



Sorbonne Université

École Doctorale des Sciences de la Terre et de l'Environnement et Physique de l'Univers, Paris

LPNHE / ATLAS Group — IBM Research France

Advancing Machine Learning for High Energy Physics: From Jet Calibration Optimisation to Time Series Anomaly Detection

Laura BOGGIA

Doctoral Thesis in Physics of the Universe

Directed by Bogdan MALAESCU

Co-Supervised by Rafael TEIXEIRA DE LIMA and Shubham GUPTA

Presented and defended on the 29th of September 2025 in front of the committee:

Caterina DOGLIONI, Professor, University of Manchester

Referee

Tobias GOLLING, Professor, University of Geneva

Referee

Dorothea VOM BRUCH, CNRS Researcher, CPPM

Examiner

Delphine HARDIN, Professor, Sorbonne Université

Jury President

Bogdan MALAESCU, CNRS Researcher, LPNHE

Thesis Director

Rafael TEIXEIRA DE LIMA, Research Scientist, IBM Research

Thesis Co-Supervisor

Mélissa RIDEL, Professor, Sorbonne Université

Guest



Copyright:

Except where otherwise noted, this work is licensed under [CC BY-NC-ND 4.0 License](https://creativecommons.org/licenses/by-nc-nd/4.0/).

Abstract

The ATLAS experiment is one of the two multi-purpose particle collision detectors at the Large Hadron Collider (LHC) at CERN. It generates unprecedented amounts of data, recording proton-proton and nuclei collisions at rates up to 40 MHz. To properly exploit the data and measure the parameters of the Standard Model of particle physics, or even search for new phenomena, we need efficient and precise algorithms. This thesis investigates machine learning-based methods to address two central challenges in ATLAS data processing: jet energy calibration and data quality monitoring of the calorimeter system. These are important issues as they constitute some of the first bottlenecks to collecting high-quality data in hadron collider experiments.

First, we propose a unified approach to jet calibration by merging several steps of the current ATLAS calibration chain. We combine a neural network-based calibration with a loss function that optimises the jet energy resolution (JER) directly in data, using the dijet asymmetry as a proxy. This approach aims to reduce the resources required for deriving the final jet calibration while maintaining or improving the JER performance.

Second, we explore anomaly detection in multivariate time series for data quality monitoring. Accurately identifying the instances when defects occur is essential but challenging, as the types of anomalies are unknown beforehand and reliably labelled data are scarce. We evaluate unsupervised transformer-based models and benchmark their performance against traditional methods on public and ATLAS liquid argon calorimeter data. To address the lack of reliable labels, we use the Lorenzetti simulator – a general-purpose framework for simulating high-energy calorimeters – where we introduce artificial defects to evaluate the sensitivity of various detection methods. As part of the SMARTEP European Training Network, this project was done in collaboration with IBM Research, who share an interest in time series anomaly detection due to its potential applications in fraud detection.

These contributions highlight how modern machine learning techniques can enhance detector performance and data reliability in particle physics experiments while demonstrating broader interdisciplinary potential.

Résumé

ATLAS est un des deux détecteurs de particules polyvalents du Grand Collisionneur de Hadrons (Large Hadron Collider (LHC)) au CERN. En enregistrant des collisions proton-proton et des collisions de noyaux à des fréquences allant jusqu'à 40 MHz, le volume de données générées par ATLAS est sans précédent. Afin d'exploiter correctement ces données et de mesurer les paramètres du Modèle Standard de la physique des particules, mais aussi de rechercher de nouveaux phénomènes, nous avons besoin d'algorithmes efficaces et précis. Cette thèse étudie les méthodes basées sur l'apprentissage automatique pour relever deux défis centraux dans le traitement des données d'ATLAS : l'étalonnage de l'énergie des jets et le contrôle de la qualité des données du système calorimétrique. Ces questions sont importantes car elles constituent les premières difficultés de la collecte de données de haute qualité dans les expériences de collisionneurs de hadrons.

Premièrement, nous proposons une approche unifiée de l'étalonnage des jets en fusionnant plusieurs étapes de la chaîne d'étalonnage actuelle d'ATLAS. Nous combinons un étalonnage basé sur un réseau de neurones avec une fonction de perte qui optimise la résolution de l'énergie des jets directement dans les données, en utilisant l'asymétrie des dijets comme proxy. Cette approche vise à réduire les ressources nécessaires à la dérivation de l'étalonnage final des jets, tout en maintenant ou en améliorant la résolution de l'énergie des jets.

Deuxièmement, nous étudions la détection des anomalies dans les séries temporelles multivariées pour le contrôle de la qualité des données. L'identification précise des moments auxquels des défauts se produisent est essentielle mais difficile, car les types d'anomalies sont inconnus à l'avance et les données étiquetées de manière fiable sont rares. Nous évaluons des modèles non-supervisés basés sur les transformeurs et comparons leurs performances aux méthodes traditionnelles sur des données publiques et des données du calorimètre à argon liquide d'ATLAS. Pour adresser le manque d'étiquettes fiables, nous utilisons le simulateur *Lorenzetti* – un cadre général pour la simulation de calorimètres pour des particules de haute énergie – où nous introduisons des défauts artificiels pour évaluer la sensibilité et la robustesse de diverses méthodes de détection. Dans le contexte du projet Européen SMARTHEP, ce travail a été réalisé en collaboration avec IBM Research, qui partage un intérêt pour la détection des anomalies dans les séries temporelles en raison de ses applications potentielles à la détection des fraudes.

Ensemble, ces contributions soulignent comment des techniques modernes d'apprentissage automatique peuvent améliorer les performances des détecteurs et la fiabilité des données dans les expériences de physique des particules, tout en démontrant également leur potentiel interdisciplinaire plus large.

*as someone told me lately,
"everyone deserves the chance to fly"*

– Defying Gravity, Stephen Schwartz

Acknowledgments

There is a great number of people involved that led me to start, continue and actually conclude my PhD, whether that be through direct advice, subtle encouragements or technical discussions. All of them are incredibly valuable to me and I will try to express my gratitude for the most prominent ones, which doesn't make the others any less important, of course!

First and foremost, I would like to thank my supervisors for accompanying me on this journey, for guiding me and always having an open ear to discuss my many questions.

I had many changes in my supervision, but the person who stuck with me through my entire PhD and even travelled to Brazil with me is *Bogdan*, whom I want to thank a lot! I really appreciated our discussions and that you always tried your best to be available, no matter your workload.

Next, I want to say thank you to *Rafael*, who took over my supervision for my last PhD year despite his full work schedule, and whose inputs I valued enormously. It was a real pleasure working with you!

Big thanks also to *Shubham*, who guided me through the many challenges I faced during the first two years. I also want to thank *Christian* for guiding me through the first few months and *Mélissa* for helping us deal with doctoral school formalities.

I also thank *Ilyas Toumlilt*[\[1\]](#) for this thesis template.

A big thank you to the people I met at IBM: the research group, the colleagues that could be found at every social event (including the ping pong games during lunch break and the outdoor treasure hunt), in particular *Pierre*, *Jean-Christophe*, *Nicolas*, who helped me declare my (many) travel expenses, and *Xavier*, who even found the time to give me feedback on my defence slides! Of course, also huge thanks to *Marine* for all our conversations (on work and random topics), for taking me to dance class with you and for helping me prepare for my PhD defence!

I want to thank my friends and colleagues from ATLAS, that I had the pleasure of meeting during these three years. In particular, I am grateful for working with the JetEtmis group, for the hadronic calibration workshops, for the many exchanges with *Arnau*, for the ATLAS anomaly detection for DQM group, for the

discussions with *Vilius* and the ML for ATLAS initiatives; all of you played a huge part in motivating me and helping me tackle various technical challenges!

Special shout-out to the Lorenzetti group; I really enjoyed our collaboration and especially the unforgettable time in Salvador! Muito obrigada to *Edmar* and *Lucas* for the sightseeing and the Samba evenings!

To SMARTHEP, the European Training Network, that enabled this great adventure that my PhD was for me; to all the people involved in SMARTHEP that guided and encouraged us along the way: Thank you!

The same goes for my fellow SMARTHEP students, with whom every SMARTHEP week passed way too fast: I am really glad I got to meet you (even *Pratik*!).

Next, I want to emphasise how much I appreciated the atmosphere at LPNHE, more specifically, the people who made the lab an amazing place to work. In particular, I thank my thesis monitoring committee: *Pauline* and *Vava*, *Fred* for volunteering to read a first version of my ATLAS chapter, *Bertrand* for the DQM brainstorming sessions, *Anja*, for sharing technical and life advice, and *Daniel* for organising our amazing ski trip (where I finally got to discover the French Alps).

To all the other PhDs and Postdocs, thank you not just for being an amazing support group but also my friends! Special thanks to *Lavinia*, *Fotis*, *Pablo*, *Enya*, *Claudia*, *Line*, my office buddies *Anaëlle*, *Dylan* and *Leander*, also *Thierry*, *Renaud*, *Artur*, *Anastasiia*, *Chloé*, *Paul*, *Romain*, *Louis*, *William* and *Gonzalo*, and everybody else, who made these three years fun!

Finally, I don't think I can put into words how grateful I am for the people closest to me, but I'll still try.

I really want to thank my *parents*, my *brother* and my *nonna* from the bottom of my heart for always being there for me, no matter how close or far! Thanks for letting me rant when needed, but also for making me laugh and reminding me not to take life too seriously. Huge thanks also to *Thomas* and *Sandra* for coming to Paris for my defence, and to my extended family, thank you for all the love and support!

The very same goes for my longest and dearest friend: *Flavia*; I couldn't have done this without you (and I had the best time discovering Paris with you)! Same for *Nadine*: I am so glad we reconnected just before I started my PhD, and it still makes me laugh to think about our adventures in the south of France! And last but not least, a big merci to *Guillaume*; you were a very unexpected, but highly appreciated bonus of this PhD (and of course so much more)!

Funding Acknowledgments

This work is part of the SMARTHEP network and is funded by the European Union's Horizon 2020 research and innovation program, call H2020-MSCA-ITN-2020, under Grant Agreement n. 956086.



Contents

Abstract	iii
Résumé	v
Acknowledgments	viii
Funding Acknowledgments	ix
Contents	xiv
List of Figures	xviii
List of Tables	xx
List of Acronyms	xxi
Publications and Original Work	xxv
Introduction	1
I Theoretical and Experimental Background	3
1 Physics Background	4
1.1 Standard Model of Particle Physics	4
1.1.1 Quantum Chromodynamics	6
1.1.2 Open Questions	9
1.2 Jet Physics	10
1.3 Simulation	13
1.3.1 Hard Parton Scattering	13
1.3.2 Parton Showering and Hadronisation	15
1.3.3 Detector Simulation	16
2 The ATLAS Experiment at CERN	17
2.1 The Large Hadron Collider (LHC)	17
2.2 The ATLAS Detector	20

2.2.1	Inner Detector	22
2.2.2	Calorimeters	23
2.2.3	Muon Spectrometer	28
2.2.4	Trigger and Data Acquisition	28
2.2.5	Object Reconstruction	30
2.3	ATLAS Jet Reconstruction and Calibration	32
2.3.1	Jet Reconstruction	32
2.3.2	Jet Calibration	34
2.3.3	Dijet Events and Asymmetry	38
2.4	ATLAS Data Quality Monitoring (DQM)	39
2.4.1	Liquid Argon Calorimeter DQM	41
3	Machine Learning	44
3.1	Introduction to Machine Learning	44
3.1.1	Learning Paradigms	46
3.1.2	Loss Function	48
3.1.3	Generalisation, Overfitting and Underfitting	49
3.2	Machine Learning for Particle Physics	50
3.3	Deep Learning	52
3.3.1	Neural Networks	52
3.3.2	Learning Weights through Backpropagation	55
3.3.3	Model Architectures and Physics Applications	56
3.4	Anomaly Detection	62
3.4.1	Anomaly Detection in High Energy Physics	63
3.4.2	Performance Metrics	65
3.5	Issues and Challenges	66
II	Simultaneous Jet Calibration with Machine Learning Including <i>in situ</i> Jet Energy Resolution Measurement	68
4	Introduction	69
5	Input Data and Preprocessing	70
5.1	Monte Carlo Simulation Samples	70
5.2	Jet Selection Criteria	71
5.3	Scaling of Inputs	72
5.4	Normalisation of Targets	73
5.5	Flattening of the p_T Distribution	74
6	Machine Learning Model	76
6.1	Deep Sets for HEP	76
6.2	Regression Targets	77

6.3	Truth-Based Loss Function	78
7	Results Using Truth Information	79
8	Introducing Dijet Asymmetry	82
8.1	Asymmetry-Based Loss Function	82
8.2	Combined Loss Function	83
8.3	Implementation and Challenges	84
8.3.1	Unphysical Behaviour of Asymmetry Loss Function	84
8.3.2	Possible Constraint Terms	86
8.3.3	Alternative Loss Function	88
9	Results Using Truth and Truth-Independent Information	89
9.1	Results with Alternative Loss Function	89
9.1.1	Keeping Average Jet Energy Invariant	90
9.2	Truth-Independent Training	92
9.2.1	Results for JETM2 JZ7	93
9.2.2	Results for JETM2 JZ2	96
9.3	Partially Truth-Based Training	98
9.3.1	Results for JETM2 JZ7	99
9.3.2	Results for JETM2 JZ2	102
9.4	Iterative Approach	103
10	Summary and Outlook	107
III	Time Series Anomaly Detection (TSAD)	108
11	Introduction	109
12	Related Work	111
12.1	TSAD Algorithms	111
12.2	Benchmark Datasets	112
12.3	Evaluation Protocols	112
12.4	TSAD for Physics	113
13	Unsupervised Strategies for Multivariate TSAD	115
13.1	Datasets	115
13.1.1	Public Data for TSAD	115
13.1.2	Time Series Anomaly Types	116
13.2	Unsupervised Algorithms	117
13.2.1	Baseline Approach	118
13.2.2	Deep Learning Baseline Approach	119
13.2.3	Transformer-Based Anomaly Detection	119

13.3 Anomaly Label Extraction Strategies	121
13.3.1 Peak Over Threshold	121
13.3.2 Anomaly Labels	124
13.4 Experimental Setup	126
13.4.1 Sliding Windows	126
13.4.2 Detection Performance Metrics	126
13.4.3 Alternative Loss Functions	127
13.4.4 Model Configurations	127
13.5 Results	128
13.5.1 iTransformer Model Configurations	128
13.5.2 Studies on Anomaly Label Extraction	129
13.5.3 Influences of Anomalies in Training Data	131
13.5.4 Benchmarking Results for Transformer-Based TSAD	132
13.6 Summary and Outlook	136
14 Synthetic Data Generation with Lorenzetti for TSAD in HEP Calorimeters	137
14.1 Introduction	138
14.2 Lorenzetti Showers Simulation	138
14.3 Dataset with Synthetic Detector Defects	140
14.4 Data Preparation and Model Training	142
14.5 Results	148
14.6 Summary and Outlook	152
15 TSAD for Data Quality Monitoring in the ATLAS LAr Calorimeter	154
15.1 Introduction	154
15.2 Data Preparation and Model Training	154
15.3 Results	156
15.3.1 Cosmic Calo Stream	156
15.3.2 Main Stream	158
15.3.3 Hard Probes Stream	161
15.4 Summary and Outlook	163
Conclusion	164
Résumé Substantiel	167
Appendices	172
A Appendix on Jet Calibration	172
A.1 Jet Hierarchy	172
A.2 Feature Dependence	174

A.3	Additional Results	176
A.4	Iterative Approach: Additional Results	178
B	Appendix on TSAD	183
B.1	Descriptions of Datasets for TSAD	183
B.2	iTransformer Model Configurations	185
B.3	Results with Alternative Loss Functions	187
B.4	Model Configurations	189
B.5	Additional Results for TSAD on Lorenzetti Data	191
	Bibliography	194

List of Figures

1.1	The fundamental particles described by the Standard Model (SM) of particle physics.	5
1.2	The three fundamental Quantum Chromodynamics (QCD) interaction vertices.	7
1.3	Evaluations of α_S as a function of the energy scale Q	8
1.4	Schema of jet formation in proton-proton collisions.	10
1.5	Schema of truth and reconstructed jets.	11
1.6	Schema of the anti- k_t algorithm.	12
1.7	SM cross section measurements for the Large Hadron Collider (LHC).	13
1.8	Schema of a $pp \rightarrow t\bar{t}$ event simulated by PYTHIA.	14
1.9	Illustration of the string potential between a $q\bar{q}$ pair in (a) and the string breaking by $q\bar{q}$ pair creation in (b).	15
2.1	CERN accelerator complex, including various physics experiments.	18
2.2	Average number of interactions per bunch-crossing (averaged over luminosity) $\langle\mu\rangle$ and most probable value for the number of interactions μ_{MPV} for the three data taking periods.	20
2.3	Schema of the ATLAS detector at the LHC.	21
2.4	Schema of the ATLAS calorimeters.	23
2.5	Schema of the ATLAS calorimeters along the $y - z$ plane with pseudorapidity η for reference.	24
2.6	Coverage plot showing the cell energy sums of the Liquid Argon (LAr) calorimeter distributed in a hypothetical tower grid for 137 proton-proton collision events at $\sqrt{s} = 13.6$ TeV in run 427394 from 5-6 July 2022.	25
2.7	Pulse shapes and signals for the LAr calorimeter.	26
2.8	Schematic slice of the ATLAS detector and how particles interact with the detector layers.	30
2.9	Schema of the Particle Flow (PFlow) algorithm.	33
2.10	Schema of the ATLAS calibration chain and potential shortcuts with a Machine Learning (ML)-based jet calibration.	35
2.11	Schema of operations workflow for the ATLAS data quality assessment.	40
2.12	Data rejection by defect assignment for the datasets of run 2 for the most optimal processing.	41

2.13	Example of a typical noise burst in the LAr calorimeter.	42
2.14	Distributions characterising noise bursts.	43
3.1	Schema of Artificial Intelligence (AI) and its sub-fields.	44
3.2	Illustration of data sizes for several big data players for data stream- ing, storage and production.	50
3.3	The evolution of ML papers in the High Energy Physics (HEP) community over the years.	51
3.4	Schema of a Neural Network (NN) with k hidden layers of variable dimensions $\{m_1, m_2, \dots, m_k\}$	53
3.5	Popular activation functions for a NN.	54
3.6	Gradient-descent technique visualised in a two-dimensional hyper- space.	56
3.7	Schema of deep sets architecture.	57
3.8	Schema of Autoencoder (AE) architecture including encoder and decoder.	58
3.9	Schema of Recurrent Neural Network (RNN) architecture visualised folded (left) and unfolded (right).	60
3.10	Schema of transformer architecture.	61
5.1	Histograms of input p_T distribution for the two Monte Carlo (MC) samples before any selection cuts.	71
5.2	Histograms of input p_T distribution for the two MC samples used as inputs for the calibration model after selection cuts.	72
5.3	Schema of input data and their scaling.	73
7.1	Results after ML training with Mean Squared Error (MSE) loss from Eq.(6.6) comparing true and predicted calibration factors.	81
8.1	Schema illustrating inputs and workflow of the ML jet calibration model.	84
8.2	Results after ML training only with asymmetry-based loss (from Eq.(8.3)).	85
9.1	Results after ML training with $f_t = f_r = 0$ and $f_a = 1$	91
9.2	Mean response and resolution per bin after re-scaling calibration factors.	92
9.3	Training loss functions and their components for each combination of hyperparameters f_a and $f_r = 1 - f_a$	94
9.4	Final training loss terms for each hyperparameter configuration shown as a function of f_a for JETM2 JZ7.	95
9.5	Results of standard deviation of dijet asymmetry as a function of f_a using JETM2 JZ7 samples.	96

9.6	Final training loss terms for each hyperparameter configuration shown as a function of f_a for JETM2 JZ2.	97
9.7	Results of standard deviation of dijet asymmetry as a function of f_a using JETM2 JZ2 samples.	98
9.8	Final training loss terms for each hyperparameter configuration shown as a function of f_a for JETM2 JZ7.	99
9.9	Training loss functions and their components for each combination of hyperparameters f_a and $f_t = 1 - f_a$ for JETM2 JZ7.	100
9.10	Results of standard deviation of dijet asymmetry as a function of f_a using JETM2 JZ7 samples.	101
9.11	Final training loss terms for each hyperparameter configuration shown as a function of f_a for JETM2 JZ2.	102
9.12	Results of standard deviation of dijet asymmetry as a function of f_a using JETM2 JZ2 samples.	103
9.13	Schema of iterative approach of jet calibration with the previously introduced ML model (see Section 8.3.3).	104
9.14	Iterative approach with two iterations using JETM2 JZ7 samples with $f_a = 0.1, f_t = 0.9$	105
9.15	Iterative approach with two iterations using JETM2 JZ2 samples with $f_a = 0.1, f_t = 0.9$	106
13.1	Taxonomy for anomalies in time series.	117
13.2	Schema of reconstruction-based Time Series Anomaly Detection (TSAD).	118
13.3	Comparison of the time series encoding for the (vanilla) transformer and the iTransformer.	121
13.4	Schema of the Peak-Over-Threshold (POT) method.	123
13.5	Illustration of different anomaly label extraction methods for multivariate time series.	125
13.6	Sliding window protocol for training and testing.	126
13.7	The accuracy of anomaly predictions for each of the anomaly label extraction methods and for different anomaly lengths l	130
13.8	The false positive rate for each of the anomaly label extraction methods and for different lengths of non-anomalous sequences l	131
14.1	Cross-sectional view of the general-purpose calorimeter modelled by the Lorenzetti Showers framework.	139
14.2	Schema of the Lorenzetti Showers framework, including the insertion of artificial detector defects during the digitisation step.	140
14.3	Phase (η, ϕ) space shows energy values for simulated detector cells after the digitisation step and aggregated over 10 events.	141

14.4	Distribution plots with dead detector modules and without pileup compared to normal feature distributions.	145
14.5	Distribution plots with increased noise, using noise factor 20, and without pileup for various anomaly rates.	146
14.6	Distribution plots with increased noise, using noise factor 20, and with pileup $\langle\mu\rangle = 40$ for various anomaly rates.	147
14.7	Applying TranAD to test data produced with Lorenzetti, which contains artificial anomalies associated with inactive detector modules.	150
14.8	Matthews Correlation Coefficient (MCC) scores as a function of the anomaly rate in the test dataset with and without pileup.	151
15.1	Applying iTransformer-reco to a run from the Cosmic Calo stream, which is different from the streams used for training.	157
15.2	Applying iTransformer-reco to a run from the main stream with a High Voltage (HV) issue shortly after 10'000 events.	159
15.3	Applying iTransformer-reco to a run from the main stream with an issue shortly before 1'000 events that has not been detected by the standard algorithm.	160
15.4	Applying iTransformer-reco to a run from the hard probes stream.	162
A.1	Results with loss function from Eq.(8.12) with $f_t = 0, f_a = f_r = 1$ and additional Global Sequential Calibration (GSC) variables as input.	173
A.2	Results with loss function from Eq.(8.12) with $f_t = 0, f_a = f_r = 1$ and additional GSC variables as input after fixing the jet hierarchy issue.	174
A.3	Permutation feature importance.	175
A.4	Training loss functions and their components for each combination of hyperparameters f_a and $f_r = 1 - f_a$	176
A.5	Training loss functions and their components for each combination of hyperparameters f_a and $f_t = 1 - f_a$	177
A.6	Iterative approach with two iterations using JETM2 JZ7 samples with $f_a = f_t = 0.5$	179
A.7	Iterative approach with two iterations using JETM2 JZ2 samples with $f_a = f_t = 0.5$	180
A.8	Iterative approach with two iterations using JETM2 JZ7 samples with $f_a = 0.9, f_t = 0.1$	181
A.9	Iterative approach with two iterations using JETM2 JZ2 samples with $f_a = 0.9, f_t = 0.1$	182

List of Tables

2.1	List of variables used as input for the Global Neural Network Calibration (GNNC).	37
5.1	Variables used as inputs for ML-based calibration model.	70
5.2	Selection criteria applied to dijet events.	72
5.3	Mean μ_c and standard deviation σ_c of the calibration targets c_{true} for different training sets.	74
13.1	Public datasets used for TSAD.	116
13.2	Anomaly label extraction producing the best results for reconstruction- or forecasting-based iTransformer models.	129
13.3	MCC and F1 scores (averaged over five random seeds) for different loss functions during training, with and without anomalies in training data.	132
13.4	Best MCC and F1 scores (averaged over five random seeds) for forecasting-based, reconstruction-based iTransformer, vanilla Transformer, AnomalyTransformer, TranAD and USAD model on various datasets. The baseline is deterministic, so only a single score is reported. The letters in the brackets denote the anomaly label extraction method. For the univariate time series SWaT_1D and UCR, all anomaly label extraction methods yield the same results.	135
14.1	List of shower shape variables.	143
14.2	MCC scores for reconstruction-based iTransformer, TranAD and USAD models on test datasets generated with Lorenzetti, under varying pileup conditions, anomaly types, and anomaly rates.	149
A.1	Configurations and observations when studying feature dependence.	174
B.1	Best iTransformer configurations for reconstruction-based anomaly detection for each dataset.	186
B.2	Best iTransformer configurations for forecasting-based anomaly detection for each dataset.	186

B.3	MCC scores (averaged over five random seeds) for reconstruction-based iTransformer using either the mean-squared error (MSE), Huber loss or the soft dynamic time warping (Soft-DTW) loss during training and the MSE during testing. For the univariate time series SWaT_1D and UCR, all anomaly label extraction methods yield the same results.	188
B.4	Number of variates N and trainable parameters for forecasting-based, reconstruction-based iTransformer, vanilla Transformer, AnomalyTransformer, TranAD and USAD model on various datasets. . .	190
B.5	MCC scores for reconstruction-based iTransformer, TranAD and USAD model on datasets generated with Lorenzetti without pileup.	191
B.6	MCC scores for reconstruction-based iTransformer, TranAD and USAD model on datasets generated with Lorenzetti with pileup. .	192

List of Acronyms

ACAT	Advanced Computing and Analysis Techniques in Physics Research
AE	Autoencoder
AI	Artificial Intelligence
AIHENP	International Workshop on Software Engineering, Artificial Intelligence and Expert Systems for High-energy and Nuclear Physics
ANTELOPE	ANomaly deTEction on particLe fLOW latent sPacE
AOD	Analysis Object Data
AUC	Area Under Curve
AXOL1TL	Anomaly eXtraction Online Level-1 Trigger aLgorithm
BDT	Boosted Decision Tree
BSM	Beyond the Standard Model
CATHODE	Classifying Anomalies THrough Outer Density Estimation
CERN	European Organization for Nuclear Research
CHS	Charged Hadron Subtraction
CICADA	Calorimeter Image Convolutional Anomaly Detection Algorithm
CNN	Convolutional Neural Network
CWoLa	Classification Without Labels
DAOD	Derived Analysis Object Data
DL	Deep Learning
DQM	Data Quality Monitoring
DTW	Dynamic Time Warping
ELU	Exponential Linear Unit
EMB	Electromagnetic Barrel
EMEC	Electromagnetic End-Cap
ESD	Event Summary Data
EuCAIF	European Coalition for AI in Fundamental Physics

EVT	Extreme Value Theory
FCal	Forward Calorimeters
FEB	Front End Board
FN	False Negative
FP	False Positive
FPGA	Field Programmable Gate Array
FSR	Final State Radiation
GAN	Generative Adversarial Network
GNN	Graph Neural Network
GNNC	Global Neural Network Calibration
GPD	Generalised Pareto Distribution
GRL	Good Runs List
GSC	Global Sequential Calibration
HEC	Hadronic Endcap
HEP	High Energy Physics
HL-LHC	High-Luminosity Large Hadron Collider
HLT	High Level Trigger
HV	High Voltage
IF	Isolation Forest
iid.	independent and identically distributed
IoT	Internet of Things
IRC	Infrared and collinear
ISR	Initial State Radiation
iTK	Inner Tracker
JER	Jet Energy Resolution
JES	Jet Energy Scale
KL	Kullback-Leibler
kNN	k-Nearest Neighbour
L1	Level 1
LAr	Liquid Argon
LB	Luminosity Block
LCW	Local Weighted Cell
LHC	Large Hadron Collider
LLM	Large Language Model

LSTM	Long-Short Term Memory
MC	Monte Carlo
MCC	Matthews Correlation Coefficient
ML	Machine Learning
MLP	Multilayer Perceptron
MPI	Multiple Parton Interaction
MSE	Mean Squared Error
NLP	Natural Language Processing
NN	Neural Network
OCSVM	One-Class Support Vector Machine
PCA	Principal Component Analysis
PDF	Parton Distribution Function
PFlow	Particle Flow
PMT	Photomultiplier Tube
POT	Peak-Over-Threshold
PV	Primary Vertex
QCD	Quantum Chromodynamics
QED	Quantum Electrodynamics
QFT	Quantum Field Theory
reco	reconstructed
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
ROC	Receiver Operator Characteristic
SCT	Semi-Conductor Tracker
SM	Standard Model
Soft-DTW	Soft Dynamic Time Warping
SVM	Support Vector Machine
TCC	Track Calorimeter Cluster
TDAQ	Trigger and Data Acquisition
TN	True Negative
TP	True Positive
TRT	Transition Radiation Tracker
TSAD	Time Series Anomaly Detection
UFO	Unified Flow Object

UPC	Ultra-Peripheral Collisions
VAE	Variational Autoencoder
WLCG	Worldwide LHC Computing Grid

Publications and Original Work

Publications

During the PhD, I contributed to the following publications, workshops and conferences.

Journals

- ‘*Unravelling Physics beyond the Standard Model with Classical and Quantum Anomaly Detection*’, Machine Learning: Science and Technology, Nov 2023 [2].

Preprints

- ‘*Benchmarking Unsupervised Strategies for Anomaly Detection in Multivariate Time Series*’, arXiv, Jun 2025 [3].
- ‘*Review of Machine Learning for Real-Time Analysis at the Large Hadron Collider Experiments ALICE, ATLAS, CMS and LHCb*’, arXiv, Jun 2025 [4]. Currently under review for a journal.

International Conferences and Workshops

- ‘*Synthetic Data Generation with Lorenzetti for Time Series Anomaly Detection in High-Energy Physics Calorimeters*’, The 2nd European AI for Fundamental Physics Conference (EuCAIFCon 2025), Cagliari, Jun 2025 [5, 6]. Proceedings will be published in *SciPost Physics Proceedings*.
- ‘*Unsupervised Anomaly Detection in Multivariate Time Series Using Public Benchmarks and Synthetic Data from Lorenzetti*’, COMETA Workshop on Uncertainty Quantification for Machine Learning (UQ4ML), Saclay, Sep 2025 [7].

ATLAS Workshops

- ‘*Simultaneous Jet Calibration with ML including in situ JER Measurement*’, Lightning Talk, ATLAS Hadronic Calibration Workshop 2023, Valencia, Sep 2023.
- ‘*Jet Energy Calibration and Resolution Optimisation with in-situ ML Training*’, ATLAS Machine Learning Workshop, CERN, May 2024.
- ‘*Jet Energy Calibration and Resolution Optimisation with in-situ ML Training*’, ATLAS Hadronic Calibration Workshop 2024, Ottawa, Aug 2024.
- ‘*Time Series Anomaly Detection for DQM in ATLAS*’, ATLAS Machine Learning Workshop, CERN, Apr 2025.
- ‘*Time Series Anomaly Detection for DQM in ATLAS and Lorenzetti*’, Physique ATLAS France, Annecy, May 2025.

Communications and Seminars

- ‘*Time Series Anomaly Detection*’, ATLAS Machine Learning Forum on Anomaly Detection, online, Dec 2024.
- ‘*Time Series Anomaly Detection*’, PhD Student’s Seminar at APC, Université Paris-Cité, Paris, Dec 2024.
- ‘*Time Series Anomaly Detection for DQM in ATLAS and Lorenzetti*’, Student’s Seminar, Universidade Federal da Bahia (UFBA), Salvador, May 2025.
- ‘*Time Series Anomaly Detection for High Energy Physics and Fraud Detection*’, SMARTHEP Meets Industry, CERN, May 2025 [8].

Original Work

My original contributions are described in Part II and Part III.

The work in Part II constitutes my ATLAS Authorship Qualification Project, developed in collaboration with ATLAS experts, particularly from LPNHE and the *in-situ* jet calibration group.

Part III addresses time series anomaly detection, in collaboration with IBM Research and ATLAS, applying the same methods to different contexts. Researchers from ATLAS Brazil were heavily involved in the synthetic data generation.

In parallel, I contributed to the SMARTHEP network paper on applications of machine learning for real-time analysis at the LHC.

I confirm that this document is entirely my own work.

Introduction

‘New directions in science are launched by new tools much more often than by new concepts. The effect of a concept-driven revolution is to explain old things in new ways. The effect of a tool-driven revolution is to discover new things that have to be explained.’

– Freeman Dyson

As [Machine Learning \(ML\)](#) is gaining importance worldwide, it is also establishing itself as an essential tool for data science and analysis across applications. Meanwhile, experiments in [High Energy Physics \(HEP\)](#) are dealing with unprecedented amounts of data, which allow them to conduct searches with statistical precision that was never seen before. However, as the data keeps growing, more efficient and precise algorithms are desperately needed to maintain the data quality and make progress on unveiling new processes in physics. Especially with the arrival of the [High-Luminosity Large Hadron Collider \(HL-LHC\)](#), where the collision rate is expected to be up to five times higher, the data-taking and processing rates are expected to shoot up to unrivalled levels.

The [Standard Model \(SM\)](#) of particle physics, so far, has provided an excellent description of fundamental particles and their interactions, and its predictions have been confirmed repeatedly through several precision measurements. The last missing puzzle piece, needed to experimentally confirm the [SM](#), had been the existence of a scalar particle, called the Higgs particle. With its experimental discovery in 2012 by the ATLAS and CMS experiments at the [Large Hadron Collider \(LHC\)](#) [9, 10], a long series of fundamental particle discoveries – derived from theoretical [SM](#) predictions – was finally concluded.

Despite its success, the [SM](#) fails at explaining phenomena such as dark matter or dark energy, or experimental observations such as non-zero neutrino masses; all of which hint at the existence of new fundamental particles or laws of physics. As currently no clear direction for new physics theories is apparent, the best approach is to search the data for unexpected occurrences or unusual measurements that deviate from theoretical predictions. Instead of relying on theory, as done so far, data-driven strategies might identify new, promising research directions that have not been thought of yet. [ML](#) provides a crucial tool to implement such searches while maintaining the quality of the data and potentially improving the precision of the physics analyses based on these data. As seen before, with new technologies allowing for ground-breaking discoveries and

breakthroughs in physics, the idea is that in the current era of big data, [ML](#) can be the tool needed to fully exploit the information contained within the huge amounts of collected physics data.

Similarly to [HEP](#), also other fields in fundamental physics such as gravitational waves, cosmology, nuclear physics or astrophysics, have started to employ [ML](#) for accelerating or improving challenging data analysis tasks. As these fields share many of their needs and challenges, interdisciplinary solutions and collaborations are more important than ever.

This thesis is organised as follows: Part [I](#) provides the necessary theoretical and experimental background to delve into the projects presented later and consists of the following chapters: Chapter [1](#) gives an introduction to [HEP](#) with a focus on jet physics, including the importance of simulation. Chapter [2](#) presents the ATLAS experiment at the [LHC](#), as a member of whose collaboration I carried out this thesis. Chapter [3](#) provides an extended introduction to [ML](#), presenting various examples of applications in [HEP](#).

In Part [II](#), we propose tackling jet calibration with a deep sets [ML](#) model which allows merging several sequential calibration steps, while trying to maintain or further improve the [Jet Energy Resolution \(JER\)](#).

Within the ATLAS experiment at the [LHC](#), the quality of the recorded data is verified in various stages. An essential task consists in the energy calibration of jets, which are particle-like objects corresponding to the showers of particles produced by the hadronisation of strongly interacting particles such as quarks and gluons. They are the most produced objects at the [LHC](#) and an essential part for many downstream physics analyses. Therefore, the precise calibration of jet energies is crucial for high-quality physics results.

Part [III](#) focuses on applications of [Time Series Anomaly Detection \(TSAD\)](#) for various monitoring tasks.

Data quality can also be ensured by identifying and filtering out events affected by detector defects. This is, for instance, done during [Data Quality Monitoring \(DQM\)](#) procedures at the ATLAS experiment.

In Chapter [13](#), various unsupervised [TSAD](#) methods are studied and benchmarked on publicly available data. In a next step, the developed methods are applied to simulated calorimeter data (Chapter [14](#)), where artificial detector defects had been carefully injected, and to real ATLAS calorimeter data (Chapter [15](#)), where the proposed methods even identified a detector issue that had been missed by the standard approach.

Part I

Theoretical and Experimental
Background

Physics Background

Particle physics focuses on finding and describing the most fundamental constituents of matter and their interactions with each other. The [SM](#) of particle physics embodies the current theoretical understanding of this field. It is a remarkable achievement of the 20th century and is one of the most successful theories in physics. Some of its parameters like the electron magnetic moment have been measured with unmatched precision [\[11\]](#), making the [SM](#) also one of the best experimentally tested theories in physics.

This chapter introduces the necessary theoretical concepts to explain and motivate the physics projects described in this thesis.

1.1 Standard Model of Particle Physics

The [SM](#) of particle physics is based on [Quantum Field Theory \(QFT\)](#), which combines the theories of special relativity and quantum mechanics. Each fundamental interaction corresponds to a quantum field (*gauge bosons*) that couples with fundamental particles (*fermions*). It is based on the *gauge invariance principle* which states that every interaction is invariant under a *gauge symmetry*, i.e. a local and continuous symmetry. Therefore, each quantum field must satisfy a corresponding gauge symmetry. Based on [QFT](#), the [SM](#) unifies three out of four fundamental forces, and their combination is associated with a $SU(3) \times SU(2) \times U(1)$ gauge symmetry.

Electromagnetic Interaction The electromagnetic interaction describes the interaction between electrically charged particles by the exchange of *photons*. It is invariant to a $U(1)$ gauge symmetry. The [QFT](#) focusing solely on such interactions is denoted as [Quantum Electrodynamics \(QED\)](#).

Weak Interaction The weak interaction manifests in nuclear β -decays and plays a key role in nuclear fusion and the decay of fission fragments. It corresponds to a $SU(2)$ gauge symmetry and the associated gauge bosons are the charged W^+ , W^- and the neutral Z^0 bosons.

Strong Interaction The strong interaction is responsible for the forces holding together the atomic nuclei. It is invariant under a $SU(3)$ gauge symmetry. The interaction is mediated by eight gauge bosons (so-called *gluons*) and the fermions interacting with them are the so-called *quarks*. The theory of strong interactions is commonly referred to as [Quantum Chromodynamics \(QCD\)](#) and is discussed further in Section 1.1.1.

