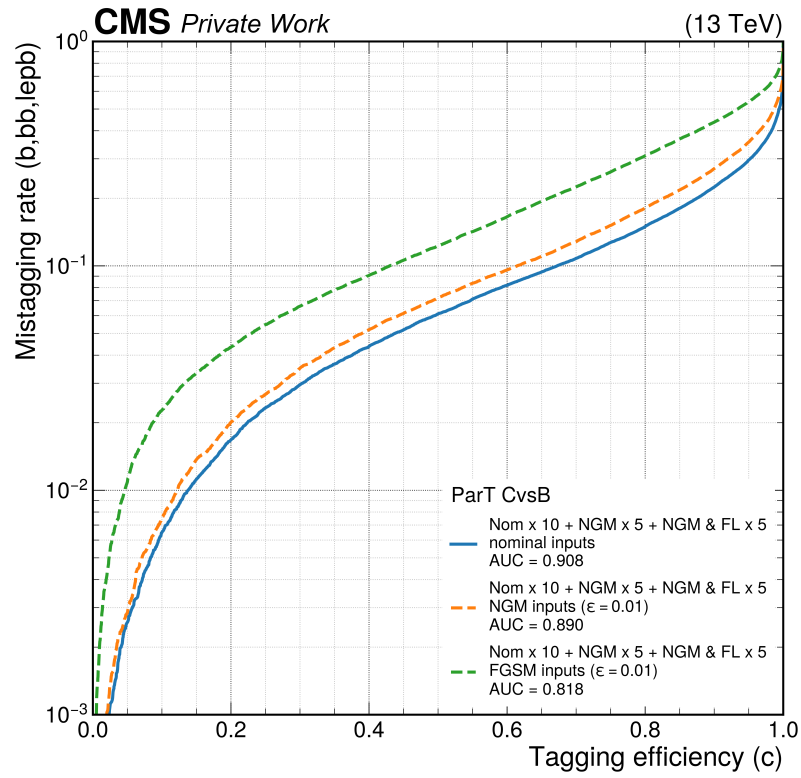


Figure F.5: CvsB performance for a pre-trained ParT NN with subsequent adversarial training phase and moving to focal loss, evaluated on nominal samples (blue), NGM-attacked samples (orange), and FGSM-attacked samples (green) with parameter $\epsilon = 0.01$ for attacks.

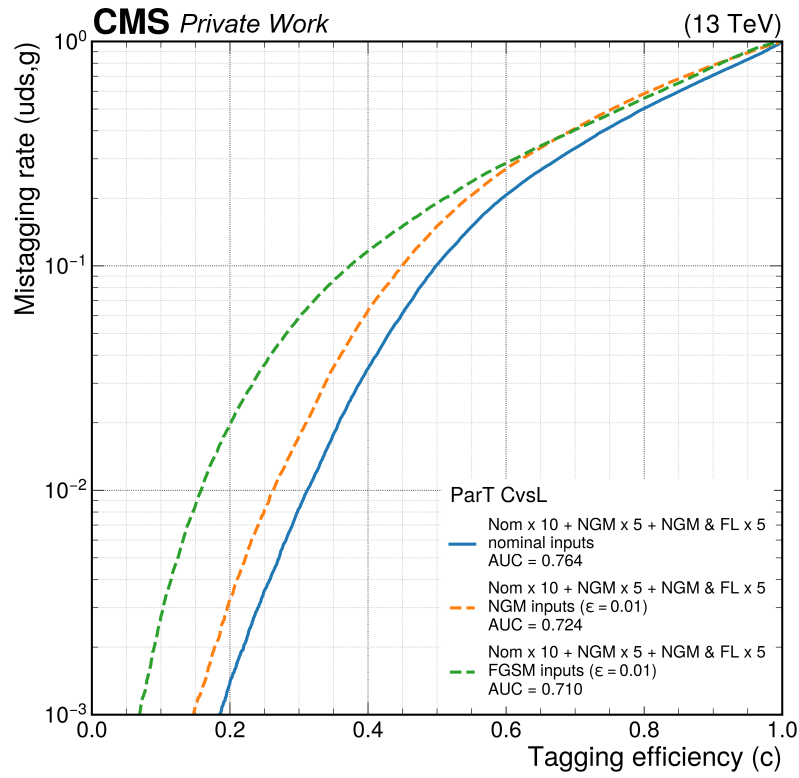


Figure F.6: CvsL performance for a pre-trained ParT NN with subsequent adversarial training phase and moving to focal loss, evaluated on nominal samples (blue), NGM-attacked samples (orange), and FGSM-attacked samples (green) with parameter $\epsilon = 0.01$ for attacks.

F.3 Technical Details: Modifying RobustParT for Run 3

F.3.1 Architecture and Input Changes compared to Run 2 RobustParT

During Run 3 and beyond, every inference speedup for ML tagging algorithms that are part of object reconstruction matters [156]. For this purpose, it was decided to perform feature ablation with respect to the Run 2 version, and construct energy-momentum tensors in a different coordinate system (which saves one step during inference), all without noticeably affecting the performance (see Table B.12 for the individual input feature changes between eras). In particular, the Run 2 model architecture involved calculating the cartesian coordinates of constituent four-vectors from their polar coordinate representation. In the Run 3 variant, this step is removed by accessing those alternative PF candidate properties directly.



The DataFormats/BTauReco^a package was prepared as a first step to introduce all variants of input features that support working in different coordinate systems [160]. This allowed for an on-the-fly transition after integrating a new version of RobustParTAK4 in the RecoBTag-Combined externals, without requiring a patch-release. Hence, such a piecewise integration decouples the data structure changes from the inference implementation and therefore has left enough room for performance improvements before commencing data-taking and prompt reconstruction.

^aSee also <https://github.com/cms-sw/cmssw/pull/40706>.

All modifications regarding newly scaled parameters via standard deviations over central quantiles for adversarial training are used. As mentioned already when discussing the architecture and setup for adversarial training for the Run 2 studies, also for Run 3 the NGM attack is only applied in every second batch.

F.3.2 Integration in CMSSW

The author has provided a first compilation of classes and scripts to introduce this algorithm in CMSSW [160]. This requires introduction of the necessary data structure (Reco), multiple producers (for features and outputs) and eventually new branches that can be used for calibration or usage in analyses (MiniAOD, NanoAOD). The initial version was presented 01/2023 (at this point, for version 12_6_0_patch1), in collaboration with the Reconstruction, JME (especially for what concerns the switch to PUPPI, and introduction of other algorithms in parallel), XPOG, and core software teams [160]; and with modifications from Alexandre de Moor (inference), the final version was integrated in the CMSSW/master branch in May 2023 and backported to relevant CMSSW versions thereafter*. There are R&D activities geared towards processing data in batches [156], the implementation of RobustParticleTransformer follows a similar paradigm by working with dynamic batch sizes and variable sized input collections. The following technical documentation ensures reproducibility with the same setup, noting how small variations in the internal operator representation could be a cause for output score discrepancies.

*See also <https://github.com/cms-sw/cmssw/pull/41275>.



Interlude: technical aspects of using a PyTorch transformer model in CMSSW

In CMSSW, neural networks are accessed via ONNX [368], where inference sessions need to be started. Such sessions can be used with a single sample or in batch mode, but the exported model needs to be compatible with such applications. During development, the model is trained using a specific version of PyTorch and inference is done with a specific version of the ONNXRuntime in C++ (it was 1.10.0 when RobustParTAK4 was integrated), each featuring a set of supported operators. Dynamic input sizes are another difficulty that must be treated when exporting a model for interchange, as not all operators behave same when given different lengths of input vectors. Some of the originally used operators therefore had to be rewritten and adapted to what is available in the experiment software, which uses ONNX as an external package. It can not be updated to a new version on-the-fly, especially not when deadlines for reconstruction changes have passed for a targeted release. To allow a multitude of use cases of a given model in an unambiguous way, the computation steps for model evaluation can be translated into “script”-format, exploiting just-in-time compilation via `torch.jit.script` with the Python decorator syntax before calling `onnx.export` [248]. In order to test locally and mimic the behavior that would be observed when running inference, PyTorch’s own code was modified by the author to include all requirements, none needing a new build of the software^a:

- Modifications inside `torch/nn/modules/activation.py`: the `MultiheadAttention` class receives type hints for its constructor, string formatting operators are replaced, strong typing of booleans as `True` or `False` replace non-empty strings
- Changes for `torch/nn/functional.py`: dynamically extracted tuples for linear layers as part of the forward call for MHA are replaced by fixed list components that can be accessed sequentially as separate entries (for which ONNX has the required operator structure)

Converted models were inspected with Netron [380] to validate applicability of operators on dynamic input sizes.

After the model was prepared, PyTorch received a major version update (to version 2), where compiled models, similar to TensorFlow [365], became possible. Especially for MHA-based blocks, operators were added and improved internally to profit from speedup through compilation. With this update, modifications as presented above are nowadays not necessary anymore. Also, in the meantime, the ONNXRuntime has been updated in CMSSW (to version 1.14.1), now supporting more operators.

^aSee also <https://github.com/pytorch/pytorch/issues/94675>.

F.3.3 Cross-Check of ONNX Evaluation Pipeline in CMSSW with PyTorch

As outlined in the technical details regarding implementation of novel machine learning algorithms for CMS, the software and data structures used for training are different from those for inference with the experiment-specific tools. The major difference is a switch from a native PyTorch model [248, 255] to the flexible ONNX format [368], and the model then runs on different variants of derived or parent datasets. One variant injects flat ntuples in machine learning-readable tensors for the PyTorch model, the other runs inside the PAT process

in CMSSW on top of multiple object collections with cross-references [160]. To prove the C++ implementation in CMSSW corresponds to what is done in the "fully pythonic" ML setup, the two pipelines are run on the same parent dataset and the outputs are compared against each other. Neglecting numerical precision differences, positive correlation is expected. Figure F.7

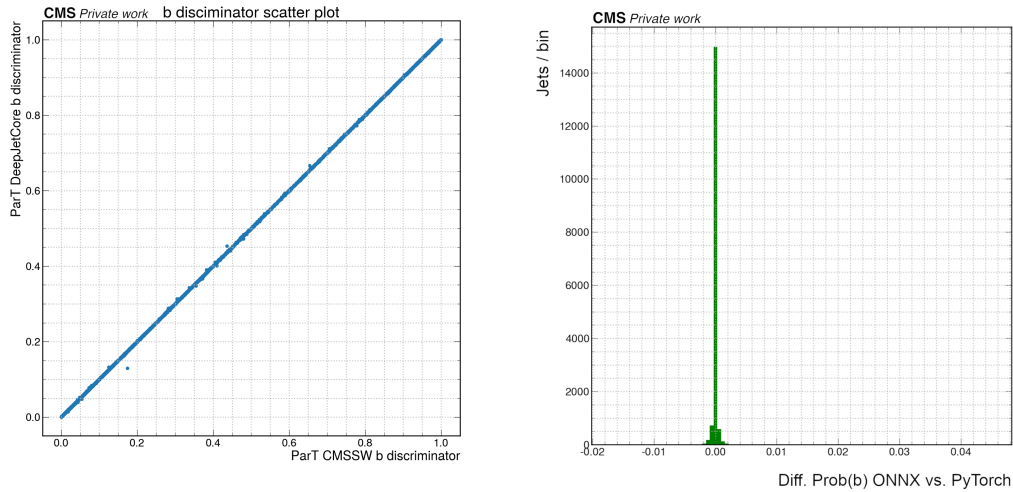


Figure F.7: Validating RobustParticleTransformer PyTorch model output scores against ONNX converted outputs as present in CMSSW. Correlation between the PyTorch output scores obtained from tuples in the DeepJetCore framework and the ONNX model in CMSSW are shown on the left, with the distribution of individual differences between the two configurations given on the right. Figures obtained in collaboration with Alexandre de Moor, VUB.

confirms the expectation for an exemplary tagger score (separating b jets from other jets), the correlation is almost perfect ($\rho \approx 1$). Numerical differences are not frequent and for jets where they exist, the difference never exceeds 0.002, thus the granularity of the typical bin width for scale factor derivation could be increased by a factor of 20, before this difference becomes visible. Similar results have been obtained for other output score and discriminant comparisons. It is thus safe to proceed with the integration, no software-induced deviations are expected from model conversion, feature preparation and inference.

F.3.4 Timing Performance for Feature Production and Inference in CMSSW

Tests prior to inclusion of RobustParT in CMSSW include investigating the time consumption to produce the input features and run inference on top of them, for example when producing MiniAOD. The comparison is done between ParticleNetAK4 and ParticleTransformerAK4, because the jet identification performance difference between them is not as large as the difference to less complex models like DeepJet or DeepFlavour / DeepCSV. Using the built-in TimeReport functionality of CMSSW [160] version 13_0_0_pre3, each producer can be timed and the event throughput investigated. The two tasks with TagInfos or JetTags suffix describe how long it takes to build the input features and then create the tagger outputs, respectively. The standard sequence is measured on an lxplus7 node and yields the results presented in Tab. F.1. For RobustParTAK4, inference speed increases by more than 50% over ParticleNetAK4, building the inputs from jet constituent information needs about 15% less time. Similar

Table F.1: Runtime comparison for production of different TagInfos and JetTags associated with ParticleNetAK4 and ParticleTransformerAK4, respectively, when investigating the producers in the PatAlgos/slimming (MiniAOD) step. Values obtained in collaboration with Alexandre de Moor, VUB.

Product	Time [s]
pfParticleNetAK4JetTags	0.020176
pfParticleNetAK4TagInfos	0.002436
pfParticleTransformerAK4JetTags	0.017060
pfParticleTransformerAK4TagInfos	0.001172

observations regarding inference speed when comparing ParT versus PNet for quark gluon separation are reported in Ref. [\[224\]](#).



Extended PFNano Samples and Estimated Relevance of Robust Algorithms Applied to Physics Analysis

The framework for training and evaluating novel jet flavour tagging algorithms on PFNano samples has been introduced in Sec. D. It can be extended to include either multiple jets at once, or a full picture of an event in terms of the available constituents and vertices. For this cautionary tale related to a physics use case, legacy PFNano samples for conditions matching the year 2017 have been generated to exemplify the relevance of robust algorithms for Higgs boson decay into charm (bottom) quark-antiquark pairs as one of the group's primary interests [199]*. Other than that, the choice of physics process is arbitrary and related solely to the availability of existing extended samples at the time of writing. Statements related to higher jet multiplicities contributing to NN susceptibility effects when multiple jets are used for training and inference at the same time also apply to other topologies, for example involving top quark pair production. When such high-dimensional inputs are taken into account in highly performing models, robust algorithms become more important, as all previous investigations in this thesis have shown, and for which Ref. [13] presents a possible explanation. Examples are shown in Fig. G.1, where the vast majority of available PF candidates of a single event are not clustered in a jet, but are spread through the entire tracker volume suitable for ML purposes and beyond. Besides positions of candidates, PFNano also records relative positions with respect to jets and vertices, such that expert variables are constructed from multiple, possibly susceptible individual low-level features. While such exemplary "event pictures" with only one property, like transverse momentum, shown as colour, do not allow classification of the underlying processes by humans, they are typical particle clouds to be investigated by GNNs [241] or transformers [20]. Similar patterns are presented in Fig. G.2 and may be taken into account for SVs (left) or features even closer to detector level related to hit patterns obtained from silicon modules in the tracker, like the property `lostInnerHits`[†] (right). In any case, whether the input collection content is kept at (multiple) jet level to enable calibration, or whether the entire event with all its low-level information is taken into account for even better exploitation of available correlations across particles, it can be estimated that the number of inputs increases by about a factor of two for events with at least two

*As discussed as a possible application of modern ML techniques at the DPG Spring Conference in March 2023: "Direct search for Higgs boson decay to a pair of charm quarks in the vector boson associated production mode at CMS", see also <https://www.dpg-verhandlungen.de/year/2023/conference/smuk/part/t/session/30/contribution/1> and "Development of novel machine learning algorithms for robust jet flavour classification for Run3 at CMS", <https://www.dpg-verhandlungen.de/year/2023/conference/smuk/part/t/session/86/contribution/2>.

[†]This variable is defined in MiniAOD and documented here: <https://twiki.cern.ch/twiki/bin/view/CMSPublic/WorkbookTrackAnalysis#HitPattern>.

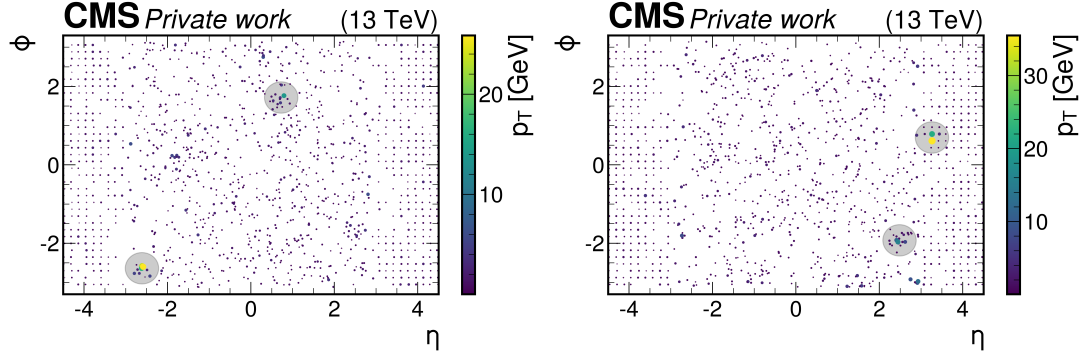


Figure G.1: Newer versions of PFNano contain low-level information for every PF candidate in a single exemplary event. This can be overlaid with shapes representing position and radius of signal jets from Higgs boson decay into charm (bottom) quark-antiquark pairs on the left (right) to demonstrate how many more dimensions can potentially contribute to NN susceptibility in a realistic use case. Particles identified at high pseudorapidities appear with less granular patterns than those inside the tracker acceptance region ($|\eta| < 2.5$). Usually, jet flavour tagging algorithms receive only a small fraction of the available information at once, associated with particles (or SVs, not shown here) in a single grey cone.