The first two interactions can be unified into the *electroweak interaction*. The resulting interaction is associated with a $SU(2) \times U(1)$ gauge symmetry, which is spontaneously broken. This spontaneous symmetry breaking implies the existence of three massive gauge bosons ($W^\pm$ and Z_0) and a massless, spin-0 gauge boson, the so-called *Higgs boson*. Initially, all fermions and bosons in the [SM](#) are massless, only their interaction with the Higgs field attributes them a non-zero mass. Its existence has been theoretically predicted since the 1960s but it was only experimentally discovered in 2012 by the ATLAS and CMS experiments at the [LHC](#) [9, 10].

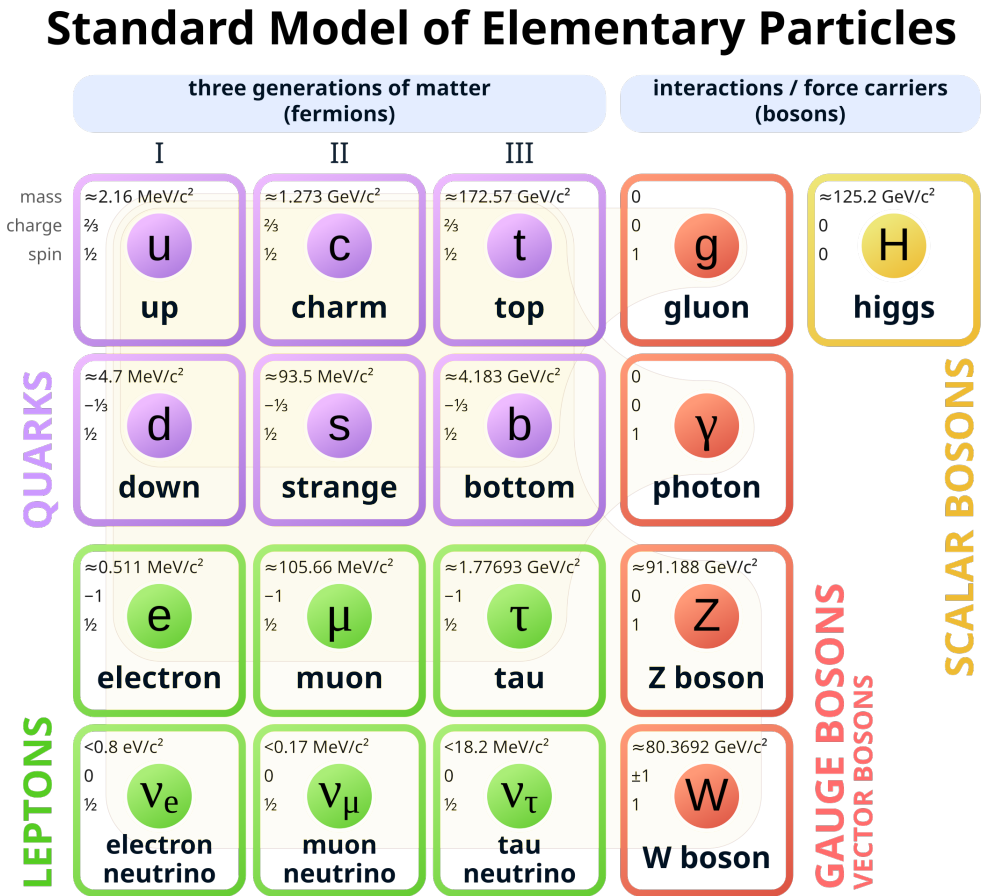


Fig. 1.1.: The fundamental particles described by the [SM](#) of particle physics. Figure from [12].

The fundamental particles that are currently known and described by the [SM](#), are shown in Figure 1.1. Bosons always have an integer spin, whereas spins of fermionic particles are multiples of $1/2$. There are three generations of particles, corresponding to the first three columns in Figure 1.1. The particles of the different generations are almost identical except for their masses, which increase with the generations [13]. The anti-particles of each fundamental particle are also described by the [SM](#); they share the same properties as their corresponding particles but with opposite quantum numbers.

1.1.1 Quantum Chromodynamics

[QCD](#) is the theory of strong interactions, which is invariant under $SU(3)$ local phase transformations. The strong force is mediated by eight gauge bosons (because $SU(3)$ has dimension eight), the *gluons*, and acts on six fermions (plus their anti-particles), the *quarks*. [QCD](#) is a *non-abelian* gauge theory, meaning that the Lie group generators t^a are not commutative, which makes it fundamentally different from [QED](#). Instead of just one conserved charge (electric) as in [QED](#), [QCD](#) has three conserved ‘colour’ charges, which can be either blue, red or green. A particle is neutral in [QCD](#) if it is colourless, which can be achieved by combining particles of all three colours or a particle with its anti-particle (that has the ‘anti-colour’). All particles interacting via the strongly (i.e. quarks and gluons) are called *partons*. Particles composed of multiple partons are referred to as *hadrons*.

The [QCD](#) Lagrangian corresponds to:

$$\mathcal{L}_{QCD} = \bar{\psi} i \gamma^\mu (\partial_\mu + i g_s t^a G_\mu^a) \psi - \frac{1}{4} G_{\mu\nu}^a G^{a,\mu\nu}, \quad (1.1)$$

$$\text{with } G_{\mu\nu}^a = \partial_\mu G_\nu^a - \partial_\nu G_\mu^a - g_s f^{abc} G_\mu^b G_\nu^c, \quad (1.2)$$

where ψ are the triplet wave functions (quarks), G_μ^a the gauge fields (gluons), $G_{\mu\nu}^a$ the field strength tensor, t^a the $SU(3)$ generators, f^{abc} the structure constants of $SU(3)$ and g the [QCD](#) coupling constant. The structure constants are defined by the Lie algebra $[t^a, t^b] = i f_{abc} t^c$. This is called a *non-abelian* Lie algebra because the group generators do not commute. Another important variable is the coupling $\alpha_S = g_S^2/(4\pi)$, which quantifies the strength of the strong interaction between partons. Its actual value depends on the energy scale of the interacting particles.

The fundamental [QCD](#) interaction vertices are shown in Figure 1.2. The second term of the QCD Lagrangian (also called the *Yang-Mills Lagrangian*) allows for the existence of vertices coupling together three or four gluons (drawn

in Figure 1.2b and Figure 1.2c). This implies that it is possible for gluons to interact with themselves, which is a fundamental difference compared to abelian gauge theories. For instance, in QED, it is impossible for photons to engage in self-interaction, while light-by-light processes can be produced through loop diagrams.

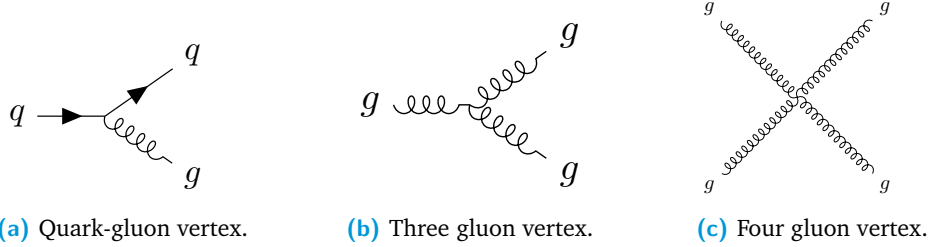


Fig. 1.2.: The three fundamental QCD interaction vertices.

To solve QCD, there are two principal approaches: *Perturbation theory* and *lattice QCD*.

Perturbative QCD is only applicable at energy scales μ satisfying $\alpha_S(\mu^2) \ll 1$, as it relies on order-by-order expansion, where terms are weighted by increasing powers of α_S . The cut-off scale Λ_{QCD} is the energy scale at which the condition $\alpha_S(\mu^2) \ll 1$ is no longer satisfied. Depending on the energy scale, at which perturbation theory is applied, more or less higher-order terms need to be taken into account to obtain precise predictions. Figure 1.3 shows the experimental measurements of the running of α_S for different energy scales and what degree of higher order computations had to be considered for perturbative QCD.

The second approach entails the discretisation of space-time and considering the values of quark and gluon fields on the vertices of this 4D lattice. This method has provided interesting results for a number of particle physics studies but is computationally very expensive. For instance, to resolve interactions at high-energy colliders such as the LHC more than 10^{34} nodes would be needed. Furthermore, it is challenging to employ lattice QCD for processes with high particle multiplicities, as expected for such a collider. Therefore, lattice QCD is not applicable for high-energy reactions as the ones discussed in the following [13–17].

Colour Confinement Colour confinement describes the phenomenon that quarks or gluons are not observed individually in nature. They are only observed in bound, colourless states such as baryons (bound states with three quarks) or mesons (bound states with a quark and anti-quark). By consequence, only states with a zero colour charge can propagate as free particles. There is no exact analytic proof describing this. However, intuitively, this can be best described with the example of pulling apart two quarks. In that case, the interaction

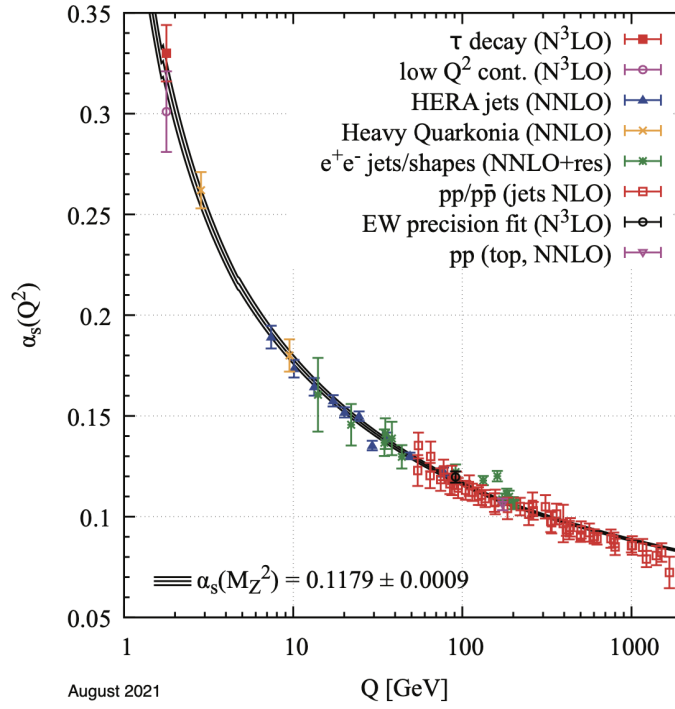


Fig. 1.3.: Summary of evaluations of α_S as a function of the energy scale Q . The respective degree of perturbative QCD used in the extraction of α_S is indicated in brackets (NLO: next-to-leading order; NNLO: next-to-next-to-leading order; NNLO+res.: NNLO matched to a resummed calculation; N3LO: next-to-NNLO). Figure from [18].

between the two quarks is given by the exchange of virtual gluons. Since gluons have a colour charge, they are attracted by themselves. This makes the colour field between the two quarks become ‘tube’-like and the energy stored in the field increases linearly with the distance between the two quarks. Hence, to separate the two quarks completely, an infinite amount of energy would be needed. This approximation is referred to as the *string model* [19], where the potential is assumed to be linear and quantified by $V(r) = \kappa r$ with the string tension $\kappa \approx 1 \text{ GeV}$ and r the distance between the quarks [13, 14].

Asymptotic Freedom The strength of the QCD coupling α_S varies significantly across the energy scales relevant to particle physics (see Figure 1.3) due to the running of the coupling constant. As the energy decreases, α_S grows asymptotically. For partons with an energy $|q| \sim 1 \text{ GeV}$, the coupling becomes strong ($\alpha_S \sim 1$) and perturbative approaches cannot be used, since higher order terms, proportional to powers of α_S , are not negligible. In contrast, for partons with higher energies ($|q| > 100 \text{ GeV}$), the coupling decreases ($\alpha_S \sim 0.1$) and perturbation theory becomes applicable (because $\alpha_S < 1$). At very high energies, α_S tends to zero and partons behave as if they were free, which is called asymptotic freedom.

Mathematically, the running of the coupling constant can be derived from renormalisation. This is done to cope with ultraviolet divergences from loop diagrams.

1.1.2 Open Questions

Despite its remarkable success at explaining and predicting experimental observations, the [SM](#) leaves a long list of questions unanswered and even raises some more. Theories that try to extend or replace the [SM](#) are called [Beyond the Standard Model \(BSM\)](#) theories. Some of the most pertinent questions that [BSM](#) theories try to answer are[[13](#), [20](#)]:

- **Dark Matter:** What is dark matter and what does it consist of?
- **Matter-Antimatter Asymmetry:** Why do we observe so much matter than antimatter everywhere? What causes this imbalance?
- **Quantisation:** Why are charges quantised despite the absence of such a constraint in the [SM](#)?
- **Neutrino Masses:** Why are neutrino masses so small and are there more neutrinos than the ones we currently know?
- **Gravity:** How can gravity be integrated into our model of the universe? Is there a grand unified theory that describes all fundamental interactions?
- **Hierarchy Problem:** Why is the Higgs particle relatively light?

There exist several [BSM](#) theories which are often based on the existence of new gauge symmetries or string theory. However, up to date no predictions derived from these theories have been experimentally confirmed. For this reason, there is currently no dominating theoretical framework for [BSM](#) physics which makes anomaly detection one of the main focuses of [HEP](#) research. Significant deviations from the [SM](#) can indicate the existence of new physics and help develop a [BSM](#) theory which agrees with experimental observations[[20](#)].

These are the main motivations for further improving data acquisition, processing, and analysis in particle physics experiments, as presented in this thesis.

1.2 Jet Physics

Jets correspond to the spray of particles produced by the hadronisation of strongly interacting particles such as quarks and gluons.

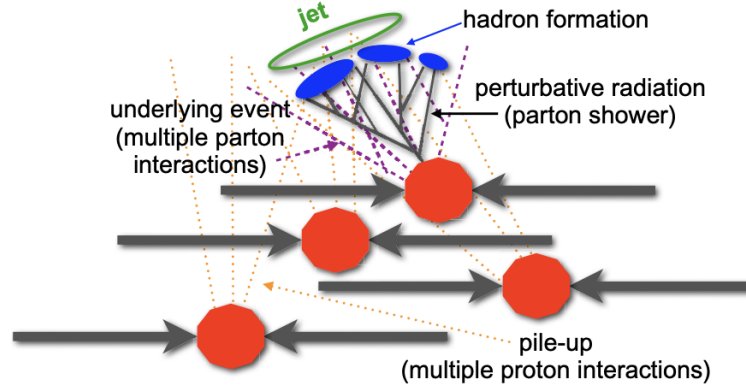


Fig. 1.4.: Schema of jet formation in proton-proton collisions. Figure from [21].

Concretely, highly energetic quarks or gluons, originating from proton-proton collisions, are considered asymptotically free and their interactions can be described by perturbative QCD. They radiate partons, which gradually reduces their energy. This phenomenon – called a *parton shower* – continues until the partons reach the hadron formation scale, where confinement forces the partons to group into hadrons (*hadronisation*). Due to the properties of QCD, the radiative splittings are more probable to happen at small angles, resulting in a collimated spray of quarks and gluons, which is named a *jet*.

The formation of jets is schematically represented in Figure 1.4. Individual jets can be schematically represented as shown in Figure 1.5.

Jets are particularly interesting as they function as a natural connection between theory and experiment by providing a common definition for partons described by perturbative QCD and clusters of energy deposits in particle detectors. Their definition is fundamentally ambiguous because they link coloured partons to uncoloured objects. Therefore, the exact definition of a jet depends on the *jet algorithm* which is used to cluster the interacting particles [21].

In the following paragraphs, we introduce the key definitions related to jets that are essential for the context of this thesis.

Jet Algorithm In the context of LHC experiments (see Section 2.1), the dominant jet algorithm is the so-called anti- k_t algorithm [23], which is a *sequential clustering algorithm* that starts by grouping the most energetic particles together and then iteratively adds softer particles. Clustering algorithms used to be

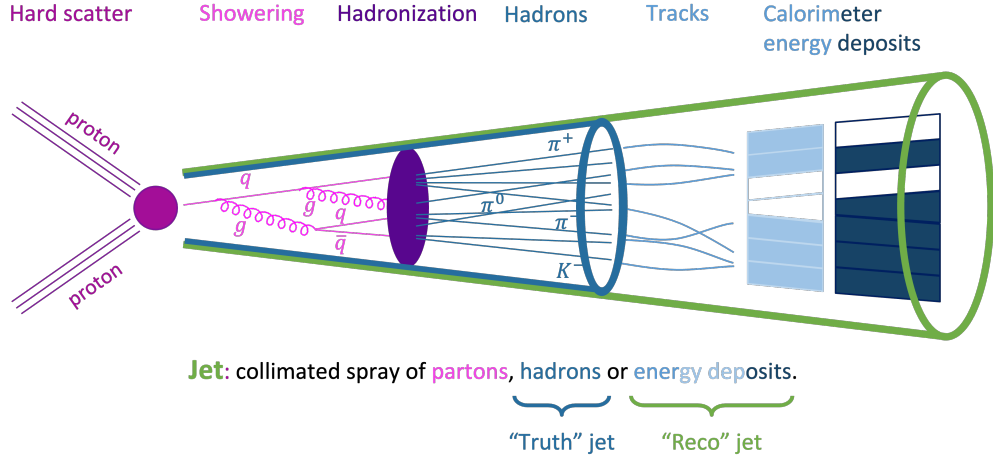


Fig. 1.5.: Schematic representation of a jet distinguishing truth and reconstructed jets. Inspired by [22].

unpopular within the experimental physics community due to their slow computational performance. However, this problem was solved with the introduction of the FastJet [24] program and clustering algorithms have become the standard approach for jet definition.

In the case of the anti- k_t algorithm, the distance measures are defined as:

$$d_{ij} = \min\left(\frac{1}{p_{T_i}^2}, \frac{1}{p_{T_j}^2}\right) \times \frac{R_{ij}^2}{R}, \quad (1.3)$$

$$d_{iB} = \frac{1}{p_{T_i}^2}, \quad (1.4)$$

where $R_{ij} = \sqrt{(\eta_i - \eta_j)^2 + (\phi_i - \phi_j)^2}$ is the phase space distance between two clusters i and j , R is a parameter of the algorithm and p_{T_i} is the momentum space distance between the beam axis and cluster i .

The anti- k_t algorithm proceeds as follows: Starting from two clusters, it first finds the minimum between d_{ij} and d_{iB} . If the minimum is realised by d_{ij} , clusters i and j are combined into one by summing up their corresponding four-vectors. The resulting cluster ij is kept in the list of clusters but not the two individual clusters i and j . If d_{iB} is the minimum, cluster i is labelled a final jet and removed from the list of clusters. The definition of the distance measure ensures that high p_T clusters dominate and the algorithm thus first focuses on hard (i.e. very energetic) objects. The clusters then further grow by the iterative addition of soft particles around the hard core until the jet i is so large that d_{iB} becomes the minimum. An additional consequence of this distance is that anti- k_t jets are rather insensitive to soft pileup and have circular shapes in the phase space (with area πR^2). If two jets overlap, the harder jets take all the soft contributions within the radius R and the other jet remains ‘moon-shaped’ [21, 25]. This is visualised in Figure 1.6.

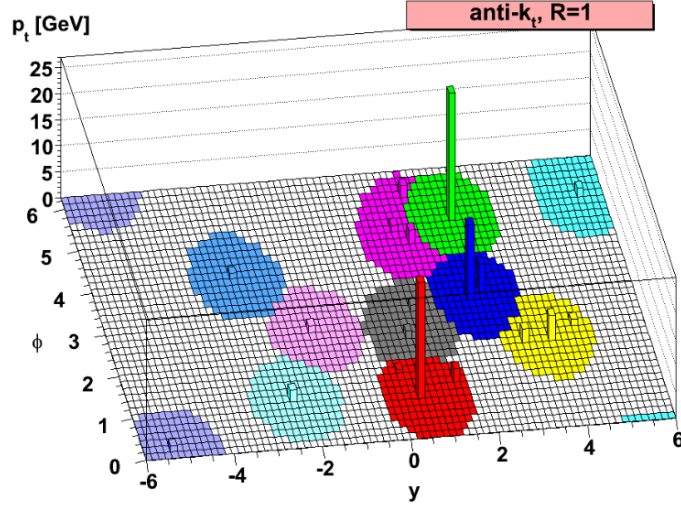


Fig. 1.6.: Schema of the anti- k_t algorithm. Figure from [23].

Infrared and Collinear (IRC) Safety Infrared and collinear (IRC) safety is essential when dealing with QCD processes, which is naturally the case with jets. A jet algorithm is said to be IRC-safe if its output does not change when a soft particle is added (infrared limit) or the collinear splitting of the particles is modified (collinear limit). This is an important property that must be satisfied to directly compare experimental results to theoretical predictions. All sequential clustering algorithms, including the anti- k_t algorithm, are by construction IRC-safe [25].

Truth vs Reconstructed (Reco) Jets In experimental data, one only observes reconstructed (reco) jets, which are – as the name suggests – reconstructed from tracks and/or energy deposits in the calorimeters. However, in simulation, one can access the hadron-level particles that constitute the jet. Such hadron jets are referred to as *truth jets*.

Jet Production Based on the SM, it is possible to derive the total and fiducial cross sections for the productions of various particles at particle colliders such as the LHC. Fiducial cross sections restrict the cross section measurements to regions where the measured objects have good detection and reconstruction efficiencies given a specific physics experiment. Here, this is done for the ATLAS experiment. Figure 1.7 summarises the cross sections for different energies of proton-proton collisions. It is evident that jets have the highest cross section and thus the dominant production rate at the LHC.



Monte Carlo (MC) event generators are the primary choice for simulations in HEP to provide concrete theoretical predictions, as it would be unfeasible to provide analytically computed predictions. To reflect the probabilistic nature of quantum mechanics, MC generators use pseudorandom numbers to sample from the probability distributions of physics events based on the SM. Starting from a hard process, corresponding to a highly energetic particle collision, the generators simulate the interactions of the constituents of the colliding particles up to the appearance of the final outgoing particles [27]. A schematic representation of a simulated $pp \rightarrow t\bar{t}$ event using the MC-based PYTHIA [28, 29] event generator, a widely used framework within and beyond HEP.

In the context of this work, we focus on proton-proton collisions and the actual interactions between gluons and quarks that make up a proton. Every simulation starts by modelling a highly energetic collision of the partons of the colliding

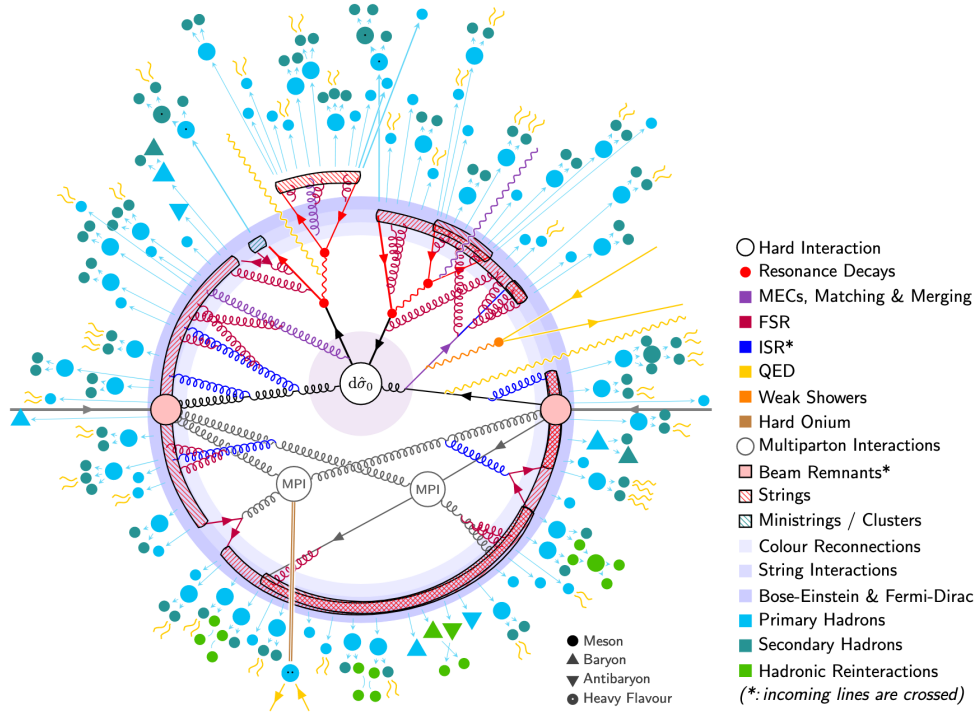


Fig. 1.8.: Schema of a $pp \rightarrow t\bar{t}$ event simulated by PYTHIA. Figure from [30].

proton beams. In a hard parton scatter event, the cross section for a proton-proton collision to produce a final particle X is given by:

$$\sigma(pp \rightarrow X) = \sum_{a,b \in \{q,g\}} \int_0^1 \int_0^1 \text{PDF}(x_1, f_a, \mu_F) \text{PDF}(x_2, f_b, \mu_F) \times \sigma(f_a f_b \rightarrow X) dx_1 dx_2, \quad (1.5)$$

where a, b are the involved partons within the colliding protons, μ_F the factorisation scale and $\sigma(f_i f_j \rightarrow X)$ the cross section at parton-level. The **Parton Distribution Function (PDF)** corresponds to the probability of finding a parton a with momentum fraction x_1 within parton p at the factorisation scale μ_F .

The parton-level cross section can be expressed as:

$$\sigma(f_i f_j \rightarrow X) = \int \frac{d\hat{\sigma}_{ab}(\Phi_{ab \rightarrow X}, \mu_F^2)}{d\Phi_{ab \rightarrow X}} d\Phi_{ab \rightarrow X} \quad (1.6)$$

$$\approx \int \frac{1}{2x_a x_b s} |\mathcal{M}_{ab \rightarrow X}|^2 d\Phi_{ab \rightarrow X}, \quad (1.7)$$

where we used that the differential cross section $\frac{d\hat{\sigma}_{ab}(\Phi_{ab \rightarrow X}, \mu_F^2)}{d\Phi_{ab \rightarrow X}}$ can be approximated using the centre-of-mass energy $s = (p_a + p_b)^2$ and the matrix element $\mathcal{M}_{ab \rightarrow X}$, which is computed up to some order in perturbation theory. The phase space integral can then be performed using **MC** sampling techniques [30–32].

It is also possible for the remaining partons of the colliding protons to interact despite not taking part in the hard scattering event. Such interactions are referred to as **Multiple Parton Interactions (MPIs)**. The colliding partons might also emit radiation before the hard scattering, which is denoted as **Initial State Radiation (ISR)**. Further, the radiation emitted by the partons after the hard scattering is called **Final State Radiation (FSR)**. MPIs together with ISR and FSR make up the *underlying event*, a term which encompasses any additional interaction besides the hard scattering.

1.3.2 Parton Showering and Hadronisation

After the hard scattering, the resulting, highly energetic partons are not colour-neutral and radiate gluons. Due to this and the self-coupling of gluons, a *parton shower* is induced and evolves until the energies of the final state particles fall below the hadronisation scale.

To simulate this, parton shower algorithms are used. Their task is to connect different energy scales (like re-normalisation), from the high energy scale to the hadronisation scale. These algorithms evolve step by step the initial partons into resulting many-parton objects, splitting their energy at each branching, until they reach hadronisation scale. Parton shower algorithms form an integral part of modern event generators such as HERWIG, PYTHIA, and SHERPA [14, 32].

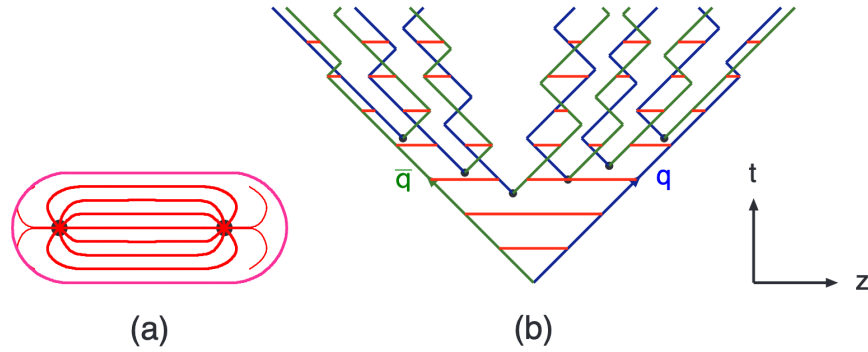


Fig. 1.9.: Illustration of the string potential between a $q\bar{q}$ pair in (a) and the string breaking by $q\bar{q}$ pair creation in (b). Figure from [27].

As the final partons reach the hadronisation level, it is no longer possible to rely on perturbative QCD. In this regime, semi-classical hadronisation models are used, such as the previously introduced *string model* (see Section 1.1.1) [19, 31, 33]. For instance, starting from a coloured $q\bar{q}$ pair, the simulation describes the quarks separating under the string-like potential (red lines in Figure 1.9 (a)) until the accumulated energy is sufficient to generate a new pair through string breaking (see Figure 1.9 (b)). This goes on until the partons' energies

are converted into $q\bar{q}$ pairs, which then begin to form hadrons. This process is visualised in Figure 1.9.

To account for the presence of multiple proton-proton interactions occurring (almost) simultaneously with a hard parton scattering, so-called *pileup*, minimum bias events are overlaid onto the simulated event.

1.3.3 Detector Simulation

The final and crucial part of event generation is simulating the propagation of the final state particles through the detector and modelling the corresponding detector response. This includes the interaction of particles with different detector materials, the electromagnetic fields applied to specific detector regions, and the corresponding detector response. All these steps are simulated with Geant4 [34]. The digital processing of the associated detector signal is simulated separately in a subsequent *digitisation* step.

The ATLAS Experiment at CERN

The ATLAS¹ experiment at the LHC is a general-purpose particle detector that is part of the European Organization for Nuclear Research (CERN). CERN was officially introduced in 1951 at a UNESCO meeting in Paris and formally established with the signing of the CERN convention in 1954, to encourage cutting-edge research in Europe and to unite European scientists after the war [35]. Initially consisting of twelve founding states, it has been constantly growing and acquiring new member states. Currently, there are 24 member states and 10 associate member states. This creates a unique and international environment that fosters cultural and scientific exchange worldwide and has produced many groundbreaking results in the last 74 years.

Namely, this includes the discovery of the W, Z and Higgs bosons [9, 10], the production and trapping of anti-hydrogen atoms [36, 37] and observing gravity's effect on them [38]. The research conducted at CERN has also led to significant technological progress such as the development of new particle detector types, the creation of the world wide web [39], touchscreen technology, improvements in superconducting magnet technology, and the Worldwide LHC Computing Grid (WLCG), the world's largest computing grid [40, 41]. Through knowledge transfer, many of these innovations have had impacts beyond particle physics, particularly in healthcare technologies like particle accelerator-based cancer treatment and advanced medical imaging [35, 42].

2.1 The Large Hadron Collider (LHC)

The LHC is a particle accelerator and collider installed at CERN. It is currently the world's largest and most powerful particle collider, achieving the highest collision energies in an experimental setting. The LHC either collides protons or heavy ions, which both fall into the category of hadrons, hence its name. It has a circumference of approximately 27 km and is located in a ring-shaped tunnel, on average 100 m underground, in the Swiss-French border region near Geneva. To get the colliding particles up to speed before injecting them into the LHC, the particles are sent through various stages of the CERN accelerator complex,

¹A Toroidal LHC Apparatus.

mainly consisting of former particle accelerators. A schematic representation of the CERN accelerator complex is shown in Figure 2.1.

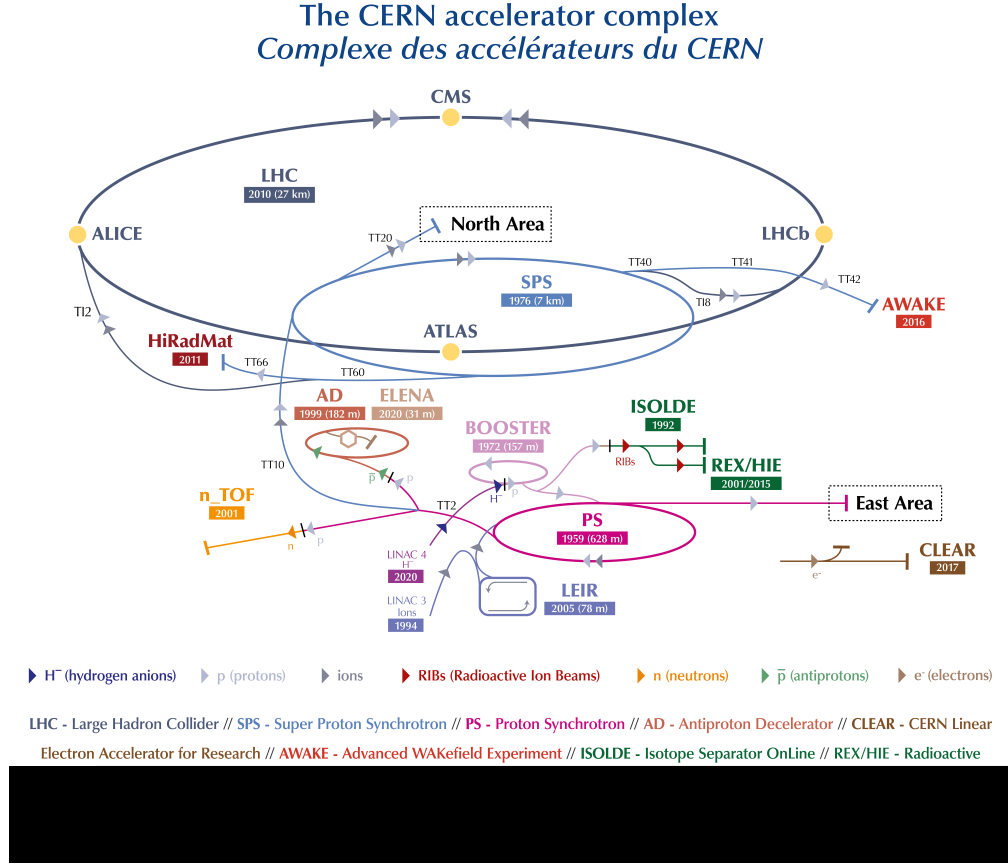


Fig. 2.1.: CERN accelerator complex, including various physics experiments. Figure from [43].

The **LHC** accelerates protons to 0.999999991 times the speed of light while circulating them through the tunnels. At the moment of collision, they reach a centre-of-mass of $\sqrt{s} = 13.6$ TeV up to now, which is the world's highest collision energy. The accelerated particles are typically protons, but for a short period each year, the protons are replaced by lead ions. Such heavy ion runs typically reach an energy of up to $\sqrt{s_{NN}} = 5.36$ TeV per nucleon pair.

The circulating particle beams travel in two separate tubes and only cross each other at four well-defined interaction points that are shown as yellow dots in Figure 2.1. Each of these dots corresponds to one of the **LHC** physics experiments, namely ALICE [44], ATLAS [45], CMS [46] and LHCb [47]. The two experiments ATLAS and CMS consist of general-purpose detectors that have been built to cover a wide range of possible physics studies with the **LHC**. This includes, for instance, the search for the Higgs boson and potential new particles beyond the **SM** of particle physics, but also precise measurements of the properties of the

SM. Next, the LHCb experiment focuses on matter-antimatter asymmetry and CP violation. Finally, the ALICE experiment studies heavy ion collisions to study quark-gluon plasmas, motivated by the fact that they might resemble the state of matter shortly after the Big Bang.

Operational periods of the **LHC** during which particles are collided and data taking is possible, are called runs. The proton collision energy has been gradually increased over these runs: during run 1 (2009-2013) a centre-of-mass energy of $\sqrt{s} = 8$ TeV was reached, during run 2 (2015-2018) it grew to $\sqrt{s} = 13$ TeV and during the ongoing run 3 (2022-present) $\sqrt{s} = 13.6$ TeV was achieved [48–51].

Luminosity Besides the energy of the colliding particles, an important quantity to describe the functioning of the **LHC** is the *luminosity*. The luminosity for a certain number of physics events N per unit of time t is defined as follows:

$$\mathcal{L} = \frac{1}{\sigma} \frac{dN}{dt}, \quad (2.1)$$

where σ represents the cross-section for the production of these events. The instantaneous luminosity depends on the number of particles in the beam, as well as other beam properties, and can therefore vary during a run, as the number of circulating particles gradually decreases due to the collisions. Therefore, one typically refers to the integrated luminosity over some time T :

$$\mathcal{L}_{int} = \int_0^T \mathcal{L} dt. \quad (2.2)$$

So far, the integrated luminosity of proton-proton collision data that has been recorded by ATLAS during run 3 amounts to 183 fb^{-1} [52].

Pileup During a typical run of the **LHC**, protons circulate in bunches of 10^{11} protons and the bunches collide every 25 ns in all of the four interaction points. The proton-proton collision with the largest transverse momentum transfer is considered the primary interaction, and all other collisions from the same bunch crossing are referred to as pileup interactions. The average number of interactions per bunch-crossing (averaged over the luminosity) has been increased over the different runs of the **LHC**. For run 2, the mean number of interactions per bunch-crossing (averaged over luminosity) $\langle \mu \rangle = 34$, which increased to $\langle \mu \rangle = 54$ for run 3.

Pileup mainly consists of soft-QCD processes, where two types can be distinguished: *In-time pileup* refers to proton-proton collisions occurring in the same

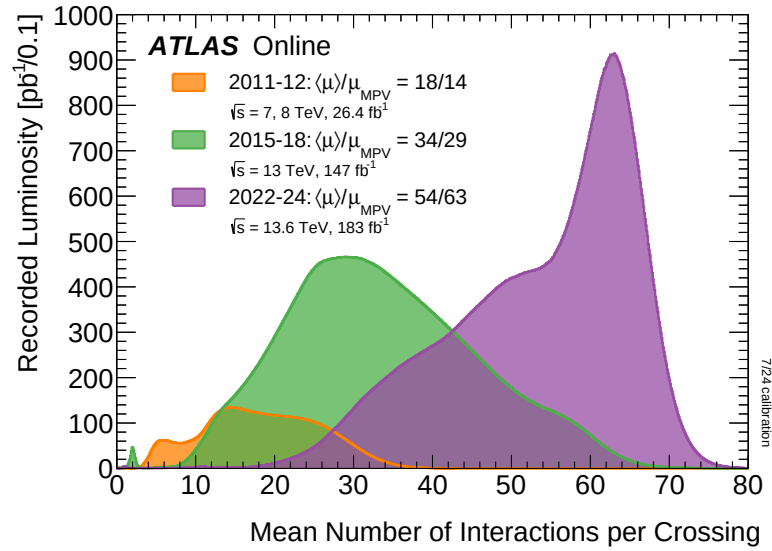


Fig. 2.2.: Average number of interactions per bunch-crossing (averaged over luminosity) $\langle\mu\rangle$ and most probable value for the number of interactions μ_{MPV} for the three data taking periods. Figure from [53].

bunch crossing, whereas *out-of-time pileup* describes residual signals from previous or subsequent bunch crossings. In-time pileup can be quantified using the number of reconstructed primary vertices N_{PV} per bunch crossing. Different, out-of-time pileup is better characterised using the average number of interactions per bunch-crossing $\langle\mu\rangle$. The evolution of $\langle\mu\rangle$ for the different LHC data taking runs is shown in Figure 2.2 [54, 55].

High-Luminosity LHC For the upgrade of the collider, the so-called HL-LHC, the average number of interactions per bunch-crossing is expected to amount to $\langle\mu\rangle = 200$ [56, 57]. The expected peak luminosity for the HL-LHC is $5 \cdot 10^{34} \text{ cm}^{-2} \text{ s}^{-1}$, which adds up to an integrated luminosity of 250 fb^{-1} per year. This will bring an entirely new challenge to the data filtering, monitoring and subsequent processing. The HL-LHC is planned to start operating in 2030 [58].

2.2 The ATLAS Detector

The ATLAS detector is a general-purpose detector. To cover almost the whole 4π around the collision point without obstructing the beam line, the detector is built in a cylindrical shape. With a height of around 25 m and a length of 46 m, ATLAS is also the biggest detector at the LHC. A schematic representation of the current detector configuration is shown in Figure 2.3.

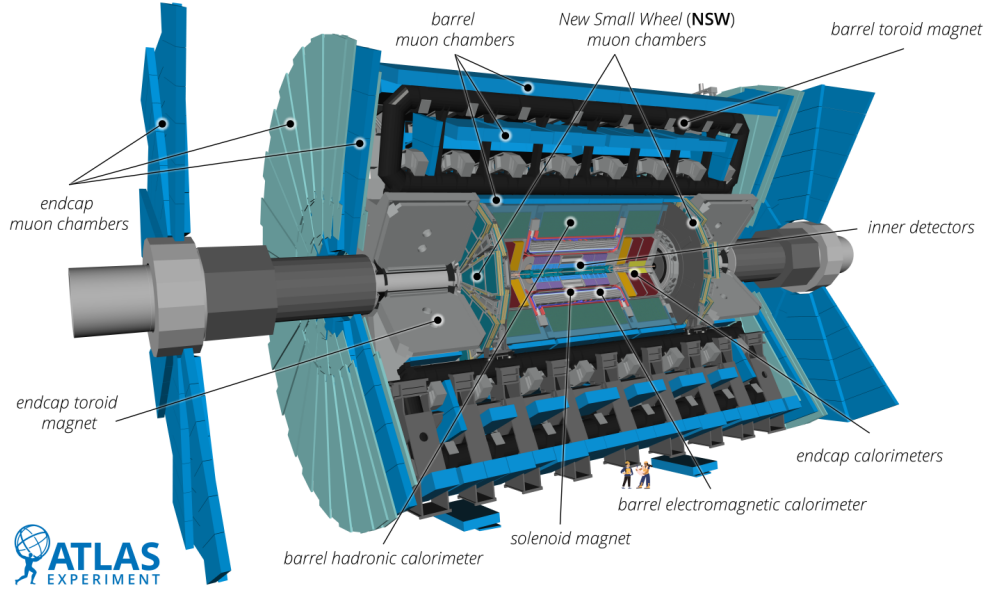


Fig. 2.3.: Schema of the ATLAS detector at the LHC. Figure from [59].

The collision point is ideally located at the centre of the detector, which is where the origin of the LHC coordinate system is set. However, shifts and stochastic fluctuations, especially along the beam axis, are present and small corrections have to be applied to account for them. The z -axis coincides with the beam line (or the collision axis), the x -axis points towards the centre of the LHC and the y -axis upwards. Since the detector is cylindrical, one can also consider expressing the coordinates with the azimuthal angle ϕ and the polar angle θ . At hadron colliders, it is common practice to use the pseudorapidity η because it is invariant to boosts along the z -axis. It is defined as $\eta = -\ln(\tanh \theta/2)$.

The ATLAS detector has an onion-like structure consisting of multiple sub-detectors that are each optimised to measure specific properties of the decaying particles. The combination of these properties makes it possible to distinguish the most common particles. The detector parts can roughly be categorised into three parts: the inner tracking detector, the electromagnetic and hadronic calorimeter, and the muon spectrometer. Their respective locations are indicated in Figure 2.3.

In the following, our main focus will lie on the ATLAS calorimeters, as they are essential for the jet and data quality studies presented in this thesis. The tracking and muon detectors, which are also relevant for these studies, will be briefly described too [45, 48, 50].

2.2.1 Inner Detector

The inner detector refers to the inner-most part of the ATLAS detector and consists of the pixel detector, the [Semi-Conductor Tracker \(SCT\)](#) and the [Transition Radiation Tracker \(TRT\)](#) as shown in Figure 2.3. Each of these three sub-detectors is specialised for tracking a charged particle's trajectory, charge and momentum. For this, the inner detector is enveloped in a 2 T solenoidal magnetic field. The detector is designed to have very good momentum resolution for charged particles with $p_T > 500$ MeV within $|\eta| < 2.5$ and a complete coverage in ϕ . As charged particles pass through the detector, they ionise the material and leave so-called *hits* in the detector. From the hits, the particle trajectories can be reconstructed, and the trajectory's curvature, due to the magnetic field, allows one to derive the particle's momentum and charge (see particle trajectories in Figure 2.8). The expected relative resolution for the transverse momentum p_T of a reconstructed track in the inner detector corresponds to:

$$\frac{\sigma_{p_T}}{p_T} = 0.05\% \times p_T[\text{GeV}] \oplus 1\%. \quad (2.3)$$

The inner detector will be completely replaced for the start of the [HL-LHC](#). The new, all-silicon [Inner Tracker \(iTK\)](#) is designed to withstand higher pileup conditions and to improve the coverage in $|\eta|$ up to 4, while maintaining a similar or better tracking performance than the current inner detector [48, 50].

Pixel Detector The silicon pixel detector is closest to the collision vertex and has the highest granularity to manage the very high track density in this sector. It covers the region with pseudorapidity $|\eta| < 2.5$. As the name implies, the pixel detector consists of silicon pixels that are segmented both parallel and vertically to the beam axis. They are arranged in four barrel layers with pixel size of $50\mu\text{m} \times 400\mu\text{m}$ in the central part, and three discs, perpendicular to the beam axis, in each end-cap region (higher $|\eta|$ region). In total, the pixel detector provides approximately 80.4 million readout channels.

Semi-conductor Tracker (SCT) The [SCT](#) consists of silicon microstrips, surrounding the pixel detector, covering again the region described by $|\eta| < 2.5$. It consists of four cylindrical layers in the barrel region and nine disks in each end-cap. Each cylindrical layer consists of two strip layers that are layered perpendicularly to create a two-dimensional net. This amounts to approximately 6.3 million readout channels.

Transition Radiation Tracker (TRT) The outermost part of the inner detector is the TRT which encloses the pixel detector and the SCT, while covering the region given by $|\eta| < 2$. It is built by alternating between gas-filled straw tubes and transition radiation material. The energy of the radiation depends on the mass of the passing particle, which helps with particle identification, in particular to distinguish electrons from charged hadrons. Each straw tube corresponds to a cylindrical gas-filled drift chamber, containing an anode wire in the centre and the outer shell corresponding to the cathode. While initially working with a Xenon-based gas mixture, for run 3, most of the straw tubes contain an Argon-based mixture. In the barrel region, the straws are aligned along the beam axis, whereas in both end-cap regions, the straws are arranged radially. There are about 351'000 TRT readout channels.

2.2.2 Calorimeters

The ATLAS calorimeters consist of two main parts, namely the *electromagnetic* and the *hadronic* calorimeters. The latter can be further decomposed into the Tile calorimeter, Hadronic Endcap (HEC) and Forward Calorimeters (FCal). All calorimeter parts are shown in Figure 2.4. The ATLAS calorimeters cover a range of $|\eta| < 4.9$ and aim to measure the energy and direction of decaying particles. For this, the calorimeters combine an active material for energy measurements with layers of high-density material for energy absorption. Figure 2.5 provides a complementary illustration of the ATLAS calorimeter, including the pseudorapidity η ranges for the different sub-detectors.

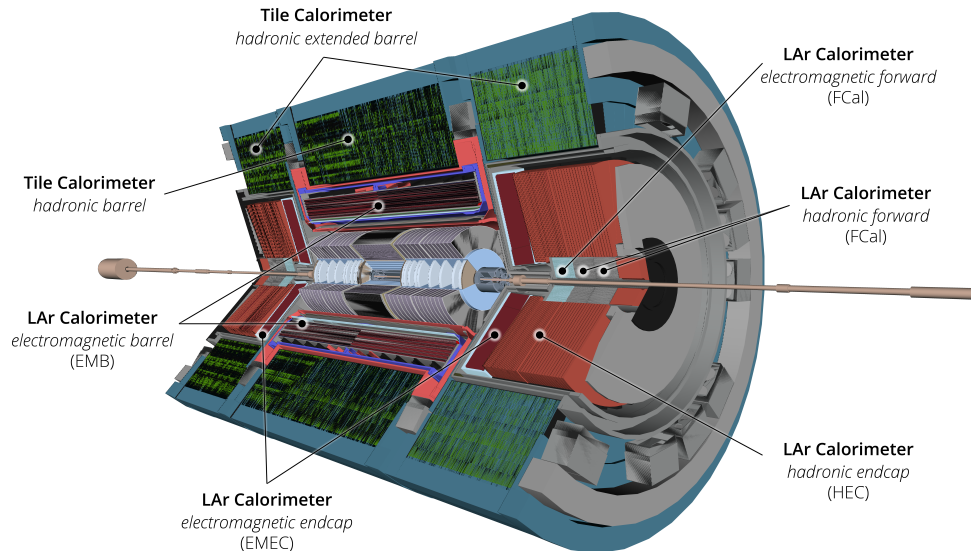


Fig. 2.4.: Schema of the ATLAS calorimeters. Figure from [48].

Most passing particles interact heavily with the calorimeter, depositing their full energy in the absorbing material of the calorimeter. Very few particles have such high energies or such little interaction with the calorimeter that they travel beyond this detector part. If they do, this is called a *punch-through*.

The ATLAS calorimeter is *non-compensating*. This means that the detector response for electromagnetic showers differs from that for hadronic showers with identical energies. Electromagnetic particles fully ionise the active material and lead to a signal proportional to the full energy of the electromagnetic shower. Differently, a hadronic shower produce a cascade that includes an electromagnetic and a non-electromagnetic component, which interacts with the calorimeter's nuclei. Therefore, part of the energy of the hadronic shower is not transmitted and the signal is not proportional to the full energy of the shower. Hence, an electromagnetic and hadronic shower with similar energies can elicit different detector responses which is taken into account and corrected for during jet calibration.

The energy resolution $\sigma(E)$ in the calorimeters is parametrised as follows:

$$\frac{\sigma(E)}{E} = \frac{N}{E} \oplus \frac{S}{\sqrt{E}} \oplus C, \quad (2.4)$$

where N the noise term, S the stochastic term and C the constant term [45, 50].

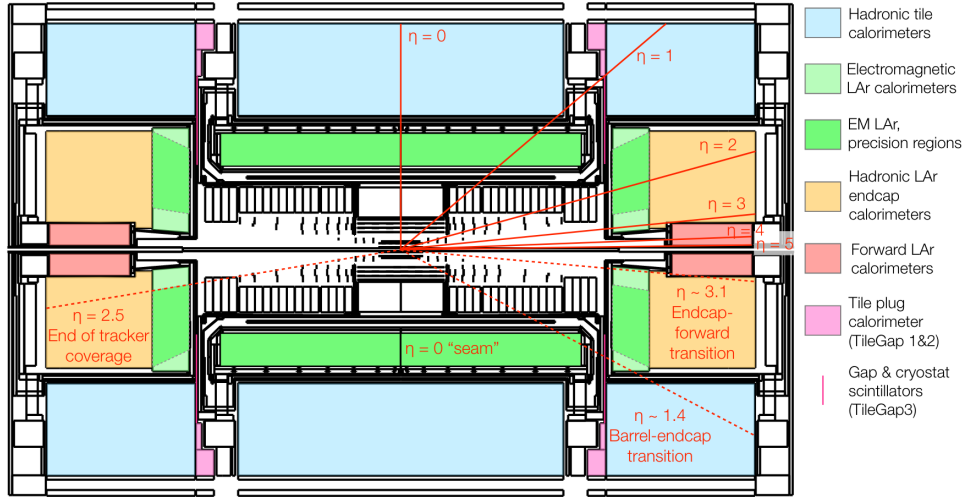


Fig. 2.5.: Schematic illustration of the ATLAS calorimeters along the $y - z$ plane with pseudorapidity η for reference. Figure from [60].

Electromagnetic Calorimeter The electromagnetic calorimeter (shown in green in Figure 2.5) consists of a barrel part ($|\eta| < 1.475$), called the **Electromagnetic**

Barrel (EMB), and two end-cap components ($1.375 < |\eta| < 3.2$), commonly referred to as **Electromagnetic End-Cap (EMEC)**. All of these sub-detectors are using **Liquid Argon (LAr)** as an active material, copper electrodes for readout and lead as absorbers. Furthermore, all of them are equipped with their separate cryostat system. The solid materials are accordion-shaped and provide a full coverage over ϕ . The **EMB** consists of three layers, whereas the **EMEC** consists of two. For the **EMB**, each layer has a different granularity, the first layer has a resolution of $\Delta\eta \times \Delta\phi = 0.025/8 \times 0.1$, the second of $\Delta\eta \times \Delta\phi = 0.025 \times 0.025$, and the third of $\Delta\eta \times \Delta\phi = 0.050 \times 0.025$. For the **EMEC**, the granularity is significantly coarser starting from 0.050×0.1 up to 0.1×0.1 .

Figure 2.6 gives an example the coverage of the **LAr** end-cap C calorimeter during a data-taking run in July 2022.

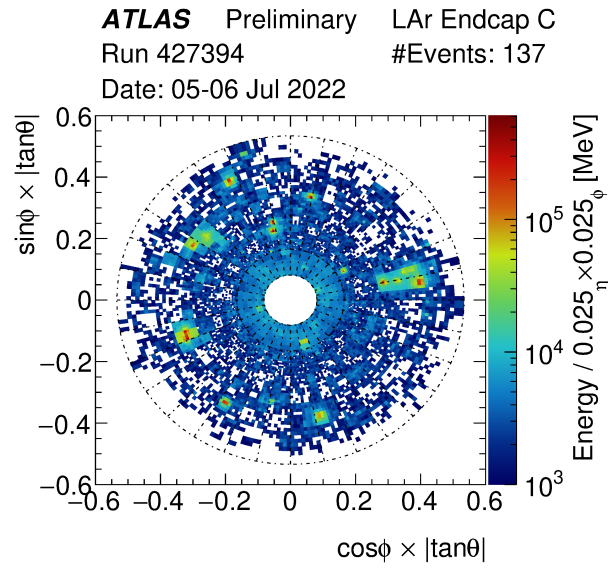


Fig. 2.6.: Coverage plot showing the cell energy sums of the **LAr** calorimeter (without **FCal**) distributed in a hypothetical tower grid with $\Delta\eta \times \Delta\phi = 0.025 \times 0.025$ for 137 proton-proton collision events at $\sqrt{s} = 13.6$ TeV in run 427394 from 5-6 July 2022. Figure from [61].

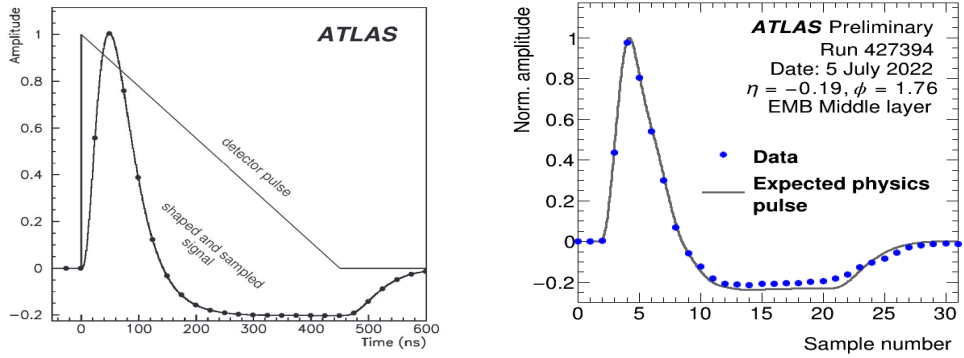
Furthermore, before the **EMB**, there is a thin *pre-sampling* detector covering $|\eta| < 1.8$ that corrects the energy lost in the material upstream of the calorimeters, especially in the solenoid magnet. It is sometimes also referred to as the 0-th layer of the **EMB**. Finally, between the **EMB** and **EMEC** is a transition region, containing the so-called *gap scintillators*. They are placed there to measure the energy lost in the inactive material between the calorimeters. Including the presampler, the electromagnetic calorimeters have up to 173'000 readout channels.

When a high-energy electromagnetic particle such as an electron or photon passes through the detector, it causes a narrow particle shower. Usually, most of the electromagnetic shower is contained within the second layer of the

electromagnetic calorimeter, and its intensity is proportional to the energy of the incident particle [45, 50].

The **EMB** is subjected to a constant voltage of 2000 V to power the readout electrodes. Differently, for the **EMEC** a varying voltage is applied to have an almost η -independent detector response.

The ATLAS **LAr** calorimeters have a relatively slow response time compared to the frequency of proton-proton collisions. This is because the drift time of the **EMB** corresponds to approximately 450 ns at nominal voltage. Therefore, the measurements in the **LAr** calorimeters typically integrate over multiple bunch-crossings, which leads to the previously introduced *out-of-time pileup*. The influence of pileup on the measurements is mitigated and corrected during jet calibration, which is detailed in Section 2.3.



(a) Triangular pulse shape of the signal in the **LAr** calorimeter overlaid with the sampled pulse shape. Figure from [62]. (b) Pulse signal in the **EMB** calorimeter overlaid with the expected pulse shape from run 3. Figure from [61].

Fig. 2.7.: Pulse shapes and signals for the **LAr** calorimeter.

By design, the energy resolution of the electromagnetic calorimeter corresponds to:

$$\frac{\sigma(E)}{E} = \frac{10\%}{\sqrt{E}} \oplus 0.7\%. \quad (2.5)$$

Front End Boards (FEBs) are the main components of the readout electronics that are directly mounted onto the **LAr** calorimeters. They amplify and adjust the incoming signal such that it is bipolar-shaped in the end and integrates to zero over its duration. A typical pulse shape of a signal in the **LAr** calorimeter is shown in Figure 2.7a and a real pulse signal from the **EMB** in Figure 2.7b.

Hadronic Calorimeter The hadronic calorimeter is split into the Tile calorimeter (shown in blue in Figure 2.5), the **HEC** (in orange in Figure 2.5) and the **FCal** (in red in Figure 2.5). The Tile calorimeter covers the barrel region corresponding to $|\eta| < 1.7$, while the **HEC** covers $1.5 < |\eta| < 3.2$ and the **FCal**

covers $3.1 < |\eta| < 4.9$. Furthermore, the Tile calorimeter is based on scintillating tiles and steel plates as absorbers, whereas the other two sub-detectors again rely on the [LAr](#) sampling technology. However, the [HEC](#) and [FCal](#) do not use lead as the absorber but copper and copper-tungsten, respectively.

Hadrons interact with the nuclei in the absorber material via the strong force and produce cascades of gluons and quarks that do form hadronic showers. The nuclear interaction length λ of these showers is typically an order of magnitude larger than the radiation length, characterising the length of electromagnetic showers. Therefore, the hadronic calorimeters mainly focus on hadronic particles and their showers, since most electromagnetic particles have already been absorbed in the previous detector layers. The measurements from the hadronic detector are essential for jet energy and missing p_T estimations. As previously mentioned, the ATLAS calorimeters are *non-compensating*, meaning that the calorimeter response for electromagnetic showers is greater than for non-electromagnetic ones. This is because some parts of non-electromagnetic components of hadronic showers, such as backscattering and nuclear excitation processes, do not lead to an observable result in the calorimeter.

The Tile calorimeter is further split into a central barrel ($|\eta| < 1$) and extended barrels ($0.8 < |\eta| < 1.7$), consisting of 64 modules of steel plates and scintillating tiles with $\Delta\phi \sim 0.1$. It consists of three layers, the first two having a granularity of $\Delta\eta \times \Delta\phi = 0.1 \times 0.1$ and the last one of $\Delta\eta \times \Delta\phi = 0.2 \times 0.1$. It has a radial depth of up to 7.4λ (interaction lengths), capturing a big part of the hadronic shower. The photons produced in the scintillating tiles are collected at the edge of the tiles, transmitted through two wavelength-shifting fibres and coupled to two [Photomultiplier Tubes \(PMTs\)](#). Two [PMTs](#) are used to provide redundant signals that can be compared and used to equalise the signals. The Tile calorimeter has almost 10'000 readout channels.

The [HEC](#) consists of two independent wheels per end-cap, again consisting of two segments each. It is directly placed after the [EMEC](#) and shares the cryostat with the [EMEC](#) and [FCal](#). The [HEC](#) combines copper leaves with [LAr](#)-filled gaps and has a granularity of $\Delta\eta \times \Delta\phi = 0.1 \times 0.1$ for the inner segments ($1.5 < |\eta| < 2.5$) and $\Delta\eta \times \Delta\phi = 0.2 \times 0.2$ for the outer segments ($2.5 < |\eta| < 3.2$). Its radial depth adds up to 9.7λ , and it has roughly 5'600 readout channels.

Finally, the [FCal](#) covers $3.1 < |\eta| < 4.9$ that previous detector layers do not cover and is therefore subjected to a high particle flux. It consists of three 45 cm modules, one electromagnetic module ([FCal 1](#)), using copper as absorber material, and two hadronic modules ([FCal 2](#) and [FCal 3](#)), using tungsten as the absorber. In total, this amounts to a radial depth of 10λ and about 3'500 readout channels.

The design energy resolution of the different hadronic calorimeters is:

$$\frac{\sigma(E)}{E} = \begin{cases} \frac{50\%}{\sqrt{E}} \oplus 3\%, & \text{for the Tile calorimeter and HEC,} \\ \frac{100\%}{\sqrt{E}} \oplus 10\%, & \text{for the FCal.} \end{cases} \quad (2.6)$$

2.2.3 Muon Spectrometer

The muon spectrometer is the outermost sub-detector, covering the range $|\eta| < 2.7$. Only a few particles reach this far; typically neutrinos or muons, or in rare cases, very energetic jets or potential new particles not described by the SM. The muon spectrometer measures the trajectory of charged particles that have not been absorbed by the calorimeters. Its primary purpose is the identification and momentum estimation of muons; neutrinos, being uncharged, do not leave any tracks in the muon spectrometer. It is also used to detect punch-through signals, i.e. hadronic showers that are not fully contained inside the hadronic calorimeter, which is essential for correcting the energy deposits measured in the calorimeters. In addition, it serves as a muon trigger in the region $|\eta| < 2.4$. Similarly to the inner detector, the muon spectrometer is immersed in a magnetic field, bending the tracks of charged particles passing through. In the barrel region ($|\eta| < 1.4$), the magnetic field is provided by a large air-core superconducting toroid, whereas in the end-cap ($1.6 < |\eta| < 2.7$), smaller magnets are used.

For precise tracking, the muon spectrometer consists of monitored drift tubes and cathode-strip chambers. For fast triggering, this is completed by resistive plate chambers and thin gap chambers. For run 3, part of the inner end-cap layers were replaced by new small wheels (see Figure 2.3) to improve triggering and tracking at higher background rates. In total, the muon spectrometer has roughly 350'000 readout channels.

All of this combined with the tracks from the inner detector make it possible to reach the following relative resolution on the transverse momentum of muons:

$$\frac{\sigma(p_T)}{p_T} = \begin{cases} 4\%, & \text{for } p_T = 50 \text{ GeV,} \\ 10\%, & \text{for } p_T = 1 \text{ TeV.} \end{cases} \quad (2.7)$$

2.2.4 Trigger and Data Acquisition

The ATLAS Trigger and Data Acquisition (TDAQ) system is responsible for selecting and processing ATLAS data.

The trigger system consists of two stages: a hardware trigger followed by a software trigger. First, the Level 1 (L1) trigger is hardware-based and reduces the

incoming data stream from 40 MHz to maximally 100 kHz. Various sub-detectors have their associated [L1](#), which together form the [L1](#) trigger system. Second, the [High Level Trigger \(HLT\)](#), which is software-based, further filters the data stream down to 3 kHz. For small selections of events, the event rate is downsampled to 200 Hz because more time (approximately 4 s) is dedicated to reconstruction for each of these events.

After the [L1](#) trigger, the events are buffered by the data acquisition system, but they are only fully reconstructed if they pass the selections of the [HLT](#) [45, 48].

This step includes detector calibration, alignment, object reconstruction (such as particles, calorimeter clusters, and jets), track reconstruction, and more. Each event in RAW format typically occupies about 1.6 MB.

Simultaneously, a separate workflow handles simulated physics events. These begin with [MC](#) generation based on established physical laws and possible extensions from [BSM](#) theories (see Section 1.3). The simulation mimics how such events would interact with the detector, including signal digitisation, as if they occurred in a real collision. The resulting [MC](#) samples resemble actual collision events, but include additional ‘*truth*’ information from the generator, providing a reference for validation. These simulated events undergo the same reconstruction steps as real data.

[HLT](#) output is stored permanently in byte-stream format as RAW files in the [WLCG](#). These are reconstructed into [Event Summary Data \(ESD\)](#) files – C++ object formats that store sufficient detail for tasks like particle identification, jet calibration, and track refitting. [ESD](#) files are typically used transiently, retained on disk only when needed and for a limited set of analyses. They form the basis for more compact analysis-friendly formats: [Analysis Object Data \(AOD\)](#) and [Derived Analysis Object Data \(DAOD\)](#). [ESDs](#) range from 1 – 2 MB per event, [AODs](#) from 200 – 500 KB, and [DAODs](#) from 50 – 100 KB depending on the analysis. For efficient, simplified analysis, these datasets are often further converted into [NTuples](#), which are flat, tabular representations of event data where each row corresponds to an event and columns represent physics variables, facilitating fast access.

Both recorded and simulated events are distributed across the [WLCG](#) for full reconstruction. The grid is organized hierarchically: Tier-0 at CERN handles initial data storage and processing, followed by about ten Tier-1 centres for additional processing and long-term tape storage, and over one hundred Tier-2 centres for analysis and simulation tasks. In total, around 350'000 CPU cores and more than 1 EB ($1 \text{ EB} = 10^6 \text{ TB} = 10^9 \text{ GB}$) of storage are available to the ATLAS collaboration. Additional resources include cloud computing services

and High-Performance Computing facilities, some including access to large-scale GPU infrastructures [63].

2.2.5 Object Reconstruction

Most particles produced after the proton-proton collisions are very unstable and decay before or inside the detector into secondary particles. These particles are more stable and correspond to well-known particles such as electrons, photons, muons or jets. Figure 2.8 visualises how these particles interact with the different ATLAS sub-detectors. Starting from the signals in the sub-detectors, it is then possible to reconstruct the physics event, initiated by the proton-proton collision.

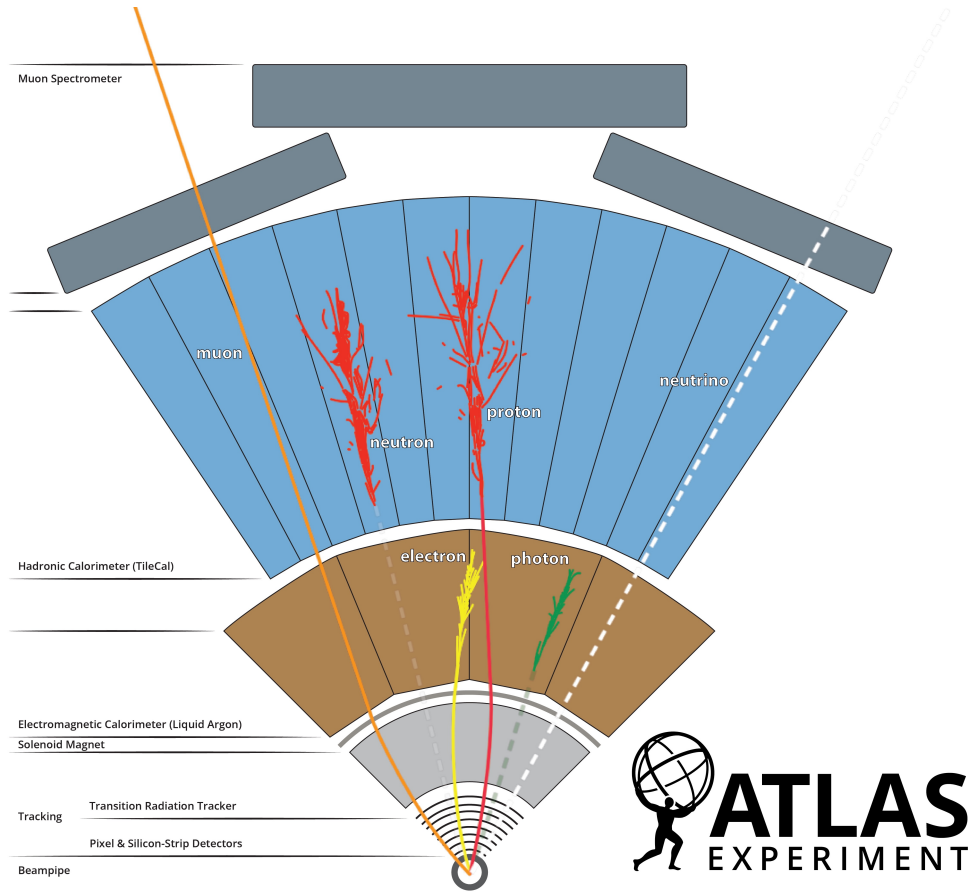


Fig. 2.8.: Schematic slice of the ATLAS detector and how particles interact with the detector layers. Figure from [64].

Track and Vertex Reconstruction Track reconstruction is based on the hits recorded in the inner detector. In the standard ATLAS tracking [65] triplets of hits in the pixel and/or the SCT detectors are used to seed the track finding. By

adding hits along the trajectories, track candidates are fitted and subsequently extrapolated into the [TRT](#). The tracking algorithm places constraints on the transverse d_0 and longitudinal z_0 impact parameters of track candidates for the [Primary Vertex \(PV\)](#). The [PV](#) corresponds to the point where the initial proton-proton collision occurred. [PVs](#) in a collision event are formed from reconstructed tracks such that they are spatially compatible with the luminous interaction region. The *hard-scatter* [PV₀](#) is chosen to be the vertex with the highest $\sum_{p_T > 0.5 \text{ GeV}} p_T^2$, where the sum is over the charged tracks [66]. Secondary vertices are the points where a long-lived particle decays into other particles and are important for particle identification. For instance, *b*-tagging relies on the fact that most *B* mesons decay within the ATLAS detector and form a secondary vertex due to their relatively short lifetime ($\approx 1.6 \text{ ps}$).

Jets Jets represent the spray of particles produced by the hadronisation of strongly interacting particles such as quarks and gluons. As discussed in Section 1.2, jets are the physics objects with the dominant production at the [LHC](#). Therefore, their precise reconstruction and calibration are of uttermost importance for many subsequent physics analyses. The details of jet reconstruction and calibration within ATLAS are discussed in Section 2.3.

Electrons and Photons Since electrons and photons are electromagnetic particles, their reconstruction is very similar. For electrons and positrons, which are charged particles, it is further possible to match the energy deposits to tracks in the inner detector. Electrons and positrons can be distinguished based on the curvature of their tracks, as the magnetic field affects them differently. This is impossible for the uncharged photons, as they leave no tracks in the inner detector, except if they convert into e^-e^+ pairs.

Electron candidates are reconstructed by matching an inner detector track to an isolated electromagnetic calorimeter energy deposit, within the fiducial region of transverse energy $E_T > 25 \text{ GeV}$ ($> 26 \text{ GeV}$ for 2016-2018) and pseudorapidity $|\eta| < 2.47$, where η is the pseudorapidity of the calorimeter cluster associated with the electron candidate. Those within the transition region between the barrel and end-cap electromagnetic calorimeters, $1.37 < |\eta| < 1.52$, are excluded. Electrons are selected using a multivariate algorithm and are required to satisfy the likelihood-based quality criterion ‘*LH Tight*’ [67].

Muons Muons are some of the rare particles that can leave tracks in every detector layer (inner detector, calorimeters and muon spectrometer). Muon candidates are reconstructed from track segments in the various layers of the muon spectrometer and matched with tracks found in the inner detector. Next,

the muon candidates are refitted using the complete track information from both detector systems and are required to satisfy $p_T > 25 \text{ GeV}$ ($> 27 \text{ GeV}$ for 2016-2018) and $|\eta| < 2.5$ [68]. Tracks of muon candidates must have a transverse impact parameter significance $|d_0^{sig}| < 3$ and longitudinal impact parameter $|\Delta z_0 \sin \theta| < 0.5 \text{ mm}$. Muon candidates are selected using the offline medium working point [69]. Finally, to reduce the background from heavy flavour decays inside jets, muons are required to satisfy some strict isolation criteria as done for photon and electron identification.

Taus Tau particles, having a shorter lifetime ($\approx 290 \text{ fs}$) than the previously mentioned particles, often decay before reaching the ATLAS detector. They can be detected and reconstructed based on their decay products, which are either hadrons or other leptons.

Missing p_T Neutrinos, being weakly-interacting particles, or potential BSM particles are expected to pass the ATLAS detector without interacting with it. In this case, a transverse momentum imbalance is observed. Due to the conservation of laws of particle physics, the transverse momentum before and after the proton-proton collision should be preserved. Hence, if the vectorial sum of reconstructed transverse momenta associated with a collision event, does not add up to the transverse momentum of the colliding particles, which is essentially zero, this hints at the presence of one or several undetected particles.

2.3 ATLAS Jet Reconstruction and Calibration

As mentioned in Section 1.2, jets are the most produced objects from proton-proton collisions at the LHC. It is therefore crucial for many ATLAS physics analyses to have the best possible reconstruction and calibration of jets.

There are multiple jet definitions (discussed in Section 1.2) and jet radii that are frequently used. However, for this thesis, the focus lies on *small-radius* jets with radius $R = 0.4$.

2.3.1 Jet Reconstruction

In experimental data, jets are reconstructed starting from tracks in the inner detector and energy deposits in the calorimeters. The resulting objects are *reco jets*. In simulated data, it is further possible to access the hadronised particles (*truth jets*) in addition to *reco jets* [55].

Topocluster First, seed cells, with energy deposits greater than $4\sigma_{noise,cell}$, are identified. The average expected noise is defined as $\sigma_{noise,cell} = \sqrt{\sigma_{electr}^2 + \sigma_{pileup}^2}$ where σ_{electr} is the electronic noise in a cell and σ_{pileup} the noise due to the number of expected interactions per bunch crossing $\langle\mu\rangle$. Second, cells neighbouring the seed cells are recursively added to form clusters if their signal satisfies $\geq 2\sigma_{noise,cell}$. Finally, all neighbouring cells regardless of their signal significance are added to the cluster and clusters with multiple local minima are split into separate clusters.

The selection criteria based on $\sigma_{noise,cell}$ automatically enable noise suppression and pileup mitigation (for low p_T pileup) at topocluster-level. Residual out-of-time pileup is further suppressed by placing a cut on the cell timing, enforcing $|t| < 12.5$ ns for the seed cells [70]. The topoclusters can be constructed at the same energy scale as the energy deposits in the cells (electromagnetic scale) or at a recalibrated scale, the so-called **Local Weighted Cell (LCW)** scale. The latter is mainly done for *large-radius* jets (i.e. $R > 0.4$) [55].

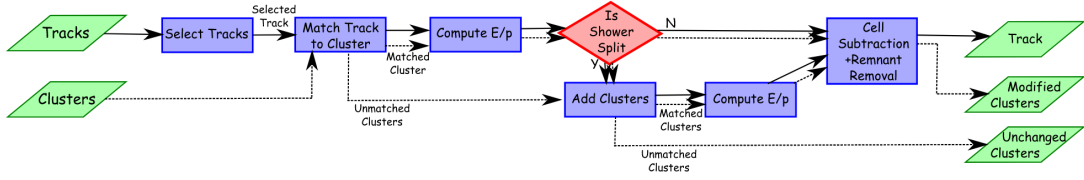


Fig. 2.9.: Schematic illustration of the **Particle Flow (PFlow)** algorithm. Figure from [55].

Particle Flow (PFlow) The **PFlow** algorithm [55] associates tracks with topoclusters to construct particle objects. A flow chart visualising the different sub-steps of the algorithm is shown in Figure 2.9. Initially, the **PFlow** algorithm ensures that each track is matched to a single topocluster. Then, the expected energy deposit in the calorimeters is computed, and it is decided whether a track contributed to multiple topoclusters. If this is the case, the energy deposits are subtracted from the cells of the corresponding topoclusters. This **Charged Hadron Subtraction (CHS)** already removes pileup tracks associated with charged particles that do not stem from the primary vertex. The final **PFlow** objects consist of the selected tracks and topoclusters.

The **PFlow** achieves good results for tracks in low p_T ranges ($\lesssim 50$ GeV). However, highly-boosted particles (> 500 GeV) are badly reconstructed by the **PFlow**. Therefore, for higher track p_T , the so-called **Track Calorimeter Cluster (TCC)** algorithm [71] is preferred, as it delivers a better resolution by resolving the spatial structure within the clusters using tracking information. Recently, the **Unified Flow Object (UFO)** algorithm [72] was introduced to combine both

methods across a wider p_T range to leverage their advantages in each p_T region. It has already become the standard algorithm for large-radius jets [72].

Jet Clustering Starting from the particle-like objects constructed from the tracking and calorimeter information, a jet algorithm is applied to group them into jets (see Section 1.2). In ATLAS, the main jet algorithm is the `anti- $k_t$` algorithm [23], as it offers the following advantages: It is IRC-safe, i.e. it doesn't change with the addition of soft radiation and collinear splitting, produces circular cones and is not very sensitive to pileup. The typical radius parameter is $R = 0.4$ for small-radius jets or $R = 1.0$ for large-radius jets.

Pileup Mitigation A first pileup removal is effected during the formation of the topoclusters. Additionally, it is possible to remove pileup contributions at the constituent-level. Since CHS is limited to charged PFlow objects, other pileup mitigation techniques are needed for neutral objects. However, both of the algorithms described in the following, are currently only used for large-radius jets [73]. The first algorithm is inspired by a pileup correction that is part of the jet calibration chain for small-radius jets (see *residual pileup correction* in Section 2.3.2).