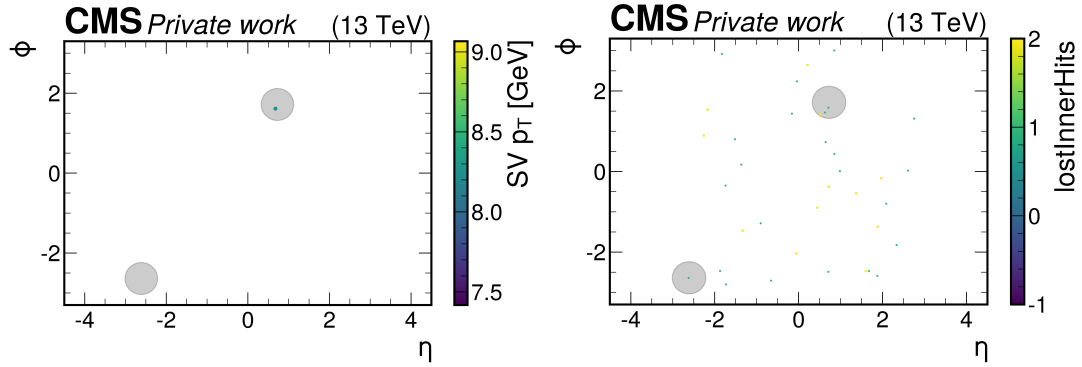


Figure G.2: More examples of additional (low-level) input features that can be studied with PFNano, using a $H \rightarrow c\bar{c}$ event as an example. What is possible for PF candidates can also be done for secondary vertices (left), or integer quantities like the number of lost inner hits (right).

well-reconstructed jets, if more than one jet is used as input to a NN. Besides computational complexity, this comes with another caveat: higher susceptibility towards small modifications, taking the dimensionality argument of Ref. [13] into account, with unclear impact of generalization gap between training on simulated samples and application to detector data. From all previous studies, it can be assumed that this use case will show once more the relevance of the performance and robustness trade-off, and the performance – data/MC agreement trade-off. It has been discussed how also after application of calibration methods, there can be phase space regions where agreement within uncertainties is not reached (see Sec. 4.2.4). In typical calibration pipelines, flavour tagging algorithms are not re-run on top of systematically shifted inputs, but the contribution of samples passing the selection criteria is modified or reweighted with uncertainties, up- and downward shifts. The unknown component has been studied in Chapters 6 and 7, where adversarial techniques proved to be useful as a coun-

term measure when introducing systematic shifts. The latest published result on the search for Higgs boson decay into a charm quark-antiquark pair [17] lists c jet identification efficiencies with 23 % as second largest experimental uncertainty source on the signal strength modifier. New generations of algorithms, such as the one introduced in Sec. 7.3, outperform previous variants with which the latest analysis has been performed. Therefore, it is very likely that analyzers will then need to assess carefully, how much performance (in terms of achieving better separation between signal and background) can be sacrificed, to not introduce larger experimental uncertainties related to jet tagging. Greater importance should be given to robust algorithms, should the architectures include more inputs in the future.

References

- [1] R. L. Workman et al. (Particle Data Group), “Review of Particle Physics”, *PTEP* **2022** (2022) 083C01, <http://dx.doi.org/10.1093/ptep/ptac097>. With 2023 updates.
- [2] D. Guest, K. Cranmer, and D. Whiteson, “Deep Learning and its Application to LHC Physics”, *Ann. Rev. Nucl. Part. Sci.* **68** (2018) 161–181, <http://dx.doi.org/10.1146/annurev-nucl-101917-021019>, [arXiv:1806.11484](https://arxiv.org/abs/1806.11484).
- [3] A. J. Larkoski, I. Moult, and B. Nachman, “Jet substructure at the Large Hadron Collider: A review of recent advances in theory and machine learning”, *Physics Reports* **841** (Jan, 2020) 1–63, <http://dx.doi.org/10.1016/j.physrep.2019.11.001>, [arXiv:1709.04464](https://arxiv.org/abs/1709.04464).
- [4] HEP ML Community, “A Living Review of Machine Learning for Particle Physics”, 2023. <https://iml-wg.github.io/HEPML-LivingReview/> (last accessed: 06.11.2023).
- [5] LHC Study Group, “The Large Hadron Collider: Conceptual design”, technical report, October, 1995. CERN-AC-95-05-LHC. <http://cds.cern.ch/record/291782> (last accessed: 05.12.2023).
- [6] O. S. Brüning et al., “LHC Design Report”. CERN Yellow Reports: Monographs. CERN, Geneva, 2004. CERN-2004-003. <https://cds.cern.ch/record/782076> (last accessed: 05.12.2023). <http://dx.doi.org/10.5170/CERN-2004-003-V-1>.
- [7] CMS Collaboration, “The CMS Experiment at the CERN LHC”, *JINST* **3** (2008) S08004, <http://dx.doi.org/10.1088/1748-0221/3/08/S08004>.
- [8] D. Guest et al., “Jet Flavor Classification in High-Energy Physics with Deep Neural Networks”, *Phys. Rev. D* **94** (2016), no. 11, 112002, <http://dx.doi.org/10.1103/PhysRevD.94.112002>, [arXiv:1607.08633](https://arxiv.org/abs/1607.08633).
- [9] CMS Collaboration, “Identification of heavy-flavour jets with the CMS detector in pp collisions at 13 TeV”, *JINST* **13** (2018), no. 05, P05011, <http://dx.doi.org/10.1088/1748-0221/13/05/P05011>, [arXiv:1712.07158](https://arxiv.org/abs/1712.07158).
- [10] ATLAS Collaboration, “ATLAS b-jet identification performance and efficiency measurement with $t\bar{t}$ events in pp collisions at $\sqrt{s} = 13$ TeV”, *Eur. Phys. J. C* **79** (2019), no. 11, 970, <http://dx.doi.org/10.1140/epjc/s10052-019-7450-8>, [arXiv:1907.05120](https://arxiv.org/abs/1907.05120).
- [11] A. Stein et al., “Improving Robustness of Jet Tagging Algorithms with Adversarial Training”, *Comput. Softw. Big Sci.* **6** (2022), no. 1, 15, <http://dx.doi.org/10.1007/s41781-022-00087-1>, [arXiv:2203.13890](https://arxiv.org/abs/2203.13890).
- [12] C. Szegedy et al., “Intriguing properties of neural networks”, *arXiv e-prints* (2014) [arXiv:1312.6199](https://arxiv.org/abs/1312.6199).
- [13] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and Harnessing Adversarial Examples”, *arXiv e-prints* (2015) [arXiv:1412.6572](https://arxiv.org/abs/1412.6572).
- [14] B. Nachman and C. Shimmin, “AI Safety for High Energy Physics”, *arXiv e-prints* (10, 2019) [arXiv:1910.08606](https://arxiv.org/abs/1910.08606).
- [15] cshimmin / C. Shimmin, “advjets-mlhep2020”, *GitHub repository* (2020). <https://github.com/cshimmin/advjets-mlhep2020> (last accessed: 04.12.2023).

- [16] A. Stein, “Improving robustness of jet tagging algorithms with adversarial training: exploring the loss surface”, in *21th International Workshop on Advanced Computing and Analysis Techniques in Physics Research: AI meets Reality*. 3, 2023. [arXiv:2303.14511](#).
- [17] CMS Collaboration, “A new calibration method for charm jet identification validated with proton-proton collision events at $\sqrt{s}=13$ TeV”, *JINST* **17** (2022), no. 03, P03014, [http://dx.doi.org/10.1088/1748-0221/17/03/P03014](#), [arXiv:2111.03027](#).
- [18] E. Bols et al., “Jet Flavour Classification Using DeepJet”, *JINST* **15** (2020), no. 12, P12012, [http://dx.doi.org/10.1088/1748-0221/15/12/P12012](#), [arXiv:2008.10519](#).
- [19] CMS Collaboration, “Adversarial training for b-tagging algorithms in CMS”, CMS Detector Performance Summary CMS-DP-2022-049, CERN, Geneva, 2022. [http://cds.cern.ch/record/2839919](#) (last accessed: 04.11.2023).
- [20] H. Qu, C. Li, and S. Qian, “Particle Transformer for Jet Tagging”, *arXiv e-prints* (2, 2022) [arXiv:2202.03772](#).
- [21] CMS Collaboration, “Transformer models for heavy flavor jet identification”, CMS Detector Performance Summary CMS-DP-2022-050, CERN, Geneva, 2022. [https://cds.cern.ch/record/2839920](#) (last accessed: 16.11.2023).
- [22] A. Stein and A. de Moor, “Jet flavour identification for Run 3 with the RobustParTAK4 algorithm”, 2023. CMS-AN-23-062.
- [23] B. Nachman, “When, Where, and How to Open Data: A Personal Perspective”, in *Snowmass 2021*. 8, 2022. [arXiv:2208.07910](#).
- [24] M. Gell-Mann, “A Schematic Model of Baryons and Mesons”, *Phys. Lett.* **8** (1964) 214–215, [http://dx.doi.org/10.1016/S0031-9163\(64\)92001-3](#).
- [25] S. Weinberg, “A Model of Leptons”, *Phys. Rev. Lett.* **19** (1967) 1264–1266, [http://dx.doi.org/10.1103/PhysRevLett.19.1264](#).
- [26] M. Thomson, “Modern Particle Physics”. Modern Particle Physics. Cambridge University Press, 2013. ISBN 9781107034266.
- [27] F. Halzen and A. D. Martin, “QUARKS AND LEPTONS: AN INTRODUCTORY COURSE IN MODERN PARTICLE PHYSICS”. John Wiley & Sons, 1984. ISBN 978-0-471-88741-6.
- [28] D. Griffiths, “Introduction to elementary particles”. John Wiley & Sons, 2008. ISBN 978-3-527-40601-2.
- [29] M. E. Peskin and D. V. Schroeder, “An Introduction to quantum field theory”. Addison-Wesley, Reading, USA, 1995. ISBN 978-0-201-50397-5.
- [30] S. Weinberg, “The Quantum theory of fields. Vol. 1: Foundations”. Cambridge University Press, 6, 2005. [http://dx.doi.org/10.1017/CB09781139644167](#), ISBN 978-0-521-67053-1, 978-0-511-25204-4.
- [31] S. Weinberg, “The quantum theory of fields. Vol. 2: Modern applications”. Cambridge University Press, 8, 2013. [http://dx.doi.org/10.1017/CB09781139644174](#), ISBN 978-1-139-63247-8, 978-0-521-67054-8, 978-0-521-55002-4.
- [32] M. D. Schwartz, “Quantum Field Theory and the Standard Model”. Cambridge University Press, 3, 2014. ISBN 978-1-107-03473-0, 978-1-107-03473-0.
- [33] C. Jarlskog, “QUANTUM FLAVOR DYNAMICS”, *Phys. Scripta* **24** (1981) 867–872,

- <http://dx.doi.org/10.1088/0031-8949/24/5/010>.
- [34] S. L. Glashow, “Partial Symmetries of Weak Interactions”, *Nucl. Phys.* **22** (1961) 579–588, [http://dx.doi.org/10.1016/0029-5582\(61\)90469-2](http://dx.doi.org/10.1016/0029-5582(61)90469-2).
- [35] KATRIN Collaboration, “Direct neutrino-mass measurement with sub-electronvolt sensitivity”, *Nature Phys.* **18** (2022), no. 2, 160–166, <http://dx.doi.org/10.1038/s41567-021-01463-1>, [arXiv:2105.08533](https://arxiv.org/abs/2105.08533).
- [36] K. Assamagan et al., “Upper limit of the muon-neutrino mass and charged pion mass from momentum analysis of a surface muon beam”, *Phys. Rev. D* **53** (1996) 6065–6077, <http://dx.doi.org/10.1103/PhysRevD.53.6065>.
- [37] ALEPH Collaboration, “An Upper limit on the tau-neutrino mass from three-prong and five-prong tau decays”, *Eur. Phys. J. C* **2** (1998) 395–406, <http://dx.doi.org/10.1007/s100520050149>.
- [38] K. G. Wilson, “Confinement of quarks”, *Phys. Rev. D* **10** (Oct, 1974) 2445–2459, <http://dx.doi.org/10.1103/PhysRevD.10.2445>.
- [39] LHCb Collaboration, “The LHCb Detector at the LHC”, *JINST* **3** (2008) S08005, <http://dx.doi.org/10.1088/1748-0221/3/08/S08005>.
- [40] LHCb Collaboration, “Observation of $J/\psi p$ Resonances Consistent with Pentaquark States in $\Lambda_b^0 \rightarrow J/\psi K^- p$ Decays”, *Phys. Rev. Lett.* **115** (2015) 072001, <http://dx.doi.org/10.1103/PhysRevLett.115.072001>, [arXiv:1507.03414](https://arxiv.org/abs/1507.03414).
- [41] LHCb Collaboration, “Observation of structure in the J/ψ -pair mass spectrum”, *Sci. Bull.* **65** (2020), no. 23, 1983–1993, <http://dx.doi.org/10.1016/j.scib.2020.08.032>, [arXiv:2006.16957](https://arxiv.org/abs/2006.16957).
- [42] C. N. Yang and R. L. Mills, “Conservation of isotopic spin and isotopic gauge invariance”, *Phys. Rev.* **96** (Oct, 1954) 191–195, <http://dx.doi.org/10.1103/PhysRev.96.191>.
- [43] E. Noether, “Invariant Variation Problems”, *Gott. Nachr.* **1918** (1918) 235–257, <http://dx.doi.org/10.1080/00411457108231446>, [arXiv:physics/0503066](https://arxiv.org/abs/physics/0503066).
- [44] J. C. Ward, “An Identity in Quantum Electrodynamics”, *Phys. Rev.* **78** (1950) 182, <http://dx.doi.org/10.1103/PhysRev.78.182>.
- [45] Y. Takahashi, “On the generalized Ward identity”, *Nuovo Cim.* **6** (1957) 371, <http://dx.doi.org/10.1007/BF02832514>.
- [46] A. J. Buras, J. R. Ellis, M. K. Gaillard, and D. V. Nanopoulos, “Aspects of the Grand Unification of Strong, Weak and Electromagnetic Interactions”, *Nucl. Phys. B* **135** (1978) 66–92, [http://dx.doi.org/10.1016/0550-3213\(78\)90214-6](http://dx.doi.org/10.1016/0550-3213(78)90214-6).
- [47] H. Fritzsch, M. Gell-Mann, and H. Leutwyler, “Advantages of the Color Octet Gluon Picture”, *Phys. Lett. B* **47** (1973) 365–368, [http://dx.doi.org/10.1016/0370-2693\(73\)90625-4](http://dx.doi.org/10.1016/0370-2693(73)90625-4).
- [48] A. Salam, “Weak and electromagnetic interactions”, in *Elementary particle physics: relativistic groups and analyticity*, N. Svartholm, ed., p. 367. Almqvist & Wiksell, Stockholm, 1968. Proceedings of the eighth Nobel symposium. http://dx.doi.org/10.1142/9789812795915_0034.
- [49] C. S. Wu et al., “Experimental Test of Parity Conservation in β Decay”, *Phys. Rev.* **105**

- (1957) 1413–1414, <http://dx.doi.org/10.1103/PhysRev.105.1413>.
- [50] T. Nakano and K. Nishijima, “Charge Independence for V-particles”, *Prog. Theor. Phys.* **10** (1953) 581–582, <http://dx.doi.org/10.1143/PTP.10.581>.
- [51] K. Nishijima, “Charge Independence Theory of V Particles”, *Prog. Theor. Phys.* **13** (1955), no. 3, 285–304, <http://dx.doi.org/10.1143/PTP.13.285>.
- [52] M. Gell-Mann, “The interpretation of the new particles as displaced charge multiplets”, *Nuovo Cim.* **4** (1956), no. S2, 848–866, <http://dx.doi.org/10.1007/BF02748000>.
- [53] J. Goldstone, “Field Theories with Superconductor Solutions”, *Nuovo Cim.* **19** (1961) 154–164, <http://dx.doi.org/10.1007/BF02812722>.
- [54] F. Englert and R. Brout, “Broken Symmetry and the Mass of Gauge Vector Mesons”, *Phys. Rev. Lett.* **13** (1964) 321–323, <http://dx.doi.org/10.1103/PhysRevLett.13.321>.
- [55] P. W. Higgs, “Broken symmetries, massless particles and gauge fields”, *Phys. Lett.* **12** (1964) 132–133, [http://dx.doi.org/10.1016/0031-9163\(64\)91136-9](http://dx.doi.org/10.1016/0031-9163(64)91136-9).
- [56] P. W. Higgs, “Spontaneous Symmetry Breakdown without Massless Bosons”, *Phys. Rev.* **145** (1966) 1156–1163, <http://dx.doi.org/10.1103/PhysRev.145.1156>.
- [57] CMS Collaboration, “A portrait of the Higgs boson by the CMS experiment ten years after the discovery”, *Nature* **607** (2022), no. 7917, 60–68, <http://dx.doi.org/10.1038/s41586-022-04892-x>, [arXiv:2207.00043](https://arxiv.org/abs/2207.00043).
- [58] ATLAS Collaboration, “A detailed map of Higgs boson interactions by the ATLAS experiment ten years after the discovery”, *Nature* **607** (2022), no. 7917, 52–59, <http://dx.doi.org/10.1038/s41586-022-04893-w>, [arXiv:2207.00092](https://arxiv.org/abs/2207.00092). [Erratum: *Nature* 612, E24 (2022)].
- [59] N. Cabibbo, “Unitary Symmetry and Leptonic Decays”, *Phys. Rev. Lett.* **10** (1963) 531–533, <http://dx.doi.org/10.1103/PhysRevLett.10.531>.
- [60] M. Kobayashi and T. Maskawa, “CP Violation in the Renormalizable Theory of Weak Interaction”, *Prog. Theor. Phys.* **49** (1973) 652–657, <http://dx.doi.org/10.1143/PTP.49.652>.
- [61] B. Pontecorvo, “Inverse beta processes and nonconservation of lepton charge”, *Zh. Eksp. Teor. Fiz.* **34** (1957) 247. <http://jetp.ras.ru/cgi-bin/e/index/e/7/1/p172?a=list> (last accessed: 05.12.2023).
- [62] Z. Maki, M. Nakagawa, and S. Sakata, “Remarks on the unified model of elementary particles”, *Prog. Theor. Phys.* **28** (1962) 870–880, <http://dx.doi.org/10.1143/PTP.28.870>.
- [63] D. Kar, “Experimental Particle Physics”. 2053-2563. IOP Publishing, 2019. <http://dx.doi.org/10.1088/2053-2563/ab1be6>, ISBN 978-0-7503-2112-9.
- [64] J. M. Campbell, J. W. Huston, and W. J. Stirling, “Hard Interactions of Quarks and Gluons: A Primer for LHC Physics”, *Rept. Prog. Phys.* **70** (2007) 89, <http://dx.doi.org/10.1088/0034-4885/70/1/R02>, [arXiv:hep-ph/0611148](https://arxiv.org/abs/hep-ph/0611148).
- [65] J. Butterworth et al., “PDF4LHC recommendations for LHC Run II”, *J. Phys. G* **43** (2016) 023001, <http://dx.doi.org/10.1088/0954-3899/43/2/023001>, [arXiv:1510.03865](https://arxiv.org/abs/1510.03865).
- [66] NNPDF Collaboration, “Parton distributions from high-precision collider data”, *Eur. Phys. J. C* **77** (2017), no. 10, 663, <http://dx.doi.org/10.1140/epjc/>