Constituent (area) subtraction [74] adds ghost particles to a collision event that are uniformly distributed in the (η, ϕ) plane. These ghost particles are massless and have very low p_T^g , which is defined as $p_T^g = A_g \times \rho$ with $A_g = \eta \times \phi = 0.1 \times 0.1$ the ghost area and ρ the median energy density (estimated from all the clustered jets in the event). The p_T -contributions of the ghosts are iteratively subtracted from the closest constituents, which removes soft contributions and therefore contributions from pileup.

The *soft-killer* [75] algorithm places a p_T cut on all particles on an event-by-event basis to remove low p_T constituents.

2.3.2 Jet Calibration

The current calibration scheme for small-radius jets measured in the ATLAS detector follows several sequential steps as shown in Figure 2.10. The calibration of jet energies is essential due to the non-compensating nature of the ATLAS calorimeters. Therefore, the energy deposits stemming from hadronic showers need to be adjusted to be at the same scale as the energy deposits for electromagnetic showers. In simulation, this can be done by comparing the energy of the *reco jets* to the one of the *truth jets*. For experimental data, this is challenging as no truth-level information is available. There exist dedicated

approaches to this such as [LCW](#) topoclusters but they face additional challenges especially regarding pileup. Therefore, for small-radius jets, the majority of this correction is applied through the *absolute MC-based calibration*, while the other calibration steps are dedicated to mitigating pileup or reducing discrepancies between simulation and data.

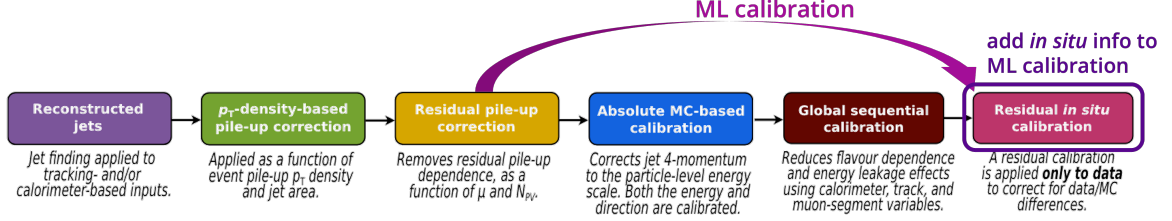


Fig. 2.10.: Schema of the ATLAS calibration chain and potential shortcuts with a [ML](#)-based jet calibration. Figure adapted from [60].

The ATLAS jet calibration can be roughly split into two phases: The first phase focuses on adjusting the [Jet Energy Scale \(JES\)](#), corresponding to the jet energy and momentum, while the second phase works on improving the [JER](#). For this, the measured quantities are corrected with the help of [MC](#) simulations. In the final step (‘pink’ box in Figure 2.10), experimental data is used to account for poorly simulated detector and physics effects. This data-driven step is referred to as *in-situ* calibration [54, 60, 76, 77].

Each of these sub-calibration steps must be done separately and in sequential order. Furthermore, each step can easily take jet calibration experts multiple months. Therefore, it can easily take a year for the jet calibration group to prepare the recommended procedure and calibration factors for each data-taking run.

Pileup Corrections Starting from the reconstructed jets, the *pileup density correction* (green box in Figure 2.10) applies an average correction to all jets within an event. Concretely, it subtracts the average pileup energy from the transverse momentum of the reconstructed jets denoted by p_T^{reco} . Mathematically, this corresponds to:

$$p_T^{corr,1} = p_T^{reco} - \rho \times A, \quad (2.8)$$

where ρ is the pileup density per area and A is the jet area. The jet area A is determined by ‘ghost association’, which is a process where infinitesimally soft ghost particles are overlaid on top of the event, and their transverse momentum is summed up if they are associated with a jet after clustering.

Next, the *residual pileup correction* (yellow box in Figure 2.10) removes any

remaining dependency on N_{PV} and μ . This correction is applied on average for all jets, depending just on p_T , μ and N_{PV} :

$$p_T^{corr,2} = p_T^{corr,1} - \alpha(N_{PV} - 1) - \beta\mu, \quad (2.9)$$

with parameters α, β being derived from MC simulation.

Absolute MC-based Calibration The *absolute MC-based calibration* (blue box in Figure 2.10) focuses on correcting the absolute JES and pseudorapidity η of the reco jets to the level of the truth jets on average. Thus, the correction factors are determined by comparing the ratio between the energy of reco and truth jets in various p_T bins. The energy response of the calibrated jets, given by $\mathcal{R} = E^{reco}/E^{true}$, is equal to 1 on average per bin. The bins are defined as a function of E^{true} and η_{det} , which is the η of the jet relative to the geometric centre of the detector. For the absolute η correction, the bias $\eta_{bias} = \langle \eta^{reco} - \eta^{true} \rangle$ is computed and subtracted from η^{reco} .

Since the truth level is only available in simulation, the correction factors are necessarily derived based on MC simulations and later applied to experimental or simulated data.

Global Sequential Calibration (GSC) The *Global Sequential Calibration (GSC)* (brown box in Figure 2.10) aims at improving the JER without affecting the previously corrected JES. Many factors such as the precise energy distribution of the jet, and the types of hadrons produced in the jet can influence the jet response. They are described in Table 2.1.

Traditionally, this calibration step must be limited to uncorrelated variables (marked with * in Table 2.1), as otherwise, the corrections could potentially interfere with previous corrections based on another variable. Furthermore, the GSC is applied sequentially: Each variable at a time, the energy response is adjusted to no longer depend on this variable, which positively affects the JER. However, the recently introduced *Global Neural Network Calibration (GNNC)* is not subjected to the same limitations [77]. Since a *Neural Network (NN)* can handle the correlated input features, they can all be treated simultaneously, and it is even possible to add additional variables to the calibration. This led to a JER improvement of over 15% on average and maximally 25% compared to the GSC.

Residual *in situ* Calibration Finally, to account for physics processes that cannot be simulated efficiently or are only present in experimental data, several steps of the *in-situ* calibration (‘pink’ box in Figure 2.10) have been developed. This is

Tab. 2.1.: List of variables used as input for the **GNNC**, variables denoted with * are also used for the **GSC** [77].

Category	Variable	Description
Calorimeter	$f_{LAr,0-3}$	The E_{frac} measured in the 0th-3rd layer of the LAr calorimeter.
	$f_{Tile,0*-2}$	The E_{frac} measured in the 0th-2nd layer of the Tile calorimeter.
	$f_{HEC,0-3*}$	The E_{frac} measured in the 0th-3rd layer of the HEC .
	$f_{FCAL,0-2}$	The E_{frac} measured in the 0th-2nd layer of the FCal .
	$N_{90\%}$	The minimum number of clusters containing 90% of the jet energy.
Jet Kinematics	η^{det}	The η of the jet relative to the geometric centre of the detector.
Tracking	w_{track}^*	The average p_T -weighted transverse distance in the $\eta - \phi$ plane between the jet axis and all tracks of $p_T > 1$ GeV ghost-associated with the jet.
	N_{track}^*	The number of tracks with $p_T > 1$ GeV ghost-associated with the jet.
	$f_{charged}^*$	The fraction of the jet p_T measured from ghost-associated tracks.
Muon Segments	$N_{segments}^*$	The number of muon track segments ghost-associated with the jet.
Pileup	μ	The average number of interactions per bunch crossing.
	N_{PV}	The number of reconstructed primary vertices.

referred to as a *relative* calibration because, unlike the absolute calibration the **reco** quantities are corrected with respect to other **reco**-level and not truth-level quantities. Furthermore, it is also termed *relative* because it is applied only to data, to account for discrepancies between data and **MC** samples, whereas previous calibration steps are applied to both data and simulation.

Therefore, the p_T response is given by:

$$\mathcal{R} = \left\langle \frac{p_T^{probe}}{p_T^{ref}} \right\rangle, \quad (2.10)$$

where both the *reference* and *probe* jet are **reco** jets. To do so, it is common practice to work with special event topologies, which enable leveraging momentum conservation.

For instance, when a jet recoils against a well-calibrated object such as a photon or Z -boson, the jet's energy can be related to the well-calibrated object's energy. The reference object can also be a jet in the central region of the detector ($|\eta| < 0.8$) compared with a probe jet in the forward region ($0.8 < |\eta| < 4.5$), which is the η -*intercalibration*. It aims to reduce the dependency in η of the **JES**. Finally, the *multijet balance* technique calibrates a high- p_T probe jet against multiple jets with lower p_T .

In summary, jet calibration is a tedious, time-consuming yet essential part of the ATLAS data preparation chain, demanding constant efforts to maintain and improve the efficiency and precision of these methods. In particular, the use of **ML** seems like a promising approach to merge several of the standard calibration

steps or include additional (often correlated) variables in the calibration (as shown with the [GNNC](#)).

2.3.3 Dijet Events and Asymmetry

Dijet events are specific event topologies that cover the entire p_T range relevant for the study of proton-proton collisions at the [LHC](#), making them important objects for jet calibration, [JER](#) estimation and a wide range of physics analyses. Therefore, the jet calibration project described in Part [II](#) will focus on dijet events.

Dijet events describe events containing two jets with large transverse momentum that dominate over other (softer) jets. To have well-defined and clean dijet events, one needs two leading jets to take up most of the collision energy because that implies that they have approximately the same transverse momentum (due to energy and momentum conservation).

In practice, there are almost always some additional jets present in a proton-proton collision event. Therefore, it is necessary to set selection criteria on how much energy can be attributed to the third-most energetic jet and on how large the opening angle between the two leading jets should be. The tighter the cut, the cleaner the dijet event, but the fewer events are available to work with.

Dijet Asymmetry The *dijet asymmetry* is defined as follows:

$$A = \frac{p_T^{prob} - p_T^{ref}}{p_T^{avg}}, \quad \text{where } p_T^{avg} = (p_T^{prob} + p_T^{ref})/2. \quad (2.11)$$

If the dijet events were perfectly clean and no third jet (or more) were present, the dijet asymmetry would always be zero by momentum conservation. However, in reality, the asymmetry values are distributed around zero with a non-zero standard deviation. Starting from Eq.(2.11), one can write the asymmetry's standard deviation using the relative [JER](#) of the probe and reference jet:

$$\left(\sigma_A^{prob}\right)^2 = \frac{\left(\sigma_{p_T}^{prob}\right)^2 + \left(\sigma_{p_T}^{ref}\right)^2}{\langle p_T^{avg} \rangle^2} = \left\langle \frac{\sigma_{p_T}}{p_T} \right\rangle_{prob}^2 + \left\langle \frac{\sigma_{p_T}}{p_T} \right\rangle_{ref}^2, \quad (2.12)$$

where the last step is valid because $\langle p_T^{probe} \rangle = \langle p_T^{ref} \rangle = \langle p_T^{avg} \rangle$ after calibration and in a given p_T bin. Usually, the probe object is chosen to be in the central part of the detector (i.e. $|\eta^{prob}| \in [0.2, 0.7]$). If both the reference and probe

object are in the central part (i.e. $|\eta^{prob,ref}| \in [0.2, 0.7]$), reference and probe jets can be interchanged and Eq.(2.12) simplifies to

$$\sigma_A = \sqrt{2} \left\langle \frac{\sigma_{pT}}{p_T} \right\rangle, \quad \forall |\eta| \in [0.2, 0.7]. \quad (2.13)$$

Furthermore, the asymmetry's standard deviation due to detector effects is given by the difference between the reconstructed and the true asymmetry standard deviation:

$$\left(\sigma_A^{det}\right)^2 = \left(\sigma_A^{reco}\right)^2 - \left(\sigma_A^{truth}\right)^2. \quad (2.14)$$

This distinction is due to various physics effects such as out-of-cone radiation [54]. Replacing σ_A^{reco} by the expression derived in Eq.(2.13) and rearranging the terms, one finds:

$$\left\langle \frac{\sigma_{pT}}{p_T} \right\rangle^2 = \frac{1}{2} \left(\sigma_A^{det}\right)^2 + \frac{1}{2} \left(\sigma_A^{truth}\right)^2, \quad \forall |\eta| \in [0.2, 0.7]. \quad (2.15)$$

JER Estimation In the central part of the detector, the standard deviation of the dijet asymmetry is directly related to the relative **JER**. It follows that the spread of the dijet asymmetry distribution can be used as an estimator of the **JER**.

2.4 ATLAS Data Quality Monitoring (DQM)

To ensure the data collected with the ATLAS detector satisfies the high-quality requirements of downstream physics analyses, **DQM** and assessment procedures are necessary. The **LHC** collides bunches of protons or heavy ions, and physics data is registered when the particle beam is stable, and the ATLAS sub-detectors are ramped up and ready. A data sequence that has been recorded without interruption is called a run and can be further split into **Luminosity Blocks (LBs)**. A **LB** is the standard unity of granularity for **DQM** activities and lasts approximately 60 s during which the detector and collider configurations are assumed to be stable [78–80].

The standard ATLAS data flow, after triggering, is visualised in Figure 2.11. Three types of data streams are distinguished: The *calibration stream* (green in Figure 2.11) consists of data needed for specific calibration purposes such as electronic noise monitoring or detector alignment.

Next, the *physics streams* (black in Figure 2.11) contain all events that passed trigger requirements and might be relevant for physics analyses. This includes the *main stream*, consisting of pp collisions, the *hard probes* [81] stream and the

Ultra-Peripheral Collisions (UPC) [82] stream, both containing heavy-ion events. The hard probes stream refers to head-on collisions of the heavy ions inducing a hard scattering event between the partons of the two nuclei. Differently, the *UPC* stream contains heavy-ion collisions, where no hadronic interaction occurs, because the impact parameter between the ions is larger than the sum of their radii. In this case, the event is dominated by electromagnetic interactions. Finally, the *express stream* (orange in Figure 2.11) is a representative subset of data from the physics and calibration stream. Events in the express stream are rapidly reconstructed at the ATLAS Tier 0 computing facility and provided for offline *DQM*. A specific type of express stream is the *Cosmic Calo* stream, which contains events triggered during empty bunch crossings when no collisions are expected.

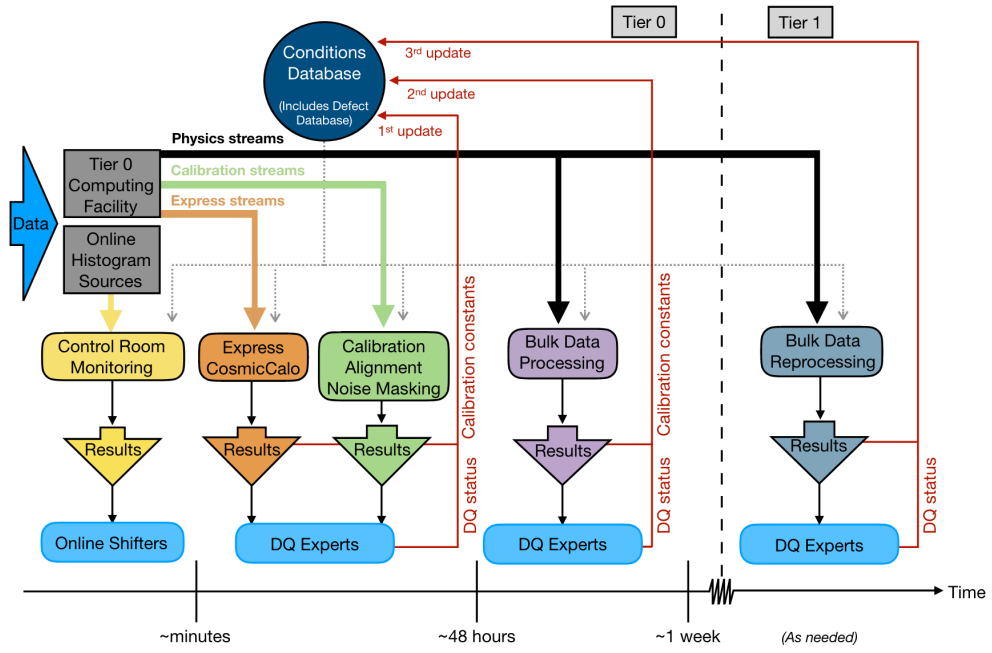


Fig. 2.11.: Schematic representation of operations workflow for the ATLAS data quality assessment. Figure from [79].

The incoming data are continuously monitored by online shifters. Small subsets of data are then rapidly reconstructed and assessed by data quality experts. Finally, after updating the calibration, the entire data are reconstructed in bulk processing. Any problem affecting the data quality is logged in a dedicated defect database. *LBs* that are not problematic are declared ‘good’ and added to the *Good Runs List (GRL)* and can then be employed for physics studies [78, 79].

For the work discussed in this thesis, we focus on the challenges associated with the *DQM* of the *LAr* calorimeter.

2.4.1 Liquid Argon Calorimeter DQM

In general, the ATLAS LAr calorimeter records high-quality data. Over 99% of the recorded data are considered of good quality, making detector defects and issues a problem that contaminates less than 1% of the data as shown in Figure 2.12. However, these issues need to be identified and the associated events discarded (with a padded window around them) as their impact on important variables such as missing p_T is non-negligible.

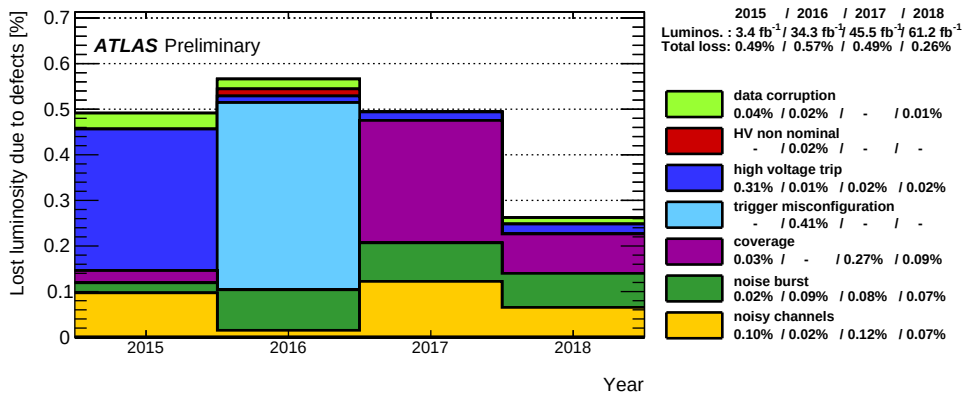


Fig. 2.12.: Data rejection by defect assignment for the datasets of run 2 for the most optimal processing. Figure from [83]

Common detector defects for the LAr calorimeter entail High Voltage (HV) trips, limited coverage, noisy channels, noise bursts and data corruption issues. Since some of these issues last much shorter than a LB, it is possible to only reject small time windows instead of the entire LB. This is called the *time window veto* procedure and is done to reduce data loss.

Since most of the detector issues are recurrent and well understood, it can be envisioned to automate part of the detector monitoring. So far, this has only been partially implemented in practice through the use of automated techniques comparing histograms of specific variable distributions against their expected shapes. However, such approaches overlook the temporal correlations between events that are still present in detector-level data. As a result, current DQM procedures still demand a lot of time and effort from human shifters and some defects might be missed [62, 78, 80].

High Voltage (HV) Trips The voltage of the electrodes in the LAr calorimeter strongly affects the number of collected signals. The most common defect is due to a current spike potentially leading to a sudden drop in HV. After such a trip, the HV has to be reduced and then gradually ramped back up to nominal values.

Coverage The [LAr](#) is designed to provide complete cover in azimuth and up to $|\eta| = 4.9$ in pseudorapidity. A faulty sector can reduce the coverage of the calorimeter and the corresponding data are rejected due to a coverage defect.

Data Corruption The recorded data can be corrupted due to issues in the [FEB](#) such as desynchronisation between the [FEB](#) and the central [LHC](#) clock. Such events are discarded and the [FEBs](#) must be reconfigured.

Noisy Channels Special calibration runs determine the electronic calibration constants for the calorimeter channels and thus help identify problematic channels. Such channels can be masked and subtracted accordingly, before forming topoclusters.

Noise Bursts The [LAr](#) calorimeter is periodically affected by large bursts of coherent noise, which become more frequent as the instantaneous luminosity increases. Such *noise bursts* can last up to 1000 ns, and in rare cases, up to a few μs (see Figure 2.14b). A typical noise burst event is shown in Figure 2.13. Furthermore, Figure 2.14a shows the relation between the number of noise

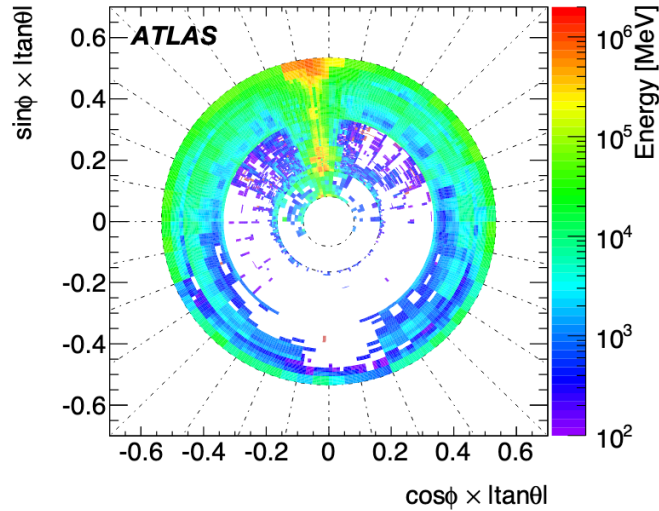
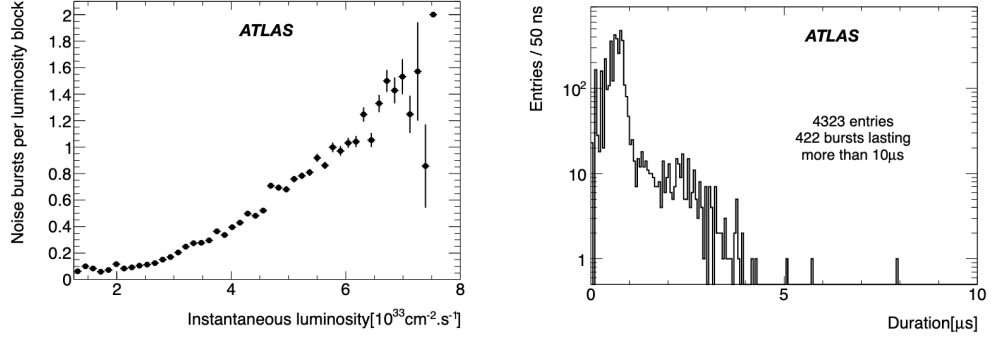


Fig. 2.13.: Example of a typical noise burst, observed during a Cosmic Calo stream, where no collisions are expected. The energy deposits are shown in the transverse plane of the [EMEC](#) partition A. Figure from [80].

bursts and the instantaneous luminosity and Figure 2.14b the distribution of noise burst durations.

The standard algorithm that was developed to identify noise bursts, checks two boolean flags: First, if more than five [FEBs](#) contain more than 30 channels with

a quality factor $Q > 4000$, then the *standard flag* is set to True. Second, if more than 20 channels have an energy > 1 GeV and a saturated Q value, the *saturated flag* is set to True. Both flags are sensitive to different types of noise bursts; the standard flag identifies widely spread noise bursts better, whereas the saturated flag is specialised for locally confined noise bursts. Such noisy events can be rejected with the previously mentioned *time window veto* procedure. A rejected time window is typically extended by $100\mu\text{s}$ in both directions to capture peripheral events of the noise burst.



(a) Distribution of the number of noise bursts as a function of instantaneous luminosity. (b) Distribution of the number of noise bursts as a function of their duration limited at $10\mu\text{s}$.

Fig. 2.14.: Distributions characterising noise bursts. Figures from [80].

Identifying and correctly categorising such detector issues, requires substantial human input. In this context, ML-based anomaly detection offers a promising avenue to alleviate this effort. Our approach to this problem is detailed in Chapter 15.

Machine Learning

ML and **Deep Learning (DL)** are fundamentally transforming data analysis, computing, and are thus also affecting the natural sciences. **HEP**, in particular, generates enormous volumes of high-dimensional data, making it an ideal domain for the application of **ML** techniques. These methods have therefore become essential tools for a wide range of tasks in **HEP**: from real-time event triggering and data reconstruction to object selection, background estimation, and physics simulations [84].

The projects discussed in this thesis all involve the implementation and exploration of **ML** methods, emphasising their central role in modern **HEP**. The following sections provide a more detailed discussion of these techniques, their origins, underlying concepts and applications in physics.

3.1 Introduction to Machine Learning

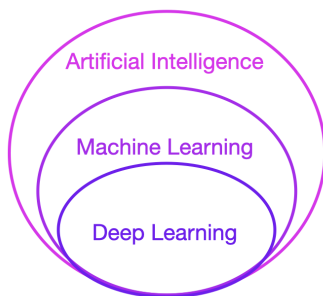


Fig. 3.1.: Schema of **Artificial Intelligence (AI)** and its sub-fields.

ML is commonly described as the field of computers learning how to solve problems ‘without explicitly being programmed’. It is unclear who exactly first coined this expression, but the formulation started appearing in the middle of the last century and is still widely used. It is often attributed to Samuel [85], who also introduced the term **ML** for his work on programming computers to play checkers. Slightly differently, Murphy [86] described **ML** as the task of analysing and finding patterns in data, using them to uncover more patterns or predict future patterns.

In general, **ML** is considered a sub-field of **AI** and then again **DL** a sub-field of **ML**, as visualised in Figure 3.1, even though some of these terms are sometimes used interchangeably.

One usually distinguishes between *classical/probabilistic* and *deep/modern ML*. Classical **ML** encompasses various optimisation methods not based on **NNs** such as linear regression [87], k-nearest neighbours algorithms [88], **Boosted**

Decision Trees (BDTs) [89], Support Vector Machines (SVMs) [90] to name a few. They are also called statistical models, as most of them have their origins in statistics. One might notice that some of these algorithms existed long before the term ML came into use. An example would be linear regression with least squares optimisation that was formally introduced by Legendre [87] in 1805 but was already applied before. There are still new classical algorithms being developed, but what has been creating most of the recent excitement in ML are so-called modern ML or DL algorithms.

DL focuses on multi-layered NNs, commonly referred to as *deep NNs*. The field evolved out of efforts to imitate a human brain, hence why nodes in a NN are called neurons, even though current NNs and biological neurons share little resemblance. However, some concepts in DL have been inspired by neuroscience, such as the need to have neurons representing non-linear functions (because human brain signals resemble spikes) and the concept that multiple neurons working together are ‘more intelligent’ than a single neuron.

The beginnings of DL are also often linked to Rosenblatt [91], who introduced the Multilayer Perceptron (MLP) in 1958, which corresponds to something similar to a three-layer NN although the weights are not yet learned in the same way as today. The next important step for DL was backpropagation, introduced by Rumelhart et al. [92] in 1986, which allows for adjusting the weights of a NN to optimise a cost function.

Even though many important concepts of DL were already described approximately 50 years ago, most of these algorithms could not be successfully implemented due to the lack of training data and limited computational resources. Only with the increase of computational power and the growing availability of data, as seen towards the end of the last century, did those concepts become applicable and start showing impressive performances in image and speech recognition.

It was during this phase in the 1990s that LeCun et al. [93] developed the Convolutional Neural Network (CNN) for automated image classification. This was largely possible thanks to the MNIST database, a benchmarking database consisting of handwritten digits, that allowed for the efficient development and comparison of classification algorithms. Furthermore, this unlocked progress in many ML classification tasks and despite being relatively easy to solve for modern techniques, MNIST remains a widely used benchmark for a proof-of-concept of new classification models. Therefore, this database is a prime example of how crucial well-prepared benchmark data are for the development of learning algorithms regardless of their application fields [86, 94, 95].

There are countless examples impressively showcasing the progress learning algorithms have made in the last 50 years, but there are a few worth mentioning.

Among them are certainly the DL models that beat the world champion of chess in 1997 [96] and the world champion of Go in 2016 [97], respectively. Both appeared to be near impossible learning tasks, given the countless possible moves in each game and were studied for years until finally succeeding.

Furthermore, recently, a lot of talk has originated from progress made in generative AI and Natural Language Processing (NLP). This was largely possible due a novel model architecture called *transformer* which was introduced in 2017 [98]. Its success is commonly attributed to its self-attention mechanism, which allows the model to learn which parts of data are most relevant. More details to transformer models are discussed in Section 3.3.3.

Based on transformers, Large Language Models (LLMs) have been developed, forming models with billions of parameters (hence the ‘large’). The currently most popular example is the AI-chatbot ‘ChatGPT’ [99], which is astonishingly good at answering various questions and solving reasoning problems, while interacting with users in a shockingly human-like way. Ever since the introduction of ‘ChatGPT’ in 2022, the field of generative AI has been rapidly evolving whilst regularly presenting new, shocking results of its growing capacities in generating text, image, music or software [100]. This increasingly fast development also raised many critical voices, questioning the safety of these models and the potential risks that come with their use [101]. There is still much research left needed to improve our understanding of such models.

3.1.1 Learning Paradigms

For the learning aspect, one typically distinguishes between *supervised*, *unsupervised* and *reinforcement* learning algorithms [86, 94, 95, 102, 103].

Supervised Learning In *supervised learning*, the goal of a learning model is to correctly assign or predict target variables for previously unseen testing data. During training, the target variables are available and the learning model’s output can be compared to the true target variables to guide the learning.

Supervised learning is often used for *classification* or *regression* but is also employed for tasks like information retrieval, or fine-tuning of LLMs. In the context of classification the target variables are typically referred to as (*class*) *labels*. Binary classification models sort data into two categories (e.g. an image recognition model distinguishing cats from dogs), whereas multi-class classification models attribute N different class labels to the data (e.g. an image recognition model identifying ten different kinds of animals). The output variable of classification algorithms is always categorical. Differently, regression

models are tasked with predicting continuous target variables (e.g. predicting housing prices based on location, surface and other relevant features).

Unsupervised Learning Different from supervised learning, in the *unsupervised* approach, there are no labels or target variables available in the training data. Therefore, an unsupervised learning model aims to unveil hidden categories or patterns of data without prior knowledge about them.

A classical example is [Principal Component Analysis \(PCA\)](#), which learns a linear projection of data that is often useful for dimensionality reduction [104]. For this, the first principal component is chosen such that it maximises the variance in the data. The next principal component must then be linearly independent of the first one (i.e. orthogonal) and again have the highest possible variance. This can be continued until the number of components corresponds to the dimensionality of the initial data, but by construction, only the first few principal components contain important information.

A more modern example would be a [Generative Adversarial Network \(GAN\)](#) which consists of a generator and a discriminator model [94]. The generator tries to generate new data, looking like was produced by the same mechanism as the input data, while the discriminator tries to distinguish between the true input data and the generator's output. Despite being difficult to train, [GANs](#) are very popular for data generation and augmentation.

Depending on the exact application, there also exist intermediate learning approaches such as *weakly-supervised* learning where only limited amounts of target data are available in training or *semi-supervised* learning, where the training data only contain one class of data, while the testing data contain more classes. Finally, one speaks of *self-supervised* learning, when the learning model creates its own target variables from the input data and trains with those [94, 105].

Reinforcement Learning In *reinforcement* learning, the goal is to teach an autonomous agent, using punishments and rewards, how to solve a task without (or very little) human intervention. In the context of [HEP](#), reinforcement learning is studied for continuous particle accelerator tuning [106, 107] or online detector monitoring [108]. Besides these first applications, reinforcement learning is mainly of interest for the field of robotics and will not be discussed any further in this thesis.

However, since reinforcement learning keeps providing remarkable results, two further examples are included here. First, the previously introduced AlphaGo, the first computer program to beat a human professional player in a full-sized game of Go, was trained with reinforcement learning [97]. Second, the most

recent use of reinforcement learning might be the DeepSeek model [109] that was released at the beginning of 2025. With the use of reinforcement and ensemble learning, the authors managed to train a LLM at a fraction of the training cost required for established LLMs as OpenAI-o1-1217, whilst achieving a comparable performance on reasoning tasks.

3.1.2 Loss Function

In general, the objective of learning models is to solve an optimisation problem by minimising a cost function. In ML, one typically refers to this function as the *loss function* which the model aims to minimise. It can also be regarded as a measure of the quality of the model's prediction.

The effectiveness, performance, and overall utility of a learning model are strongly influenced by the choice of its loss function. The optimal loss function depends on the nature of the data and the specific problem being addressed. Further details on loss functions are provided within the context of each specific application.

Maximum Likelihood Estimation Depending on the problem or the data one is working with, it is not always obvious what kind of loss function is optimal for a given learning problem. However, many supervised ML problems can be interpreted as maximum likelihood estimations to analytically motivate the choice of the loss function [94, 95, 102].

Maximum likelihood estimation provides a statistical method to relate the underlying probability distribution of the target variable y given some data x and parameters θ . This is useful when facing a regression problem with observed data $x = (x_1, \dots, x_n)^T$ and one wants to predict the target variable $y = (y_1, \dots, y_n)^T$ given x . For this, we define the likelihood function as $L(\theta) = p(y|x, \theta)$ where θ are parameters of the conditional probability distribution. Now, the maximum likelihood principle states that the best choice of parameters θ to relate y to x is the one that maximises $L(\theta)$. Formally, we are searching for:

$$\theta^* = \arg \max_{\theta} L(\theta) = \arg \max_{\theta} p(y|x, \theta) \stackrel{iid.}{=} \arg \max_{\theta} \prod_i p(y_i|x_i, \theta) \quad (3.1)$$

The last equality is allowed because the data are assumed to be **independent and identically distributed (iid.)**, which is a common assumption in ML. Since the maximisation of the likelihood is more prone to computational issues such as numerical underflow, it is common practice to opt for the mathematically equivalent solution of minimising the logarithm of the likelihood. This transformation

is allowed because the logarithm is a monotonically increasing function. Finding the maximum likelihood then becomes minimising the negative logarithmic likelihood:

$$\theta^* = -\arg \min_{\theta} \log L(\theta) = -\arg \min_{\theta} \sum_i \log p(y_i|x_i, \theta). \quad (3.2)$$

To provide a common example, let us look at linear regression. First, the target variables $\mathbf{y} = (y_1, \dots, y_n)$ are assumed to depend linearly on the inputs $\mathbf{x} = (\mathbf{x}_1, \dots, \mathbf{x}_n)$, i.e. there are parameters $\mathbf{w}$ such that $\mathbf{y} = \mathbf{w}^T \mathbf{x} + \epsilon$ where the noise $\epsilon \sim \mathcal{N}(0, \sigma)$ is Gaussian with mean 0 and variance σ^2 . Then,

$$p(\mathbf{y}|\mathbf{x}, \mathbf{w}, \sigma) \stackrel{iid.}{=} \prod_{i=1}^n \mathcal{N}(y_i|\mathbf{w}^T \mathbf{x}_i, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \prod_{i=1}^n \exp\left(-\frac{(y_i - \mathbf{w}^T \mathbf{x}_i)^2}{2\sigma^2}\right) \quad (3.3)$$

where N is the dimension of the input and target data. Next, we take the negative logarithm of $p(y_i|\mathbf{x}, \mathbf{w}, \sigma)$:

$$-\log p(y_i|\mathbf{x}_i, \mathbf{w}, \sigma) = \frac{1}{2} \log(2\pi\sigma^2) + \frac{1}{2\sigma^2} (y_i - \mathbf{w}^T \mathbf{x}_i)^2 \quad (3.4)$$

and insert it into Eq.(3.2) to find the optimal estimator:

$$\mathbf{w}^* = \arg \min_{\mathbf{w}} \left[\frac{n}{2} \log(2\pi\sigma^2) + \frac{1}{2\sigma^2} \sum_{i=1}^n (y_i - \mathbf{w}^T \mathbf{x}_i)^2 \right] \quad (3.5)$$

$$= \arg \min_{\mathbf{w}} \sum_{i=1}^n (y_i - \mathbf{w}^T \mathbf{x}_i)^2 \quad (3.6)$$

where the first term and the constant factor of the second term are omitted because they are irrelevant for the optimisation in $\mathbf{w}$. The final result in Eq.(3.6) indicates that the maximum likelihood estimator $\mathbf{w}^*$ coincides with the least squares solution. Therefore, under the assumed linear Gaussian conditions, the least squares estimation is the optimal loss function for linear regression [94, 95, 102].

3.1.3 Generalisation, Overfitting and Underfitting

The central challenge of ML is to train an algorithm on training data such that it performs well on new, unseen data stemming from the same data-generation process. This is referred to as *generalisation*.

The degree of generalisation can be estimated by evaluating the loss function on the training data (*training error*) and the test data (*test error*). If the model does not generalise well, one speaks of *over-* or *underfitting*. Underfitting occurs when the model does not have a sufficient number of parameters to capture

the underlying data structure, leading to a large training error. Meanwhile, the model is overfitting if the training error is low but the test error is much larger than the training error. This corresponds to the situation where the model has more parameters than needed to describe the data structure, thereby learning about residual variations (i.e the noise) in the training data [94, 95].

3.2 Machine Learning for Particle Physics

Machine learning has been actively used, adjusted and tested in HEP for years, but its use further surged with the launch of the LHC. With roughly a billion proton-proton collisions per second, the LHC produces unprecedented amounts of data which are comparable with some of the largest industry datasets [50, 110, 111]. Initially, the operating LHC was recording around 15 PB (1 PB = 10^3 TB = 10^6 GB) of data per year after applying online data reduction techniques [50]. With the growing luminosity and beam energy of the LHC, this number has significantly increased; the average amount of recorded collision data by the LHC experiments in 2024 added up to almost 3PB per day [112]. This pushes current data acquisition, trigger and data monitoring systems to their limits but also makes the LHC a perfect ‘big data’ playground for developing and testing new ML algorithms. A illustration of these dataset sizes in recent years is shown in Figure 3.2.