- s10052-017-5199-5, [arXiv:1706.00428](#).
- [67] PDF4LHC Working Group Collaboration, “The PDF4LHC21 combination of global PDF fits for the LHC Run III”, *J. Phys. G* **49** (2022), no. 8, 080501, <http://dx.doi.org/10.1088/1361-6471/ac7216>, [arXiv:2203.05506](#).
- [68] S. Forte, L. Garrido, J. I. Latorre, and A. Piccione, “Neural network parametrization of deep inelastic structure functions”, *JHEP* **05** (2002) 062, <http://dx.doi.org/10.1088/1126-6708/2002/05/062>, [arXiv:hep-ph/0204232](#).
- [69] R. D. Ball et al., “A first unbiased global NLO determination of parton distributions and their uncertainties”, *Nucl. Phys. B* **838** (2010) 136–206, <http://dx.doi.org/10.1016/j.nuclphysb.2010.05.008>, [arXiv:1002.4407](#).
- [70] NNPDF Collaboration, “The path to proton structure at 1% accuracy”, *Eur. Phys. J. C* **82** (2022), no. 5, 428, <http://dx.doi.org/10.1140/epjc/s10052-022-10328-7>, [arXiv:2109.02653](#).
- [71] NNPDF Collaboration, “An open-source machine learning framework for global analyses of parton distributions”, *Eur. Phys. J. C* **81** (2021), no. 10, 958, <http://dx.doi.org/10.1140/epjc/s10052-021-09747-9>, [arXiv:2109.02671](#).
- [72] NNPDF Collaboration, “Unbiased global determination of parton distributions and their uncertainties at NNLO and at LO”, *Nucl. Phys. B* **855** (2012) 153–221, <http://dx.doi.org/10.1016/j.nuclphysb.2011.09.024>, [arXiv:1107.2652](#).
- [73] Y. L. Dokshitzer, “Calculation of the Structure Functions for Deep Inelastic Scattering and e^+e^- Annihilation by Perturbation Theory in Quantum Chromodynamics.”, *Sov. Phys. JETP* **46** (1977) 641–653. http://www.jetp.ras.ru/cgi-bin/dn/e_046_04_0641 (last accessed: 05.12.2023).
- [74] V. N. Gribov and L. N. Lipatov, “Deep inelastic $e p$ scattering in perturbation theory”, *Sov. J. Nucl. Phys.* **15** (1972) 438–450.
- [75] G. Altarelli and G. Parisi, “Asymptotic Freedom in Parton Language”, *Nucl. Phys. B* **126** (1977) 298–318, [http://dx.doi.org/10.1016/0550-3213\(77\)90384-4](http://dx.doi.org/10.1016/0550-3213(77)90384-4).
- [76] S. Höche, “Introduction to parton-shower event generators”, in *Theoretical Advanced Study Institute in Elementary Particle Physics: Journeys Through the Precision Frontier: Amplitudes for Colliders*, pp. 235–295. 2015. [arXiv:1411.4085](#). http://dx.doi.org/10.1142/9789814678766_0005.
- [77] D. Amati and G. Veneziano, “Preconfinement as a Property of Perturbative QCD”, *Phys. Lett. B* **83** (1979) 87–92, [http://dx.doi.org/10.1016/0370-2693\(79\)90896-7](http://dx.doi.org/10.1016/0370-2693(79)90896-7).
- [78] G. C. Fox and S. Wolfram, “A Model for Parton Showers in QCD”, *Nucl. Phys. B* **168** (1980) 285–295, [http://dx.doi.org/10.1016/0550-3213\(80\)90111-X](http://dx.doi.org/10.1016/0550-3213(80)90111-X).
- [79] R. D. Field and S. Wolfram, “A QCD Model for e^+e^- Annihilation”, *Nucl. Phys. B* **213** (1983) 65–84, [http://dx.doi.org/10.1016/0550-3213\(83\)90175-X](http://dx.doi.org/10.1016/0550-3213(83)90175-X).
- [80] B. R. Webber, “A QCD Model for Jet Fragmentation Including Soft Gluon Interference”, *Nucl. Phys. B* **238** (1984) 492–528, [http://dx.doi.org/10.1016/0550-3213\(84\)90333-X](http://dx.doi.org/10.1016/0550-3213(84)90333-X).
- [81] J.-C. Winter, F. Krauss, and G. Soff, “A Modified cluster hadronization model”, *Eur. Phys. J. C* **36** (2004) 381–395, <http://dx.doi.org/10.1140/epjc/s2004-01960-8>,

- [arXiv:hep-ph/0311085](#).
- [82] X. Artru and G. Mennessier, “String model and multiproduction”, *Nucl. Phys. B* **70** (1974) 93–115, [http://dx.doi.org/10.1016/0550-3213\(74\)90360-5](http://dx.doi.org/10.1016/0550-3213(74)90360-5).
 - [83] B. Andersson, G. Gustafson, G. Ingelman, and T. Sjostrand, “Parton Fragmentation and String Dynamics”, *Phys. Rept.* **97** (1983) 31–145, [http://dx.doi.org/10.1016/0370-1573\(83\)90080-7](http://dx.doi.org/10.1016/0370-1573(83)90080-7).
 - [84] B. Andersson, “The Lund model”, volume 7. Cambridge University Press, 7, 2005. <http://dx.doi.org/10.1017/CB09780511524363>, ISBN 978-0-521-01734-3, 978-0-521-42094-5, 978-0-511-88149-7.
 - [85] A. Buckley et al., “General-purpose event generators for LHC physics”, *Phys. Rept.* **504** (2011) 145–233, <http://dx.doi.org/10.1016/j.physrep.2011.03.005>, [arXiv:1101.2599](#).
 - [86] J. Alwall et al., “MadGraph 5 : Going Beyond”, *JHEP* **06** (2011) 128, [http://dx.doi.org/10.1007/JHEP06\(2011\)128](http://dx.doi.org/10.1007/JHEP06(2011)128), [arXiv:1106.0522](#).
 - [87] R. Frederix and S. Frixione, “Merging meets matching in MC@NLO”, *JHEP* **12** (2012) 061, [http://dx.doi.org/10.1007/JHEP12\(2012\)061](http://dx.doi.org/10.1007/JHEP12(2012)061), [arXiv:1209.6215](#).
 - [88] S. Frixione, P. Nason, and C. Oleari, “Matching NLO QCD computations with Parton Shower simulations: the POWHEG method”, *JHEP* **11** (2007) 070, <http://dx.doi.org/10.1088/1126-6708/2007/11/070>, [arXiv:0709.2092](#).
 - [89] T. Sjöstrand, S. Mrenna, and P. Skands, “PYTHIA 6.4 physics and manual”, *JHEP* **05** (2006) 026, <http://dx.doi.org/10.1088/1126-6708/2006/05/026>, [arXiv:hep-ph/0603175](#).
 - [90] T. Sjöstrand et al., “An introduction to PYTHIA 8.2”, *Computer Physics Communications* **191** (Jun, 2015) 159–177, <http://dx.doi.org/10.1016/j.cpc.2015.01.024>, [arXiv:1410.3012](#).
 - [91] M. Bähr et al., “Herwig++ physics and manual”, *Eur. Phys. J. C* **58** (2008) 639, <http://dx.doi.org/10.1140/epjc/s10052-008-0798-9>, [arXiv:0803.0883](#).
 - [92] G. Corcella et al., “HERWIG 6: an event generator for hadron emission reactions with interfering gluons (including supersymmetric processes)”, *JHEP* **01** (2001) 010, <http://dx.doi.org/10.1088/1126-6708/2001/01/010>, [arXiv:hep-ph/0011363](#).
 - [93] T. Gleisberg et al., “Event generation with SHERPA 1.1”, *JHEP* **02** (2009) 007, <http://dx.doi.org/10.1088/1126-6708/2009/02/007>, [arXiv:0811.4622](#).
 - [94] J. Alwall et al., “A Standard format for Les Houches event files”, *Comput. Phys. Commun.* **176** (2007) 300–304, <http://dx.doi.org/10.1016/j.cpc.2006.11.010>, [arXiv:hep-ph/0609017](#).
 - [95] M. R. Whalley, D. Bourilkov, and R. C. Group, “The Les Houches accord PDFs (LHAPDF) and LHAGLUE”, in *HERA and the LHC: A Workshop on the Implications of HERA and LHC Physics (Startup Meeting, CERN, 26-27 March 2004; Midterm Meeting, CERN, 11-13 October 2004)*, pp. 575–581. 8, 2005. [arXiv:hep-ph/0508110](#).
 - [96] D. Bourilkov, R. C. Group, and M. R. Whalley, “LHAPDF: PDF use from the Tevatron to the LHC”, in *TeV4LHC Workshop - 4th meeting*. 5, 2006. [arXiv:hep-ph/0605240](#).
 - [97] S. Carrazza, S. Forte, and J. Rojo, “Parton Distributions and Event Generators”, in *43rd*

- International Symposium on Multiparticle Dynamics*, pp. 89–96. 2013. [arXiv:1311.5887](https://arxiv.org/abs/1311.5887).
- [98] M. Schönherr, “Parton shower matching and merging”, *PoS CORFU2017* (2018) 110, [http://dx.doi.org/10.22323/1.318.0110](https://dx.doi.org/10.22323/1.318.0110), [arXiv:1804.10643](https://arxiv.org/abs/1804.10643).
- [99] M. L. Mangano, M. Moretti, and R. Pittau, “Multijet matrix elements and shower evolution in hadronic collisions: $Wb\bar{b} + n$ jets as a case study”, *Nucl. Phys. B* **632** (2002) 343–362, [http://dx.doi.org/10.1016/S0550-3213\(02\)00249-3](https://dx.doi.org/10.1016/S0550-3213(02)00249-3), [arXiv:hep-ph/0108069](https://arxiv.org/abs/hep-ph/0108069).
- [100] P. Nason, “A new method for combining NLO QCD with shower Monte Carlo algorithms”, *JHEP* **11** (2004) 040, [http://dx.doi.org/10.1088/1126-6708/2004/11/040](https://dx.doi.org/10.1088/1126-6708/2004/11/040), [arXiv:hep-ph/0409146](https://arxiv.org/abs/hep-ph/0409146).
- [101] S. Alioli, P. Nason, C. Oleari, and E. Re, “A general framework for implementing NLO calculations in shower Monte Carlo programs: the POWHEG BOX”, *JHEP* **06** (2010) 043, [http://dx.doi.org/10.1007/JHEP06\(2010\)043](https://dx.doi.org/10.1007/JHEP06(2010)043), [arXiv:1002.2581](https://arxiv.org/abs/1002.2581).
- [102] J. Alwall et al., “The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations”, *JHEP* **07** (2014) 079, [http://dx.doi.org/10.1007/JHEP07\(2014\)079](https://dx.doi.org/10.1007/JHEP07(2014)079), [arXiv:1405.0301](https://arxiv.org/abs/1405.0301).
- [103] CMS Collaboration, “Extraction and validation of a new set of CMS PYTHIA8 tunes from underlying-event measurements”, *Eur. Phys. J. C* **80** (2020), no. 1, 4, [http://dx.doi.org/10.1140/epjc/s10052-019-7499-4](https://dx.doi.org/10.1140/epjc/s10052-019-7499-4), [arXiv:1903.12179](https://arxiv.org/abs/1903.12179).
- [104] CMS Collaboration, “CMS pythia 8 colour reconnection tunes based on underlying-event data”, *Eur. Phys. J. C* **83** (2023), no. 7, 587, [http://dx.doi.org/10.1140/epjc/s10052-023-11630-8](https://dx.doi.org/10.1140/epjc/s10052-023-11630-8), [arXiv:2205.02905](https://arxiv.org/abs/2205.02905).
- [105] K. Ehatäht and C. Veelken, “Stitching Monte Carlo samples”, *Eur. Phys. J. C* **82** (2022), no. 5, 484, [http://dx.doi.org/10.1140/epjc/s10052-022-10407-9](https://dx.doi.org/10.1140/epjc/s10052-022-10407-9), [arXiv:2106.04360](https://arxiv.org/abs/2106.04360).
- [106] P. Artoisenet, R. Frederix, O. Mattelaer, and R. Rietkerk, “Automatic spin-entangled decays of heavy resonances in Monte Carlo simulations”, *JHEP* **03** (2013) 015, [http://dx.doi.org/10.1007/JHEP03\(2013\)015](https://dx.doi.org/10.1007/JHEP03(2013)015), [arXiv:1212.3460](https://arxiv.org/abs/1212.3460).
- [107] M. Dobbs and J. B. Hansen, “The HepMC C++ Monte Carlo event record for High Energy Physics”, *Comput. Phys. Commun.* **134** (2001) 41–46, [http://dx.doi.org/10.1016/S0010-4655\(00\)00189-2](https://dx.doi.org/10.1016/S0010-4655(00)00189-2).
- [108] O. Buning et al., “LHC Design Report. 2. The LHC infrastructure and general services”, [http://dx.doi.org/10.5170/CERN-2004-003-V-2](https://dx.doi.org/10.5170/CERN-2004-003-V-2).
- [109] M. Benedikt et al., “LHC Design Report. 3. The LHC injector chain”, [http://dx.doi.org/10.5170/CERN-2004-003-V-3](https://dx.doi.org/10.5170/CERN-2004-003-V-3).
- [110] L. R. Evans and P. Bryant, “LHC Machine”, *JINST* **3** (2008) S08001. 164 p, [http://dx.doi.org/10.1088/1748-0221/3/08/S08001](https://dx.doi.org/10.1088/1748-0221/3/08/S08001). <http://cds.cern.ch/record/1129806> (last accessed: 05.12.2023). This report is an abridged version of the LHC Design Report (CERN-2004-003).
- [111] E. Lopienska, “The CERN accelerator complex, layout in 2022. Complexe des accélérateurs du CERN en janvier 2022”, *CERN Document Server* (2, 2022). General Photo. CERN-GRAPHICS-2022-001-1. <https://cds.cern.ch/record/2800984> (last accessed: 06.11.2023).

- [112] CERN Education, Communications and Outreach Group, “LHC Guide”, *CERN Document Server* (03, 2017). CERN-Brochure-2017-002-Eng. <http://cds.cern.ch/record/2255762> (last accessed: 05.12.2023).
- [113] R. Bruce et al., “Review of LHC Run 2 Machine Configurations”, in *9th LHC Operations Evian Workshop*, pp. 187–197. Geneva, Switzerland, 2019. <https://cds.cern.ch/record/2750415> (last accessed: 05.12.2023).
- [114] S. Fartoukh et al., “LHC Configuration and Operational Scenario for Run 3”, technical report, CERN, Geneva, 2021. <https://cds.cern.ch/record/2790409> (last accessed: 05.12.2023).
- [115] R. Assmann et al., “The LHC run 2022”, *JACoW IPAC2023* (2023) MOPL020, <http://dx.doi.org/10.18429/JACoW-IPAC2023-MOPL020>.
- [116] ATLAS Collaboration, “The ATLAS Experiment at the CERN Large Hadron Collider”, *JINST* **3** (2008) S08003, <http://dx.doi.org/10.1088/1748-0221/3/08/S08003>.
- [117] ATLAS Collaboration, “Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC”, *Physics Letters B* **716** (Sep, 2012) 1–29, <http://dx.doi.org/10.1016/j.physletb.2012.08.020>, [arXiv:1207.7214](https://arxiv.org/abs/1207.7214).
- [118] CMS Collaboration, “Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC”, *Physics Letters B* **716** (Sep, 2012) 30–61, <http://dx.doi.org/10.1016/j.physletb.2012.08.021>, [arXiv:1207.7235](https://arxiv.org/abs/1207.7235).
- [119] LHCf Collaboration, “The LHCf detector at the CERN Large Hadron Collider”, *JINST* **3** (2008) S08006, <http://dx.doi.org/10.1088/1748-0221/3/08/S08006>.
- [120] LHCf Collaboration, “Measurement of zero degree single photon energy spectra for $\sqrt{s} = 7$ TeV proton-proton collisions at LHC”, *Phys. Lett. B* **703** (2011) 128–134, <http://dx.doi.org/10.1016/j.physletb.2011.07.077>, [arXiv:1104.5294](https://arxiv.org/abs/1104.5294).
- [121] FASER Collaboration, “Technical Proposal for FASER: ForwArd Search ExpeRiment at the LHC”, [arXiv:1812.09139](https://arxiv.org/abs/1812.09139).
- [122] FASER Collaboration, “First Direct Observation of Collider Neutrinos with FASER at the LHC”, *Phys. Rev. Lett.* **131** (2023), no. 3, 031801, <http://dx.doi.org/10.1103/PhysRevLett.131.031801>, [arXiv:2303.14185](https://arxiv.org/abs/2303.14185).
- [123] SND@LHC Collaboration, “SND@LHC: The Scattering and Neutrino Detector at the LHC”, [arXiv:2210.02784](https://arxiv.org/abs/2210.02784).
- [124] SND@LHC Collaboration, “Observation of Collider Muon Neutrinos with the SND@LHC Experiment”, *Phys. Rev. Lett.* **131** (2023), no. 3, 031802, <http://dx.doi.org/10.1103/PhysRevLett.131.031802>, [arXiv:2305.09383](https://arxiv.org/abs/2305.09383).
- [125] ALICE Collaboration, “The ALICE experiment at the CERN LHC”, *JINST* **3** (2008) S08002, <http://dx.doi.org/10.1088/1748-0221/3/08/S08002>.
- [126] ALICE Collaboration, “Direct observation of the dead-cone effect in quantum chromodynamics”, *Nature* **605** (2022), no. 7910, 440–446, <http://dx.doi.org/10.1038/s41586-022-04572-w>, [arXiv:2106.05713](https://arxiv.org/abs/2106.05713). [Erratum: *Nature* 607, E22 (2022)].
- [127] TOTEM Collaboration, “The TOTEM experiment at the CERN Large Hadron Collider”, *JINST* **3** (2008) S08007, <http://dx.doi.org/10.1088/1748-0221/3/08/S08007>.

- [128] TOTEM Collaboration, “First measurement of elastic, inelastic and total cross-section at $\sqrt{s} = 13$ TeV by TOTEM and overview of cross-section data at LHC energies”, *Eur. Phys. J. C* **79** (2019), no. 2, 103, <http://dx.doi.org/10.1140/epjc/s10052-019-6567-0>, [arXiv:1712.06153](https://arxiv.org/abs/1712.06153).
- [129] milliQan Collaboration, “Sensitivity to millicharged particles in future proton-proton collisions at the LHC with the milliQan detector”, *Phys. Rev. D* **104** (2021), no. 3, 032002, <http://dx.doi.org/10.1103/PhysRevD.104.032002>, [arXiv:2104.07151](https://arxiv.org/abs/2104.07151).
- [130] R. A. Millikan, “On the Elementary Electrical Charge and the Avogadro Constant”, *Phys. Rev.* **2** (1913) 109–143, <http://dx.doi.org/10.1103/PhysRev.2.109>.
- [131] MoEDAL Collaboration, “Technical Design Report of the MoEDAL Experiment”, technical report, 6, 2009. CERN-LHCC-2009-006, MoEDAL-TDR-001. <https://cds.cern.ch/record/1181486> (last accessed: 01.11.2023).
- [132] MoEDAL Collaboration, “Search for Magnetic Monopoles with the MoEDAL Forward Trapping Detector in 13 TeV Proton-Proton Collisions at the LHC”, *Phys. Rev. Lett.* **118** (2017), no. 6, 061801, <http://dx.doi.org/10.1103/PhysRevLett.118.061801>, [arXiv:1611.06817](https://arxiv.org/abs/1611.06817).
- [133] CMS Collaboration, “Public CMS Luminosity Information”, 2023. <https://twiki.cern.ch/twiki/bin/view/CMSPublic/LumiPublicResults> (last accessed: 01.11.2023).
- [134] CMS Collaboration, “Precision luminosity measurement in proton-proton collisions at $\sqrt{s} = 13$ TeV in 2015 and 2016 at CMS”, *Eur. Phys. J. C* **81** (2021) 800, <http://dx.doi.org/10.1140/epjc/s10052-021-09538-2>, [arXiv:2104.01927](https://arxiv.org/abs/2104.01927).
- [135] G. Apollinari et al., “High-Luminosity Large Hadron Collider (HL-LHC): Preliminary Design Report”. CERN Yellow Reports: Monographs. CERN, Geneva, 2015. CERN-2015-005. <https://cds.cern.ch/record/2116337> (last accessed: 05.12.2023). <http://dx.doi.org/10.5170/CERN-2015-005>.
- [136] O. Aberle et al., “High-Luminosity Large Hadron Collider (HL-LHC): Technical design report”. CERN Yellow Reports: Monographs. CERN, Geneva, 2020. CYRM-2020-0010. <https://cds.cern.ch/record/2749422> (last accessed: 05.12.2023). <http://dx.doi.org/10.23731/CYRM-2020-0010>.
- [137] O. Brüning et al., “The scientific potential and technological challenges of the High-Luminosity Large Hadron Collider program”, *Rept. Prog. Phys.* **85** (2022), no. 4, 046201, <http://dx.doi.org/10.1088/1361-6633/ac5106>.
- [138] CMS Collaboration, “CMS Physics: Technical Design Report Volume 1: Detector Performance and Software”, *CERN Document Server* (2006). CERN-LHCC-2006-001, CMS-TDR-8-1, CERN-LHCC-2006-001, CMS-TDR-8-1. <http://cds.cern.ch/record/922757> (last accessed: 01.11.2023).
- [139] CMS Collaboration, “Development of the CMS detector for the CERN LHC Run 3”, [arXiv:2309.05466](https://arxiv.org/abs/2309.05466).
- [140] T. Sakuma and T. McCauley, “Detector and Event Visualization with SketchUp at the CMS Experiment”, *J. Phys. Conf. Ser.* **513** (2014) 022032, <http://dx.doi.org/10.1088/1742-6596/513/2/022032>, [arXiv:1311.4942](https://arxiv.org/abs/1311.4942). Cutaway diagrams of CMS detector. <https://cds.cern.ch/record/2665537> (last accessed: 02.11.2023).
- [141] CMS Collaboration, “The CMS Phase-1 Pixel Detector Upgrade”, *JINST* **16**

- (2021), no. 02, P02027, <http://dx.doi.org/10.1088/1748-0221/16/02/P02027>, [arXiv:2012.14304](https://arxiv.org/abs/2012.14304).
- [142] CMS Collaboration, “Description and performance of track and primary-vertex reconstruction with the CMS tracker”, *Journal of Instrumentation* **9** (Oct, 2014) P10009–P10009, <http://dx.doi.org/10.1088/1748-0221/9/10/p10009>, [arXiv:1405.6569](https://arxiv.org/abs/1405.6569).
- [143] CMS Collaboration, “CMS Technical Design Report for the Pixel Detector Upgrade”, technical report, Sep, 2012. CERN-LHCC-2012-016, CMS-TDR-11. <https://cds.cern.ch/record/1481838> (last accessed: 05.12.2023).
- [144] CMS Tracker Collaboration, “The CMS Tracker Phase-2 Upgrade for the HL-LHC Era”, *Springer Proc. Phys.* **213** (2018) 344–348, http://dx.doi.org/10.1007/978-981-13-1316-5_64.
- [145] CMS Collaboration, “The CMS electromagnetic calorimeter project: Technical Design Report”, cms technical design report, Geneva, 1997. CERN-LHCC-97-033, CMS-TDR-4, CERN-LHCC-97-033, CMS-TDR-4. <http://cds.cern.ch/record/349375> (last accessed: 29.11.2023).
- [146] CMS Collaboration, “Energy Calibration and Resolution of the CMS Electromagnetic Calorimeter in pp Collisions at $\sqrt{s} = 7$ TeV”, *JINST* **8** (2013) P09009, <http://dx.doi.org/10.1088/1748-0221/8/09/P09009>, [arXiv:1306.2016](https://arxiv.org/abs/1306.2016).
- [147] CMS Collaboration, “Particle-flow reconstruction and global event description with the CMS detector”, *Journal of Instrumentation* **12** (Oct, 2017) P10003–P10003, <http://dx.doi.org/10.1088/1748-0221/12/10/p10003>, [arXiv:1706.04965](https://arxiv.org/abs/1706.04965).
- [148] J. Mans et al., “CMS Technical Design Report for the Phase 1 Upgrade of the Hadron Calorimeter”, cms technical design report, Geneva, 2012. CERN-LHCC-2012-015, CMS-TDR-10. <https://cds.cern.ch/record/1481837> (last accessed: 29.11.2023).
- [149] CMS Collaboration, “Calibration of the CMS hadron calorimeters using proton-proton collision data at $\sqrt{s} = 13$ TeV”, *JINST* **15** (2020), no. 05, P05002, <http://dx.doi.org/10.1088/1748-0221/15/05/P05002>, [arXiv:1910.00079](https://arxiv.org/abs/1910.00079).
- [150] CMS Collaboration, “Performance of the CMS Hadron Calorimeter with Cosmic Ray Muons and LHC Beam Data”, *JINST* **5** (2010) T03012, <http://dx.doi.org/10.1088/1748-0221/5/03/T03012>, [arXiv:0911.4991](https://arxiv.org/abs/0911.4991).
- [151] CMS Collaboration, “HCAL Energy Reconstruction Performance”, CMS Detector Performance Summary CMS-DP-2016-071, CERN, Geneva, 2016. <https://cds.cern.ch/record/2235509> (last accessed: 04.11.2023).
- [152] CMS Collaboration, “Performance of the CMS muon detector and muon reconstruction with proton-proton collisions at $\sqrt{s} = 13$ TeV”, *JINST* **13** (2018), no. 06, P06015, <http://dx.doi.org/10.1088/1748-0221/13/06/P06015>, [arXiv:1804.04528](https://arxiv.org/abs/1804.04528).
- [153] CMS Collaboration, “CMS TriDAS project: Technical Design Report, Volume 1: The Trigger Systems”, CMS Technical Design Report CERN-LHCC-2000-038. CMS-TDR-6-1, CERN, Geneva, 2000. <https://cds.cern.ch/record/706847> (last accessed: 05.12.2023).
- [154] CMS Collaboration, “The CMS trigger system”, *JINST* **12** (2017), no. 01, P01020, <http://dx.doi.org/10.1088/1748-0221/12/01/P01020>, [arXiv:1609.02366](https://arxiv.org/abs/1609.02366).
- [155] CMS Collaboration, “CMS: The TriDAS project. Technical design report, Vol. 2: Data acquisition and high-level trigger”, CMS Technical Design Report CERN-LHCC-2002-

026. CMS-TDR-6, CERN, Geneva, 12, 2002. <https://cds.cern.ch/record/578006> (last accessed: 05.12.2023).
- [156] CMS Offline Software and Computing, “CMS Phase-2 Computing Model: Update Document”, technical report, CERN, Geneva, 2022. <https://cds.cern.ch/record/2815292> (last accessed: 24.11.2023).
- [157] S. Agostinelli et al., “GEANT4—a simulation toolkit”, *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **506** (2003), no. 3, 250–303, [http://dx.doi.org/10.1016/S0168-9002\(03\)01368-8](http://dx.doi.org/10.1016/S0168-9002(03)01368-8).
- [158] V. Ivanchenko et al., “CMS Full Simulation for Run 3”, *EPJ Web of Conferences* **251** (01, 2021) 03016, <http://dx.doi.org/10.1051/epjconf/202125103016>.
- [159] DELPHES 3 Collaboration, “DELPHES 3, A modular framework for fast simulation of a generic collider experiment”, *JHEP* **02** (2014) 057, [http://dx.doi.org/10.1007/JHEP02\(2014\)057](http://dx.doi.org/10.1007/JHEP02(2014)057), [arXiv:1307.6346](https://arxiv.org/abs/1307.6346).
- [160] cms-sw / CMS Collaboration, “CMSSW”, *GitHub repository* (2023). <https://github.com/cms-sw/cmssw> (last accessed: 06.11.2023).
- [161] R. Fruhwirth, “Application of Kalman filtering to track and vertex fitting”, *Nucl. Instrum. Meth. A* **262** (1987) 444–450, [http://dx.doi.org/10.1016/0168-9002\(87\)90887-4](http://dx.doi.org/10.1016/0168-9002(87)90887-4).
- [162] P. Billoir and S. Qian, “Simultaneous pattern recognition and track fitting by the Kalman filtering method”, *Nucl. Instrum. Meth. A* **294** (1990) 219–228, [http://dx.doi.org/10.1016/0168-9002\(90\)91835-Y](http://dx.doi.org/10.1016/0168-9002(90)91835-Y).
- [163] CMS Collaboration, “CMS Tracking Performance Results from Early LHC Operation”, *Eur. Phys. J. C* **70** (2010) 1165–1192, <http://dx.doi.org/10.1140/epjc/s10052-010-1491-3>, [arXiv:1007.1988](https://arxiv.org/abs/1007.1988).
- [164] CMS Collaboration, “Performance of Run 3 track reconstruction with the mkFit algorithm”, CMS Detector Performance Summary CMS-DP-2022-018, CERN, Geneva, 2022. <http://cds.cern.ch/record/2814000> (last accessed: 06.11.2023).
- [165] K. Rose, “Deterministic annealing for clustering, compression, classification, regression, and related optimization problems”, *Proceedings of the IEEE* **86** (1998), no. 11, 2210–2239, <http://dx.doi.org/10.1109/5.726788>.
- [166] R. Fruhwirth, W. Waltenberger, and P. Vanlaer, “Adaptive vertex fitting”, *J. Phys. G* **34** (2007) N343, <http://dx.doi.org/10.1088/0954-3899/34/12/N01>.
- [167] CMS Collaboration, “Muon Identification and Isolation efficiency on full 2016 dataset”, CMS Detector Performance Summary CMS-DP-2017-007, CERN, Geneva, 2017. <https://cds.cern.ch/record/2257968> (last accessed: 04.11.2023).
- [168] CMS Collaboration, “Performance of electron reconstruction and selection with the CMS detector in proton-proton collisions at $\sqrt{s} = 8$ TeV”, *JINST* **10** (2015) P06005, <http://dx.doi.org/10.1088/1748-0221/10/06/P06005>, [arXiv:1502.02701](https://arxiv.org/abs/1502.02701).
- [169] CMS Collaboration, “Electron and photon reconstruction and identification with the CMS experiment at the CERN LHC”, *JINST* **16** (2021), no. 05, P05014, <http://dx.doi.org/10.1088/1748-0221/16/05/P05014>, [arXiv:2012.06888](https://arxiv.org/abs/2012.06888).
- [170] W. Adam, R. Fruhwirth, A. Strandlie, and T. Todorov, “Reconstruction of electrons with

- the Gaussian sum filter in the CMS tracker at LHC”, *eConf* **C0303241** (2003) TULT009, <http://dx.doi.org/10.1088/0954-3899/31/9/N01>, [arXiv:physics/0306087](https://arxiv.org/abs/hep-ph/0306087).
- [171] T. Chen and C. Guestrin, “Xgboost: A scalable tree boosting system”, in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD ’16, p. 785–794. Association for Computing Machinery, New York, NY, USA, 2016. <http://dx.doi.org/10.1145/2939672.2939785>.
- [172] CMS Collaboration, “Performance of photon reconstruction and identification with the CMS detector in proton-proton collisions at $\sqrt{s}=8$ TeV”, *JINST* **10** (2015) P08010, <http://dx.doi.org/10.1088/1748-0221/10/08/P08010>, [arXiv:1502.02702](https://arxiv.org/abs/1502.02702).
- [173] CMS Collaboration, “Reconstruction and identification of τ lepton decays to hadrons and ν_τ at CMS”, *JINST* **11** (2016), no. 01, P01019, <http://dx.doi.org/10.1088/1748-0221/11/01/P01019>, [arXiv:1510.07488](https://arxiv.org/abs/1510.07488).
- [174] CMS Collaboration, “Performance of reconstruction and identification of τ leptons decaying to hadrons and ν_τ in pp collisions at $\sqrt{s} = 13$ TeV”, *JINST* **13** (2018), no. 10, P10005, <http://dx.doi.org/10.1088/1748-0221/13/10/P10005>, [arXiv:1809.02816](https://arxiv.org/abs/1809.02816).
- [175] CMS Collaboration, “Identification of hadronic tau lepton decays using a deep neural network”, *JINST* **17** (2022) P07023, <http://dx.doi.org/10.1088/1748-0221/17/07/P07023>, [arXiv:2201.08458](https://arxiv.org/abs/2201.08458).
- [176] B. Edwards et al., “Analysis of nuclear interactions of energies between 1000 and 100 000 bev”, *The Philosophical Magazine: A Journal of Theoretical Experimental and Applied Physics* **3** (1958), no. 27, 237–266, <http://dx.doi.org/10.1080/14786435808238218>.
- [177] G. Sterman and S. Weinberg, “Jets from quantum chromodynamics”, *Phys. Rev. Lett.* **39** (Dec, 1977) 1436–1439, <http://dx.doi.org/10.1103/PhysRevLett.39.1436>.
- [178] R. D. Field and R. P. Feynman, “A Parametrization of the Properties of Quark Jets”, *Nucl. Phys. B* **136** (1978) 1, [http://dx.doi.org/10.1016/0550-3213\(78\)90015-9](http://dx.doi.org/10.1016/0550-3213(78)90015-9).
- [179] G. C. Fox and S. Wolfram, “Observables for the Analysis of Event Shapes in e^+e^- Annihilation and Other Processes”, *Phys. Rev. Lett.* **41** (1978) 1581, <http://dx.doi.org/10.1103/PhysRevLett.41.1581>.
- [180] CMS Collaboration, “Jet energy scale and resolution in the CMS experiment in pp collisions at 8 TeV”, *Journal of Instrumentation* **12** (Feb, 2017) P02014–P02014, <http://dx.doi.org/10.1088/1748-0221/12/02/p02014>, [arXiv:1607.03663](https://arxiv.org/abs/1607.03663).
- [181] CDF Collaboration, “The Topology of three jet events in $\bar{p}p$ collisions at $\sqrt{s} = 1.8$ TeV”, *Phys. Rev. D* **45** (1992) 1448–1458, <http://dx.doi.org/10.1103/PhysRevD.45.1448>.
- [182] G. C. Blazey et al., “Run II jet physics”, in *Physics at Run II: QCD and Weak Boson Physics Workshop: Final General Meeting*, pp. 47–77. 5, 2000. [arXiv:hep-ex/0005012](https://arxiv.org/abs/hep-ex/0005012).
- [183] S. D. Ellis, J. Huston, and M. Tonnesmann, “On building better cone jet algorithms”, *eConf* **C010630** (2001) 513, [arXiv:hep-ph/0111434](https://arxiv.org/abs/hep-ph/0111434).
- [184] G. P. Salam and G. Soyez, “A Practical Seedless Infrared-Safe Cone jet algorithm”, *JHEP* **05** (2007) 086, <http://dx.doi.org/10.1088/1126-6708/2007/05/086>, [arXiv:0704.0292](https://arxiv.org/abs/0704.0292).
- [185] G. Soyez, “The SIScone and anti-k(t) jet algorithms”, in *16th International Workshop on Deep Inelastic Scattering and Related Subjects*, p. 178. 7, 2008. [arXiv:0807.0021](https://arxiv.org/abs/0807.0021). <http://dx.doi.org/10.1088/1748-0221/7/07/P07178>.

- [//dx.doi.org/10.3360/dis.2008.178](https://dx.doi.org/10.3360/dis.2008.178).
- [186] I. W. Stewart et al., “XCone: N-jettiness as an Exclusive Cone Jet Algorithm”, *JHEP* **11** (2015) 072, [http://dx.doi.org/10.1007/JHEP11\(2015\)072](http://dx.doi.org/10.1007/JHEP11(2015)072), [arXiv:1508.01516](https://arxiv.org/abs/1508.01516).
- [187] M. Cacciari and G. P. Salam, “Dispelling the N^3 myth for the k_t jet-finder”, *Phys. Lett. B* **641** (2006) 57–61, <http://dx.doi.org/10.1016/j.physletb.2006.08.037>, [arXiv:hep-ph/0512210](https://arxiv.org/abs/hep-ph/0512210).
- [188] M. Cacciari, G. P. Salam, and G. Soyez, “The anti- k_t jet clustering algorithm”, *JHEP* **04** (2008) 063, <http://dx.doi.org/10.1088/1126-6708/2008/04/063>, [arXiv:0802.1189](https://arxiv.org/abs/0802.1189).
- [189] M. Cacciari, G. P. Salam, and G. Soyez, “FastJet User Manual”, *Eur. Phys. J. C* **72** (2012) 1896, <http://dx.doi.org/10.1140/epjc/s10052-012-1896-2>, [arXiv:1111.6097](https://arxiv.org/abs/1111.6097).
- [190] JADE Collaboration, “Experimental Study of Jets in electron - Positron Annihilation”, *Phys. Lett. B* **101** (1981) 129–134, [http://dx.doi.org/10.1016/0370-2693\(81\)90505-0](http://dx.doi.org/10.1016/0370-2693(81)90505-0).
- [191] JADE Collaboration, “Particle Distribution in Three Jet Events Produced by e^+e^- Annihilation”, *Z. Phys. C* **21** (1983) 37, <http://dx.doi.org/10.1007/BF01648774>.
- [192] S. D. Ellis and D. E. Soper, “Successive combination jet algorithm for hadron collisions”, *Phys. Rev. D* **48** (1993) 3160–3166, <http://dx.doi.org/10.1103/PhysRevD.48.3160>, [arXiv:hep-ph/9305266](https://arxiv.org/abs/hep-ph/9305266).
- [193] Y. L. Dokshitzer, G. D. Leder, S. Moretti, and B. R. Webber, “Better jet clustering algorithms”, *JHEP* **08** (1997) 001, <http://dx.doi.org/10.1088/1126-6708/1997/08/001>, [arXiv:hep-ph/9707323](https://arxiv.org/abs/hep-ph/9707323).
- [194] M. Wobisch and T. Wengler, “Hadronization corrections to jet cross-sections in deep inelastic scattering”, in *Workshop on Monte Carlo Generators for HERA Physics (Plenary Starting Meeting)*, pp. 270–279. 4, 1998. [arXiv:hep-ph/9907280](https://arxiv.org/abs/hep-ph/9907280).
- [195] CMS Collaboration, “A further reduction in CMS event data for analysis: the NANO AOD format”, *EPJ Web Conf.* **214** (2019) 06021, <http://dx.doi.org/10.1051/epjconf/201921406021>.
- [196] cms-jet / CMS Collaboration, “PFNano”, *GitHub repository* (2023). <https://github.com/cms-jet/PFNano> (last accessed: 03.11.2023).
- [197] T. Plehn, G. P. Salam, and M. Spannowsky, “Fat Jets for a Light Higgs”, *Phys. Rev. Lett.* **104** (2010) 111801, <http://dx.doi.org/10.1103/PhysRevLett.104.111801>, [arXiv:0910.5472](https://arxiv.org/abs/0910.5472).
- [198] CMS Collaboration, “A search for the standard model Higgs boson decaying to charm quarks”, *JHEP* **03** (2020) 131, [http://dx.doi.org/10.1007/JHEP03\(2020\)131](http://dx.doi.org/10.1007/JHEP03(2020)131), [arXiv:1912.01662](https://arxiv.org/abs/1912.01662).
- [199] CMS Collaboration, “Search for Higgs Boson Decay to a Charm Quark-Antiquark Pair in Proton-Proton Collisions at $\sqrt{s}=13$ TeV”, *Phys. Rev. Lett.* **131** (2023), no. 6, 061801, <http://dx.doi.org/10.1103/PhysRevLett.131.061801>, [arXiv:2205.05550](https://arxiv.org/abs/2205.05550).
- [200] CMS Collaboration, “Search for Higgs Boson and Observation of Z Boson through their Decay into a Charm Quark-Antiquark Pair in Boosted Topologies in Proton-Proton Collisions at $\sqrt{s}=13$ TeV”, *Phys. Rev. Lett.* **131** (2023), no. 4, 041801, <http://dx.doi.org/10.1103/PhysRevLett.131.041801>, [arXiv:2211.14181](https://arxiv.org/abs/2211.14181).