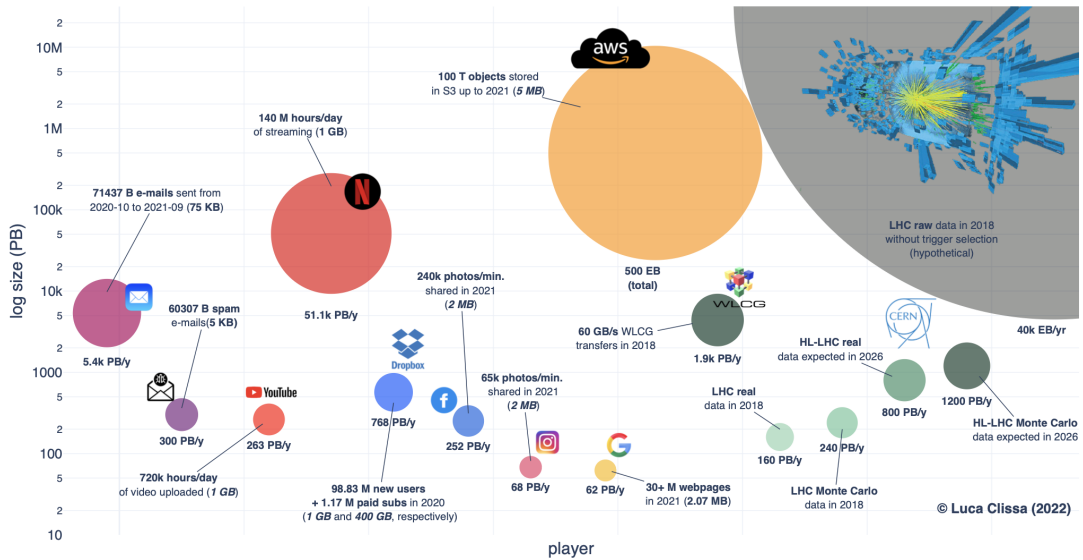


Fig. 3.2.: Illustration of data sizes for several big data players for data streaming, storage and production. Figure from [111].

The HEP community has always put a strong emphasis on statistics and data analysis techniques, and it is not surprising that they have continuously made

efforts to keep up with developments in ML. The first [International Workshop on Software Engineering, Artificial Intelligence and Expert Systems for High-energy and Nuclear Physics \(AIHENP\)](#) was already held in 1990. It has since then evolved into the [Advanced Computing and Analysis Techniques in Physics Research \(ACAT\)](#) workshop, whose 23rd edition is taking place in 2025. During the 1990s, particle physicists at the Tevatron and LEP colliders applied early NNs to track and cluster finding [113, 114], jet identification and jet triggering [115–117] and particle identification [118–120].

Starting from the 2000s, BDTs got heavily introduced to particle physics, they were used for particle identifications in MiniBooNE [121, 122], helped with signal-background separation in the D0 experiment [123, 124] and BaBar experiment [125] and they were also involved in the discovery of the Higgs boson by the ATLAS and CMS experiments in 2012 [9, 10].

In the last ten years, ML algorithms have steadily been developed and applied to almost every aspect of data processing, monitoring and analysing at the LHC. The living review of ML for particle physics [126] provides a regularly updated list of references applying ML to experimental, phenomenological or theoretical HEP, and also documents the recent surge in publications on ML applications for physics (see Figure 3.3) [84, 110, 127–129].

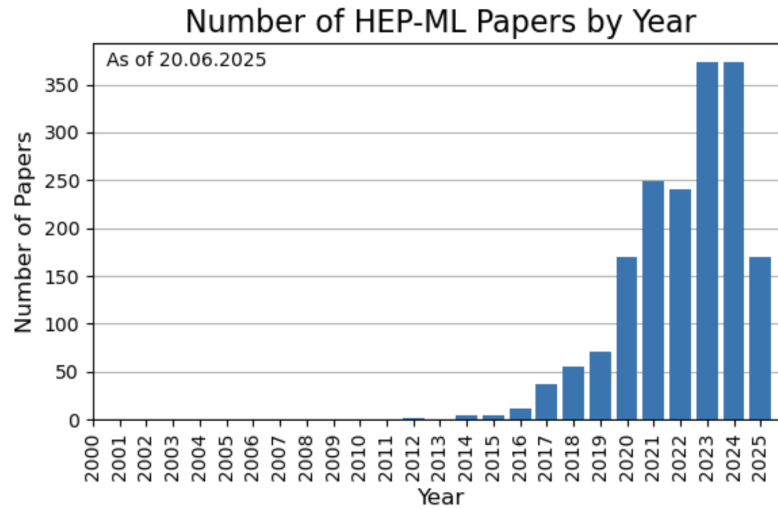


Fig. 3.3.: The evolution of ML papers in the HEP community over the years. Figure from [126].

The need for more efficient and better performing algorithms is currently even more acute, given that the HL-LHC is planned to start operating in 2030 [58]. The HL-LHC is an upgraded version of the LHC with a much higher density of collisions (up to five times the luminosity of the current LHC) and therefore, a massively increased data flux. The ATLAS HL-LHC conceptual design report [56, 57] estimates that ATLAS can only handle the expected amounts of data with aggressive R&D and that ML might be the key to tackle this [127, 128].

The effort on integrating ML for physics' applications extends well beyond particle physics; on the contrary, ML has also become an established tool across various sub-fields of physics, such as astrophysics, cosmology, gravitational waves physics and theoretical physics [130]. Initiatives have been started to unify these efforts and develop common strategies for addressing the shared challenges faced by these domains. On the European level, the [European Coalition for AI in Fundamental Physics \(EuCAIF\)](#) was established recently to serve this purpose [131].

3.3 Deep Learning

As mentioned previously, DL is a sub-field of ML that employs multi-layered NNs. One of the biggest conceptual advantages that DL has over classical learning is that DL models can learn how to best represent data themselves [132]. While some classical models work very well when combined with feature extraction methods or with an adequately chosen latent space representation, these choices must be made by the user and depend on the user's domain knowledge. Differently, DL methods can learn an internal representation of the raw input data that is the most useful for completing their learning task. This representation learning has allowed DL methods to make huge progress in image and speech recognition, biological problems such as protein folding (which is crucial for drug discovery) [133] and more recently also NLP [99].

3.3.1 Neural Networks

DL is, as its name implies, based on the use of deep NN for learning tasks. Each layer of a NN can be represented by a non-linear function $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$, written as:

$$f(\mathbf{x}; W, \mathbf{b}) = g(W^T \mathbf{x} + \mathbf{b}), \quad \mathbf{x} \in \mathbb{R}^n, W \in \mathbb{R}^{m \times n}, \mathbf{b} \in \mathbb{R}^m \quad (3.7)$$

where W is the so-called weight matrix, $\mathbf{b}$ the bias, g the activation function, n the input dimension and m the output dimension of the network. Both W and $\mathbf{b}$ are parameters of the network and can be modified.

A NN usually consists of multiple concatenated layers, where the first layer is referred to as the input layer, the last as the output layer and all intermediate ones as hidden layers. Each node of a layer is connected to each node of the next layer, and computations are typically done in the *forward* direction, i.e. from the

input nodes across the hidden layers to the output layer. Mathematically, the k -th layer then corresponds to:

$$f_k = g(W_k^T f_{k-1} + \mathbf{b}_k), \quad \text{where } k > 0, f_0 = \mathbf{x}, \quad (3.8)$$

where $f_k : \mathbb{R}^{m_{k-1}} \rightarrow \mathbb{R}^{m_k}$ and the weight matrices $W_k \in \mathbb{R}^{m_k \times m_{k-1}}$ and biases $\mathbf{b}_k \in \mathbb{R}^{m_k}$ are chosen to have compatible dimensions. Also, the activation function is typically not applied to the output layer or corresponds to the identity function in that case.

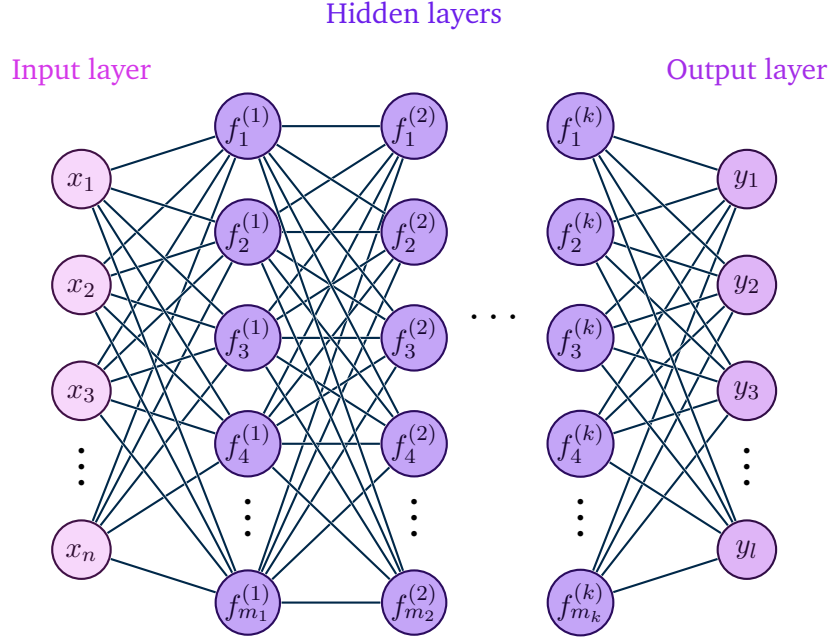


Fig. 3.4.: Schema of a NN with k hidden layers of variable dimensions $\{m_1, m_2, \dots, m_k\}$. The computations are done from left to right, i.e. the forward direction, whereas the differentiation with back-propagation is done from right to left. Inspired by [134].

Schematically, a NN can be represented as shown in Figure 3.4, where each layer represents the function f_k of the network and each node is the i -th element $f_k^{(i)}$. The depth of a neural network depends on the number of hidden layers it has: networks with more hidden layers are considered deep, whereas those with fewer are deemed shallow.

A NN as described above and shown in Figure 3.4 is also called a MLP, an *artificial neural network*, a *dense neural network* or a *fully-connected neural network* since every node is connected to every node of the following layer.

Most of a NN's generalisation power is based on its *activation function*. The activation function must be non-linear, otherwise the function f_k becomes a linear combination of linear functions, which is again a linear function and

severely limits the expressiveness of a **NN**. However, it should also be (mostly) differentiable to be able to apply the gradient descent algorithms. A popular activation function choice is the so-called **Rectified Linear Units (ReLU)**, which is described by $g(z) = \max(0, z)$ [135]. They got popular after their successful application to Boltzmann machines in 2010 [136]. Since these functions are either linear or constant, their derivatives are very simple except for $z = 0$ where the derivative is not defined. In practice, this is acceptable, as one point barely affects the network training and software implementations usually settle for one-sided derivatives in such cases. Another, more important drawback of **ReLU** activation functions is that they cannot learn from inputs equal to zero. To mitigate these issues, there exist many slight variations of **ReLU**s that are currently, such as leaky **ReLU** and **Exponential Linear Units (ELUs)**. Other popular choices for activation functions before **ReLU**s were the hyperbolic tangent function $g(z) = \tanh(z)$ and the sigmoid function $g(z) = 1/(z + e^{-z})$ which are visualised in Figure 3.5 together with **ReLU** activation functions. The softsign function given by $1/(z + |z|)$ is very similar to the hyperbolic tangent function but converges polynomially instead of exponentially.

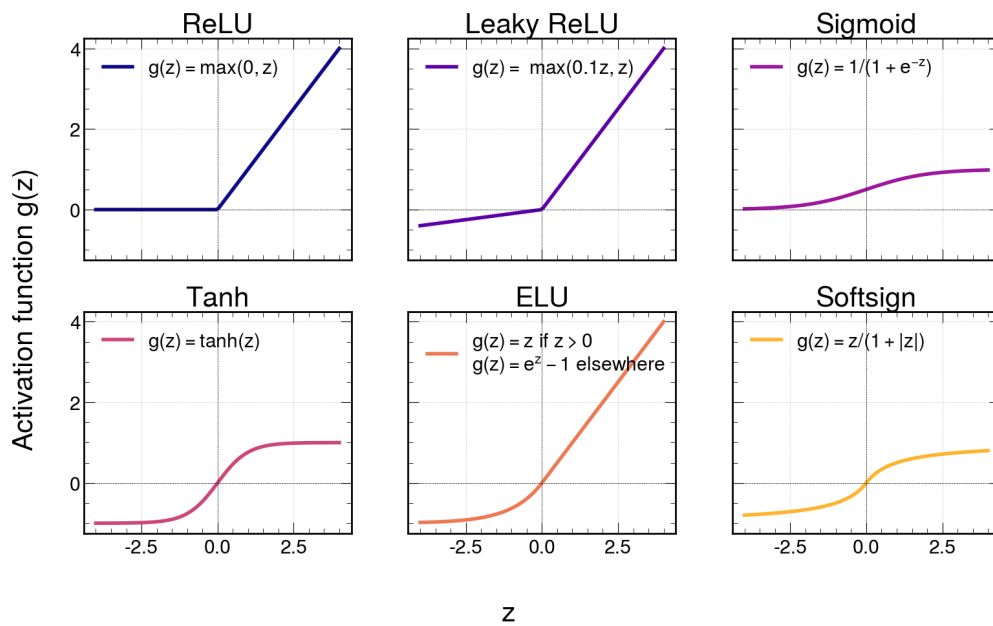


Fig. 3.5.: Popular activation functions for a **NN**.

It has been proven that **NN** can arbitrarily well approximate any continuous function under certain conditions [137, 138]. There are still ongoing efforts to prove the universal approximation power of **NN** in more general settings, since there are still many remaining gaps in the theoretical understanding of **NN**s.

3.3.2 Learning Weights through Backpropagation

Given a problem with a cost function that needs to be optimised, it is essential to adjust the weights W_k, b_k of the network to achieve the desired output. For this, *back-propagation* is needed. Since the computations always evolve in the forward direction (from left to right in Figure 3.4), a layer's output depends on the outputs from the previous layer. By using the chain rule of derivation, it can be shown that the gradient of a layer f_k solely depends on the derivations of the previous layers and therefore the gradient computation goes *backwards* through the layers. The introduction of the back-propagation algorithm [92] was essential as it allowed to efficiently compute the gradient of a NN and to use gradient-based techniques for finding the minimum of the loss function.

A *gradient descent* technique is an iterative method to find a function's extremum by taking small steps into the opposite direction of the derivative. This algorithm was already introduced in the 19th century Cauchy [139]. After each step, the new parameters θ' are set to $\theta' = \theta - \alpha \nabla_{\theta} J(\theta)$ where J is the loss function and α the *learning rate* which is given by a small, positive scalar. For intuition, the algorithm can be compared to wandering around blindly in the mountains and trying to find the valley by always taking a step in the opposite direction of the steepest ascent which corresponds to the the steepest descent and repeating this assessment for every step. *Batch* or *deterministic gradient descent* processes the entire training set at every optimisation step, i.e. whenever computing θ' . This can become computationally expensive for large training sets, which is typically the case in DL. Furthermore, gradient-descent strongly depends on the starting point (initialisation) and step size of the algorithm. This illustrated for a two-dimensional hyperspace in Figure 3.6, where different trajectories lead to different outcomes.

A computationally more efficient alternative is the so-called *stochastic (minibatch) optimisation*, which only evaluates the gradient on a randomly chosen sample of the training data (a so-called minibatch) and then averages over it. Stochastic methods only treat one data instance at a time but most DL models work with minibatch stochastic methods. *Minibatch stochastic gradient descent* is currently one of the preferred optimisation methods for DL [94, 95].

For a successful optimisation, the choice of learning rate α is crucial as a too big rate might miss an extremum, whereas with a too small rate, the method might take too much time to converge. Hence, the learning rate is an important hyperparameter of a DL model as it strongly affects its performance. For this reason, several optimisation algorithms with adaptive learning rates have been introduced in the last decade, with the most popular one being the *Adam* optimisation algorithm [141]. The name Adam comes from the adaptive moment

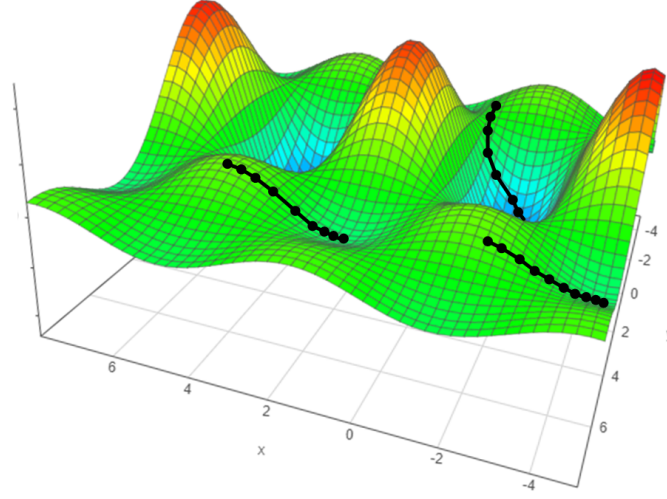


Fig. 3.6.: Gradient-descent technique visualised in a two-dimensional hyperspace. Different trajectories may lead to different local minima and different quantitative results depending on the initialisation or optimiser. Figure from [140].

estimation on which the algorithm is based. The parameter updates assigned by the algorithm are

$$\alpha_t = \alpha \cdot \sqrt{1 - (\beta_2)^t} / (1 - (\beta_1)^t), \quad (3.9)$$

$$\theta_t = \theta_{t-1} - \alpha_t \cdot m_t / (\sqrt{v_t} + \hat{\epsilon}), \quad (3.10)$$

where β_1, β_2 are decay rates which are inputs to the algorithm and t is the number of the optimisation step. Further, the algorithm also estimates the first moment (mean) m_t of the gradient and of the second moment (un-centred variance) v_t of the gradient.

3.3.3 Model Architectures and Physics Applications

By combining and concatenating different **NNs** with various functions, one can construct **DL** models tailored to specific tasks. The following paragraphs present a selection of well-known and influential model architectures that are widely used in **HEP** and relevant to this thesis [84].

Deep Sets

A deep sets model, which was first introduced in 2018 by Zaheer et al. [142], is a **NN** architecture taking sets as inputs. It can be visually represented as shown in Figure 3.7. The most important property of this architecture is that it can

handle inputs of variable length, and the order of the inputs is irrelevant. The latter is assured by the addition of a permutation-invariant layer in the network. The model's architecture is based on a theorem stating that a permutation invariant function $S(x)$, where x are elements of a set X , can be written as

$$S(X) = \rho\left(\sum_{x \in X} \phi(x)\right), \quad (3.11)$$

for suitable transformations ρ and ϕ [142]. Here, the sum guarantees that the order of the elements $x \in X$ does not matter; it follows that $S(X)$ is permutation invariant. Based on the universal approximation theorem [137], one can assume that the two functions ρ and ϕ can be well approximated by two NNs.

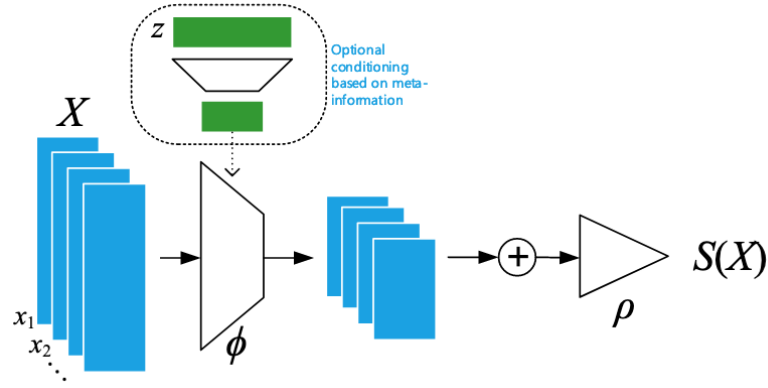


Fig. 3.7.: Schema of deep sets architecture. Figure from [142].

At the LHC, the proton-proton collisions are often interpreted as data instances with variable-length entries per event and no inherent order. This is because the collision events are generally assumed to be independent from one another. The sequential order of the events only can play a role for DQM tasks, where detector issues or might induce some temporal correlations. Given these properties, it is a natural choice to represent collision events as sets. Komiske et al. [143] apply a deep sets model to jet events, showing that the model learns a latent event representation and is able to differentiate between quark and gluon jets. Their Energy flow network also has the advantage of being IRC-safe by construction.

Deep sets are also at the heart of the DIPS algorithm which is used for jet flavour tagging in ATLAS [144, 145]. A faster version of this tagger (called FastDIPS) has been implemented at the trigger level [146]. It runs right after the hardware trigger (L1) but before the remaining HLT reconstruction and reduces the computational costs associated with the track reconstruction needed for b -tagging.

Similarly, LHCb recently implemented a fast inclusive flavour tagger based on

deep sets [147]. Its main task is to identify B mesons at the primary interaction point while considering all particles within an event simultaneously.

Autoencoder (AE)

An **Autoencoder (AE)** model is a type of model architecture that compresses its inputs into a lower-dimensional latent space and then reconstructs them back to the input space [148–150]. The model’s objective is to produce outputs identical to the inputs, hence the typical loss function is the **Mean Squared Error (MSE)** comparing inputs to outputs. A schematic illustration of the **AE** model architecture is shown in Figure 3.8.

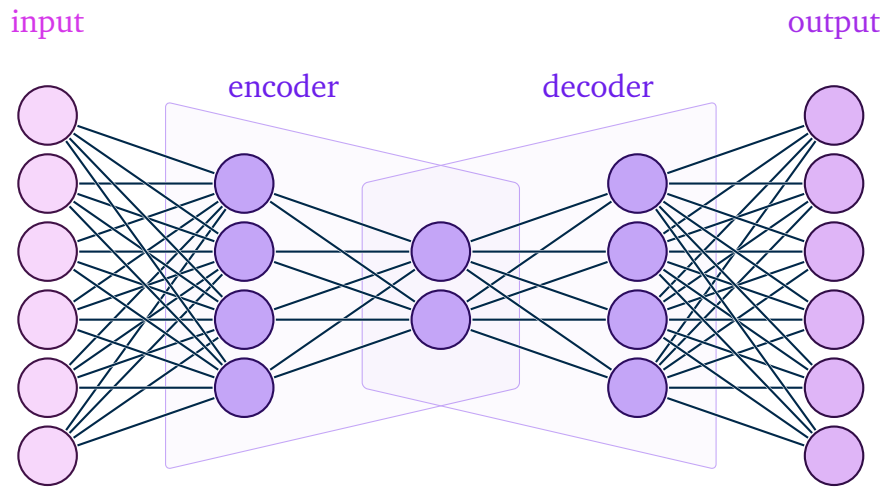


Fig. 3.8.: Schema of **AE** architecture including encoder and decoder (shaded regions) with a latent space dimension $d_l = 2$. Inspired by [134].

An **AE** can be decomposed into an encoder and decoder model: The encoder projects an d_N -dimensional input into a latent space of dimension d_l , whereas the decoder reconstructs the latent space representation back to a variable of dimension d_N . The encoder and decoder models usually consist of a multi-layered **NN** with d_N (d_l) input nodes and d_l (d_N) output nodes, respectively. However, they can also be built using model architectures more adapted for a specific task. Typically, d_l is chosen such that $d_l < d_N$, thus forcing the model to learn an imperfect compression. Therefore, **AEs** are popular methods for dimensionality reduction and feature learning.

Furthermore, they are frequently used for anomaly detection, where one assumes that the **AE** model learns well to compress normal data, but obtains high reconstruction errors for rare anomalous data. By placing a threshold on the reconstruction error, it is possible to distinguish anomalous from normal instances.

It is possible not merely to map an input to a single point in the latent space but into a latent space distribution. To implement this, a [Kullback-Leibler \(KL\)](#) divergence [151] term is added to the loss function. The KL divergence compares two probability distributions $\mathcal{P}$, $\mathcal{Q}$ and is minimal if they coincide. Mathematically, it is given by:

$$\mathcal{D}_{KL}(\mathcal{P}|\mathcal{Q}) = \sum_{x \in \mathcal{X}} \mathcal{P}(x) \log \frac{\mathcal{P}(x)}{\mathcal{Q}(x)}. \quad (3.12)$$

If the latent space distribution is a Gaussian distribution, the model is called a [Variational Autoencoder \(VAE\)](#) [152].

In [HEP](#), autoencoders are extensively used for anomaly detection, be it as a part of a physics analysis or a data quality assessment [153].

For data quality, both, ATLAS [78] and CMS [154, 155] investigate the use of autoencoder-based models for monitoring their respective electromagnetic calorimeters. CMS already deployed this algorithm during 2022 during the [LHC](#) run 3 in the online [DQM](#) workflow.

For physics analyses, [AEs](#) are employed for model-agnostic new physics searches such as semi-visible jets searches [156–159]. These projects introduce different parametrisations of the [AE](#)’s latent space, which are also discussed for QCD vs. dark jet studies [160, 161]. Parametrising the latent space improves the [AE](#)’s ability to identify out-of-distribution data, a task at which it usually fails. It is also possible to conduct anomaly detection in the feature space of a deep sets model using a [VAE](#) such as done for the [ANomaly deTEction on particle fLOW latent sPacE \(ANTELOPE\)](#) architecture [162]. Weakly-supervised methods used for model-agnostic searches like the [Classification Without Labels \(CWoLa\)](#) [163] are also often combined with [AEs](#).

Even for anomaly detection at the trigger level, the CMS collaboration implemented two hardware triggers based on autoencoding algorithms called [Anomaly eXtraction Online Level-1 Trigger aLgorithm \(AXOL1TL\)](#) and [Calorimeter Image Convolutional Anomaly Detection Algorithm \(CICADA\)](#) [164, 165]. Further details of these algorithms are discussed in Section 3.4. Finally, LHCb also implemented anomaly detection for a trigger line in the muon detectors based on a normalised [AE](#) [166].

Besides anomaly detection, [AEs](#) are also important for data compression in [HEP](#). The [Baler](#) package [167] provides a lossy but efficient data compression approach based on [AEs](#).

Recurrent Neural Network (RNN)

A **Recurrent Neural Network (RNN)** is a type of **NN**-based model that is specialised for sequential data $x^{(t)} \in (1, \dots, \tau)$, where t indicates the time stamp index [92, 168]. Its schematic representation is shown in Figure 3.9. It creates a hidden state $h^{(t)}$, based on previous inputs $x^{(t)}$ and previous hidden states $h^{(t-1)}$, that is shared with different parts of the model. This *self-loop* allows the model to effectively keep track of contextual relations between data instances. An immediate drawback is that this cannot be easily parallelised, as data has to be processed in sequential order. Furthermore, these models struggle with vanishing or exploding gradients and are therefore hard to train effectively [94].

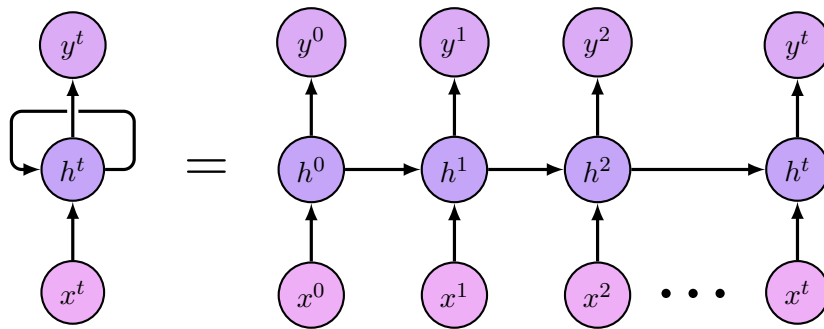


Fig. 3.9.: Schema of **RNN** architecture visualised folded (left) and unfolded (right). Inspired by [134].

Long-Short Term Memory (LSTM) A **Long-Short Term Memory (LSTM)** model is a type of gated **RNN** that avoids the problem of vanishing gradients by learning how much of the hidden state should be remembered [169]. This is done by allowing the model to control the weights of the self-loop using a forget gate. **LSTMs** networks are observed to be better at handling long-term dependencies compared to simple recurrent units [170], and have become the dominant choice for processing sequential data before transformers [94].

Standard **RNNs** are less omnipresent in **HEP** than the previously discussed ones. However, there are some studies of applying **RNNs** to flavour tagging in jets [171] or to vertex finding at a future lepton collider [172].

More popular are **LSTM** models that are mainly employed for monitoring purposes: For instance, the monitoring of the superconducting **LHC** magnets could be done with **LSTMs** [173]. The ATLAS **AE** approach for monitoring the **LAr** calorimeter, mentioned in the previous paragraph, is built with a **LSTM** encoder and decoder [78]. Furthermore, **LSTMs** have been studied for monitoring the data quality of the ATLAS data management system **rucio** [174].

Concerning more physics-related topics, **LSTMs** have been proposed for tagging strange jets [175], tagging boosted top particles [176] or identifying taus [177]. ATLAS also investigates real-time energy reconstruction on **Field Programmable Gate Arrays (FPGAs)** employing a **CNN** or **LSTM** for the processing of the ATLAS **LAr** calorimeter signals during the **HL-LHC** [178]. LHCb further considers the use of **LSTMs** for track reconstruction in the LHCb vertex locator [179].

Transformer

The transformer is a model architecture that has gained widespread recognition due to its success in **NLP** tasks [98]. It is built upon two key components, the *self-attention mechanism*, which allows the model to learn contextual relationships, and *skip connections*, which facilitates the information flow by allowing unfiltered signals to pass through the model. A schematic illustration of the transformer architecture is given in Figure 3.10.

The attention mechanism relies on *scaled dot-product attention*. To compute this, one needs a key vector and a query vector, both of dimension d_k , and a value vector, of dimension d_v . The value vector typically contains the input values, after having passed through an embedding step. In practice, multiple query vectors are stacked into a query matrix Q , and similarly, one gets a key matrix K and a value matrix V . The attention is then computed as follows:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) V \quad (3.13)$$

Instead of setting d_k equal to the internal model size d_{model} , one can split the inputs onto h linear projections using projection matrices W_i . To each projection,

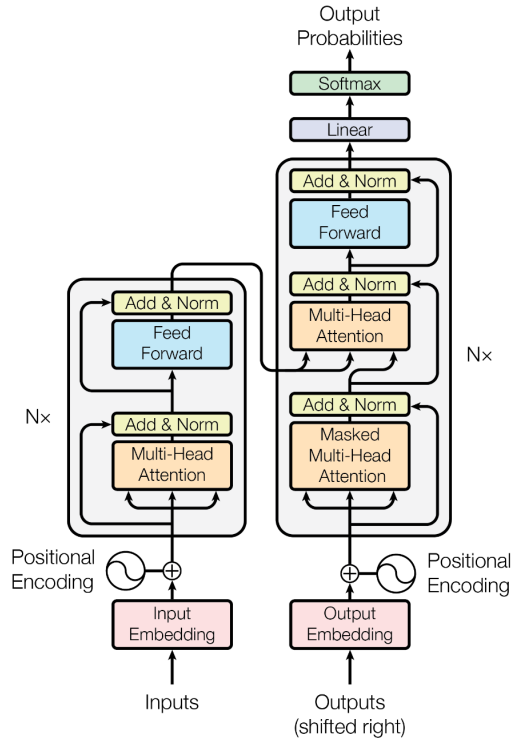


Fig. 3.10.: Schema of transformer architecture. Figure from [98].

one can apply a key, query and value matrix of dimension $d_k = d_v = d_{model}/h$. This is called *multi-headed attention* and is mathematically described by:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \text{head}_2, \dots, \text{head}_h)W^O \quad (3.14)$$

$$\text{with head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^K), \quad (3.15)$$

$$W_i^Q, W_i^K \in \mathbb{R}^{d_{model} \times d_k}, W_i^V \in \mathbb{R}^{d_{model} \times d_v}, W^O \in \mathbb{R}^{hd_v \times d_{model}} \quad (3.16)$$

where h denotes the number of heads and the projections W_i^Q, W_i^K, W_i^V, W^O are parameter matrices.

On one hand, the self-attention layer scales as $\mathcal{O}(n^2 \cdot d_{model})$, where n is the length of the input sequence and d_{model} the dimension of the model's internal representation. On the other hand, a recurrent layer has a complexity of $\mathcal{O}(n \cdot d_{model}^2)$. However, self-attention only requires a constant number of sequentially executed operations, i.e. $\mathcal{O}(1)$, whereas a [RNN](#) needs $\mathcal{O}(n)$ operations. Therefore, as long as $n < d_{model}$, self-attention layers offer an advantage over recurrent layers in terms of computational complexity.

In [HEP](#), transformers have been tested for a variety of applications from flavour tagging over energy calibration to detector simulations. ATLAS updated their jet flavour tagging algorithm with a transformer-based model because it outperformed the previous, graph-based algorithm [\[180\]](#). Also for ATLAS, transformers have been used for identifying boosted Higgs particles decaying into $b\bar{b}$ and $c\bar{c}$ pairs [\[181\]](#) and for calibrating energies in the electromagnetic calorimeters [\[182\]](#).

Further, generative transformers have been tested for simulating calorimeter showers in high-granularity calorimeters [\[183\]](#), full-length detector-level events from parton-level information [\[184\]](#) and [LHC](#) physics data [\[185\]](#), studied on the example of leptonically decaying Z -bosons associated with a variable number of [QCD](#) jets. Finally, since transformers have been especially successful for [NLP](#) tasks, it has been investigated whether the substructures of jets could be interpreted as words of a 'jet language' and learned by transformers [\[186\]](#). The authors of this study argue that such a transformer learns the density of [QCD](#) and top jets and can generate new jet samples which are difficult to distinguish from the real data.

3.4 Anomaly Detection

Anomaly detection, i.e., the identification of specific data instances that do not conform to the vast majority of them, is an important problem across a wide range of domains. This task is challenging due to the unknown nature of these

instances, with respect to the majority, and their aforementioned rarity. However, interest in anomaly detection continues to grow due to its valuable applications for real-world problems such as fraud detection, intrusion detection, medical diagnosis, risk prediction, and many more [187].

Anomaly detection is typically approached in an *unsupervised* manner, as the nature of anomalous instances is not defined beyond their deviation from the majority of the data. However, in certain well-studied settings, where anomalies are properly characterised, the data can be labelled and *supervised* learning methods become applicable. In reality, due to the rare occurrence and ill-defined nature of anomalies, obtaining reliable labels is often difficult and costly. Consequently, supervised learning is unsuitable for many real-world applications, and one relies on *unsupervised* learning. Furthermore, one distinguishes between *unsupervised* and *self-supervised* methods. Whilst both work without anomaly labels, *self-supervised* models create their own targets from the input data and train with those. However, *unsupervised* and even *self-supervised* techniques often perform worse than the supervised approach [105, 188] when labels are available.

Despite the previously mentioned challenges, anomaly detection remains an active field of research. It is applied wherever one does not have a concrete description of the rare, unusual processes one is looking for.

A review over recent anomaly detection methods and applications in HEP is given in [153]. In HEP, anomaly detection is either used to find rare physics events in offline physics analyses to place limits on the existence of new particles [163, 189, 190], or already at the level of data taking (online) for triggering or data quality monitoring [164, 191]. In the second case, one either focuses on directly improving the trigger efficiency for interesting and rare events or on catching detector defects in an automatised way to relieve the workload of human shifters. A more detailed introduction to anomaly detection for data quality applications is given in Chapter 11 and Chapter 12.

3.4.1 Anomaly Detection in High Energy Physics

As mentioned in the description of the AE, there are two principal applications of anomaly detection in HEP: anomaly detection for new physics searches and DQM. We will provide a brief overview of the most recent projects on this domain with a special focus in the LHC experiments.

Many physics searches are biased by the BSM model, whose signatures they are searching for. Therefore, recently model-agnostic searches have been gaining popularity. Based on weakly-supervised strategies such as the CWoLa al-

gorithm [163], the [Classifying Anomalies THrough Outer Density Estimation \(CATHODE\)](#) [192], the signal density in different phase space regions are compared, while barely making any assumptions about the signal itself. CMS conducted a model-agnostic search for dijet resonances combining a [VAE](#) with the [CWoLa](#) or [CATHODE](#) algorithm. Different versions of [AE](#) with some latent space parametrisation have been employed in searches for semi-visible or dark jets [156–159]. Whilst the majority of anomaly detection strategies are based on some kind of [AE](#) model, a different approach was followed by the authors of [193], which work with adversarial training. In that case, one generative model, the generator, tries to imitate the input data, whilst the second model, the discriminator, tries to distinguish between real and imitated data. If during testing an unexpected (anomalous) event occurs, the imitation of the generator will probably be poor, allowing the discriminator to pick up on the anomaly.

There is also the possibility that exotic physics signals are falsely sorted out at the trigger level. To mitigate this risk, CMS has developed two anomaly detection algorithms [AXOL1TL](#) and [CICADA](#) that can be implemented at the level of the hardware triggers. [AXOL1TL](#) is based on a [VAE](#) and acts on objects reconstructed by the [L1](#) trigger, whereas [CICADA](#), a [CNNs-AE](#), is applied to image-like calorimeter energy regions. Both algorithms are trained with quantisation-aware frameworks, translated to high-level synthesis code using `hls4ml` [194, 195] and then deployed on [FPGAs](#). So far, only the [AXOL1TL](#) has been commissioned for real-time testing and data taking within the CMS trigger, where it has been observed to pick up on physics events different from the ones passing the traditional triggers.

Finally, there is a growing interest in automating or improving the efficiency of current [DQM](#) methods and all of the four [LHC](#) experiments are investigating [DL](#)-based approaches for this [78, 108, 191, 196]. So far, CMS has deployed a [CNN](#)-based [AE](#) in the online [DQM](#) workflow of the electromagnetic calorimeter [154]. The [AE](#) is applied to images of the electromagnetic calorimeter and the anomaly score corresponds to a pixel-by-pixel reconstruction error. ATLAS is currently investigating anomaly detection approaches based on time series for the [DQM](#) of the [LAr](#) calorimeter [78]. This topic will be further explored in one of the projects presented in this thesis, with a detailed account of the corresponding studies provided in Chapter 15. LHCb is currently using [BDTs](#) or histogram-distance measures for their [DQM](#) workflow but is considering the use of reinforcement learning from human feedback [108].

3.4.2 Performance Metrics

Since normal, i.e. non-anomalous, data are more common by definition, anomaly detection leads to a strongly imbalanced classification problem. Typically, one assigns the label 0 to normal data instances and 1 to anomalies. To quantify a classification model's predictions, it is common to use a confusion matrix. The confusion matrix compares the predicted labels $\hat{y}$ with the actual, true labels y and counts the following quantities:

- **True Positive (TP)**, where $y = \hat{y} = 1$,
- **True Negative (TN)**, where $y = \hat{y} = 0$,
- **False Positive (FP)**, where $y = 0, \hat{y} = 1$,
- **False Negative (FN)**, where $y = 1, \hat{y} = 0$.

Based on these quantities, one can derive various metrics. For balanced classification problems, the classifier's performance is quantified either with the accuracy (i.e. $(TP + TN)/(TP + TN + FP + FN)$) or the **Area Under Curve (AUC)** of the **Receiver Operator Characteristic (ROC)** curve [197].

However, for very imbalanced problems, such as most anomaly detection scenarios, these metrics tend to grossly overestimate a model's performance and should not be used. Therefore, it is often preferred to use precision ($P = TP/P$), recall ($R = TP/(TP + FN)$) or a combination of both like the F1 score which is given by:

$$F1 = \frac{2 \cdot P \cdot R}{P + R} = \frac{2 \cdot TP}{TP + FP + FN}. \quad (3.17)$$

The F1 score gives the most weight to **TPs** and can hence be a rather conservative estimation of a model's performance while neglecting the **TNs** [198].