- [201] CMS Collaboration, “Performance of heavy-flavour jet identification in boosted topologies in proton-proton collisions at $\sqrt{s} = 13$ TeV”, CMS Physics Analysis Summary, 2023. <https://cds.cern.ch/record/2866276> (last accessed: 05.12.2023).
- [202] CMS Collaboration, “Pileup Removal Algorithms”, CMS Physics Analysis Summary, 2014. <http://cds.cern.ch/record/1751454> (last accessed: 04.11.2023).
- [203] CMS Collaboration, “Pileup mitigation at CMS in 13 TeV data”, *Journal of Instrumentation* **15** (Sep, 2020) P09018–P09018, <http://dx.doi.org/10.1088/1748-0221/15/09/p09018>, [arXiv:2003.00503](https://arxiv.org/abs/2003.00503).
- [204] CMS Collaboration, “Performance of the pile up jet identification in CMS for Run 2”, CMS Detector Performance Summary CMS-DP-2020-020, CERN, Geneva, 2020. <http://cds.cern.ch/record/2715906> (last accessed: 04.11.2023).
- [205] D. Bertolini, P. Harris, M. Low, and N. Tran, “Pileup Per Particle Identification”, *JHEP* **10** (2014) 059, [http://dx.doi.org/10.1007/JHEP10\(2014\)059](http://dx.doi.org/10.1007/JHEP10(2014)059), [arXiv:1407.6013](https://arxiv.org/abs/1407.6013).
- [206] CMS Collaboration, “Pileup-per-particle identification: optimisation for Run 2 Legacy and beyond”, CMS Detector Performance Summary CMS-DP-2021-001, CERN, Geneva, 2021. <http://cds.cern.ch/record/2751563> (last accessed: 04.11.2023).
- [207] CMS Collaboration, “Jet algorithms performance in 13 TeV data”, CMS Physics Analysis Summary, CERN, Geneva, 2017. <https://cds.cern.ch/record/2256875> (last accessed: 05.12.2023).
- [208] CMS Collaboration, “Determination of Jet Energy Calibration and Transverse Momentum Resolution in CMS”, *JINST* **6** (2011) P11002, <http://dx.doi.org/10.1088/1748-0221/6/11/P11002>, [arXiv:1107.4277](https://arxiv.org/abs/1107.4277).
- [209] CMS Collaboration, “Jet energy scale and resolution performance with 13 TeV data collected by CMS in 2016-2018”, CMS Detector Performance Summary CMS-DP-2020-019, CERN, Geneva, 2020. <http://cds.cern.ch/record/2715872> (last accessed: 04.11.2023).
- [210] CMS Collaboration, “Jet energy scale and resolution measurement with Run 2 Legacy Data Collected by CMS at 13 TeV”, CMS Detector Performance Summary CMS-DP-2021-033, CERN, Geneva, 2021. <http://cds.cern.ch/record/2792322> (last accessed: 04.11.2023).
- [211] M. Cacciari and G. P. Salam, “Pileup subtraction using jet areas”, *Phys. Lett. B* **659** (2008) 119–126, <http://dx.doi.org/10.1016/j.physletb.2007.09.077>, [arXiv:0707.1378](https://arxiv.org/abs/0707.1378).
- [212] M. Cacciari, G. P. Salam, and G. Soyez, “The Catchment Area of Jets”, *JHEP* **04** (2008) 005, <http://dx.doi.org/10.1088/1126-6708/2008/04/005>, [arXiv:0802.1188](https://arxiv.org/abs/0802.1188).
- [213] CMS Collaboration, “Jet Energy Scale and Resolution Measurements Using Prompt Run3 Data Collected by CMS in the First Months of 2022 at 13.6 TeV”, CMS Detector Performance Summary CMS-DP-2022-054, CERN, Geneva, 2022. <http://cds.cern.ch/record/2841534> (last accessed: 04.11.2023).
- [214] CMS Collaboration, “Missing transverse energy performance of the CMS detector”, *JINST* **6** (2011) P09001, <http://dx.doi.org/10.1088/1748-0221/6/09/P09001>, [arXiv:1106.5048](https://arxiv.org/abs/1106.5048).
- [215] CMS Collaboration, “Performance of the CMS missing transverse momentum reconstruction in pp data at $\sqrt{s} = 8$ TeV”, *JINST* **10** (2015), no. 02, P02006, <http://dx.doi.org/10.1088/1748-0221/10/02/P02006>.

- [org/10.1088/1748-0221/10/02/P02006](https://arxiv.org/10.1088/1748-0221/10/02/P02006), [arXiv:1411.0511](https://arxiv.org/abs/1411.0511).
- [216] CMS Collaboration, “Performance of missing transverse momentum reconstruction in proton-proton collisions at $\sqrt{s} = 13$ TeV using the CMS detector”, *JINST* **14** (2019), no. 07, P07004, <http://dx.doi.org/10.1088/1748-0221/14/07/P07004>, [arXiv:1903.06078](https://arxiv.org/abs/1903.06078).
- [217] R. Kogler et al., “Jet Substructure at the Large Hadron Collider: Experimental Review”, *Rev. Mod. Phys.* **91** (2019), no. 4, 045003, <http://dx.doi.org/10.1103/RevModPhys.91.045003>, [arXiv:1803.06991](https://arxiv.org/abs/1803.06991).
- [218] ATLAS Collaboration, “ATLAS flavour-tagging algorithms for the LHC Run 2 pp collision dataset”, *arXiv e-prints* (11, 2022) [arXiv:2211.16345](https://arxiv.org/abs/2211.16345).
- [219] D0 Collaboration, “Evidence for production of single top quarks”, *Phys. Rev. D* **78** (2008) 012005, <http://dx.doi.org/10.1103/PhysRevD.78.012005>, [arXiv:0803.0739](https://arxiv.org/abs/0803.0739).
- [220] A. Rizzi, F. Palla, and G. Segneri, “Track impact parameter based b-tagging with CMS”, CMS Note CMS-NOTE-2006-019, CERN, Geneva, 2006. <https://cds.cern.ch/record/927385> (last accessed: 06.11.2023).
- [221] JADE Collaboration, “Experimental Evidence for Differences in p_T Between Quark Jets and Gluon Jets”, *Phys. Lett. B* **123** (1983) 460–466, [http://dx.doi.org/10.1016/0370-2693\(83\)90994-2](http://dx.doi.org/10.1016/0370-2693(83)90994-2).
- [222] CMS Collaboration, “Performance of quark/gluon discrimination in 8 TeV pp data”, CMS Physics Analysis Summary CMS-PAS-JME-13-002, CERN, Geneva, 2013. <https://cds.cern.ch/record/1599732> (last accessed: 05.12.2023).
- [223] A. Butter, B. M. Dillon, T. Plehn, and L. Vogel, “Performance versus Resilience in Modern Quark-Gluon Tagging”, *arXiv e-prints* (12, 2022) [arXiv:2212.10493](https://arxiv.org/abs/2212.10493).
- [224] ATLAS Collaboration, “Constituent-Based Quark Gluon Tagging using Transformers with the ATLAS detector”, ATLAS PUB Note ATL-PHYS-PUB-2023-032, CERN, Geneva, 2023. <https://cds.cern.ch/record/2878932>. All figures including auxiliary figures are available at <https://atlas.web.cern.ch/Atlas/GROUPS/PHYSICS/PUBNOTES/ATL-PHYS-PUB-2023-032> (last accessed: 05.12.2023).
- [225] CMS Collaboration, “Jet Flavour Identification (MC Truth)”, 2016. <https://twiki.cern.ch/twiki/bin/view/CMSPublic/SWGuideBTagMCTools> (last accessed: 03.11.2023).
- [226] CMS Collaboration, “Identification of b-Quark Jets with the CMS Experiment”, *JINST* **8** (2013) P04013, <http://dx.doi.org/10.1088/1748-0221/8/04/P04013>, [arXiv:1211.4462](https://arxiv.org/abs/1211.4462).
- [227] ATLAS Collaboration, “Impact parameter-based b-tagging algorithms in the 7 TeV collision data with the ATLAS detector: the TrackCounting and JetProb algorithms”, ATLAS CONF Note ATLAS-CONF-2010-041, CERN, Geneva, 2010. <https://cds.cern.ch/record/1277681> (last accessed: 05.12.2023).
- [228] CMS Collaboration, “First Measurement of the Cross Section for Top-Quark Pair Production in Proton-Proton Collisions at $\sqrt{s} = 7$ TeV”, *Phys. Lett. B* **695** (2011) 424–443, <http://dx.doi.org/10.1016/j.physletb.2010.11.058>, [arXiv:1010.5994](https://arxiv.org/abs/1010.5994).
- [229] ATLAS Collaboration, “Measurement of the top quark pair production cross-section with ATLAS in pp collisions at $\sqrt{s} = 7$ TeV in dilepton final states”, ATLAS CONF

- Note ATLAS-CONF-2011-034, CERN, Geneva, 3, 2011. <https://cds.cern.ch/record/1337784> (last accessed: 05.12.2023).
- [230] CMS Collaboration, “Measurement of differential cross sections for top quark pair production using the lepton+jets final state in proton-proton collisions at 13 TeV”, *Phys. Rev. D* **95** (2017), no. 9, 092001, <http://dx.doi.org/10.1103/PhysRevD.95.092001>, [arXiv:1610.04191](https://arxiv.org/abs/1610.04191).
- [231] CMS Collaboration, “Observation of Higgs boson decay to bottom quarks”, *Phys. Rev. Lett.* **121** (2018), no. 12, 121801, <http://dx.doi.org/10.1103/PhysRevLett.121.121801>, [arXiv:1808.08242](https://arxiv.org/abs/1808.08242).
- [232] CMS Collaboration, “Search for supersymmetry in final states with disappearing tracks in proton-proton collisions at $\sqrt{s} = 13$ TeV”, [arXiv:2309.16823](https://arxiv.org/abs/2309.16823).
- [233] A. Stein, “Investigation of robustness of b-Tagging algorithms for the CMS Experiment”, Master’s thesis, RWTH Aachen U., 2021. <http://cds.cern.ch/record/2862786>. Presented 22 Oct 2021.
- [234] CMS Collaboration, “Search for nonresonant Higgs boson pair production in final states with two bottom quarks and two photons in proton-proton collisions at $\sqrt{s} = 13$ TeV”, *JHEP* **03** (2021) 257, [http://dx.doi.org/10.1007/JHEP03\(2021\)257](http://dx.doi.org/10.1007/JHEP03(2021)257), [arXiv:2011.12373](https://arxiv.org/abs/2011.12373).
- [235] CMS Collaboration, “Observation of four top quark production in proton-proton collisions at $\sqrt{s} = 13$ TeV”, [arXiv:2305.13439](https://arxiv.org/abs/2305.13439).
- [236] CMS Collaboration, “Search for W' bosons decaying to a top and a bottom quark at $s=13\text{TeV}$ in the hadronic final state”, *Phys. Lett. B* **820** (2021) 136535, <http://dx.doi.org/10.1016/j.physletb.2021.136535>, [arXiv:2104.04831](https://arxiv.org/abs/2104.04831).
- [237] CMS Collaboration, “Search for top squark pair production in a final state with at least one hadronically decaying tau lepton in proton-proton collisions at $\sqrt{s} = 13$ TeV”, *JHEP* **07** (2023) 110, [http://dx.doi.org/10.1007/JHEP07\(2023\)110](http://dx.doi.org/10.1007/JHEP07(2023)110), [arXiv:2304.07174](https://arxiv.org/abs/2304.07174).
- [238] CMS Collaboration, “Measurement of the Higgs boson production via vector boson fusion and its decay into bottom quarks in proton-proton collisions at $\sqrt{s} = 13$ TeV”, [arXiv:2308.01253](https://arxiv.org/abs/2308.01253).
- [239] CMS Collaboration, “Performance summary of AK4 jet b tagging with data from proton-proton collisions at 13 TeV with the CMS detector”, CMS Detector Performance Summary CMS-DP-2023-005, CERN, Geneva, 2023. <http://cds.cern.ch/record/2854609> (last accessed: 08.11.2023).
- [240] CMS Collaboration, “Performance summary of AK4 jet charm tagging with the CMS Run2 Legacy dataset”, CMS Detector Performance Summary CMS-DP-2023-006, CERN, Geneva, 2023. <http://cds.cern.ch/record/2854610> (last accessed: 08.11.2023).
- [241] H. Qu and L. Gouskos, “ParticleNet: Jet Tagging via Particle Clouds”, *Phys. Rev. D* **101** (2020), no. 5, 056019, <http://dx.doi.org/10.1103/PhysRevD.101.056019>, [arXiv:1902.08570](https://arxiv.org/abs/1902.08570).
- [242] ATLAS Collaboration, “Identification of Jets Containing b -Hadrons with Recurrent Neural Networks at the ATLAS Experiment”, ATLAS PUB Note ATL-PHYS-PUB-2017-003, CERN, Geneva, 2017. <http://cds.cern.ch/record/2255226> (last accessed: 05.12.2023).

- [243] ATLAS Collaboration, “Deep Sets based Neural Networks for Impact Parameter Flavour Tagging in ATLAS”, ATLAS PUB Note ATL-PHYS-PUB-2020-014, CERN, Geneva, 2020. <http://cds.cern.ch/record/2718948> (last accessed: 05.12.2023).
- [244] ATLAS Collaboration, “Graph Neural Network Jet Flavour Tagging with the ATLAS Detector”, ATLAS PUB Note ATL-PHYS-PUB-2022-027, CERN, Geneva, 2022. <http://cds.cern.ch/record/2811135> (last accessed: 05.12.2023).
- [245] ATLAS Collaboration, “New ATLAS b -tagging algorithm for Run 3”, *PoS ICHEP2022* (2022) 1090, <http://dx.doi.org/10.22323/1.414.1090>.
- [246] ATLAS Collaboration, “Monte Carlo to Monte Carlo scale factors for flavour tagging efficiency calibration”, ATLAS PUB Note ATL-PHYS-PUB-2020-009, CERN, Geneva, 2020. <https://cds.cern.ch/record/2718610> (last accessed: 06.11.2023).
- [247] I. Goodfellow, Y. Bengio, and A. Courville, “Deep Learning”. MIT Press, 2016. <http://www.deeplearningbook.org> (last accessed: 03.11.2023).
- [248] E. Stevens, L. Antiga, and T. Viehmann, “Deep learning with PyTorch”. Manning Publications Company, 2020.
- [249] L. Wan et al., “Regularization of neural networks using dropconnect”, in *International conference on machine learning*, pp. 1058–1066, PMLR. 2013. <https://proceedings.mlr.press/v28/wan13.html> (last accessed: 04.12.2023).
- [250] N. Srivastava et al., “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”, *Journal of Machine Learning Research* **15** (2014), no. 1, 1929–1958. <https://jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf> (last accessed: 02.12.2023).
- [251] K. Fukushima, “Visual feature extraction by a multilayered network of analog threshold elements”, *IEEE Transactions on Systems Science and Cybernetics* **5** (1969), no. 4, 322–333, <http://dx.doi.org/10.1109/TSSC.1969.300225>.
- [252] X. Glorot, A. Bordes, and Y. Bengio, “Deep sparse rectifier neural networks”, in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, G. Gordon, D. Dunson, and M. Dudík, eds., volume 15 of *Proceedings of Machine Learning Research*, pp. 315–323. PMLR, Fort Lauderdale, FL, USA, 11–13 Apr, 2011. <http://proceedings.mlr.press/v15/glorot11a/glorot11a.pdf> (last accessed: 21.11.2023).
- [253] D. Hendrycks and K. Gimpel, “Gaussian Error Linear Units (GELUs)”, *CoRR* (2016) [arXiv:1606.08415](https://arxiv.org/abs/1606.08415).
- [254] J. S. Bridle, “Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters”, in *Advances in Neural Information Processing Systems 2, [NIPS Conference, Denver, Colorado, USA, November 27-30, 1989]*, D. S. Touretzky, ed., pp. 211–217. Morgan Kaufmann, 1989. https://proceedings.neurips.cc/paper_files/paper/1989/file/0336dcbab05b9d5ad24f4333c7658a0e-Paper.pdf (last accessed 05.12.2023).
- [255] A. Paszke et al., “PyTorch: An Imperative Style, High-Performance Deep Learning Library”, in *Advances in Neural Information Processing Systems 32.*, pp. 8024–8035. Curran Associates, Inc., 2019. [arXiv:1912.01703](https://arxiv.org/abs/1912.01703). NeurIPS 2019.
- [256] K. Hornik, M. Stinchcombe, and H. White, “Multilayer feedforward networks are uni-