Finally, the **Matthews Correlation Coefficient (MCC)** is an interesting metric for imbalanced problems, as it gives equal weight to the true positives and negatives. It is defined as follows:

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}. \quad (3.18)$$

Since it is a correlation coefficient, the **MCC** ranges between -1 and 1 and is not strictly positive as all the previously mentioned metrics [199].

3.5 Issues and Challenges

Finally, common issues of ML models and their applications are presented and discussed. This selection of topics is not exhaustive and only intends to give the reader some general understanding of the challenges associated with ML [94, 95, 100].

Availability of Data Training ML models effectively requires not only large volumes of data but also data that meet high-quality standards. Without such standards, models are prone to inheriting biases or exhibiting degraded performance. This is particularly relevant for LLMs, which rely on massive, self-supervised training over written language. Their performance deteriorates noticeably when the input data are insufficient or poorly curated. Recent estimations suggest that the availability of high-quality data may soon become a bottleneck, especially if the emphasis on data quality is maintained.

Besides the requirement for large amounts of well-prepared data, it is also important that the data is (correctly) labelled. It is often very costly in terms of human resources to reliably label a dataset. However, this is an essential step because any learning model is only ever as good as its input data.

Finally, high-quality benchmark datasets are essential to allow for a fair comparison among different learning models.

For instance, at the LHC, huge amounts of data are available but the filtering, calibration, labelling and storing of them require a remarkable amount of resources.

Computational Resources Modern learning models are steadily growing in size, with increasing numbers of parameters driving up memory and computational demands. Consequently, data centres require significantly more energy to support both training and inference workloads. In 2022, global data centre electricity consumption reached approximately 460 TW h [200]. This growing energy demand has notable implications for the carbon footprint and water usage associated with data centre operations.

Domain Adaptation In general, one assumes that the training and testing data are drawn from the same underlying distribution. However, in practice, this cannot always be guaranteed; there is often a *domain shift*. *Domain adaptation* then refers to the techniques that are employed to ‘adapt’ the model from one domain to another. This is crucial for many real-world applications.

For instance, in HEP, many target variables are only available in MC simulations.

Therefore, the training is often conducted with simulated data, while the final application focuses on experimental data, leading to an obvious domain shift.

Interpretability and Explainability Even though it is possible to access the weights and parameter values of a learning model, it is very hard for humans to understand how a model obtains its results, especially as the number of parameters grows. Therefore, DL models, which often have huge numbers of parameters, are frequently described as *black boxes*. This is because one does not know what is happening inside the model, and can only observe the model's outputs based on the inputs. The field of interpretable ML and the field of explainable AI focus on shedding light on this problem by providing techniques to describe a model's decision-making process [201, 202].

For these, two concepts need to be introduced. *Interpretability* can be described as the 'degree of understanding' a human has of a model's prediction. It is important to notice that interpretability is neither a completely precise nor an objective term, as it is tied to an observer's judgement of the situation. Similarly, *explainability* is also often mentioned in this context, but again, there is no exact definition. In general, it can be said that a model is explainable if a human is capable of explaining how a model reached its predictions. Researchers often use interpretability and explainability exchangeably, but some argue that for explainability, one needs an interpretable model plus some additional context, such as the application field and model objective [201–203].

Interpretability of ML models is essential for applications in healthcare, legal processes or hard sciences, to name a few. For instance, when using ML in sciences, it is not sufficient to find a model that solves a research problem; it is just as important to understand *how* the problem was solved, such that knowledge on the research topic can be further expanded [201].

Part II

Simultaneous Jet Calibration with
Machine Learning Including *in situ* Jet
Energy Resolution Measurement

Introduction

As described in Section 2.3.2, jet calibration is a complex and time-consuming yet crucial component of the ATLAS data preparation. To improve efficiency and precision, new approaches combining traditional methods with ML are being investigated. This direction is particularly relevant in anticipation of the upcoming HL-LHC data-taking period. A first possible approach explored the use of a NN for the GSC step, the so-called GNNC, which has been used for run 3 jet calibration recommendations [77]. Another ongoing project aims at merging several of the standard calibration steps based on MC simulation. Despite these promising results, all examples exclusively focus on simulation-based calibration, disregarding the important role experimental data plays in the standard ATLAS jet calibration.

Therefore, the following chapters detail the application of ML models to ATLAS jet calibration, while focusing on the potential inclusion of experimental data during training to mitigate some of the bias induced by MC simulation. The overall objective is to improve the JER, while either keeping the JES invariant or correcting it with respect to the true JES and thereby merging several calibration steps into one. For this, the dijet asymmetry was identified as the physical quantity allowing one to relate the jet resolution on a broad p_T scale to experimental data.

Due to the existence of a trivial solution of the minimisation problem, several challenges arise during the implementation of a loss function term based on the dijet asymmetry. To mitigate this, several implementations of loss functions and possible constraint terms are explored in the context of this project.

First, Chapter 5 gives information about the MC samples and selection criteria used for the training of the calibration model. Next, the ML model implementation is discussed in Chapter 6. The results obtained from training the calibration model exclusively with truth information are described in Chapter 7. While Chapter 8 introduces a loss function based on the dijet asymmetry and discusses how to implement it and what challenges might arise. This is followed by Chapter 9 presenting the results obtained from this implementation. Finally, a summary of the project and an outlook on potential future work are given in Chapter 10.

Input Data and Preprocessing

5.1 Monte Carlo Simulation Samples

As inputs for a jet calibration [ML](#) model, dijet events consisting of [PFlow](#) [55] jets at the electromagnetic scale and with a small radius ($R = 0.4$) are used. The events are based on [MC](#) event generators (as described in Section 1.3).

We focus on dijet events as these event topologies cover the entire p_T range relevant for the study of proton-proton collisions. Dijet events refer to physics events with at least two hard jets that dominate over other (softer) jets (see Section 2.3.3). More concretely, the four-vectors and constituents of the two leading jets plus the p_T of the third jet are kept in the employed datasets. The exact variables are listed in Table 5.1. Their definition is based on the coordinate system of the ATLAS detector described in Chapter 2.

Tab. 5.1.: Variables used as inputs for [ML](#)-based calibration model.

	Jet Constituents	Reco Jet	True Jet
Variables	(p_x, p_y, p_T, η) + <i>GSC Variables</i>	(p_x, p_y, p_T, η, E)	$(p_x^{true}, p_y^{true}, p_T^{true}, \eta^{true}, E^{true})$
Dimensions	(80, 4)	(5,) + (15,)	(5,)

Furthermore, some additional variables (often called [GSC](#) variables because most of them are traditionally used for the [GSC](#), see Table 2.1) are included for every jet.

Starting from a JETM2 MC20 [MC](#) sample, NTuples are produced using the Isolated Jet Tree code [204]. The jet derivation JETM2 is a specific [DAOD](#) that was chosen as it contains the jet constituents, which will be used as inputs for the calibration model. These correspond to the separate energy deposits per calorimeter cell and tracks in the detector from which the reconstructed jets have been built with the anti- k_t algorithm (for more details see Section 1.2). During the NTuple production, a first pileup calibration step is added such that the *jet area pileup correction* has been applied, but not yet the *residual pileup correction* (see the exact calibration steps in Section 2.3.2). These jets will be denoted as *reco* jets and are used as inputs to the [ML](#) model. In the next step,

using an adjusted version of the Root2H5 code repository [205], the NTuples are accessed, randomly shuffled and regrouped into a training and testing set, which are saved in an HDF5 file.

The available jet derivation JETM2 samples are divided into several smaller datasets, which are referred to as JZ slices. The JZ slices are numbered and each slice contains jets from a certain p_T range, where the energy range increases with the numbering.

The goal is to compare and validate the method on different energy ranges. Therefore, we select the JETM2 JZ2 and JZ7 MC samples and follow the previously described procedure. If not stated otherwise, the numerical experiments are realised with samples from JETM2 JZ7. Figure 5.1 shows the p_T distribution of the two different input samples that are used in the course of this project.

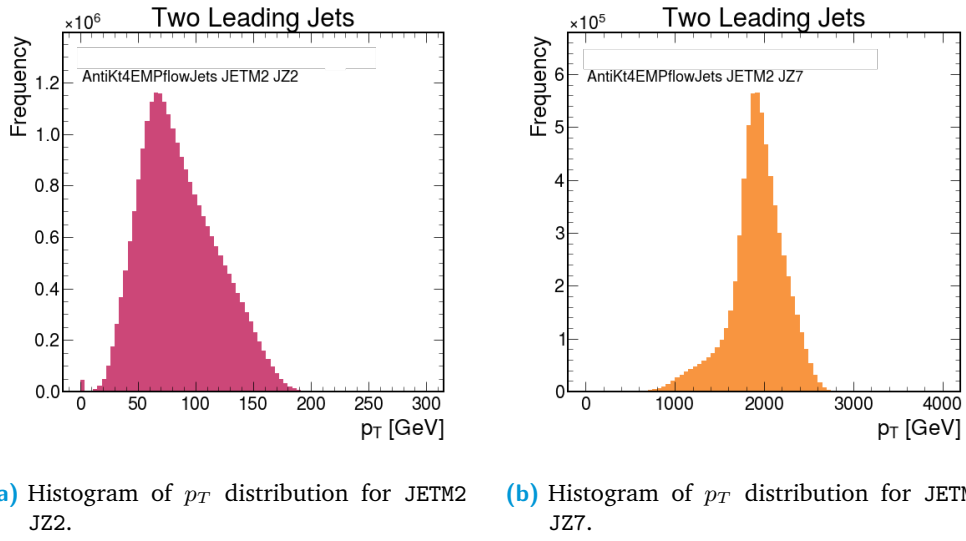


Fig. 5.1.: Histograms of input p_T distribution for the two MC samples before any selection cuts.

5.2 Jet Selection Criteria

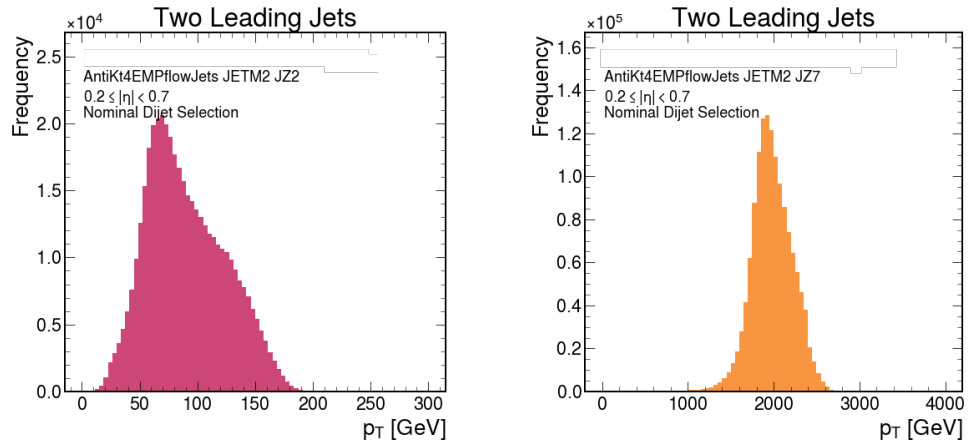
Several selection cuts have been applied to the input MC samples. They are listed in Table 5.2. Note that some of the cuts are dependent on the JZ slice (JZ2 or JZ7).

Firstly, we choose to focus on central jets, i.e. $0.2 \leq \eta \leq 0.7$. This allows for simplifying the relation between dijet asymmetry and relative JER, which is discussed in more detail in the following. Secondly, *nominal dijet selection* criteria are applied (as explained in [60]). The cuts guarantee that a dijet event is as clean as possible, with two leading jets that are (almost) back to back and

Tab. 5.2.: Selection criteria applied to dijet events.

Variable	Criteria	Definition
η	Central	$0.2 \leq \eta \leq 0.7$
Dijet Selection	Nominal	$\Delta\phi_{12} > 2.7 \text{ rad}$ $p_{T_3} < \max(25\text{GeV}, 0.25 \cdot p_{T,avg})$
p_T	JZ2	$p_T \in [20, 160] \text{ GeV}$
	JZ7	$p_T \in [1700, 2100] \text{ GeV}$

any additional jets having relatively low energy. Finally, the p_T selection cuts depend on the chosen JZ slice. Avoiding abrupt cut-offs at the edges of the chosen p_T range, the p_T cut treats the two leading jets per event differently and adds some padding to the limits. Events are included if $p_{T_1} \geq p_{T,lower} - 200 \text{ GeV}$ and $p_{T_2} \leq p_{T,upper} + 200 \text{ GeV}$ for training (where $p_{T,lower}$ and $p_{T,upper}$ are listed in Table 5.2). For testing and plotting the results, only jets with a p_T within the chosen range $[p_{T,lower}, p_{T,upper}]$ are considered. Figure 5.2 shows the p_T distributions of the JZ2 and JZ7 input samples after the selection cuts.



(a) Histogram of p_T distribution for JETM2 JZ2 after selection cuts. (b) Histogram of p_T distribution for JETM2 JZ7 after selection cuts.

Fig. 5.2.: Histograms of input p_T distribution for the two MC samples used as inputs for the calibration model after applying the selection cuts from Table 5.2.

5.3 Scaling of Inputs

It is known that the training of NNs benefits from scaled inputs to avoid artificially attributing heavier weights to some features with larger values. This is not naturally satisfied by collision data; the distributions of the different elements of

the jet four-vector alone have very different scales. To ensure that all the input features are on a comparable scale, we subjected most of them to some type of scaling before passing them to the network. A schematic summary of the data input workflow is shown in Figure 5.3.

For the jet constituents, all except the η constituents are divided by the standard deviation of the p_x jet constituents. By the definition of η , the pseudorapidity values naturally range between -4.5 and 4.5 . Furthermore, restricting our study to the central part of the detector, we only use $|\eta| \in [0.2, 0.7]$, which restricts the range further and makes any scaling of η unnecessary. Similarly, the jet four-vector variables are all divided by the standard deviation of p_x except for η . Finally, we leave the GSC variables unchanged, as most of their values are contained in a small interval.

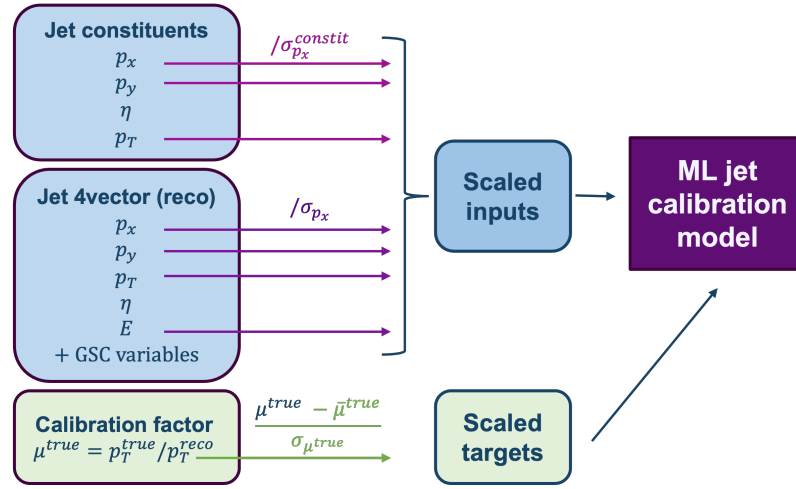


Fig. 5.3.: Schema of input data and their scaling.

5.4 Normalisation of Targets

The targets of the regression task are the calibration factors, which are defined as follows:

$$c_{true} = p_T^{true} / p_T^{reco} \quad (5.1)$$

where p_T^{reco} denotes the reconstructed jet momentum, which is used as the model's input. The targets are standardised using the mean and standard deviation of the training data, which are listed in Table 5.3. The standardised targets are given by $c'_{true} = (c_{true} - \mu_c) / \sigma_c$.

This normalisation of the calibration targets affects the computation of the asymmetry in the asymmetry-based loss function because $p_{T_i}^{pred}$ depends on the predicted calibration factor, i.e. the network output. To obtain the actual physical

Tab. 5.3.: Mean μ_c and standard deviation σ_c of the calibration targets c_{true} for different training sets.

Training Set	Mean μ_c	Standard Deviation σ_c
JZ2	0.9748008	0.1474832
JZ7	0.9976801	0.0815602

quantity, this can be corrected by simply re-scaling the network’s outputs (i.e. the predicted calibration factors):

$$c_{pred} = \sigma_c \cdot c'_{pred} + \mu_c. \quad (5.2)$$

After normalisation, it is observed that some targets have values very far from 0. As discussed in [206], this hints at a poor reconstruction of the jet. Therefore, we only keep events whose calibration targets fall within the interval $[-2, 2]$.

5.5 Flattening of the p_T Distribution

In the context of ML approaches for HEP, it has sometimes been found beneficial to work with data for which the p_T distribution has been flattened. This is even more important when working with multiple JZ slices, as each of them has a pronounced cut-off. Since DL relies on differentiation, steeply falling distributions are a challenge for the training process of such models.

Therefore, in HEP, sets of collision events are often resampled such that the p_T distribution becomes nearly flattened over the interval of interest. This is very easily done when treating each jet individually as it is done for the truth-based loss function (see Chapter 7). However, when introducing the asymmetry-based loss function in Chapter 8 – where we work with exactly two jets per event – this gets more challenging. It is no longer obvious which distribution exactly should be considered for the resampling: Should one focus on flattening the p_{T_1} or p_{T_2} distribution? Or better focus on a combined distribution such as $p_{T,avg} = \frac{1}{2}(p_{T_1} + p_{T_2})$?

After a few preliminary tests with data flattened in p_{T_1} or $p_{T,avg}$, both were found not to be beneficial but had a negative impact on the model’s training. Therefore, for the following studies, the p_T distributions have not been flattened, but we took special care in applying the p_T cuts. This was facilitated by the fact that we used the JZ slides separately.

For further studies, it would be interesting to test, whether re-weighting the jet momenta directly in the loss function could mitigate this problem. More concretely, one could assign weight factors to different p_T bins such that each p_T bin contributes equally to the total loss regardless of the actual numbers of jets available in each bin.

Machine Learning Model

This section focuses on the deep sets [ML](#) model and its characteristics. A general introduction to the deep sets architecture is given in Section [3.3.3](#). In this section, the focus lies on the implementation of this model to the jet calibration problem..

6.1 Deep Sets for HEP

Applying the deep sets architecture to problems in [HEP](#), led to the development of so-called Energy Flow Networks [\[143\]](#). They are based on the observation that one can interpret each particle collision event as an unordered set of particles of variable length.

Starting from Eq.([3.11](#)) and replacing the generic function $S(x)$ with an observable $O(p_i)$ depending on the momentum of each particle, one gets the following observable decomposition:

$$O(p_1, \dots, p_M) = F \left(\sum_{i=1}^M \Phi(p_i) \right), \quad (6.1)$$

where $\Phi : \mathbb{R}^d \rightarrow \mathbb{R}^l$, $F : \mathbb{R}^l \rightarrow \mathbb{R}$, d the dimension of the input space (i.e. the dimension of p_i) and l the dimension of the latent space. Again, based on the universal approximation theorem [\[137\]](#), it can be assumed that the two functions F and Φ can be well approximated by two [NNs](#).

Permutation Invariance Permutation invariance is an important property for [HEP](#) as collision events measured in the detector do not appear in any particular order. In principle, each collision is independent, and any particular ordering of the events (as often done for [RNNs](#)) does not make sense in this context.

This statement has some limitations, as the reconstruction in the detector might introduce dependencies on the order of the events. This is, for example, the case for out-of-time pileup, i.e. when the signals from different bunch crossings overlap in the calorimeter. However, in general and especially after pileup corrections, which constitute the first steps of the jet calibration chain, one can assume the collisions to be mostly independent. To guarantee that the [ML](#)

model respects this invariance, a permutation-invariant layer (e.g. sum-layer or mean-layer) is included in its architecture.

Since in [HEP](#) many observables can be decomposed as shown in Eq.(6.1), one can implement a deep sets model approximating such an observable. Furthermore, the observable decomposition can be slightly adjusted to work with [IRC](#)-safe observables.

Infrared and Collinear (IRC) Safety As discussed in Section 1.2, [IRC](#) safety is crucial when dealing with jets. A model is said to be [IRC](#)-safe if its predictions do not change with the addition of a soft particle or a change in the collinear splitting of the particles. This property must be satisfied to directly compare the experimental results to theoretical predictions.

Concretely, an [IRC](#)-safe observable can be decomposed in the following way [[143](#)]:

$$O(p_1, \dots, p_M) = F \left(\sum_{i=1}^M z_i \Phi(p_i) \right) \quad (6.2)$$

where F, Φ are as defined previously and z_i is either the energy, the transverse momentum or the angular information of particle i . Again, such observables can easily be approximated by a slightly adjusted deep sets model.

6.2 Regression Targets

The whole jet calibration problem is formulated as a regression task, where the targets are the calibration factors which relate the [reco](#) to the truth level. Concretely, the target calibration factors are defined as:

$$c_{true} = p_T^{true} / p_T^{reco}, \quad (6.3)$$

where the [reco](#) jets correspond to the input jets as described in Chapter 5. The calibration factors predicted by the learning model are denoted as follows:

$$c_{pred}(\theta) = p_T^{pred}(\theta) / p_T^{reco}, \quad (6.4)$$

where θ denotes the model's parameters.

6.3 Truth-Based Loss Function

Starting from the supervised context, where one has access to data corresponding to the ground truth, we define the loss function as the [MSE](#) between targets and predictions. For jet calibration, the truth data corresponds to the true jet p_T^{true} , which is only accessible in simulated data. The loss function can then be defined as follows:

$$\text{loss}(\theta) = \min_{\theta} \text{MSE}(c_{true}, c_{pred}(\theta)) \quad (6.5)$$

$$= \min_{\theta} \frac{1}{N_b} \sum_{i \in \text{batch}} (c_{true} - c_{pred}(\theta))^2, \quad (6.6)$$

where N_b is the batch size, since the loss function is computed batch-wise.

Instead of working with the calibration factors, one could also directly compare the true and predicted jet p_T in the loss function. For numerical reasons, this approach is preferred when working with several loss terms to ensure that all loss functions start at a similar scale and have somewhat comparable numerical values.

Results Using Truth Information

Using the deep sets model presented previously and with the loss function from Eq.(6.6), which is completely based on the comparison between truth and **reco** transverse jet momenta (p_T), one finds the results shown in Figure 7.1. The model was trained over 30 epochs on roughly 17k dijet events, while using a batch size of 4096 jets.

After training, the model already finds calibration factors that correct the reconstructed jet p_T very closely to the true p_T (see Figure 7.1d). However, the **JER** could still benefit from some improvement as seen in Figure 7.1e. Training the model for longer epochs or using more training data allows for a slight, albeit not drastic, improvement in resolution. Note that in this configuration, the training is entirely based on the comparison of truth to **reco** jets.

Figure 7.1a shows the evolution of the training loss as a function of the epochs, which decreases continuously, although it appears to slow down towards the end of the training.

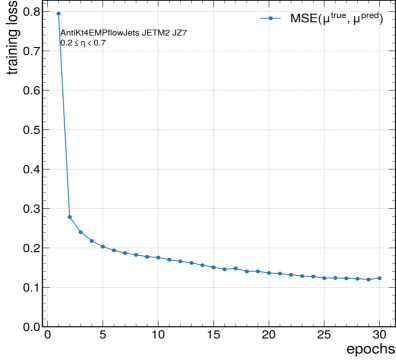
Figure 7.1b and Figure 7.1c are histograms comparing the predicted p_T distribution to the **reco** (i.e. input) distribution and true distribution, respectively. We observe that the distribution is sharply peaked around approximately 2 TeV. This could pose a challenge for the model to learn effectively, particularly because it must also account for the lower-energy regions, even though these are of less interest to us.

For the central plots, a Gaussian is fitted to the predicted (**reco**) over true p_T values, respectively, and this is done for each bin in p_T . From the fit, the mean and standard deviation including uncertainties are extracted and the means and their uncertainties are plotted in Figure 7.1d, whereas the standard deviations and their uncertainties are shown in Figure 7.1e. Figure 7.1d shows the mean response per bin of the ratio of predicted jet p_T over the true jet p_T . It can be observed that the predicted over true ratios (in cyan) are closer to one than the **reco** over true ratio. This implies that the model can predict calibration factors that approach the predicted to the true jets as intended. In addition, the **JER** has also improved, as visible in Figure 7.1e.

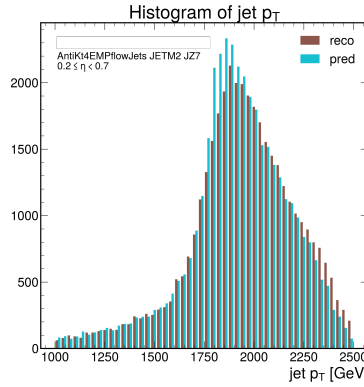
Finally, Figure 7.1f and Figure 7.1g compare the dijet asymmetry distributions (see Section 2.3.3) of the predicted, **reco** and true jets. Here, no Gaussian fit has

been done, but for comparison, we plot a Gaussian function with the same mean and standard deviation as the asymmetry distribution. The spread, quantified by the standard deviation, of the predicted dijet asymmetry is narrower than the [reco](#) distribution but still larger than the true dijet asymmetry distribution. This confirms that a reduction of the [JER](#) is related to a narrower dijet asymmetry distribution and is a key observation for our next steps.

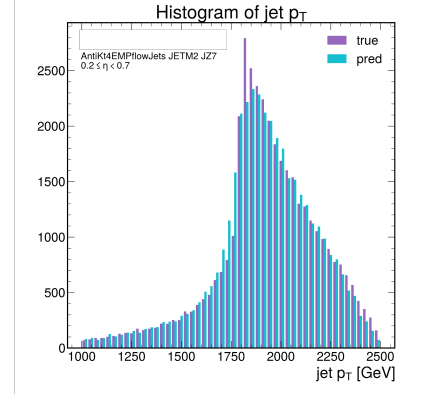
It is worth mentioning that even at truth-level, the spread of the dijet asymmetry is never completely zero because there is always some smearing induced by physics effects (e.g. additional jets present in the collision) which impact the asymmetry distribution. Any jet calibration algorithm is only supposed to correct the reconstructed quantities as close as possible to the truth-level, but not further than that. This is because the calibration algorithm cannot access any information about the physics effects for each event and can therefore not reliably account for such effects.



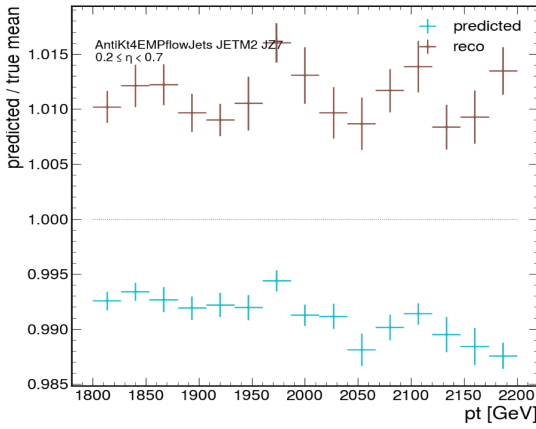
(a) Total training loss per epoch.



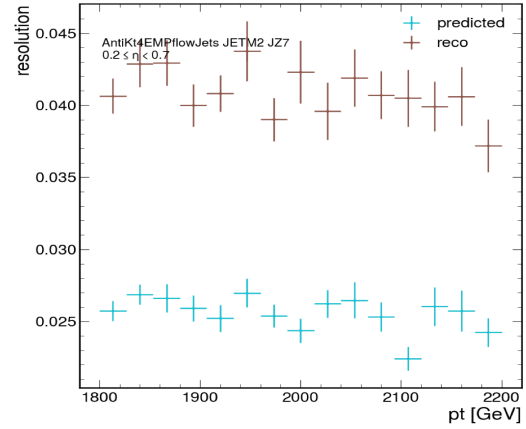
(b) Histogram of *reco* (brown) vs. predicted (cyan) p_T values.



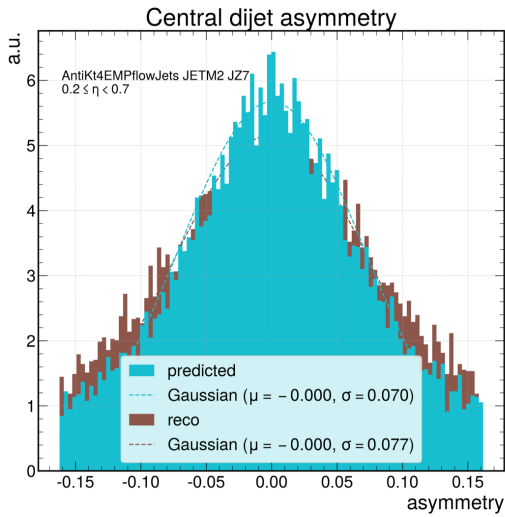
(c) Histogram of true (purple) vs. predicted (cyan) p_T values.



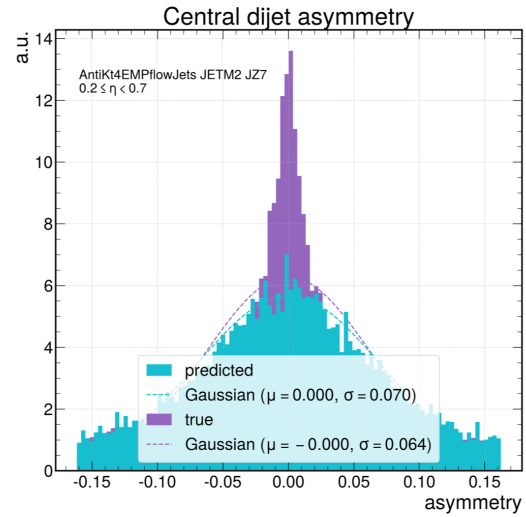
(d) Mean response per bin of ratio of predicted (cyan) or *reco* (brown) over true p_T .



(e) Resolution per bin for predicted (cyan) and *reco* (brown) jet p_T .



(f) Central dijet asymmetry of *reco* vs. predicted p_T values including a Gaussian with mean and standard deviation of the central asymmetry.



(g) Central dijet asymmetry of true vs. predicted p_T values including a Gaussian with mean and standard deviation of the central asymmetry.

Fig. 7.1.: Results after ML training with MSE loss from Eq.(6.6) comparing true and predicted calibration factors.

Introducing Dijet Asymmetry

The loss function introduced in Eq.(6.6) already allows us to correct the average **JES** quite well. However, this is not always true for the **JER**; usually, there is little to no improvement of the predicted resolution to the resolution of the input jets. In addition, there are always some physics effects that are not sufficiently described by **MC** simulations. For a successful calibration, it is therefore crucial to consider experimental data. The difficulty in this case arises from the fact that we only have access to **reco** jets, with no corresponding truth-level information. Consequently, utilising experimental data for jet calibration requires alternative techniques that rely on energy and momentum conservation.

For this project, the goal is to improve the **JER** by injecting knowledge from experimental data into the **NN**'s learning process. In the standard jet calibration chain, the **JER** is primarily improved by including information from additional variables (**GSC** variables), whereas the so-called *in situ* calibration mainly serves to quantify the **JER**.

As a first step, we choose to focus on dijet events (see Section 2.3.3) and try to integrate methods that are currently used for **JER** estimation into the model's training.

8.1 Asymmetry-Based Loss Function

Using the dijet asymmetry introduced in Section 2.3.3, the goal is to inject it into a loss function that allows for minimising the **JER** without relying on truth-level information. Starting from the Eq.(2.11) and focusing only on the two leading jets, the labels *probe* and *reference* jet are randomly assigned to the two leading jets. This simplification is valid for jets measured in the central part of the detector (i.e. $|\eta| \in [0.2, 0.7]$), and we find the following dijet asymmetry for the two leading jets:

$$A(\theta) = \frac{p_{T_1}(\theta) - p_{T_2}(\theta)}{\frac{1}{2}(p_{T_1}(\theta) + p_{T_2}(\theta))} = \frac{p_{T_1}(\theta) - p_{T_2}(\theta)}{p_{T,avg}(\theta)}. \quad (8.1)$$

It is now possible to construct a [ML](#) model predicting calibration factors (as done in Chapter 6) to improve the relative [JER](#). Starting from this idea and using Eq.(2.15) and Eq.(8.1), one can derive the following loss function:

$$\text{loss}(\theta)_{\text{asym}} = \min_{\theta} \left\langle \frac{\sigma_{p_T}(\theta)}{p_T(\theta)} \right\rangle = \min_{\theta} \frac{1}{\sqrt{2}} \left(\sigma_{A(\theta)}^{\text{det}} + \sigma_A^{\text{truth}} \right) \quad (8.2)$$

$$= \frac{1}{\sqrt{2}} \min_{\theta} \sigma_{A(\theta)}^{\text{det}} + \text{const.} = \frac{1}{\sqrt{2}} \min_{\theta} \text{std} \left(\frac{p_{T1}(\theta) - p_{T2}(\theta)}{p_{T,\text{avg}}(\theta)} \right), \quad (8.3)$$

where the smearing introduced by physics effects (i.e. the term proportional to σ_A^{truth}) is treated as a constant with respect to the calibration model's parameters θ . Hence, the term is of no importance for the minimisation problem and can be dropped. Indeed, as the model has no information to characterise these physics effects, it cannot account for them and should not aim to correct the smearing introduced by them. The 'std' in Eq.(8.3) stands for standard deviation. In the future, we will omit the multiplication factor of $\frac{1}{\sqrt{2}}$ and the constant term as neither of them influences the minimisation process. To render the notation less cumbersome, we will use $\sigma_{A(\theta)}^{\text{det}} = \sigma_{A(\theta)}$ from now on.

8.2 Combined Loss Function

Combining the truth-based and asymmetry-based loss terms, the loss function becomes:

$$\text{loss}(\theta) = \min_{\theta} f_t \cdot \text{MSE}(p_T^{\text{true}}, p_T^{\text{pred}}(\theta)) + f_a \cdot \sigma_{A(\theta)}, \quad (8.4)$$

where $f_t, f_a \in \mathbb{R}^+$ are hyperparameters determining how to weigh the two loss terms. Note that the second term of the loss function is completely independent of any truth information and can be used with experimental data instead of [MC](#) samples.

The fact that the truth-based (first) loss term compares individual jets, while the asymmetry-based (second) term works with dijet events, adds another layer of complexity to the problem. The detailed workflow is schematically represented in Figure 8.1. Concretely, we carefully select $N_b/2$ dijet events (i.e. N_b jets) for each batch and randomly shuffle the jets within each batch. To be able to re-assign the two jets belonging to the same event, we also save the event number of each jet in a one-dimensional tensor, which is passed to the asymmetry-based loss function. This information is only used within the loss function, to sort the jets and re-assign them to their corresponding event and cannot be accessed by the model for optimising its parameters. This is essential to ensure that each jet

is treated independently of the jet hierarchy (i.e. which one is the leading or sub-leading jet).

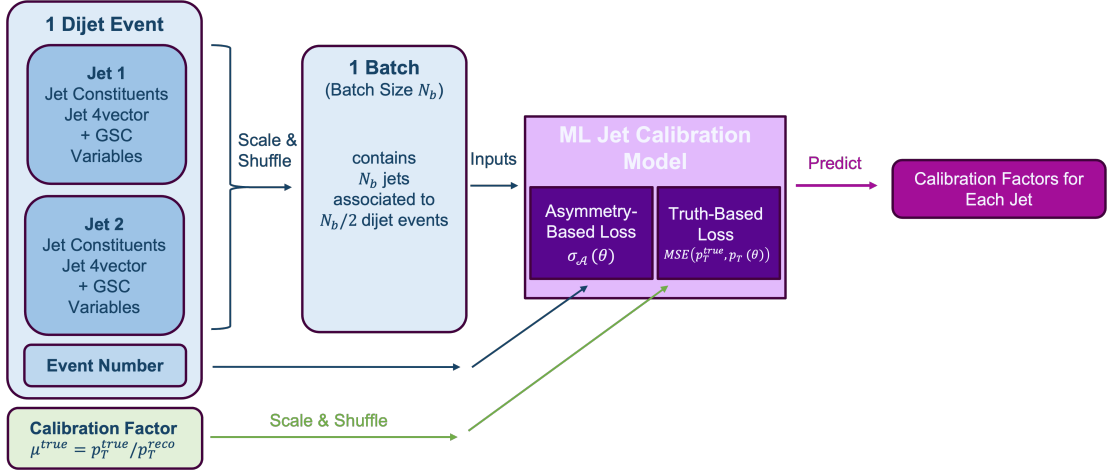


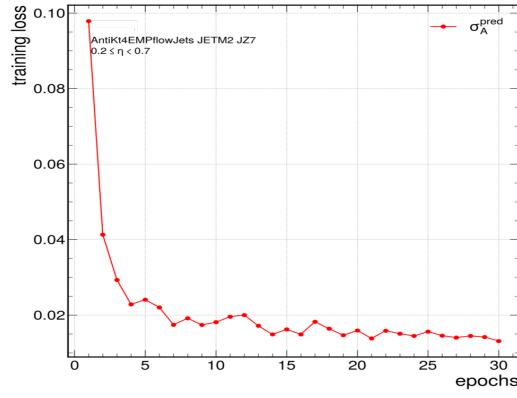
Fig. 8.1.: Schema illustrating inputs and workflow of the ML jet calibration model.

8.3 Implementation and Challenges

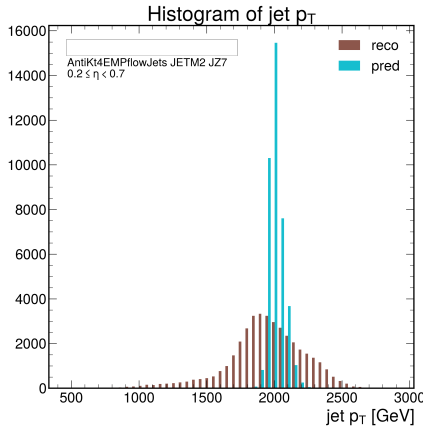
During our studies, we encountered various problems which are not all detailed here. Most of them were due to erroneous implementations of the data loader or loss function. However, for completeness, some of them are discussed in the appendix. They include problems with the jet hierarchy (Appendix A.1) and feature dependence (Appendix A.2). Finally, the next chapter explores the implementation of the asymmetry-based loss term, the challenges that come with it and how they can be avoided.