- versal approximators", *Neural Networks* **2** (1989), no. 5, 359–366, [http://dx.doi.org/10.1016/0893-6080\(89\)90020-8](http://dx.doi.org/10.1016/0893-6080(89)90020-8).
- [257] G. V. Cybenko, "Approximation by superpositions of a sigmoidal function", *Mathematics of Control, Signals and Systems* **2** (1989) 303–314, <http://dx.doi.org/10.1007/BF02551274>.
- [258] Z. Zhang and M. R. Sabuncu, "Generalized Cross Entropy Loss for Training Deep Neural Networks with Noisy Labels", in *32nd Conference on Neural Information Processing Systems (NeurIPS)*. 2018. [arXiv:1805.07836](https://arxiv.org/abs/1805.07836).
- [259] PyTorch, "CrossEntropyLoss", 2023. <https://pytorch.org/docs/stable/generated/torch.nn.CrossEntropyLoss.html> (last accessed: 05.12.2023).
- [260] T.-Y. Lin et al., "Focal Loss for Dense Object Detection", in *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2999–3007. 2017. [arXiv:1708.02002](https://arxiv.org/abs/1708.02002). <http://dx.doi.org/10.1109/ICCV.2017.324>.
- [261] A. Hassan, "Adeelh/pytorch-multi-class-focal-loss: 1.1", *Zenodo* (October, 2021) <http://dx.doi.org/10.5281/zenodo.5547584>.
- [262] CMS Collaboration, "Search for new physics using effective field theory in 13 TeV pp collision events that contain a top quark pair and a boosted Z or Higgs boson", *Phys. Rev. D* **108** (2023) 032008, <http://dx.doi.org/10.1103/PhysRevD.108.032008>, [arXiv:2208.12837](https://arxiv.org/abs/2208.12837).
- [263] hcarlens / H. Carlens, "pytorch-tabular", *GitHub repository* (2020). <https://github.com/hcarlens/pytorch-tabular> (last accessed: 04.12.2023).
- [264] H. Carlens, "Better Data Loading: 20x PyTorch Speed-Up for Tabular Data", 2020. <https://towardsdatascience.com/better-data-loading-20x-pytorch-speed-up-for-tabular-data-e264b9e34352> (last accessed: 04.12.2023).
- [265] ufoym / M. Yang, "Imbalanced Dataset Sampler", *GitHub repository* (2022). <https://github.com/ufoym/imbalanced-dataset-sampler> (last accessed: 04.12.2023).
- [266] S. Ruder, "An overview of gradient descent optimization algorithms", *arXiv e-prints* (06, 2017) [arXiv:1609.04747](https://arxiv.org/abs/1609.04747).
- [267] H. Robbins and S. Monro, "A Stochastic Approximation Method", *The Annals of Mathematical Statistics* **22** (1951), no. 3, 400 – 407, <http://dx.doi.org/10.1214/aoms/1177729586>.
- [268] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization", in *3rd International Conference for Learning Representations (ICLR)*. 2015. [arXiv:1412.6980](https://arxiv.org/abs/1412.6980).
- [269] PyTorch, "Adam", 2023. <https://pytorch.org/docs/stable/generated/torch.optim.Adam.html#torch.optim.Adam> (last accessed: 30.11.2023).
- [270] C. Darken, J. Chang, and J. Moody, "Learning rate schedules for faster stochastic gradient search", in *Neural Networks for Signal Processing II Proceedings of the 1992 IEEE Workshop*, pp. 3–12. 1992. <http://dx.doi.org/10.1109/NNSP.1992.253713>.
- [271] PyTorch, "Dropout", 2023. <https://pytorch.org/docs/stable/generated/torch.nn.Dropout.html> (last accessed: 05.12.2023).
- [272] G. E. Hinton et al., "Improving neural networks by preventing co-adaptation of feature

- detectors”, *arXiv e-prints* (07, 2012) [arXiv:1207.0580](https://arxiv.org/abs/1207.0580).
- [273] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift”, [arXiv:1502.03167](https://arxiv.org/abs/1502.03167).
- [274] L. Prechelt, G. Montavon, G. B. Orr, and K.-R. Müller, “Early Stopping — But When?”, pp. 53–67. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. http://dx.doi.org/10.1007/978-3-642-35289-8_5.
- [275] M. Kuhn and K. Johnson, “Feature Engineering and Selection: A Practical Approach for Predictive Models”. Chapman & Hall/CRC Data Science Series. CRC Press, 2019. <http://www.feats.engineering/> (last accessed: 04.12.2023).
- [276] F. Pedregosa et al., “Scikit-learn: Machine learning in python”, *Journal of Machine Learning Research* **12** (01, 2012) [arXiv:1201.0490](https://arxiv.org/abs/1201.0490). <https://jmlr.org/papers/v12/pedregosa11a.html> (last accessed: 05.12.2023).
- [277] K. Fukushima, “Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position”, *Biological cybernetics* **36** (1980), no. 4, 193–202, <http://dx.doi.org/10.1007/BF00344251>.
- [278] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition”, *Proceedings of the IEEE* **86** (1998), no. 11, 2278–2324, <http://dx.doi.org/10.1109/5.726791>.
- [279] Y. Lecun, Y. Bengio, and G. Hinton, “Deep learning”, *Nat. Cell Biol.* **521** (2015), no. 7553, 436, <http://dx.doi.org/10.1038/nature14539>.
- [280] S. Hochreiter and J. Schmidhuber, “Long short-term memory”, *Neural Comput.* **9** (nov, 1997) 1735–1780, <http://dx.doi.org/10.1162/neco.1997.9.8.1735>.
- [281] P. Branco, L. Torgo, and R. P. Ribeiro, “A Survey of Predictive Modeling on Imbalanced Domains”, *ACM Comput. Surv.* **49** (August, 2016) <http://dx.doi.org/10.1145/2907070>, [arXiv:1505.01658](https://arxiv.org/abs/1505.01658).
- [282] M. Galar et al., “A Review on Ensembles for the Class Imbalance Problem: Bagging-, Boosting-, and Hybrid-Based Approaches”, *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* **42** (2012), no. 4, 463–484, <http://dx.doi.org/10.1109/TSMCC.2011.2161285>.
- [283] J. Davis and M. Goadrich, “The Relationship between Precision-Recall and ROC Curves”, in *Proceedings of the 23rd International Conference on Machine Learning, ICML ’06*, p. 233–240. Association for Computing Machinery, New York, NY, USA, 2006. <http://dx.doi.org/10.1145/1143844.1143874>.
- [284] D. Powers, “Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation”, *arXiv e-prints* (2020) [arXiv:2010.16061](https://arxiv.org/abs/2010.16061).
- [285] Fleshgrinder, “Gaussian distribution”, *Wikimedia Commons* (Jun, 2009). Released into the public domain. https://commons.wikimedia.org/wiki/File:Gaussian_distribution.svg (last accessed: 03.12.2023).
- [286] K. Albertsson et al., “Machine Learning in High Energy Physics Community White Paper”, *J. Phys. Conf. Ser.* **1085** (2018), no. 2, 022008, <http://dx.doi.org/10.1088/1742-6596/1085/2/022008>, [arXiv:1807.02876](https://arxiv.org/abs/1807.02876).
- [287] A. Adadi and M. Berrada, “Peeking Inside the Black-Box: A Survey on Explainable

- Artificial Intelligence (XAI)", *IEEE Access* **6** (2018) 52138–52160, <http://dx.doi.org/10.1109/ACCESS.2018.2870052>.
- [288] D. Amodei et al., "Concrete Problems in AI Safety", *arXiv e-prints* (2016) [arXiv:1606.06565](https://arxiv.org/abs/1606.06565).
- [289] J. Su, D. V. Vargas, and K. Sakurai, "One Pixel Attack for Fooling Deep Neural Networks", *IEEE Transactions on Evolutionary Computation* **23** (Oct, 2019) 828–841, <http://dx.doi.org/10.1109/tevc.2019.2890858>, [arXiv:1710.08864](https://arxiv.org/abs/1710.08864).
- [290] S. Kassem, "Interpretability of b-Tagging Neural Networks for the CMS Experiment", Master's thesis, RWTH Aachen U., 2022.
- [291] A. Chakraborty et al., "Adversarial Attacks and Defences: A Survey", *arXiv e-prints* (2018) [arXiv:1810.00069](https://arxiv.org/abs/1810.00069).
- [292] U. Shaham, Y. Yamada, and S. Negahban, "Understanding adversarial training: Increasing local stability of supervised models through robust optimization", *Neurocomputing* **307** (Sep, 2018) 195–204, <http://dx.doi.org/10.1016/j.neucom.2018.04.027>, [arXiv:1511.05432](https://arxiv.org/abs/1511.05432).
- [293] A. Madry et al., "Towards Deep Learning Models Resistant to Adversarial Attacks", *arXiv e-prints* (2019) [arXiv:1706.06083](https://arxiv.org/abs/1706.06083).
- [294] G. Louppe, M. Kagan, and K. Cranmer, "Learning to Pivot with Adversarial Networks", *arXiv e-prints* (11, 2016) [arXiv:1611.01046](https://arxiv.org/abs/1611.01046).
- [295] Y. Ganin and V. Lempitsky, "Unsupervised Domain Adaptation by Backpropagation", *arXiv e-prints* (2015) [arXiv:1409.7495](https://arxiv.org/abs/1409.7495).
- [296] CMS Collaboration, "A deep neural network to search for new long-lived particles decaying to jets", *Mach. Learn. Sci. Tech.* **1** (2020) 035012, <http://dx.doi.org/10.1088/2632-2153/ab9023>, [arXiv:1912.12238](https://arxiv.org/abs/1912.12238).
- [297] A. Ćiprijanović et al., "DeepAdversaries: Examining the Robustness of Deep Learning Models for Galaxy Morphology Classification", *arXiv e-prints* (12, 2021) [arXiv:2112.14299](https://arxiv.org/abs/2112.14299).
- [298] M. Babicz, S. Alonso-Monsalve, S. Dolan, and K. Terao, "Adversarial methods to reduce simulation bias in neutrino interaction event filtering at liquid argon time projection chambers", *Phys. Rev. D* **105** (2022), no. 11, 112009, <http://dx.doi.org/10.1103/PhysRevD.105.112009>, [arXiv:2201.11009](https://arxiv.org/abs/2201.11009).
- [299] A. Ghosh, B. Nachman, and D. Whiteson, "Uncertainty-aware machine learning for high energy physics", *Phys. Rev. D* **104** (2021), no. 5, 056026, <http://dx.doi.org/10.1103/PhysRevD.104.056026>, [arXiv:2105.08742](https://arxiv.org/abs/2105.08742).
- [300] W. Waltenberger, "RAVE: A detector-independent toolkit to reconstruct vertices", *IEEE Trans. Nucl. Sci.* **58** (2011) 434–444, <http://dx.doi.org/10.1109/TNS.2011.2119492>.
- [301] dguest / D. Guest, "delphes-rave", *GitHub repository* (2016). <https://github.com/dguest/delphes-rave> (last accessed: 05.12.2023).
- [302] J. Pivarski et al., "Awkward array", *Zenodo* (November, 2023) <http://dx.doi.org/10.5281/zenodo.4341376>.
- [303] AnnikaStein / A. Stein, "Adversarial-Training-for-Jet-Tagging", *GitHub repository* (2023). <https://github.com/AnnikaStein/>

- [Adversarial-Training-for-Jet-Tagging](#) (last accessed: 05.12.2023).
- [304] RWTH Aachen University IT Center, “Cluster Aix-la-Chapelle”, 2023. <https://www.itc.rwth-aachen.de/cms/it-center/Forschung-Projekte/Forschungsschwerpunkte/Hochleistungsrechnen/~ylkds/CLAIX/lidx/1/> (last accessed: 25.11.2023).
- [305] A. Wald, “Statistical decision functions which minimize the maximum risk”, *Annals of Mathematics* **46** (1945), no. 2, 265–280. <http://www.jstor.org/stable/1969022> (last accessed: 04.12.2023).
- [306] A. Fawzi, S.-M. Moosavi-Dezfooli, and P. Frossard, “Robustness of classifiers: from adversarial to random noise”, *arXiv e-prints* (2016) [arXiv:1608.08967](#). Accepted to NIPS 2016.
- [307] A. Fawzi, O. Fawzi, and P. Frossard, “Analysis of classifiers’ robustness to adversarial perturbations”, *Machine Learning* **107** (2018), no. 3, 481–508, <http://dx.doi.org/10.1007/s10994-017-5663-3>, [arXiv:1502.02590](#).
- [308] H. Li et al., “Visualizing the Loss Landscape of Neural Nets”, *arXiv e-prints* (2018) [arXiv:1712.09913](#). NIPS 2018.
- [309] S. Fort, H. Hu, and B. Lakshminarayanan, “Deep Ensembles: A Loss Landscape Perspective”, *arXiv e-prints* (2020) [arXiv:1912.02757](#).
- [310] A. Kurakin, I. Goodfellow, and S. Bengio, “Adversarial Machine Learning at Scale”, *arXiv e-prints* (2017) [arXiv:1611.01236](#).
- [311] D. Stutz, M. Hein, and B. Schiele, “Disentangling Adversarial Robustness and Generalization”, *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019) 6969–6980, <https://doi.org/10.1109/CVPR.2019.00714>, [arXiv:1812.00740](#).
- [312] N. Papernot, P. McDaniel, and I. Goodfellow, “Transferability in machine learning: from phenomena to black-box attacks using adversarial samples”, [arXiv:1605.07277](#).
- [313] CMS Collaboration, “Identification of c-quark jets at the CMS experiment”, CMS Physics Analysis Summary CMS-PAS-BTV-16-001, CERN, 2016. <https://cds.cern.ch/record/2205149> (last accessed: 05.12.2023).
- [314] ATLAS Collaboration, “Measurement of the c-jet mistagging efficiency in $t\bar{t}$ events using pp collision data at $\sqrt{s} = 13$ TeV collected with the ATLAS detector”, *Eur. Phys. J. C* **82** (2022), no. 1, 95, <http://dx.doi.org/10.1140/epjc/s10052-021-09843-w>, [arXiv:2109.10627](#).
- [315] K. Tapp, “Differential geometry of curves and surfaces”. Springer, 2016. <http://dx.doi.org/10.1007/978-3-319-39799-3>, ISBN 978-3-319-39799-3.
- [316] A. Kurakin et al., “Adversarial Attacks and Defences Competition”, *arXiv e-prints* (04, 2018) [arXiv:1804.00097](#).
- [317] CMS Collaboration, “Mini-AOD: A New Analysis Data Format for CMS”, *J. Phys. Conf. Ser.* **664** (2015), no. 7, 7, <http://dx.doi.org/10.1088/1742-6596/664/7/072052>, [arXiv:1702.04685](#).
- [318] cms-nanoAOD / CMS Collaboration, “nanoAOD-tools”, *GitHub repository* (2023). <https://github.com/cms-nanoAOD/nanoAOD-tools> (last accessed: 05.12.2023).
- [319] cms-btv-pog / CMS Collaboration, “RecoBTag-PerformanceMeasurements”,

- GitHub repository* (2023). <https://github.com/cms-btv-pog/RecoBTag-PerformanceMeasurements> (last accessed: 05.12.2023).
- [320] CMSDeepFlavour / CMS Collaboration, “DeepNTuples”, *GitHub repository* (2023). <https://github.com/CMSDeepFlavour/DeepNTuples> (last accessed: 04.12.2023).
- [321] DL4Jets / CMS Collaboration, “DeepJetCore”, *GitHub repository* (2023). <https://github.com/DL4Jets/DeepJetCore> (last accessed: 04.12.2023).
- [322] ATLAS Collaboration, “The ATLAS Collaboration Software and Firmware”, ATLAS PUB Note ATL-SOFT-PUB-2021-001, CERN, Geneva, 2021. <https://cds.cern.ch/record/2767187> (last accessed: 06.11.2023).
- [323] atlas-flavor-tagging-tools / ATLAS Collaboration, “umami”, *GitLab repository* (2023). <https://gitlab.cern.ch/atlas-flavor-tagging-tools/algorithms/umami> (last accessed: 06.11.2023).
- [324] atlas-flavor-tagging-tools / ATLAS Collaboration, “salt”, *GitLab repository* (2023). <https://gitlab.cern.ch/atlas-flavor-tagging-tools/algorithms/salt> (last accessed: 06.11.2023).
- [325] atlas-flavor-tagging-tools / ATLAS Collaboration, “training-dataset-dumper”, *GitLab repository* (2023). <https://gitlab.cern.ch/atlas-flavor-tagging-tools/training-dataset-dumper> (last accessed: 06.11.2023).
- [326] hqucms / H. Qu, “weaver-core”, *GitHub repository* (2023). <https://github.com/hqucms/weaver-core> (last accessed: 29.11.2023).
- [327] M. Zhang, J. Lucas, J. Ba, and G. E. Hinton, “Lookahead optimizer: k steps forward, 1 step back”, *Advances in neural information processing systems* **32** (2019).
- [328] L. Liu et al., “On the variance of the adaptive learning rate and beyond”, [arXiv:1908.03265](https://arxiv.org/abs/1908.03265).
- [329] mondalspandan / CMS Collaboration, “VHcc - cTag SF”, *GitHub repository* (2023). <https://github.com/mondalspandan/VHcc-cTagSF> (last accessed: 03.11.2023).
- [330] CMS Collaboration, “The CMS NanoAOD data tier”, 2023. <https://twiki.cern.ch/twiki/bin/view/CMSPublic/WorkBookNanoAOD> (last accessed: 03.11.2023).
- [331] CMS Collaboration, “Performance of CMS muon reconstruction in pp collision events at $\sqrt{s} = 7\text{TeV}$ ”, *Journal of Instrumentation* **7** (2012), no. 10, P10002–P10002, <http://dx.doi.org/10.1088/1748-0221/7/10/p10002>.
- [332] J. Lin, “Divergence measures based on the Shannon entropy”, *IEEE Transactions on Information Theory* **37** (1991), no. 1, 145–151, <http://dx.doi.org/10.1109/18.61115>.
- [333] S. Kullback and R. A. Leibler, “On Information and Sufficiency”, *Ann. Math. Statist.* **22** (03, 1951) 79–86, <http://dx.doi.org/10.1214/aoms/1177729694>.
- [334] A. Vaswani et al., “Attention is all you need”, *CoRR* **abs/1706.03762** (2017) [arXiv:1706.03762](https://arxiv.org/abs/1706.03762). NIPS 2017.
- [335] T. Lin, Y. Wang, X. Liu, and X. Qiu, “A survey of transformers”, *AI Open* (2022) [arXiv:2106.04554](https://arxiv.org/abs/2106.04554).
- [336] OpenAI, “Gpt-4 technical report”, [arXiv:2303.08774](https://arxiv.org/abs/2303.08774).
- [337] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding”, [arXiv:1810.04805](https://arxiv.org/abs/1810.04805).