8.3.1 Unphysical Behaviour of Asymmetry Loss Function

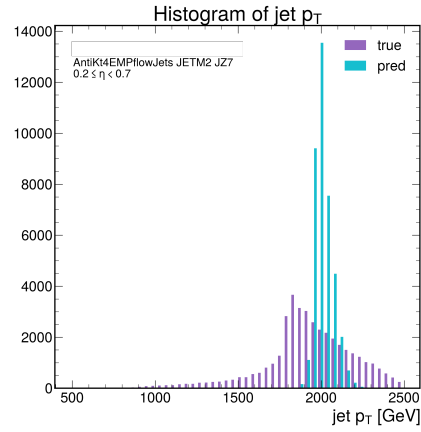
What follows is a brief description of a serious limitation of the previously introduced asymmetry-based loss term. It is observed that the model's predictions become trivial when focusing only on the asymmetry-based loss, i.e. setting $f_t = 0, f_a = 1$ in Eq.(8.4). The problem is that when only minimising σ_A , the ML model can easily solve this problem by setting all predicted jet p_T values to an arbitrary constant. To set all predicted jet momenta to the same value, the NN cannot simply predict constant calibration factors, but it has to learn how to adjust its outputs to predict them. Albeit unwanted, this is a valid solution to the optimisation task, which the model correctly identifies. Therefore, this limitation does not originate from a problem in the network's training, but from a conceptual issue with this loss function.



(a) Total training loss per epoch.



(b) Histogram of reco (brown) vs. predicted (cyan) p_T values.



(c) Histogram of true (purple) vs. predicted (cyan) p_T values.

Fig. 8.2.: Results after ML training only with asymmetry-based loss (from Eq.(8.3)).

Figure 8.2 shows an example of what the results can look like in that case. We observe that the training appears to be successful as the training loss is decreasing in Figure 8.2a. However, Figure 8.2b and Figure 8.2c show that the predicted (cyan) p_T values are all centred around 2 TeV. This centring is completely arbitrary, for other runs with the same configuration, different predictions are observed, but in every case, the predicted jet p_T distribution is very peaked around an arbitrary value. We also observe that the longer the training duration, the narrower and sharper the predicted jet p_T distribution becomes.

This trivial solution is not only unwanted but also highly unrealistic. It seems that the model has too much freedom in choosing its parameters. Therefore, the goal is to restrict them such that the previously described trivial solution is no longer a valid solution to the optimisation problem. In the following paragraphs,

we study the implementation of various constraints in the loss function to avoid or at least reduce this unwanted behaviour.

8.3.2 Possible Constraint Terms

As previously established, the goal is to optimise the unsupervised loss function from Eq.(8.3) without any ground truth. We therefore use the **reco** jet p_T^{reco} values, which are passed to the model as inputs. The simplest approach is to not allow the predicted values to stray too far from their input values. To do so, a constraint function $C(\theta)$ only depending on the model's predictions and its inputs is added to the loss function:

$$\text{loss}(\theta) = \min_{\theta} f_t \cdot \text{MSE}(p_T^{true}, p_T^{pred}(\theta)) + f_a \cdot \sigma_{A(\theta)} + f_r \cdot C(\theta), \quad (8.5)$$

where $f_t, f_a, f_r \in \mathbb{R}^+$ are treated as hyperparameters.

Event-Wise Constraint In a first step, we compare the difference between the predicted jet p_T and the **reco** values. This can be done by computing the **MSE** between them:

$$C_{event}(\theta) = \text{MSE}(p_T^{reco}, p_T^{pred}(\theta)) = \frac{1}{N_b} \sum_{i \in batch} (p_{T_i}^{pred}(\theta) - p_{T_i}^{reco})^2 \quad (8.6)$$

where N_b is the batch size and the sum goes over all jets in a batch (i.e. both leading jets of an event).

This first constraint is quite strict, and depending on the weighting of this term, it might not allow the model to find a solution that deviates from the input at all.

Batch-Wise Constraint In the next step, we compare the average p_T predictions per batch and the average **reco** p_T per batch. This is evaluated in the following way:

$$C_{batch}(\theta) = \frac{1}{(\bar{p}_T^{reco})^2} \left(\frac{1}{N_b} \sum_{i \in batch} p_{T_i}^{pred}(\theta) - \bar{p}_T^{reco} \right)^2 \quad (8.7)$$

where $\bar{p}_T^{reco}$ is the batch mean of the **reco** p_T values.

Here, we observe that the model pushes most of the predictions towards the batch mean of the **reco** values, which is still a constant value and does not constitute a satisfying calibration.

Non-Constant Constraint To avoid pushing the predictions towards a constant value, a non-constant constraint term is introduced:

$$C_{non-const}(\theta) = -\frac{1}{(\bar{p}_T^{pred}(\theta))^2} \frac{1}{N_b} \sum_{i \in batch} \left(p_{T_i}^{pred}(\theta) - \frac{1}{N_b} \sum_{j \in batch} p_{T_j}^{pred}(\theta) \right)^2 \quad (8.8)$$

$$= -\frac{1}{(\bar{p}_T^{pred}(\theta))^2} \frac{1}{N_b} \sum_{i \in batch} \left(p_{T_i}^{pred}(\theta) - \bar{p}_T^{pred}(\theta) \right)^2, \quad (8.9)$$

where $\bar{p}_T^{pred}(\theta)$ is the average p_T predicted per batch. Note the minus sign, because we want the difference between the predictions and their mean to be as big as possible.

Although this term has stopped the model from pushing towards one constant solution, the model instead found two constants to minimise the asymmetry, which is still not a satisfying constraint.

Bin-Wise Constraint Finally, we also study constraints derived from binned distributions. First, all initial dijet events are assigned a bin number depending on their $p_{T,avg}^{reco}$, which is the average p_T of the two leading jets. In the loss function, the average difference between predicted and **reco** jet p_T is computed for each bin in the batch:

$$C_{binned}(\theta) = \sum_{bin \in batch} \frac{1}{(\bar{p}_T^{reco})^2} \left(\frac{1}{N_b} \sum_{i \in bin} p_{T_i}^{pred}(\theta) - p_{T_i}^{reco} \right)^2, \quad (8.10)$$

where $\bar{p}_T^{reco}$ is the average **reco** p_T per bin and not per batch.

With this constraint, the results look more interesting, as the predictions no longer converge to very few constant values. It is also observed that the binning size influences the minimisation process. However, the asymmetry can still be minimised and depending on the duration of the training process, it is possible to reach an asymmetry distribution with a smaller spread than the true asymmetry distribution.

By varying f_r , it is possible to sufficiently suppress the tendency of the asymmetry loss term towards an unphysical solution by not letting the predictions stray too far from the initial values. However, the choice of this weighting factor f_r can only be determined empirically using a validation set. It would be more desirable to find a different loss function that does not inherently contain an unphysical solution, but this approach constitutes a valid workaround.

8.3.3 Alternative Loss Function

As an alternative to the straightforward implementation of the truth-independent loss term (see Eq.(8.3)), we slightly modify the asymmetry-based term in the loss function. Inspired by current *in-situ* techniques for Z +jet or γ +jet events [54], we assume one of the leading jets to be a ‘well-calibrated object’, i.e. its p_T^{reco} is well calibrated. Inserting the p_T^{reco} for the presumably well-calibrated jet into the dijet asymmetry calculation (see Eq.(8.1)), we observe that in this setting only one term can be modified by the model at a time. This approach effectively hinders the model from setting both jet momenta to the same constant to minimise the spread of the asymmetry. The assumption of working with a well-calibrated object is made for both leading jets separately, and we hence get the following two asymmetry terms:

$$A_1(\theta) = \frac{p_{T_1}^{pred}(\theta) - p_{T_2}^{reco}}{\frac{1}{2}(p_{T_1}^{pred}(\theta) + p_{T_2}^{reco})}, \quad A_2(\theta) = \frac{p_{T_1}^{reco} - p_{T_2}^{pred}(\theta)}{\frac{1}{2}(p_{T_1}^{reco} + p_{T_2}^{pred}(\theta))}. \quad (8.11)$$

As done in Eq.(8.1), we randomly assign the labels $i \in \{1, 2\}$ to the two leading jets.

For the loss function, we again focus on the minimisation of the standard deviation of the two updated asymmetry terms, which is now inherently constrained. This approach can also be interpreted as a self-supervised loss, similarly to what is done in AEs, where one compares the predicted output to the original input. Injecting this into the complete loss function from Eq.(8.5), we find the following new loss function:

$$\text{loss}(\theta) = \min_{\theta} f_t \cdot \text{MSE}(p_T^{true}, p_T^{pred}(\theta)) + f_a \cdot (\sigma_{A_1(\theta)} + \sigma_{A_2(\theta)}) + f_r \cdot C(\theta), \quad (8.12)$$

where $\sigma_{A_i(\theta)}$, $i \in \{1, 2\}$ denotes the standard deviation of the ‘new’ asymmetry terms defined in Eq.(8.11) and $f_t, f_a, f_r \in \mathbb{R}^+$ are hyperparameters. Again, only the first term relies on truth-level information; the other two terms depend on *reco* and predicted jets. Various studies on the importance and best choice of the hyperparameters are presented in the next chapter.

Results Using Truth and Truth-Independent Information

This chapter presents and discusses the most relevant results obtained from numerical experiments examining the importance and characteristics of the different terms in the loss function introduced in Eq.(8.12).

9.1 Results with Alternative Loss Function

To study the characteristics of the alternative asymmetry-based loss function term, we set $f_t = f_r = 0$ and $f_a = 1$ in Eq.(8.12). The model is trained over 30 epochs on roughly 17k dijet events using a batch size of 2048 events, i.e. 4096 jets.

The results of the training are presented in Figure 9.1.

Starting with Figure 9.1a, which shows the evolution of the training loss, we observe that the training loss values are significantly higher than before (see Figure 7.1a). This is because we switched from the standard deviation to the variance and refrained from dividing the asymmetry terms by their respective denominators. These numerical tricks were necessary as the model converges better in this setting.

Figure 9.1b and Figure 9.1c are histograms comparing the *reco*, predicted and true p_T distributions. The predicted distribution still has a noticeable peak, but it is mainly centred around a high-density region of the *reco* distribution.

As described in Chapter 7, the middle plots are done with the use of Gaussian fits. For each bin in p_T , a Gaussian is fitted to the predicted (*reco*) over true p_T values, respectively. From this, the mean response, the resolution and their corresponding uncertainties can be extracted. Looking at the mean response ratio of predicted over true p_T in Figure 9.1d, the predicted ratios deviate very strongly from the desired ratio of 1. This is somewhat expected, since the network was not given any truth-level information and hence could not adjust its predictions to the truth. However, the *JER* of the predicted jets has improved compared to the *reco* jets (see Figure 9.1e).

Finally, Figure 9.1f and Figure 9.1g plot the dijet asymmetry distribution of the

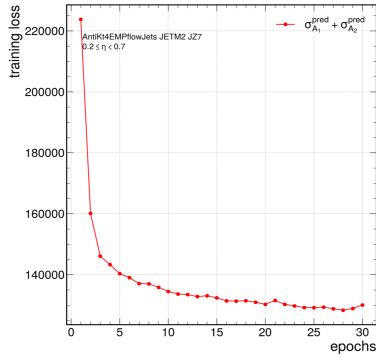
[reco](#), predicted and true jets. The spread (quantified by the standard deviation) of the predicted dijet asymmetry is observed to be smaller than the spread of the [reco](#) dijet asymmetry.

These observations confirm that minimising the alternative asymmetry-based loss function (defined in Eq.(8.11)) allows for shrinking the predicted dijet asymmetry distribution and improving the [JER](#) without any truth information. In the next step, one can either re-scale the average jet response to keep it invariant while reducing the resolution or aim at improving the [JES](#) simultaneously with the [JER](#).

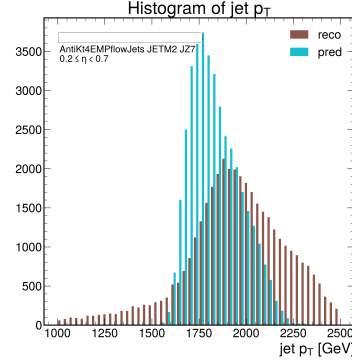
9.1.1 Keeping Average Jet Energy Invariant

Since the previous minimisation only focuses on improving the [JER](#), it is possible to keep the average p_T value per p_T bin invariant. For this, one applies a constant factor to the predicted calibration factors to keep the [JES](#) constant per bin. This method is commonly used in the [GSC](#) calibration (see Section 2.3.2), where the [JER](#) is improved by gradually decreasing the jet response's dependence on the flavour of the constituents or other effects. Furthermore, this barely affects the resolution of the predicted jets, which is still lower (and therefore better) than the resolution of the [reco](#) jets.

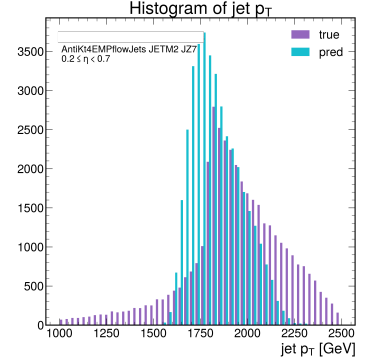
Figure 9.2 displays the jet response and resolution after the correction of the predicted calibration factors. For comparison, one can refer back to Figure 9.1d and Figure 9.1e, where no re-scaling of the calibration factors has been done. As expected, Figure 9.2b confirms that the correction barely affects the [JER](#), whereas it strongly corrects the [JES](#) (see Figure 9.2a) to keep the mean response invariant.



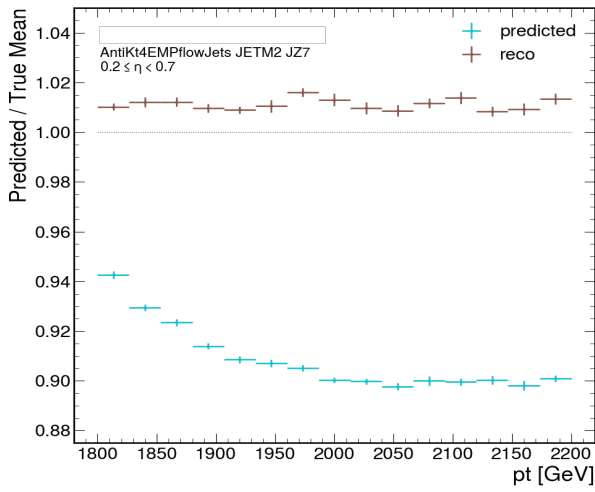
(a) Total training loss per epoch.



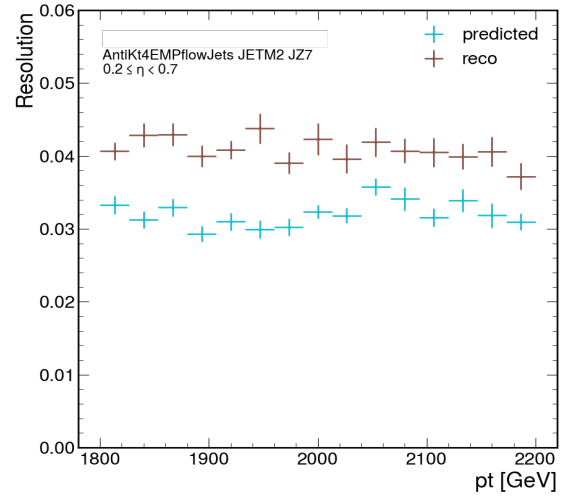
(b) Histogram of **reco** (brown) vs. predicted (cyan) p_T values.



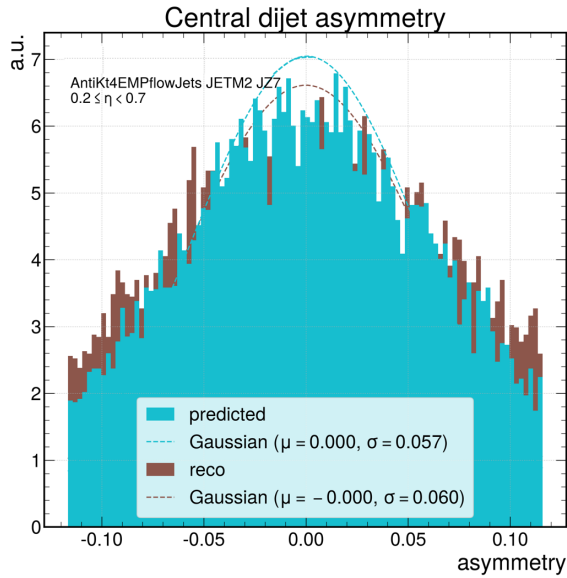
(c) Histogram of true (purple) vs. predicted (cyan) p_T values.



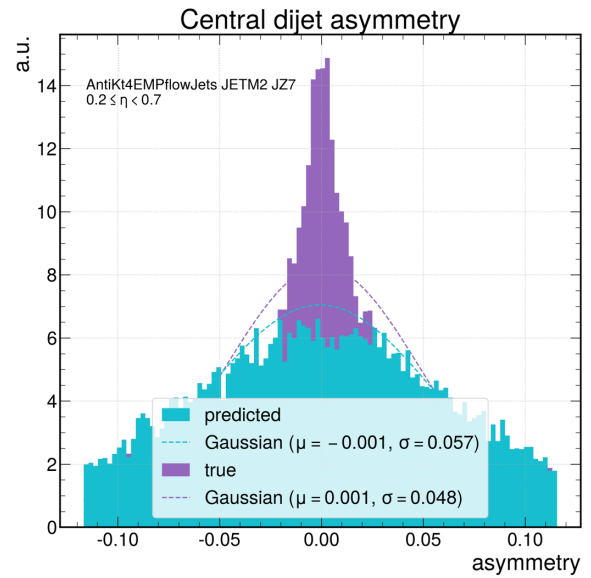
(d) Mean response per bin of ratio of predicted (cyan) or **reco** (brown) over true p_T .



(e) Resolution per bin for predicted (cyan) and **reco** (brown) jet p_T .

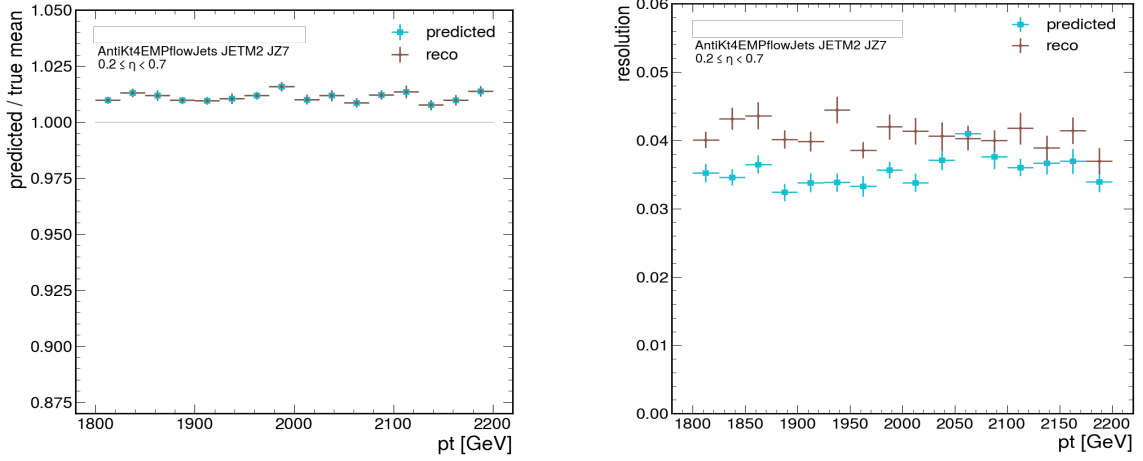


(f) Central dijet asymmetry of **reco** vs. predicted p_T values including a Gaussian with mean and standard deviation of the central asymmetry.



(g) Central dijet asymmetry of true vs. predicted p_T values including a Gaussian with mean and standard deviation of the central asymmetry.

Fig. 9.1.: Results after ML training with $f_t = f_r = 0$ and $f_a = 1$.



(a) Re-scaled mean response per bin of ratio of predicted (cyan) or reco (brown) over true p_T [GeV]. (b) Re-scaled resolution per bin for predicted (cyan) and reco (brown) jet p_T [GeV].

Fig. 9.2.: Mean response (Figure 9.2a) and resolution (Figure 9.2b) per bin after re-scaling calibration factors such that the mean response is invariant per bin.

The re-scaling of the calibration factors only depends on the chosen binning and the average reco p_T per bin. Since it does not demand any truth-level information, one can envision applying the previous steps (ML training only with asymmetry-based term plus re-scaling) after having corrected the JES with previous calibration steps (which can be achieved with the standard ATLAS calibration or a new ML-based jet calibration). In summary, this re-scaling approach provides a first, simple and straightforward application of the methods developed for this project.

9.2 Truth-Independent Training

It is further interesting to combine the asymmetry-based loss function with the MSE loss comparing the predicted to reco level. This corresponds to the constraint term proposed in Eq.(8.6). Explicitly, the loss function corresponds to:

$$\text{loss}(\theta) = f_a \cdot (\sigma_{A_1} + \sigma_{A_2}) + f_r \cdot \text{MSE}(p_T^{\text{reco}}, p_T^{\text{pred}}(\theta)), \quad (9.1)$$

where $f_a, f_r \in \mathbb{R}^+$ are hyperparameters.

To study the influence of the hyperparameters, a scan is done where parameter f_a is varied between 0 and 1 in steps of 0.1, while setting $f_r = 1 - f_a$. The goal is to find a configuration which minimises the asymmetry while still giving sufficient importance to the constraining MSE term such that the spread of the predicted dijet asymmetry will not be lower than the spread of the true asymmetry.

For each of the eleven different configurations, a model is trained over 30 epochs on roughly 47k dijet events. This is first done for the usual input samples from JZ7 and then repeated on inputs from JZ2, where we only work with 7k events though. This is partially because our selection cuts affect this sample more strongly, and partially due to efficiency reasons. It has, however, been ensured that the training still converged.

9.2.1 Results for JETM2 JZ7

All training losses and their individual components are shown in Figure 9.3. In each plot, the loss components have been scaled by their respective weighting factors. The black solid line represents their sum, corresponding to the total loss minimised by the model.

The final training loss for each run as a function of the parameter f_a is shown in Figure 9.4. The key aspects of this plot deserve further comments.

Firstly, the solid lines denote the final training loss values of the components of the loss function that enter the total loss function. The dotted line in blue additionally shows the evolution of the **MSE** between the true jet p_T and its prediction. Secondly, the more f_a grows, the more the final asymmetry-based loss (in orange) is minimised. Reversely, the final **reco**-based loss (in green) increases with f_a , since its weight is determined by $f_r = 1 - f_a$. This shows that modifying the hyperparameters indeed adjusts the importance of the loss terms in the training process. Thirdly, even though the **reco**- and truth-based **MSEs** are strongly correlated, in the limit of $f_a \rightarrow 0$ and $f_a \rightarrow 1$ they differ notably. Especially, in the limit of $f_a \rightarrow 0$ (i.e. $f_r \leftarrow 1$), it becomes obvious that minimising the **MSE** between the **reco** and predicted jets does not minimise the truth-based loss as expected.

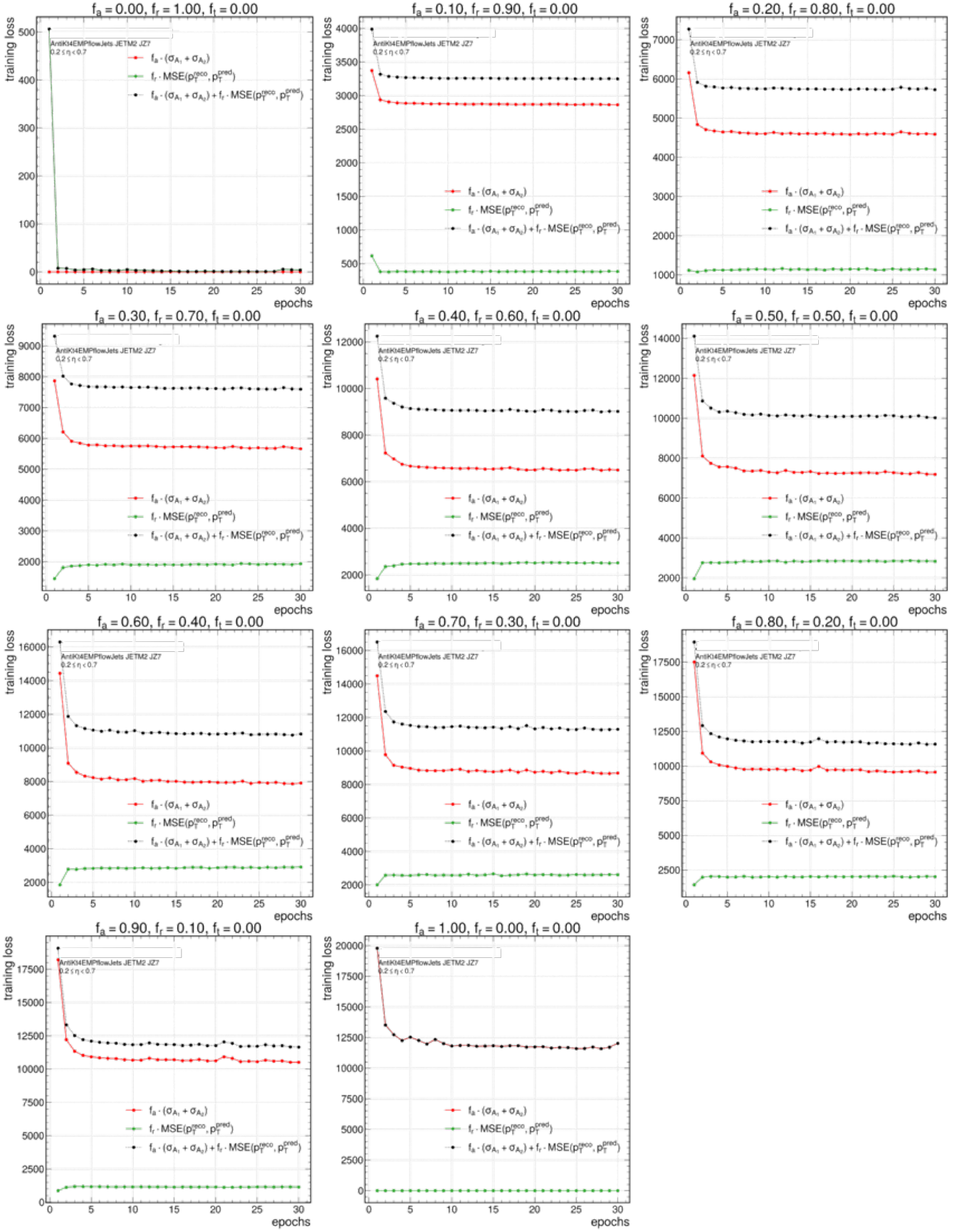


Fig. 9.3.: Training loss functions and their components for each combination of hyperparameters f_a and $f_r = 1 - f_a$.

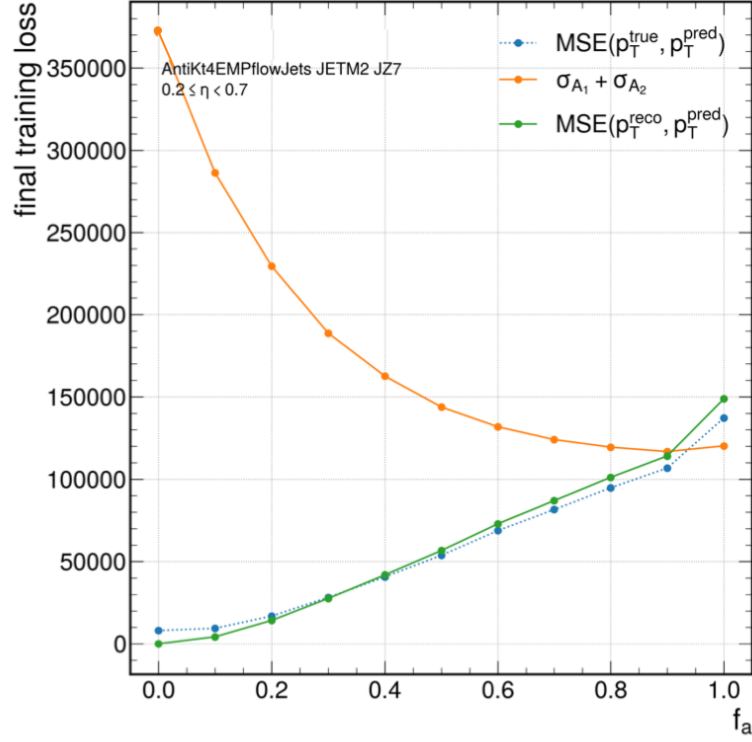
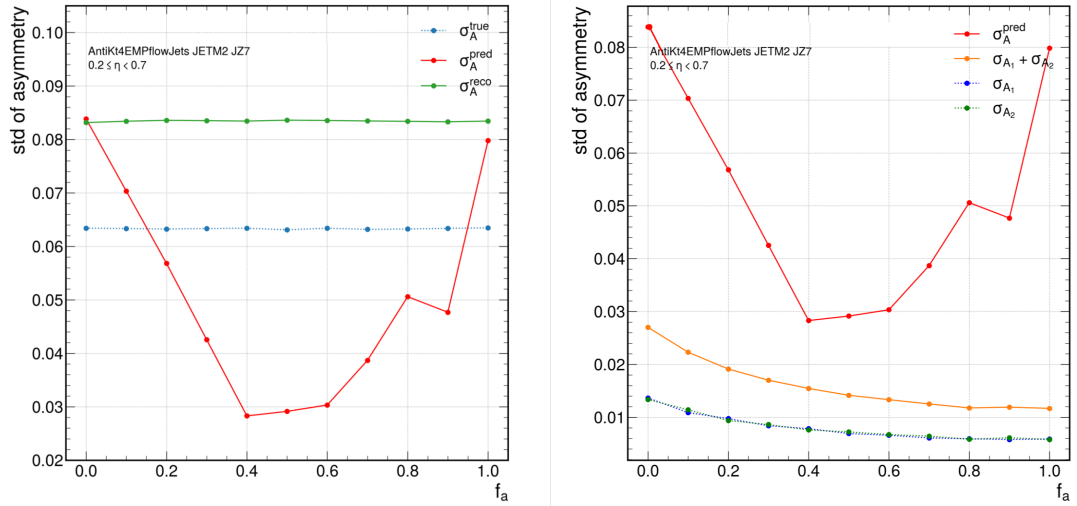


Fig. 9.4.: Final training loss terms for each hyperparameter configuration shown as a function of f_a for JETM2 JZ7. The asymmetry-based (orange) and reco-based (green) components of the total loss function are shown separately, and the MSE between true and predicted jet momenta (blue) is included as an additional metric.

Figure 9.5 gives some information on the evolution of the standard deviation of the dijet asymmetry as a function of f_a . For each configuration, the trained model is applied to the same test data. Therefore, the estimated standard deviation of the true (blue) and reco (green) asymmetry distribution does not change across the different runs.

Meanwhile, the standard deviation σ_A^{pred} of the predicted (red) dijet asymmetry varies when f_a increases. It might seem surprising that σ_A^{pred} , after initially decreasing, increases again. This can be explained by the ‘self-constraining’ nature of the new asymmetry-based loss term from Eq.(8.11), which was explicitly constructed such that the spread of the asymmetry could not become arbitrarily small. For completeness, Figure 9.5b compares the evolution of σ_A^{pred} (still red) with the evolution of $\sigma_{A_1} + \sigma_{A_2}$ (orange), which corresponds to the term minimised in the loss function. The latter is decreasing as more weight is attributed to the asymmetry-based loss term, i.e. with increasing f_a .

Finally, one notices that σ_A^{pred} gets narrower than σ_A^{true} quite fast (ca. at $f_a \approx 0.15$). This is undesired as we only wish to correct for detector effects with our calibration model and not account for any physics effects which are still present



(a) Standard deviation of true (blue), predicted (red) and reco (green) dijet asymmetry distribution as a function of f_a . (b) Standard deviation of predicted (red) dijet asymmetry distribution compared to actual asymmetry-based loss as a function of f_a .

Fig. 9.5.: Results of standard deviation of dijet asymmetry as a function of f_a using JETM2 JZ7 samples.

at the truth level. It follows that the optimal hyperparameter configuration for this loss function is to restrict $f_a < 0.15$ (and $f_r > 0.85$).

9.2.2 Results for JETM2 JZ2

The same studies detailed in the previous section have been repeated with input samples coming from the JZ2 slice, which are covering a lower p_T range ($p_T \in [20, 160]$ GeV). The evolution of the training loss for each hyperparameter configuration can be found in the Appendix A.3.1. Figure 9.6 shows the final loss for the different terms of the chosen loss function (given in Eq.(9.1)). The MSE between the true and predicted p_T (in blue) is also shown as an additional metric.

Figure 9.7 displays the evolution of the predicted dijet asymmetry distribution as a function of f_a . For each configuration, the trained model is applied to the same test data. Therefore, the estimated standard deviation of the true (blue) and reco (green) asymmetry distribution does not change across the different runs. Differently from the previous results (see Figure 9.5), we observe that σ_A^{true} and σ_A^{reco} for this test sample are higher than for JETM2 JZ7.

Furthermore, in Figure 9.7a, it is observed that σ_A^{pred} does not cross the blue line (i.e. $\sigma_A^{pred} > \sigma_A^{true}$). We attribute this to the fact that the physics effects are more dominant at a lower p_T range, making it harder for the model to decrease the

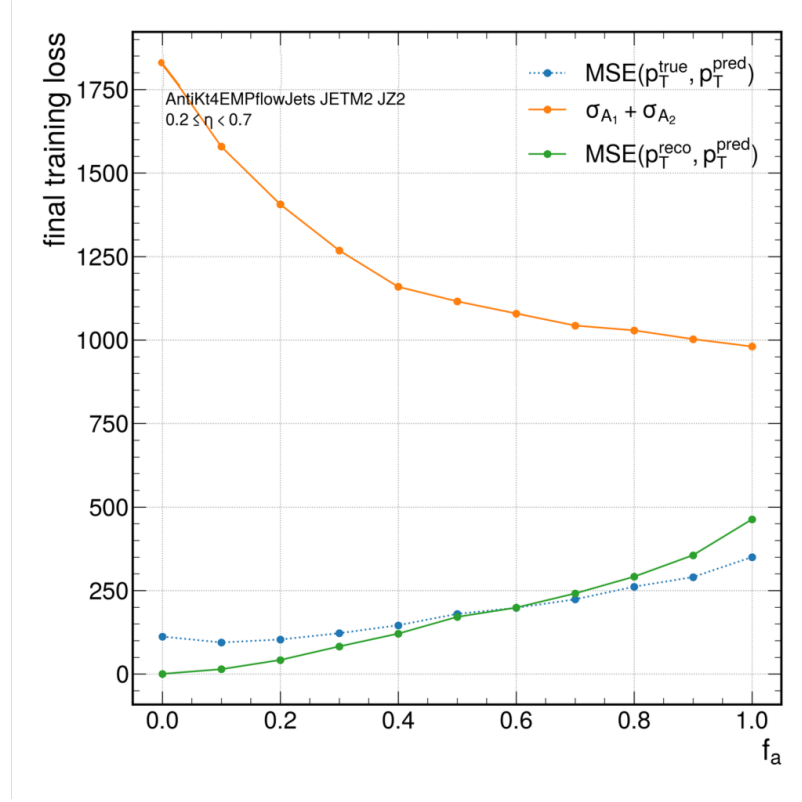
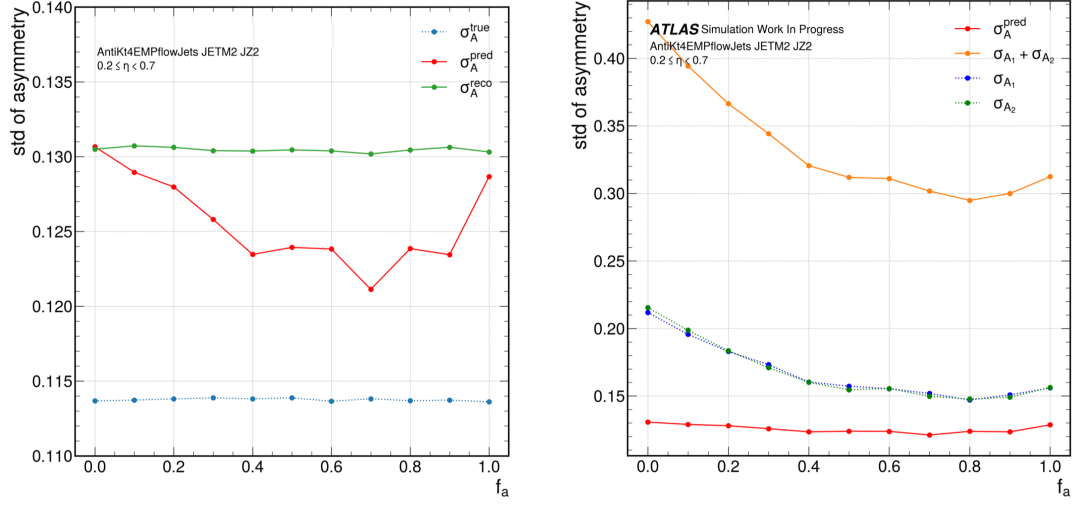


Fig. 9.6.: Final training loss terms for each hyperparameter configuration shown as a function of f_a for JETM2 JZ2. The asymmetry-based (orange) and reco-based (green) components of the total loss function are shown separately, and the MSE between true and predicted jet momenta (blue) is included as an additional metric.

spread of the predicted dijet distribution with this self-constrained asymmetry term.

Finally, Figure 9.7b compares the evolution of σ_A^{pred} (still red) with the evolution of $\sigma_{A_1} + \sigma_{A_2}$ (orange), which corresponds to the term minimised in the loss function. Compared to Figure 9.5b, the order of the two terms has been switched because the orange term is now much higher than before.

In this case, one does not need to worry about the σ_A^{pred} becoming smaller than σ_A^{true} , so the optimal configuration is given by the one with the lowest σ_A^{pred} . Based on this, the best hyperparameter choice is $f_a = 0.7$ (and $f_r = 0.3$).



(a) Standard deviation of true (blue), predicted (red) and reco (green) dijet asymmetry distribution as a function of f_a . (b) Standard deviation of predicted (red) dijet asymmetry distribution compared to actual asymmetry-based loss as a function of f_a .

Fig. 9.7.: Results of standard deviation of dijet asymmetry as a function of f_a using JETM2 JJ2 samples.

9.3 Partially Truth-Based Training

To avoid the trivial solution, one can also introduce a [MSE](#) loss term which computes the error between the predicted and true p_T values. Concretely, the loss function corresponds to:

$$\text{loss}(\theta) = f_a \cdot (\sigma_{A_1} + \sigma_{A_2}) + f_t \cdot \text{MSE}(p_T^{\text{true}}, p_T^{\text{pred}}(\theta)), \quad (9.2)$$

where $f_a, f_t \in \mathbb{R}^+$ are hyperparameters.