- [338] A. Dosovitskiy et al., “An image is worth 16x16 words: Transformers for image recognition at scale”, [arXiv:2010.11929](#).
- [339] S. Bhojanapalli et al., “Understanding robustness of transformers for image classification”, in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10231–10241. 2021. [arXiv:2103.14586](#).
- [340] J. L. Ba, J. R. Kiros, and G. E. Hinton, “Layer normalization”, [arXiv:1607.06450](#).
- [341] H. Touvron et al., “Going deeper with image transformers”, in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 32–42. 2021. [arXiv:2103.17239](#).
- [342] G. M. Kurtzer et al., “hpcng/singularity: Singularity 3.7.3”, <http://dx.doi.org/10.5281/zenodo.1310023>.
- [343] G. M. Kurtzer, V. Sochat, and M. W. Bauer, “Singularity: Scientific containers for mobility of compute”, *PLOS ONE* **12** (05, 2017) 1–20, <http://dx.doi.org/10.1371/journal.pone.0177459>.
- [344] T. Miyato et al., “Distributional smoothing with virtual adversarial training”, [arXiv:1507.00677](#).
- [345] R. Meyes, M. Lu, C. W. de Puiseau, and T. Meisen, “Ablation studies in artificial neural networks”, [arXiv:1901.08644](#).
- [346] M. Sundararajan, A. Taly, and Q. Yan, “Axiomatic attribution for deep networks”, in *International conference on machine learning*, pp. 3319–3328, PMLR. 2017. [arXiv:1703.01365](#).
- [347] F. Doshi-Velez and B. Kim, “Towards a rigorous science of interpretable machine learning”, *arXiv e-prints* (02, 2017) [arXiv:1702.08608](#).
- [348] G. Montavon, W. Samek, and K.-R. Müller, “Methods for interpreting and understanding deep neural networks”, *Digital Signal Processing* **73** (February, 2018) 1–15, <http://dx.doi.org/10.1016/j.dsp.2017.10.011>.
- [349] T. Faucett, J. Thaler, and D. Whiteson, “Mapping Machine-Learned Physics into a Human-Readable Space”, *Phys. Rev. D* **103** (2021), no. 3, 036020, <http://dx.doi.org/10.1103/PhysRevD.103.036020>, [arXiv:2010.11998](#).
- [350] F. Fan, J. Xiong, M. Li, and G. Wang, “On interpretability of artificial neural networks: A survey”, *arXiv e-prints* (01, 2021) [arXiv:2001.02522](#).
- [351] C. Grojean, A. Paul, Z. Qian, and I. Strümke, “Lessons on interpretable machine learning from particle physics”, *Nature Rev. Phys.* **4** (2022), no. 5, 284–286, <http://dx.doi.org/10.1038/s42254-022-00456-0>, [arXiv:2203.08021](#).
- [352] R. Brun and F. Rademakers, “ROOT: An object oriented data analysis framework”, *Nucl. Instrum. Meth. A* **389** (1997) 81–86, [http://dx.doi.org/10.1016/S0168-9002\(97\)00048-X](http://dx.doi.org/10.1016/S0168-9002(97)00048-X).
- [353] V. Kuznetsov, D. Evans, and S. Metson, “The cms data aggregation system”, *Procedia CS* **1** (05, 2010) 1535–1543, <http://dx.doi.org/10.1016/j.procs.2010.04.172>.
- [354] CERN Open Data / T. Simko, “opendata-client”, 2023. <https://cernopendata-client.readthedocs.io/en/latest/> (last accessed: 25.11.2023).
- [355] A. Dorigo, P. Elmer, F. Furano, and A. Hanushevsky, “Xrootd - a highly scalable architecture for data access”, *WSEAS Transactions on Computers* **4** (04, 2005) 348–353.

- [356] D. Spiga et al., “The CMS Remote Analysis Builder (CRAB)”, *Lect. Notes Comput. Sci.* **4873** (2007) 580–586, http://dx.doi.org/10.1007/978-3-540-77220-0_52.
- [357] M. Galli, E. Tejedor, and S. Wunsch, “A New PyROOT: Modern, Interoperable and More Pythonic”, *EPJ Web Conf.* **245** (2020) 06004, <http://dx.doi.org/10.1051/epjconf/202024506004>.
- [358] J. Pivarski et al., “Uproot”, *Zenodo* (June, 2023) <http://dx.doi.org/10.5281/zenodo.8062258>.
- [359] Anaconda Inc., “Anaconda Software Distribution”, Vers. 2-2.4.0 (Nov, 2016). <https://anaconda.com> (last accessed: 29.11.2023).
- [360] I. Bird et al., “LHC computing Grid. Technical design report”, Technical Report CERN-LHCC-2005-024, CERN, Geneva, 6, 2005. <https://cds.cern.ch/record/840543> (last accessed: 05.12.2023).
- [361] H. Camp, “Goal setting as teacher development practice.”, *International journal of teaching and learning in higher education* **29** (2017), no. 1, 61–72. <https://files.eric.ed.gov/fulltext/EJ1135994.pdf> (last accessed: 05.12.2023).
- [362] J. Nickolls, I. Buck, M. Garland, and K. Skadron, “Scalable parallel programming with cuda: Is cuda the parallel programming model that application developers have been waiting for?”, *Queue* **6** (mar, 2008) 40–53, <http://dx.doi.org/10.1145/1365490.1365500>.
- [363] cms-opendata-analyses / CMS Collaboration, “Physics Objects Extractor Tool (POET)”, *GitHub repository* (2023). <https://github.com/cms-opendata-analyses/PhysObjectExtractorTool> (last accessed: 06.12.2023).
- [364] cms-opendata-analyses / CMS Collaboration, “HiggsToBBNtupleProducer-Tool”, *GitHub repository* (2023). <https://github.com/cms-opendata-analyses/HiggsToBBNtupleProducerTool> (last accessed: 06.12.2023).
- [365] M. Abadi et al., “TensorFlow: Large-scale machine learning on heterogeneous systems”, 2015. Software available from <https://www.tensorflow.org/> (last accessed: 03.12.2023).
- [366] C. R. Harris et al., “Array programming with NumPy”, *Nature* **585** (September, 2020) 357–362, <http://dx.doi.org/10.1038/s41586-020-2649-2>.
- [367] Wes McKinney, “Data Structures for Statistical Computing in Python”, in *Proceedings of the 9th Python in Science Conference*, Stéfan van der Walt and Jarrod Millman, eds., pp. 56 – 61. 2010. <http://dx.doi.org/10.25080/Majora-92bf1922-00a>.
- [368] onnx, “onnx”, *GitHub repository* (2023). <https://github.com/onnx/onnx> (last accessed: 04.12.2023).
- [369] CMS Collaboration, “A first look at early 2022 proton-proton collisions at $\sqrt{s} = 13.6$ TeV for heavy-flavor jet tagging”, CMS Detector Performance Summary CMS-DP-2023-012, CERN, Geneva, 2023. <http://cds.cern.ch/record/2854698> (last accessed: 04.11.2023).
- [370] L. Gray et al., “coffea”, *Zenodo* (2023) <http://dx.doi.org/10.5281/zenodo.3266454>.
- [371] Parsl, “parsl”, *GitHub repository* (2023). <https://github.com/Parsl/parsl> (last accessed: 04.12.2023).

- [372] `dask`, “`dask`”, *GitHub repository* (2023). <https://github.com/dask/dask> (last accessed: 04.12.2023).
- [373] J. Pivarski, J. Nandi, D. Lange, and P. Elmer, “Columnar data processing for HEP analysis”, *EPJ Web Conf.* **214** (2019) 06026, <http://dx.doi.org/10.1051/epjconf/201921406026>.
- [374] J. Pivarski, P. Elmer, and D. Lange, “Awkward Arrays in Python, C++, and Numba”, *EPJ Web Conf.* **245** (2020) 05023, <http://dx.doi.org/10.1051/epjconf/202024505023>, [arXiv:2001.06307](https://arxiv.org/abs/2001.06307).
- [375] P. Virtanen et al., “SciPy 1.0: fundamental algorithms for scientific computing in Python”, *Nature Methods* **17** (Feb, 2020) 261–272, <http://dx.doi.org/10.1038/s41592-019-0686-2>, [arXiv:1907.10121](https://arxiv.org/abs/1907.10121).
- [376] Beyer, Christoph et al., “Beyond HEP: Photon and accelerator science computing infrastructure at DESY”, *EPJ Web Conf.* **245** (2020) 07036, <http://dx.doi.org/10.1051/epjconf/202024507036>.
- [377] `conda`, “`conda-pack`”, *GitHub repository* (2023). <https://github.com/conda/conda-pack> (last accessed: 04.12.2023).
- [378] H. Kim, W. Lee, and J. Lee, “Understanding catastrophic overfitting in single-step adversarial training”, in *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 8119–8127. 2021. [arXiv:2010.01799](https://arxiv.org/abs/2010.01799).
- [379] Z. He, T. Li, S. Chen, and X. Huang, “Investigating catastrophic overfitting in fast adversarial training: A self-fitting perspective”, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2313–2320. 2023. [arXiv:2302.11963](https://arxiv.org/abs/2302.11963).
- [380] L. Roeder, “Netron: Visualizer for neural network, deep learning and machine learning models.”, *GitHub repository* (2023). <https://github.com/lutzroeder/netron> (last accessed: 05.12.2023).

List of Figures

2.1	NNPDF3.1 NNLO PDFs at two different scales.	28
2.2	Graphical visualization of a hadron collision [76].	30
3.1	Sketch of the CERN accelerator complex with several experiments and accelerators, the launch year and individual circumferences, compiled at the end of LS 2 [111].	34
3.2	Peak delivered luminosity over time per fill (a), total integrated luminosity, cumulatively over time (b), average PU distribution by year (c).	37
3.3	A 3D model of the CMS detector after the Phase-1 pixel upgrade 2017 – 2018 [140, 141].	40
4.1	Jet energy fraction carried by different types of particles with simulated samples, split by flavour and grouping / merging individual types of hadrons for Z+jet events (a), at the particle level as a function of jet transverse momentum and for a QCD dijet sample (b).	45
4.2	Schematic display of infrared- (a) and collinear-safety (b).	46
4.3	Schematic view of three jets that emerge from a common primary vertex (PV).	51
4.4	Architectures of the DeepCSV (a) and DeepJet (b) taggers.	55
4.5	Performance of DeepCSV (a) and DeepJet (b) expressed via b-jet identification efficiency (x axis) versus light-jet misidentification rate (y axis), using rates in simulation and comparing them with results obtained with the help of data recorded in 2017.	57
5.1	Sketch of classifier score distributions for positive and negative classes, for a typical classifier, compared with a random classifier, and corresponding ROC curves	68
6.1	Generating adversarial inputs with the FGSM attack, simplified with one dimension only.	78
6.2	Adversarial training algorithm utilizing the FGSM attack in the training loop.	79
6.3	Comparison of the nominal and adversarial training against the FGSM attack.	80
6.4	The inference process to perform a comparison of robustness of both training strategies.	81
6.5	Distributions of raw and systematically distorted inputs, for a set of features containing high- and low-level information.	82
6.6	ROC curves for the BvsL (left) and CvsL discriminant (right), using the nominal training and applying FGSM attacks of different magnitudes.	83
6.7	ROC curves for the BvsL (left) and CvsL discriminant (right), using the nominal training and applying FGSM attacks with $\epsilon = 0.05$ at various checkpoints of the training that each come with different nominal performance.	83

6.8	ROC curves for the BvsL (left) and CvsL (right) discriminants, comparing the nominal with adversarial training when applying the FGSM attack to both trainings individually.	84
6.9	Relation between susceptibility and nominal performance for the nominal and adversarial training, tested on systematically distorted inputs with varying ϵ in different colors.	85
6.10	Signed transverse impact parameter distribution for the first track, split by flavour, before (filled histograms) and after (lines) applying the FGSM attack for the nominal (top) and adversarial (bottom) models, respectively.	86
6.11	Illustration of the potential geometry of the loss surfaces for the nominal as well as the adversarial training.	87
6.12	Visualization of the random shift of inputs by adding a Gaussian noise term.	88
6.13	ROC curves for the BvsL (left) and CvsL (right) discriminants, comparing the nominal with adversarial training when smearing the inputs with a Gaussian noise term.	88
6.14	ROC curves for the BvsL (left) and CvsL (right) discriminants, comparing the nominal with adversarial training when applying the FGSM attack to the nominal training and injecting the obtained inputs to both models.	89
6.15	ROC curves for the BvsL (left) and CvsL (right) discriminants, comparing the nominal with adversarial training when shifting inputs systematically downwards.	90
6.16	ROC curves for the BvsL (left) and CvsL (right) discriminants, comparing the nominal with adversarial training when shifting inputs systematically upwards.	90
6.17	Different geometries of loss manifolds for nominal (bottom) and adversarial (top) training. The figure and caption is taken from Ref. [16].	92
6.18	Possible directions of adversarial attacks for different models.	92
7.1	Comparison of existing (left) and new (right) pipelines incorporating NN training for jet tagging algorithms and evaluation with data.	97
7.2	The DeepJet architecture in Run 2 configuration with schematic representation of FGSM input modifications.	98
7.3	ROC curves for the DeepJet BvsL discriminant, comparing the nominal with adversarial training with and without adversarial attack.	100
7.4	ROC curves for the DeepJet CvsB discriminant, comparing the nominal (blue) with adversarial (orange) training with and without adversarial attack.	101
7.5	ROC curves for the DeepJet CvsL discriminant, comparing the nominal (blue) with adversarial (orange) training with and without adversarial attack.	102
7.6	BvsL discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the light-flavour enriched selection.	104
7.7	CvsB discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the light-flavour enriched selection.	104

7.8	CvsL discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the light-flavour enriched selection.	105
7.9	BvsL discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the charm-flavour enriched selection.	105
7.10	CvsB discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the charm-flavour enriched selection.	106
7.11	CvsL discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the charm-flavour enriched selection.	106
7.12	BvsL discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the bottom-flavour enriched selection.	107
7.13	CvsB discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the bottom-flavour enriched selection.	107
7.14	CvsL discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the bottom-flavour enriched selection.	108
7.15	Scale factors for nominal training and associated statistical uncertainties, for three different flavours (b, c, light) from top to bottom.	114
7.16	Scale factors for adversarial training and associated statistical uncertainties, for three different flavours (b, c, light) from top to bottom.	115
7.17	Pseudoangular distance between the first neutral particle flow candidate and the jet axis for simulated samples and data.	116
7.18	Changes for the pseudoangular distance between the first neutral particle flow candidate and the jet axis as introduced by the FGSM attack, split by flavour, comparing nominal and adversarial training on the left and right, respectively.	117
7.19	General overview of the ParT architecture (a), the Particle Attention Block (b) and the Class Attention Block (c). Figures (a)-(c) taken from Ref. [20].	119
7.20	Two-dimensional comparison of the FGSM and NGM attacks, which either reflect only the sign of the gradient, no matter the exact direction or magnitude of the gradient vector (left), or take the direction and relative impact of features into account (right). Figures not drawn to scale.	121
7.21	How a possible input modification with a simple PGD implementation looks like, when utilizing the ℓ_∞ norm (known from the FGSM attack) only, without (left) and with random initialization (right).	123
7.22	Performance of the first Run 2 (Robust)ParTAK4 algorithm for the BvsL classification task, comparing nominal (blue) and adversarial (orange) training on nominal inputs (solid lines), NGM inputs (dashed lines), and FGSM inputs (dotted lines).	125
7.23	Performance of the retrained Run 2 (Robust)ParTAK4 algorithm for the BvsL classification task, comparing nominal (blue) and adversarial (orange) training on nominal inputs (solid lines), NGM inputs (dashed lines), and FGSM inputs (dotted lines).	126

7.24	Performance of the first Run 2 (Robust)ParTAK4 algorithm for the CvsB classification task, comparing nominal (blue) and adversarial (orange) training on nominal inputs (solid lines), NGM inputs (dashed lines), and FGSM inputs (dotted lines).	127
7.25	Performance of the retrained Run 2 (Robust)ParTAK4 algorithm for the CvsB classification task, comparing nominal (blue) and adversarial (orange) training on nominal inputs (solid lines), NGM inputs (dashed lines), and FGSM inputs (dotted lines).	128
7.26	Performance of the first Run 2 (Robust)ParTAK4 algorithm for the CvsL classification task, comparing nominal (blue) and adversarial (orange) training on nominal inputs (solid lines), NGM inputs (dashed lines), and FGSM inputs (dotted lines).	129
7.27	Performance of the retrained Run 2 (Robust)ParTAK4 algorithm for the CvsL classification task, comparing nominal (blue) and adversarial (orange) training on nominal inputs (solid lines), NGM inputs (dashed lines), and FGSM inputs (dotted lines).	130
7.28	Distributions of pseudoangular distance of the first NPF candidate with respect to the jet (top) and 2D SIP value for the second track (bottom), comparing nominal inputs with systematically distorted inputs.	131
7.29	Correlation between nominal (x axis) and attacked (y axis) inputs using FGSM (blue) or NGM (orange), for the second charged PF candidate (track) SIP value in two dimensions, all values given in units of cm.	132
7.30	Impact of NGM attack on output scores of a nominal ParticleTransformer training, split into six ground truth flavours for b (bb) scores on the left (right) column in the first row, leptonic b (c) jets on the left (right) in the second row, uds (g) jets on the left (right) in the third row.	133
7.31	BvsL discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the light-flavour enriched selection.	134
7.32	CvsB discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the light-flavour enriched selection.	135
7.33	CvsL discriminant distributions, comparing the nominal (left) with adversarial training (right) after applying the light-flavour enriched selection.	135
7.34	Performance impact of adversarial training against NGM for the CMS Run 3 (Robust)ParticleTransformer algorithm applied to AK4 jets in semi-leptonic $t\bar{t}$ samples.	137
7.35	Performance of different b-tagging algorithms evaluated with CMS Run 3 simulation of AK4 jets from top quark pairs decaying semi-leptonically.	138
7.36	Impact of FGSM or NGM attacks on nominal training or adversarial training for (Robust)ParticleTransformer applied on AK4 jets using the Run 3 setup with simulated top quark pair events.	139
7.37	Step-by-step derivation of input shifts related to observed data/MC ratios per input feature and flavour.	141

7.38	Schematic illustration of how different choices for ϵ_{usdg} , ϵ_{charm} and ϵ_{bottom} affect performance and robustness for different flavours.	142
8.1	Proposed structure for the Open Data “AI Safety Wrapper” that reads data from various datasets and applies a set of common attacks or defenses in an enclosing framework.	146
8.2	Distributions of jet transverse momentum before pre-processing (left) and after pre-processing (right).	153
8.3	Distributions of jet transverse momentum before pre-processing (left) and after pre-processing (right).	153
B.1	Simulated PFNano samples for hadronic $t\bar{t}$ and Drell-Yan (into two leptons, with additional jets) processes, split into jet flavour and applying different selection criteria related to quality.	171
C.1	Training and validation loss of nominal Delphes training and Delphes adversarial training.	181
C.2	Discriminant distributions for nominal Delphes training with focal loss.	183
C.3	Discriminant distributions for nominal Delphes training without focal loss.	184
C.4	Output score distributions for nominal Delphes training with focal loss.	184
C.5	Output score distributions for nominal Delphes training without focal loss.	184
C.6	Confusion matrices for nominal Delphes training using focal loss with focusing parameter $\gamma = 25$ on top of the kinematic and flavour reweighting technique.	185
C.7	Confusion matrices for nominal Delphes training using only the kinematic and flavour reweighting technique.	185
C.8	BvsL ROC curve for nominal Delphes training without focal loss, showing nominal and systematically distorted inputs.	186
C.9	CvsL ROC curve for nominal Delphes training without focal loss, showing nominal and systematically distorted inputs.	186
C.10	Output scores for nominal training, exposed to nominal or systematically distorted inputs (with FGSM at $\epsilon = 0.05$, and applied restriction of 25%).	187
C.11	Output scores for adversarial training, exposed to nominal or systematically distorted inputs (with FGSM at $\epsilon = 0.05$, and applied restriction of 25%).	187
C.12	Discriminant shapes for nominal training, exposed to nominal or systematically distorted inputs (with FGSM at $\epsilon = 0.05$, and applied restriction of 25%).	188
C.13	Discriminant shapes for adversarial training, exposed to nominal or systematically distorted inputs (with FGSM at $\epsilon = 0.05$, and applied restriction of 25%).	189
E.1	Comparison of nominal (adversarial) training on the left (right), in the light-flavour enriched selection with 60 bins for histograms of the three discriminants BvsL, CvsB, CvsL from top to bottom.	200

E.2	Comparison of nominal (adversarial) training on the left (right), in the c-flavour enriched selection with 60 bins for histograms of the three discriminants BvsL, CvsB, CvsL from top to bottom.	201
E.3	Comparison of nominal (adversarial) training on the left (right), in the b-flavour enriched selection with 60 bins for histograms of the three discriminants BvsL, CvsB, CvsL from top to bottom.	202
E.4	JS divergence scores for three different selections (udsg, c, b jets from top to bottom) and two different binning setups, with 30 bins (60 bins) on the left (right).	203
E.5	χ^2 values for three different selections (udsg, c, b jets from top to bottom) and two different binning setups, with 30 bins (60 bins) on the left (right).	204
F.1	Correlation between nominal (x axis) and FGSM-attacked (y axis) inputs, split by flavour (colours), for the second charged PF candidate (track) SIP value in two dimensions, all values given in units of cm.	205
F.2	Correlation between nominal (x axis) and NGM-attacked (y axis) inputs, split by flavour (colours), for the second charged PF candidate (track) SIP value in two dimensions, all values given in units of cm.	206
F.3	Training history for an alternative training scenario of RobustParticleTransformer where adversarial training and focal loss are introduced to fine-tune the initial nominal training period with cross entropy loss.	207
F.4	BvsL performance for a pre-trained ParT NN with subsequent adversarial training phase and moving to focal loss, evaluated on nominal samples (blue), NGM-attacked samples (orange), and FGSM-attacked samples (green) with parameter $\epsilon = 0.01$ for attacks.	208
F.5	CvsB performance for a pre-trained ParT NN with subsequent adversarial training phase and moving to focal loss, evaluated on nominal samples (blue), NGM-attacked samples (orange), and FGSM-attacked samples (green) with parameter $\epsilon = 0.01$ for attacks.	209
F.6	CvsL performance for a pre-trained ParT NN with subsequent adversarial training phase and moving to focal loss, evaluated on nominal samples (blue), NGM-attacked samples (orange), and FGSM-attacked samples (green) with parameter $\epsilon = 0.01$ for attacks.	210
F.7	Validating RobustParticleTransformer PyTorch model output scores against ONNX converted outputs as present in CMSSW.	213
G.1	Newer versions of PFNano contain low-level information for every PF candidate in a single exemplary event.	216
G.2	More examples of additional (low-level) input features that can be studied with PFNano, using a $H \rightarrow c\bar{c}$ signal event as an example.	216

List of Tables

2.1	Fermions (quarks, leptons) in the SM, split into three generations each, with their masses taken from Ref. [1].	18
2.2	Fermions and their quantum numbers for electric charge Q , third component of weak isospin I^3 , and weak hypercharge Y	23
4.1	Comparison between the DeepCSV and DeepJet algorithms: differences and similarities.	56
5.1	Confusion matrix of a binary classification problem, where predictions are compared against the actual classes, taken from Refs. [233,282].	66
5.2	Terminology for selected performance metrics for binary classification tasks and alternative naming schemes for object identification such as jet flavour tagging [9,17,233,282].	67
7.1	Agreement between data and simulation, expressed through the Jensen-Shannon divergence, in three control regions (light-, charm- and bottom-enriched samples) and for three discriminants each (BvsL, CvsB, CvsL).	109
8.1	Different storage options considered for OpenNano and size of resulting files containing small-radius jet information.	151
B.1	Assigning truth flavour classes in delphes-rave to small-radius jets, ordered hierarchically.	167
B.2	Expert / high-level features for the neural network.	168
B.3	Low-level and intermediate-level features for the neural network.	169
B.4	List of simulated samples entering jet flavour tagging algorithm development and performance measurements, with usage specified in additional columns.	173
B.5	List of simulated samples for RobustParticleTransformer development for Run 3, always choosing the default puppiJet collection, for these samples with PUP-Plv15.	174
B.6	List of recorded detector data, grouped by trigger paths (triggering two muons, or a single muon).	174
B.7	Assigning truth flavour classes in PFNano to small-radius jets in CMS: the definition of the Jet_FlavSplit branch.	175
B.8	Global variables entering DeepJet.	176
B.9	Charged PF candidate variables entering DeepJet.	177
B.10	Neutral PF candidate variables entering DeepJet.	178
B.11	Secondary vertex variables entering DeepJet.	178
B.12	Variables entering RobustParticleTransformer in Run 2 or Run 3 configuration, compared with DeepJet.	179

F.1	Runtime comparison for production of different TagInfos and JetTags associated with ParticleNetAK4 and ParticleTransformerAK4, respectively, when investigating the producers in the PatAlgos/slimming (MiniAOD) step.	214
-----	--	-----

Contributions

Published Articles

- (A*) A. Stein, X. Coubez, S. Mondal, A. Novak, A. Schmidt, **"Improving Robustness of Jet Tagging Algorithms with Adversarial Training"**.

[Comput Softw Big Sci \(2022\) 6: 15. arXiv:2203.13890](#)

(The author of this thesis has conducted all data analysis steps, which included writing a new software framework to investigate the Delphes dataset, prepared all figures, wrote the initial version of the manuscript and acted as corresponding author. The introduction of a modified loss function for this classification task was proposed by the author of this thesis, which enabled the efficient application of the adversarial defense strategy in the context of jet flavour tagging for the first time on an imbalanced dataset.)

Conference Proceedings

- (B*) A. Stein, **"Improving robustness of jet tagging algorithms with adversarial training: exploring the loss surface"**.

[arXiv:2303.14511](#). Accepted for publication in Journal Of Physics: Conference Series for ACAT 2022, AI meets Reality.

(This contribution contains material previously discussed in Comput Softw Big Sci (2022) 6: 15, which has been significantly augmented by performing a loss surface scan of a jet flavour tagging algorithm for the first time. The author did all formal analysis, compared the nominal and adversarial training in terms of geometric properties of the associated loss manifold. Additionally, she prepared the graphical design and added visualizations that show and explain the findings.)

Conference Contributions (Without Proceedings)

- (C*) A. Stein, X. Coubez, S. Mondal, A. Novak, A. Schmidt, **"Improving robustness of jet tagging algorithms with adversarial training"**. Center for Data and Computing in Natural Sciences Opening Symposium, 2022.

[contribution/120862](#)

(The same statement as for Comput Softw Big Sci (2022) 6: 15 applies, as the contribution closely follows the results discussed in the paper. Additionally, the author of this thesis prepared the results for a different audience (with backgrounds in biology or chemistry), and presented the results at the conference.)

Reports

- (D) CMS Collaboration, **"A first look at early 2022 proton-proton collisions at $\sqrt{s} = 13.6$ TeV for heavy-flavor jet tagging"**.

<https://cds.cern.ch/record/2854698>. CMS DP-2023/012.

(The author of this thesis has extended the customized data tier to include relevant inputs and outputs of the investigated algorithms, adapted the customization to changing conditions and setup for Run 3 (especially the jet collections with new pileup mitigation, which became accessible for the commissioning studies with the author's work), produced samples and co-coordinated the production in the working group. As a co-developer of the analysis framework, the author of this thesis has introduced technical optimizations especially for high-throughput computing and contributed to various workflows by performing extensive code review, cross-checking selections with respect to logic, physics and calibrations, laid the groundwork for keeping an external, reusable module which contains relevant properties of the observables under investigation, and contributed to correct representation of physical quantities and style with domain knowledge.)

(E*) CMS Collaboration, "**Adversarial training for b-tagging algorithms in CMS**".

<https://cds.cern.ch/record/2839919>. CMS DP-2022/049.

(The author of this thesis has introduced the previously studied adversarial training for a more complex neural network, involving convolutional and recurrent layers, and conducted extensive studies of the utilized input features to find suitable parameters for the modified training strategy. The author of this thesis has performed the training of the investigated tagging algorithms, prepared samples for evaluation by customizing available data tiers and performing batch production, wrote the software framework to run with aforementioned custom algorithms and datasets, and conducted the data analysis to probe agreement between simulation and data. All text and figures included in this note have been prepared by the author of this thesis, who also acted as internal contact person and communicated the results, prospects and limitations to the collaboration.)

Internal Notes

(F*) A. Stein and A. de Moor, "**Jet flavour identification for Run 3 with the RobustParTAK4 algorithm**". CMS AN-2023-062.

(The author of this thesis was responsible for the jet flavour tagging reconstruction software especially for Run 3 (and for derived data tiers in collaboration with JME, XPOG) and introduced a novel training paradigm. The algorithm is documented in this note, and at the time of writing in October 2023 at v1.0.0, all text that went into the internal note was prepared by the author of this thesis. Figures were prepared by both authors. All studies with detector data were done by the author of this thesis, studies using simulation only were shared between authors.)

Published Datasets

(G*) A. Stein and T. C. Briggs, "**Jet Flavour Tagging OpenData Sample**".

DOI: 10.34740/kaggle/ds/3706085.

<https://www.kaggle.com/datasets/annikastein/jet-tagging-opendata-mockup>.

(The author of this thesis prepared the dataset with a framework to extract and calculate low-level information based on jet constituent properties. A set of plugins and scripts have been written to facilitate the production with high-performance computing infrastructures, for experts with CMSSW experience. To facilitate exchange with other (academic) communities, the author of this thesis has derived a novel file structure that is fully independent of experiment-specific

software and which does not require any experience with ROOT. The author of this thesis then executed production and conversion of the samples. The file structure has been exchanged with the co-author, whose feedback was included for the public version of the dataset. The author of this thesis modified an exemplary notebook that shows how to use the files directly on the kaggle platform, increasing the datasets's usability score.)

Software and Tutorials

- (H*) Alejandro Gomez Espinosa, Andrzej Novak, rappoccio, camclean, Ashley Marie Parker, Annika Stein, Lauren Hay, AlionaD, Nicholas Smith, Javier Duarte, & Daniel Holmberg, **"AnnikaStein/OpenNano: v1.0.0"**. Zenodo.

<https://doi.org/10.5281/zenodo.10023537>.

(The author of this thesis introduced producers that serve as plugins when processing Open Data samples in MiniAOD format, and derived a novel data format to store the resulting branches in a ML-friendly way. She modified existing functionality to ensure compatibility with early Run 2 samples that do not contain the now standard variables. Hundreds of features can be computed from several object collections, making use of first principles instead of trying to access the non-existing branches. Additionally, she added scripts that can be used with HPC clusters to automatize the sample production. Making use of a new combination of experiment-specific software and fully pythonic, independent tools all happening in the same job, she improved efficiency of the required intermediate file saving and bookkeeping, making storing any ROOT files obsolete, thus improving outreach potential to a wider audience.)

- (I) A. Stein, **"AnnikaStein/short-ex-ftag-edu: v1.0.0"**. Zenodo.

<https://doi.org/10.5281/zenodo.10005025>.

(The tutorial was created by the author of this thesis to teach graduate students in jet flavour tagging. Discussed points include theoretical background of input features with discriminating power, flavour definitions in various data tiers, performance evaluation with simulation and in data. A core novelty with respect to earlier editions of the b-tagging short exercise is the fully pythonic tutorial, where students are taught how to think in columnar way and apply functions in vectorized, efficient manner. Further, the material is meant to start from the ground up with input distributions, split by flavour. The author prepared example files, educational notebooks, the accompanying presentation and documentation, and tested the tutorial with several groups of students as lead facilitator of the flavour tagging short exercise during the CMS Data Analysis School 2023 at CERN (internal link to the session: <https://indico.cern.ch/event/1257234/timetable/#39-short-exercise-3b-flavour-t>.)

*Results from these publications are presented within this thesis.

Eidesstattliche Erklärung

Ich, **Annika Stein**, erkläre hiermit, dass diese Dissertation und die darin dargelegten Inhalte die eigenen sind und selbstständig, als Ergebnis der eigenen originären Forschung, generiert wurden.

Hiermit erkläre ich an Eides statt

1. Diese Arbeit wurde vollständig oder größtenteils in der Phase als Doktorand dieser Fakultät und Universität angefertigt;
2. Sofern irgendein Bestandteil dieser Dissertation zuvor für einen akademischen Abschluss oder eine andere Qualifikation an dieser oder einer anderen Institution verwendet wurde, wurde dies klar angezeigt;
3. Wenn immer andere eigene- oder Veröffentlichungen Dritter herangezogen wurden, wurden diese klar benannt;
4. Wenn aus anderen eigenen- oder Veröffentlichungen Dritter zitiert wurde, wurde stets die Quelle hierfür angegeben. Diese Dissertation ist vollständig meine eigene Arbeit, mit der Ausnahme solcher Zitate;
5. Alle wesentlichen Quellen von Unterstützung wurden benannt;
6. Wenn immer ein Teil dieser Dissertation auf der Zusammenarbeit mit anderen basiert, wurde von mir klar gekennzeichnet, was von anderen und was von mir selbst erarbeitet wurde;
7. Teile dieser Arbeit wurden zuvor veröffentlicht und zwar in den mit (*) gekennzeichneten Contributions (A*), (B*), (C*), (E*), (F*), (G*), (H*).

Ort, Datum

Unterschrift

Acknowledgements

First, I would like to thank Prof. Alexander Schmidt for allowing me to work on such interesting topics and giving me the opportunity to complete what I started during my Master's thesis. Thank you for your support and feedback, and for encouraging me to present my work at conferences. I would also like to thank my second advisor, Prof. Michael Krämer for feedback on various machine learning aspects and discussions. It improved deeper understanding of concepts and brought new perspectives, especially for what concerns modern ML architectures. This also leads me to the fantastic environment at the Research Training Group (GRK 2497), where I want to thank Prof. Michal Czakon for being an enthusiastic coordinator and motivator, as well as Prof. Malgorzata Worek and Prof. Robert Harlander, with whom I worked together on organizing events such as the RTG Colloquium. Being it enlightening discussions after the talks or during the Phenochat, or the annual retreats with social events, I am happy to have been part of this group. Special thanks go to Prof. Worek, who acted as a great mentor regarding careers in academia and for encouraging feedback. Thank you, Prof. Worek and Prof. Czakon for inviting me to present at the review panel, it was a real pleasure and an honour to speak there. Thanks also go to Dennis Noll and Frederic Engelke who served as student representatives and acted as great communicators between students and PIs.

Thank you to Benjamin Nachman and Chase Shimmin for great initial work on AI Safety in HEP. Thanks also go to Dan Guest and Jonathan Shlomi for answering my jet tagging, vertexing and Delphes questions, and to Philipp Gadow for an interesting discussion on software for flavour tagging. Thanks to all for your expertise and genuine interest.

Throughout this journey, I had the pleasure to work with curious students and enjoy fun moments together. Representative for those working on their theses, with Summer Kassem - your enthusiasm for the project was amazing and our conversations always a highlight. You asked the right questions and always made sure to really understand something, and through that I also gained more insights as well. Taylor Briggs, you were an incredible DAAD exchange student and it's sad this internship didn't last longer. Thank you to the DAAD Rise Germany program for making this possible, it was a great experience for everyone involved!

I would also like to thank Valentina Sarkisovi, for nice conversations (and gossip) on- and off-topic over a coffee. Thank you also to Luca Mastrolorenzo and Xavier Coubez, I am happy that I got to meet you in person in Geneva, while only brief, this was for sure an enjoyable time. Especially during the first few months of my PhD where the learning curve was steep, I could count on your experience to decipher HEP jargon in a fun way.

Thank you to our secretaries and / or non-academic staff, who had to answer a lot of requests and always had an open ear. Besides my usual working group, I also got the chance to work together with Prof. Johannes Erdmann, Jan Lukas Späh and the whole team on leading the Datenverarbeitung (DV) exercise classes. This was an instructive experience and introducing real programming tasks into electronic exams was demanding, but definitely something I won't forget so quickly! Thanks also go to Henning Gast, who provided useful tools for DV that made things a lot easier.

In Hamburg at the CDCS Opening Symposium, I met Gesa Goetzke, whose curiosity on adversarial attacks and critical thinking had a huge impact on what I developed later as the successor of the previous approach - thank you for the great questions and fun discussions on

the boat cruise. While I'm not a biologist or anything close, at the same occasion I had the opportunity to meet Tanja Laske, and want to thank her for motivating words of encouragement in my very early stages of the PhD and being so kind to let me use her office during my time of stay. It was fantastic to have the dog-picture-poster-fanclub around me while exploring the DESY campus and facilities! Thanks also to my fellow physicists from Mainz whom I met in Dresden for the DPG conference after a long time again and with whom I spent some really nice coffee breaks there, David, Saskia and colleagues. Thanks also go to the associate editors with whom I got to work with as a reviewer of physics and software papers, being given such responsibility as an early career researcher was an honour, thank you for giving me this opportunity.

It goes without saying that the speedcubing community all around the world, and especially the German Delegate- and Orga-Team, had a huge impact on me personally. Being a volunteer in such a team is great, thanks to both Lauras, Sébastien, Javid, Ricardo, Leon, Linus and Wilhelm, as well as the entire cubing community, no matter where, it always feels like a big family of twisty puzzle enthusiasts, genuinely wishing each other great results. Outside of cubing, thank you to my coffee-, Kirmes- and Weinfest-buddies, especially Vanessa and Mohini, nothing beats our (very serious!) STEM-discussions on the fairground, of course only with the correct choice of fries. On the same note, time to look back to my time in school, where I enjoyed challenging and motivating classes in Physics and Mathematics by Herr Reick and Frau Sausen.

I am deeply grateful for my family's unlimited support throughout this whole journey. It really means a lot to know I always had and have the encouragement and assistance of my parents, Marion and Eugen Stein, for whatever happens or whatever challenges arise. Show must go on! Thanks also go to my brother Tobias for helping out several times, especially with car maintenance. Alexa Dollitz and Olaf Steffens – auf der anderen Seite des Waldes – contributed to this support system, especially for what concerns proper and quick treatment of the serious illness. Indeed, this thesis certainly would not have been possible without Dr. Scholz, modern medical science and the amazing team at Klinikum Idar-Oberstein, Dr. Sklavounos and all staff at the ICU and other stations. And once again, as with all previous theses (and more), vielen Dank Mama fürs Korrekturlesen!