The two hyperparameters influence the weighting of each loss term and need to be determined empirically. For this, one can do a scan where parameter f_a is varied between 0 and 1 in steps of 0.1, while setting $f_t = 1 - f_a$ and observing the hyperparameter's influence on the performance of the model. The goal is to find a configuration which minimises the asymmetry while still giving sufficient importance to the constraining [MSE](#) term such that the spread of the predicted dijet asymmetry does not become lower than the spread of the true asymmetry. For each of the eleven different configurations, a model is trained over 50 epochs on roughly 47k dijet events. This is first done for the usual input samples from JJ7 and then repeated on inputs from JJ2, where we only work with 13k events. Again, this is because our selection cuts affect this sample more strongly and due to efficiency reasons. It was, however, assured that the training still converged.

9.3.1 Results for JETM2 JZ7

Figure 9.9 shows the evolution of the total training loss (in black) and its components for each hyperparameter configuration. The truth-based loss term (in blue) is very quickly dominated by the asymmetry-based loss term (in red) as the weight f_a of the asymmetry term increases.

Figure 9.8 compares the final loss values of each loss term for each configuration to observe how the changing weight affects the minimisation process. As one would expect, the more f_a increases, the more weight is given to the asymmetry-based loss (orange) and the lower its final value. Inversely, the bigger f_a , the smaller f_t , and the less weight is on the truth-based loss (blue). For completeness, the reco-based MSE (green) from the previous studies is also shown, although it does not enter the loss function. Again, we observe that the reco- and truth-based MSE values are strongly correlated, but in the limit of $f_a \rightarrow 0$ and $f_a \rightarrow 1$ they diverge.

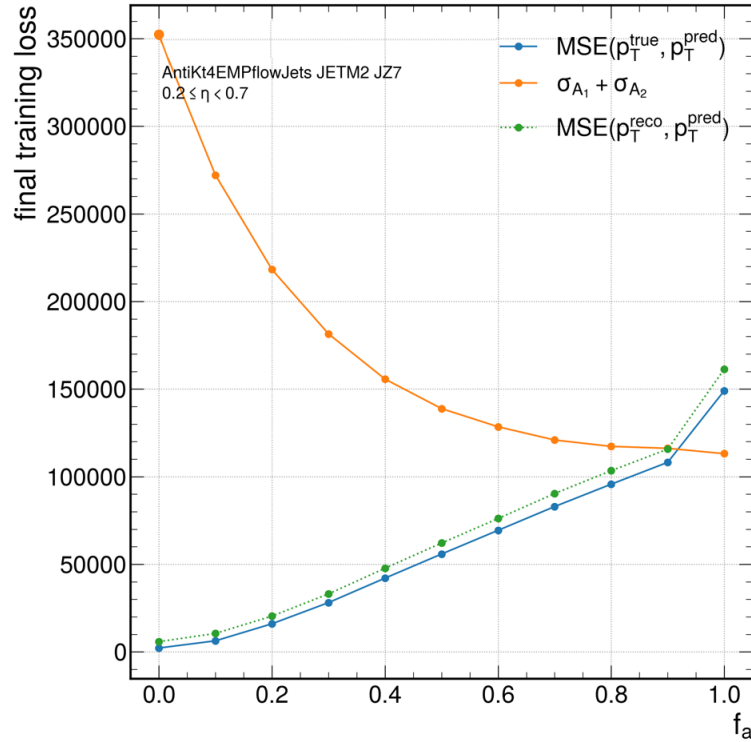


Fig. 9.8.: Final training loss terms for each hyperparameter configuration shown as a function of f_a for JETM2 JZ7. The asymmetry-based (orange) and truth-based (blue) components of the total loss function are shown separately, and the MSE between reco and predicted jet momenta (green) is included as an additional metric.

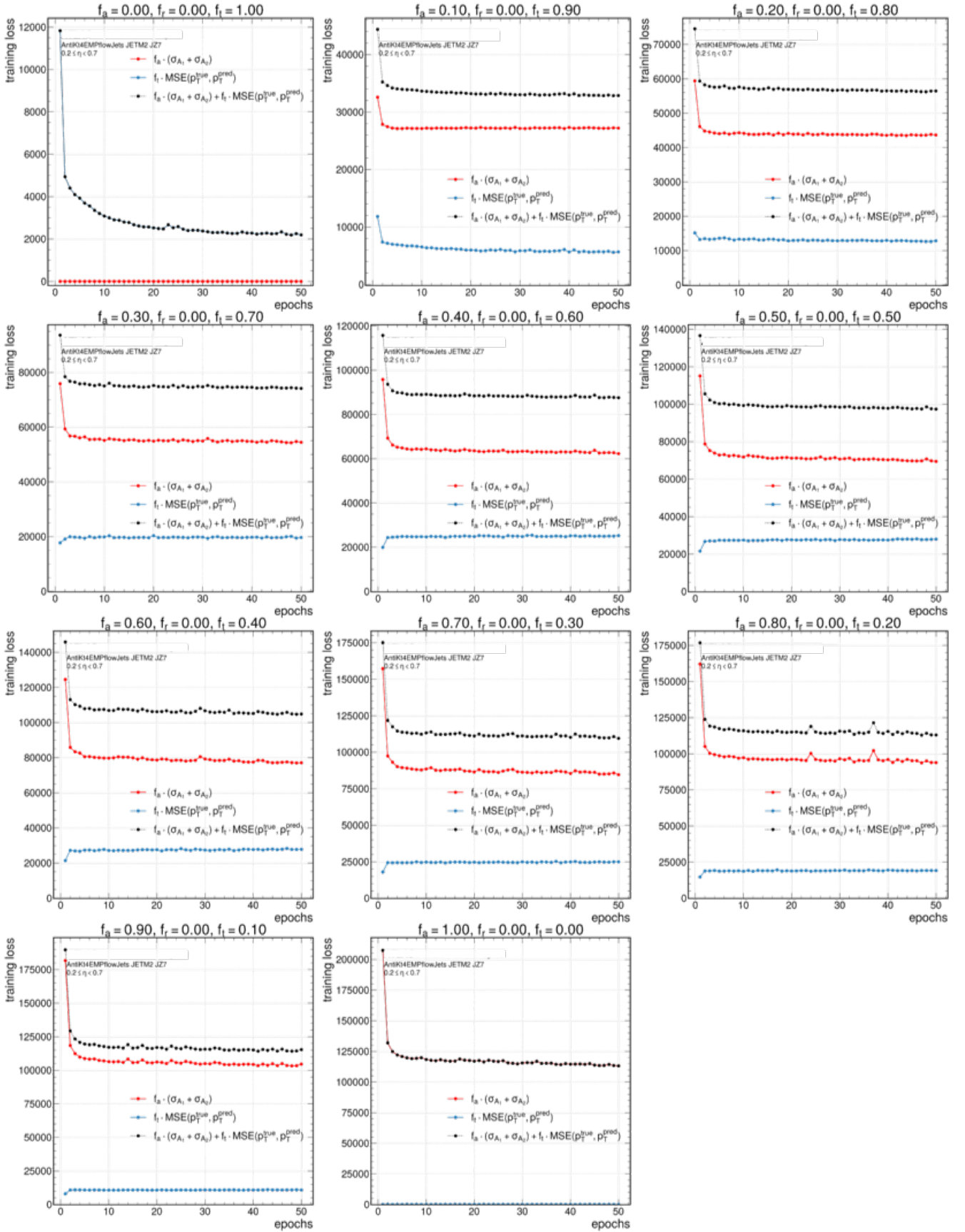
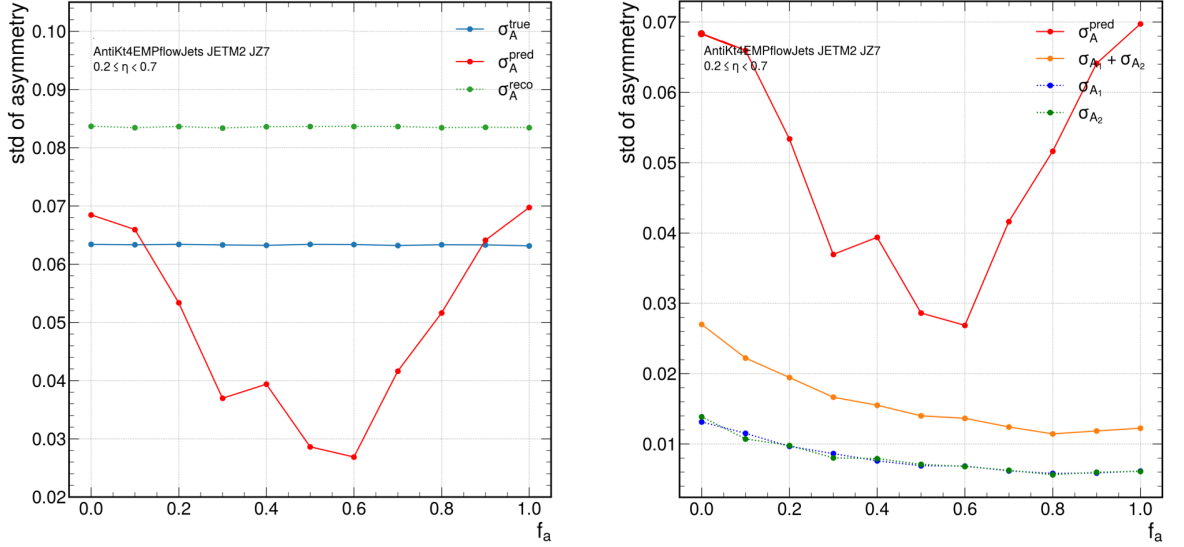


Fig. 9.9.: Training loss functions and their components for each combination of hyperparameters f_a and $f_t = 1 - f_a$ for JETM2 JJZ7.

The plots in Figure 9.10 display the evolution of the spread of the predicted dijet asymmetry distribution via its standard deviation σ_A^{pred} . For each configuration, the trained model is applied to the same test data. Therefore, the estimated standard deviation of the true (blue) and reco (green) asymmetry distribution does not change across the different runs as shown in Figure 9.10a. Differently, the standard deviation σ_A^{pred} of the predicted (red) dijet asymmetry varies when f_a increases. As for the previous studies with the reco-based loss term (see Figure 9.5), one observes that σ_A^{pred} , first decreases and then increases again while f_a grows. Again, this is explained by the ‘self-constraining’ nature of the new asymmetry-based loss term from Eq.(8.11), which was explicitly constructed such that the spread of the asymmetry could not become arbitrarily small. For completeness, Figure 9.10b compares the evolution of σ_A^{pred} (red) with the evolution of $\sigma_{A_1} + \sigma_{A_2}$ (orange), which corresponds to the term minimised in the loss function. The latter is decreasing as more weight is attributed to the asymmetry-based loss term, i.e. with increasing f_a .



(a) Standard deviation of true (blue), predicted (red) and reco (green) dijet asymmetry distribution as a function of f_a . (b) Standard deviation of predicted (red) dijet asymmetry distribution compared to actual asymmetry-based loss as a function of f_a .

Fig. 9.10.: Results of standard deviation of dijet asymmetry as a function of f_a using JETM2 JZ7 samples.

Furthermore, one notices that σ_A^{pred} gets narrower than σ_A^{true} above $f_a \approx 0.12$. This is undesired as the calibration model is only supposed to correct for detector effects and not to account for any physics effects still present at the truth level. It follows that the optimal hyperparameter configuration for this loss function is to restrict $f_a < 0.12$ (and $f_t > 0.88$).

9.3.2 Results for JETM2 JZ2

Finally, the same hyperparameter scan is repeated for samples from JETM2 JZ2, which contains jet in a lower p_T range ($p_T \in [20, 160]$ GeV). The evolution of the loss function and its components for each configuration are found in Appendix A.3.2.

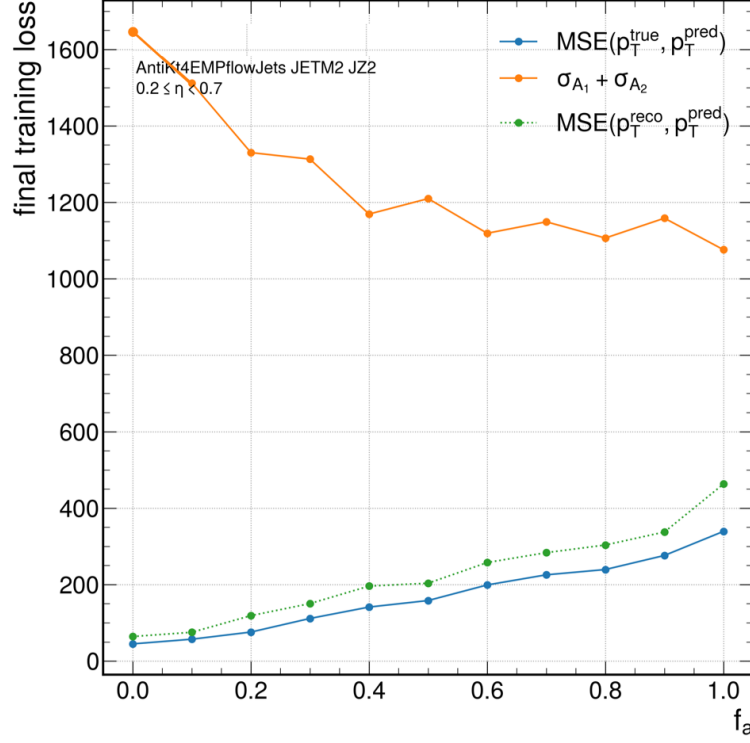


Fig. 9.11.: Final training loss terms for each hyperparameter configuration shown as a function of f_a for JETM2 JZ2. The asymmetry-based (orange) and truth-based (blue) components of the total loss function are shown separately, and the MSE between reco and predicted jet momenta (green) is included as an additional metric.

Figure 9.11 visualises the final values of the different loss terms: in orange the asymmetry-based loss and in blue the truth-based loss. The reco-based loss term (green), which was used in the previous, truth-independent studies, is shown for completeness. Again, the final asymmetry-based loss values decrease when the weighting factor f_a increases, whereas the opposite is observed for the final truth-based loss values.

Figure 9.12 presents the evolution of the predicted dijet asymmetry distribution as a function of f_a . For each configuration, the trained model is applied to the same test data. Therefore, the estimated standard deviation of the true (blue) and reco (green) asymmetry distribution does not change across the different runs. Differently from the results for JZ7 (see Figure 9.10), one notices that σ_A^{true}

and σ_A^{reco} for this test sample are higher. Further, in Figure 9.12a, it is observed that σ_A^{pred} does not cross the blue line (i.e. $\sigma_A^{pred} > \sigma_A^{true}$). We attribute this to the fact that the physics effects are more dominant at this lower p_T range, so it is harder for the model to decrease the spread of the predicted dijet distribution with this self-constrained asymmetry term.

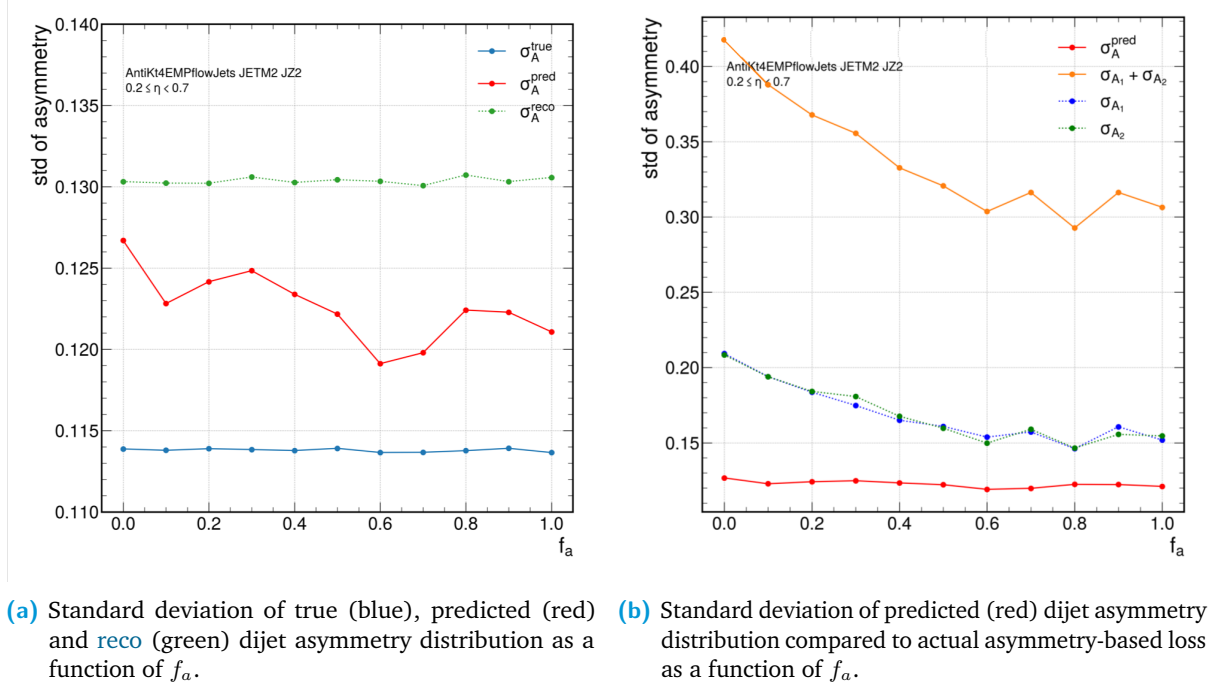


Fig. 9.12.: Results of standard deviation of dijet asymmetry as a function of f_a using JETM2 JZ2 samples.

Finally, Figure 9.12b compares the evolution of σ_A^{pred} (red) with the evolution of $\sigma_{A_1} + \sigma_{A_2}$ (orange), which corresponds to the term minimised in the loss function. Compared to Figure 9.10b, the order of the two terms has been switched because the orange term is now much higher than before.

In this case, one does not need to worry about the σ_A^{pred} becoming smaller than σ_A^{true} , so the optimal configuration is given by the one with the lowest σ_A^{pred} . Based on this, the best hyperparameter choice is $f_a = 0.6$ (and $f_t = 0.4$).

9.4 Iterative Approach

Assuming that **reco** jets could be used as ‘well-calibrated’ objects for the alternative asymmetry-based loss (introduced in Section 8.3.3) is a strong approximation and might negatively impact the quality of the model’s prediction. To somewhat mitigate this bias, one can envision doing iterations with previously predicted jets as reference objects. Concretely, one re-trains the ML model using previously predicted jets as input and also uses these predictions for computing

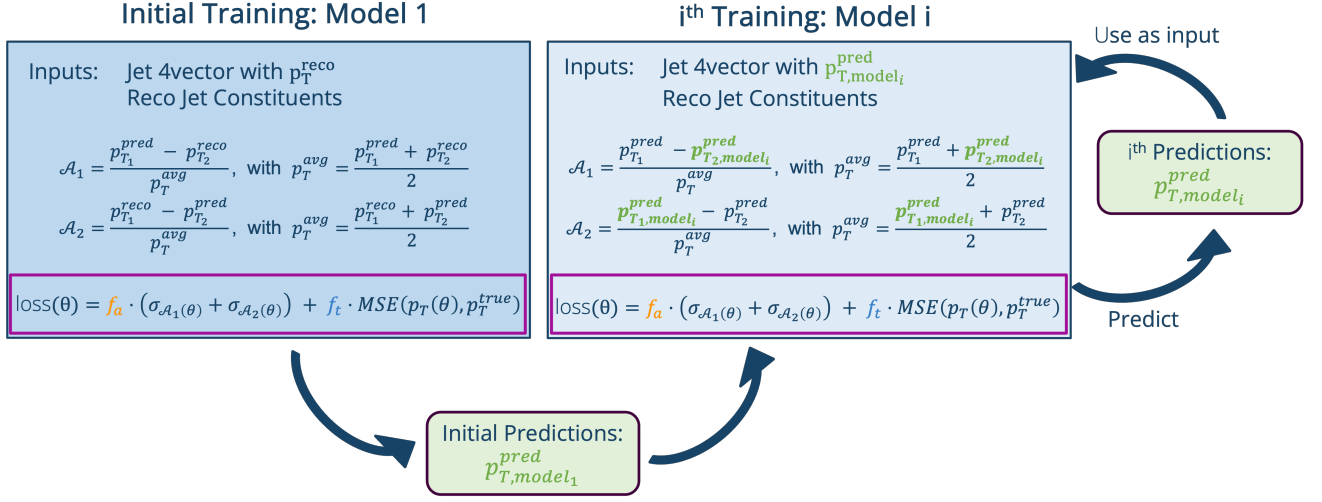


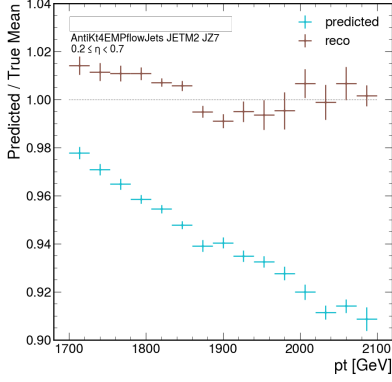
Fig. 9.13.: Schema of iterative approach of jet calibration with the previously introduced ML model (see Section 8.3.3).

the asymmetry-based loss term. The iterative approach is schematically summarised in Figure 9.13. Technically, the procedure can be repeated an arbitrary number of times, depending on what is deemed beneficial for the calibration.

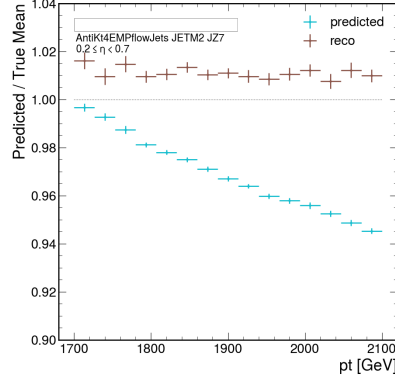
As a first study, the iterative approach is applied to a few selected configurations with the truth- and asymmetry-based loss from Eq.(9.2) for JZ7 and JZ2 samples, respectively. The hyperparameter configuration used in the following example is $f_a = 0.1, f_t = 0.9$. Other studied configurations are $f_a = f_t = 0.5$ and $f_a = 0.1, f_t = 0.9$, which can be found in Appendix A.4.

Figure 9.14 shows the JES and JER after one, two or three training cycles, respectively. It seems that the truth-based loss term somewhat dominates (at least for the second training) and therefore pushes the model to improve the JES. A third training does not improve the JES any more, whereas the JER slightly decreases with every iteration.

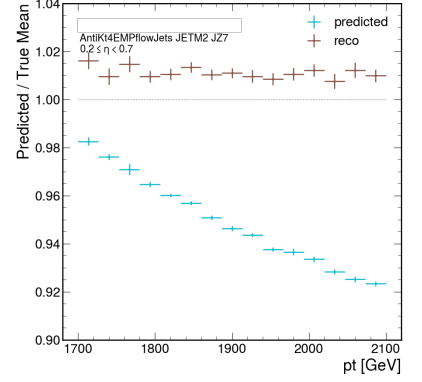
Only one iteration has been studied for JETM2 JZ2 sample with the same hyperparameter configuration. The JES and JER before and after the iteration are presented in Figure 9.15. Very little improvement can be observed in this case. However, for the lowest p_T bins, we notice that the JES improves.



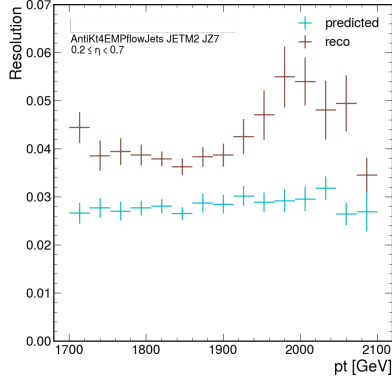
(a) Mean response per bin of ratio of predicted (cyan) or reco (brown) over true p_T after training for 50 epochs.



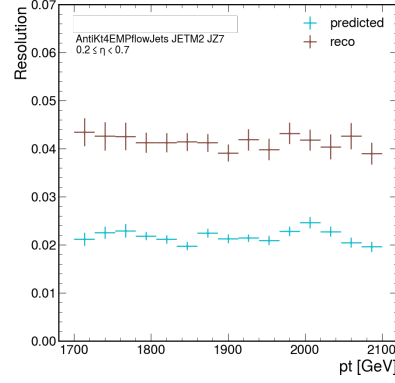
(b) Mean response per bin of ratio of predicted (cyan) or reco (brown) over true p_T after the first iteration.



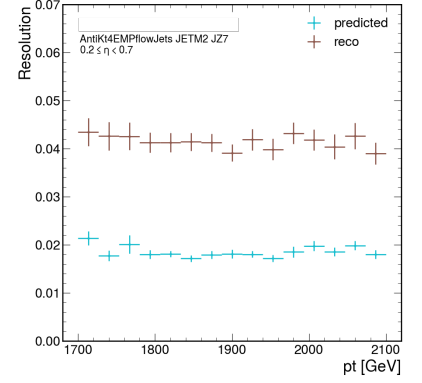
(c) Mean response per bin of ratio of predicted (cyan) or reco (brown) over true p_T after the second iteration.



(d) Resolution per bin for predicted (cyan) and reco (brown) jet p_T after training for 50 epochs.

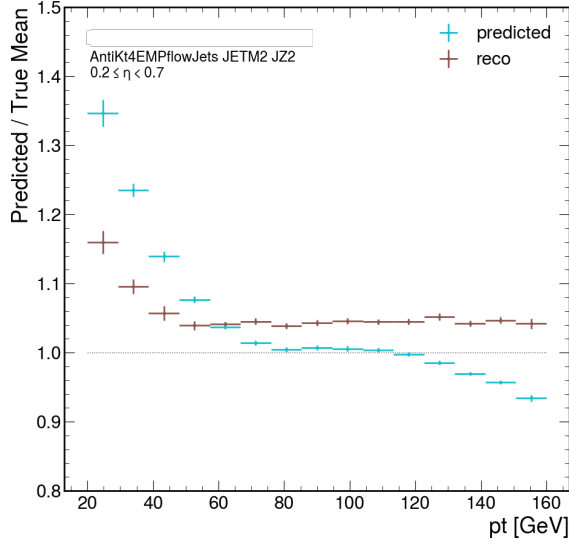


(e) Resolution per bin for predicted (cyan) and reco (brown) jet p_T after the first iteration.

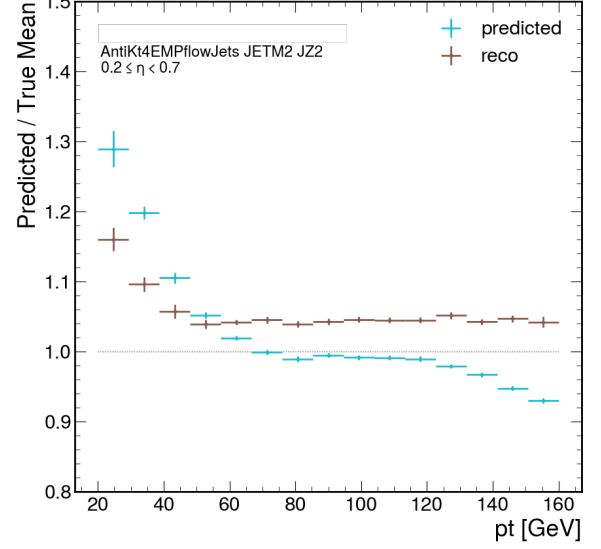


(f) Resolution per bin for predicted (cyan) and reco (brown) jet p_T after the second iteration.

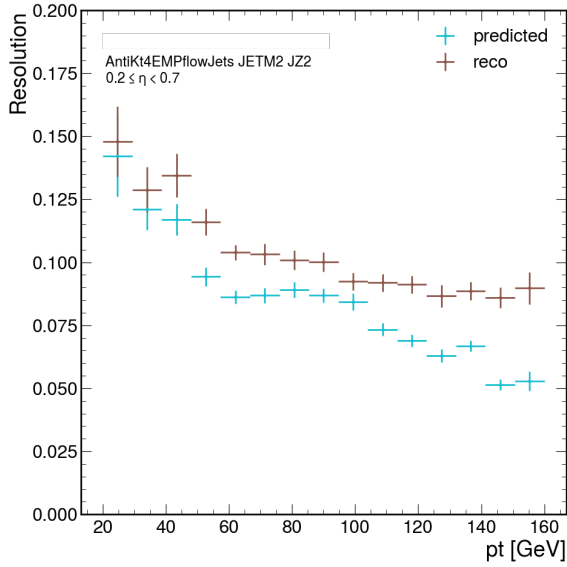
Fig. 9.14.: Iterative approach with two iterations using JETM2 JZ7 samples with $f_a = 0.1$, $f_t = 0.9$. The left two plots visualise the JES and JER after an initial training, respectively. The middle plots display the JES and JER after a first iteration with 20 epochs and the right plots after another 20 epochs.



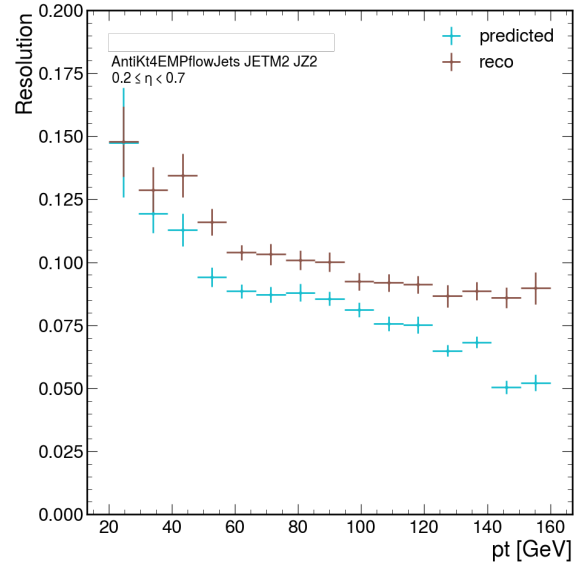
(a) Mean response per bin of ratio of predicted (cyan) or reco (brown) over true p_T after training for 50 epochs.



(b) Mean response per bin of ratio of predicted (cyan) or reco (brown) over true p_T after re-training for 20 epochs.



(c) Resolution per bin for predicted (cyan) and reco (brown) jet p_T after training for 50 epochs.



(d) Resolution per bin for predicted (cyan) and reco (brown) jet p_T after re-training for 20 epochs.

Fig. 9.15.: Iterative approach with two iterations using JETM2 JZ2 samples with $f_a = 0.1$, $f_t = 0.9$. The left two plots visualise the JES and JER after an initial training, respectively. The right plots display the JES and JER after an iteration with 20 epochs.

Summary and Outlook

In summary, the dijet *in situ* asymmetry has been chosen as a physical quantity which can be used for improving the [JER](#) directly in experimental data. However, several challenges arise when introducing a loss term that directly minimises the spread of the dijet asymmetry. The most prominent is the existence of a trivial solution in the optimisation space of the loss function. The model can effectively learn to perfectly minimise the spread of the asymmetry by setting the momentum of both leading jets to an arbitrary constant. This is a valid solution to the loss function, despite being highly unphysical and therefore unwanted.

To tackle this problem, the asymmetry-based loss term is modified to only act on one of the two leading jets at a time and other loss terms are introduced to further constrain the solution space. From there, two approaches can be derived to optimise the [JER](#): one completely independent of any kind of truth information and one partially dependent on it. Therefore, the first method can be executed entirely with experimental data, whereas the second method can be implemented using experimental data and simulation simultaneously. For both of these approaches, it is crucial to determine the best weighting of the loss terms empirically.

Additionally, the new asymmetry-based loss term is based on the assumption that some jets are well calibrated, which is likely too optimistic. To mitigate the effects of this assumption, the presumably ‘well-calibrated’ jet p_T can also be replaced with the predictions from a previous model iteration and be adjusted in an iterative manner.

Finally, this project has been presented at the ‘ATLAS Hadronic Calibration Workshop’ in 2023 and 2024, and was summarised in an internal ATLAS note [\[207\]](#).

For future studies, it could be interesting to apply this method to other event topologies such as Z +jet or γ +jet events, where Z or γ can be assumed to be ‘well-calibrated’ objects.

Furthermore, it might also be helpful to reweigh the jet events directly in the loss function to make up for artificial peaks in the input p_T distribution, since [NNs](#) train more easily on flat distributions.

Part III

Time Series Anomaly Detection (TSAD)

Introduction

With the evolution of the [Internet of Things \(IoT\)](#) and the increase in data taking in general, many relevant data from various applications come in the shape of *time series*, i.e. sequential data instances indexed over time [105]. [TSAD](#) is a critical task in diverse fields such as healthcare, financial services, cybersecurity, transportation, and monitoring systems, where it ensures the correct operation of manufacturing lines, water treatment facilities or physics detectors, to name a few. Accurate identification of the moment when unexpected errors or defects occur is essential for these applications.

One typically distinguishes between *univariate* (one-dimensional) and *multivariate* (multidimensional) time series. In this case, the correlations between different dimensions must be taken into account, which increases the complexity of the problem.

[TSAD](#) is a challenging problem that has attracted interest from both ATLAS and IBM Research. It represents a rare intersection of research priorities: ATLAS seeks to apply [TSAD](#) for data quality and detector monitoring, while IBM explores its use in fraud detection. This shared interest makes [TSAD](#) a natural focus for collaborative research, with potential for impactful outcomes in both domains.

In the context of [HEP](#), [TSAD](#) is primarily studied as a tool for [DQM](#) [153]. Specifically at detector-level, the temporal correlations among the collision events are non-negligible, and [TSAD](#) methods offer the potential to further improve and automate the detection of detector defects.

Meanwhile, IBM Research is exploring new approaches to fraud detection. In financial systems, anomalies might correspond to fraudulent credit card transactions hidden within a large volume of legitimate ones. Timely detection of such anomalies can save financial companies or institutions billions of dollars [208–210]. While IBM already provides a wide range of automation solutions to its clients, it continuously seeks to expand its catalogue with innovative and modern algorithms.

In both cases, the associated challenges are similar: Anomalies are very rare, and often, data are lacking reliable and correct labels. For instance, in 2018, while less than 1% of card transactions were fraudulent, the total amount of fraudulent card transactions in the European Union amounted to €1.8 billion [211].

Furthermore, financial data underlie strict confidentiality and privacy regulations, making it almost impossible to find realistic datasets without important information missing.

Differently, HEP drowns in huge amounts of data, but efficiently filtering out the interesting events, while maintaining high-quality standards for data selection and calibration, is just as much of a challenge. Furthermore, there is a lack of reliably labelled data; most labelling is done by human shifters, a process that is both error-prone and time-consuming.

The work is structured as follows: First, Section 12.4 provides some further introduction for TSAD in general and its applications in physics. Second, Chapter 13 focuses on the development and comparison of unsupervised strategies for anomaly detection on multivariate time series. These studies are entirely realised with publicly available data. Third, in Chapter 14, synthetic anomalies related to detector defects are introduced in a general-purpose simulation framework for HEP calorimeters, the Lorenzetti Showers simulation [212]. Then, the strategies established in Chapter 13 are employed to identify the artificial anomalies in the data. Finally, Chapter 15 applies the same strategies to experimental data from the ATLAS LAr calorimeter for DQM purposes and thereby demonstrates their utility in a real-world setting.

Related Work

12.1 TSAD Algorithms

TSAD has become an increasingly active area of research. With the growing availability of data from diverse sensors and monitoring systems, numerous new applications have emerged, giving rise to a wide range of novel research directions. Given the vast range of applications and corresponding research in these fields, we will not cover them entirely but refer the reader to more exhaustive surveys and reviews [105, 213–216]. Furthermore, for the specific topic of univariate TSAD, more information can be found in [217, 218].

By definition, anomalies correspond to unexpected and rare instances in data, often corresponding to patterns that have never occurred before, making it impossible to teach a model about them in advance. This inherent unpredictability renders supervised approaches unsuitable for TSAD, leading most TSAD methods to rely on unsupervised strategies instead.

Classical, unsupervised approaches entail methods such as Isolation Forests (IFs) [219], One-Class Support Vector Machines (OCSVMs) [220], clustering algorithms, k-Nearest Neighbour (kNN) methods [88] or algorithms like MERLIN [221] to name a few [222, 223]. However, in recent years, many of them have been outperformed by deep learning approaches, though some remain relevant. Therefore, the focus in TSAD currently mostly lies on deep learning, as numerous prior works have demonstrated its superior performance [105, 224–227].

A complete review of deep learning approaches for TSAD can be found in [214]. Unsupervised, deep learning methods for TSAD entail models such as GANs [228], AEs [148–150], VAEs [152], Graph Neural Networks (GNNs) [229] and RNNs [92, 168] including LSTM models [169]. A popular example for TSAD is DAGMM [230] that jointly trains a Gaussian mixture model with an AE. The AE helps with data compression, allowing the Gaussian mixture model to better capture multivariate correlations, whilst the combined training assures that this is done effectively. OmniAnomaly [231] is an example of a model employing stochastic RNNs for robust anomaly detection on multivariate time series. Next, USAD [224] is based on adversarially trained AEs, facilitating fast training and increased robustness. Finally, an LSTM-based AE [232] combines the advantages of AEs with the strengths of LSTMs at capturing temporal dependencies.

Recently, transformer models have received increased attention due to their ability to model long-term dependencies through self-attention mechanisms, especially when combined with positional encoding. They also enable parallel computation and help mitigate gradient-related issues that can still pose challenges for models like [LSTMs](#) on longer sequences [105, 233, 234]. Transformer-based approaches to [TSAD](#) entail models such as TranAD [225], MT-RVA [235], TransAnomaly [236], GTA [237], Anomaly Transformer [238], VTT [239], which explore the combination of transformers with adversarial training, [VAEs](#), [GNNs](#), association discrepancy or alternative embeddings, respectively.

12.2 Benchmark Datasets

Although the importance of benchmark databases is well established in the machine learning community, there is a severe lack of reliable and challenging anomaly detection data for time series. An overview of currently available datasets is presented by Lai et al. [222], but all except four of the presented datasets are synthetic. Other public datasets that have been proposed to evaluate anomaly detection in time series are the Yahoo [240], Numenta [241], NASA (MSL and SMAP) [242] and SMD [231]. However, concerns have been raised about the validity and usefulness of these datasets for benchmarking. Wu and Keogh [243] argue that most of these commonly used time series have such obvious anomalies that they can be identified by simple rules or one-line algorithms, and therefore, are not suited to benchmark a model's anomaly detection performance. Furthermore, some benchmarks also suffer from mislabelled test data and biases such as run-to-failure bias. The authors propose the UCR time series anomaly archive [243]. Whilst containing realistic anomalies, this data collection has the drawback that it is currently limited to univariate time series. Similarly, Nakamura et al. [221] assert that current benchmarks are insufficient and do not allow for a complete comparison of emerging algorithms. Finally, Han et al. [188] focuses on tabular data for anomaly detection but includes some recommendations for TSAD.

12.3 Evaluation Protocols

The methods introduced in the previous section primarily focus on attributing an anomaly score to individual data instances. How to transform these predictions into binary anomaly labels and how to compare these labels with the true labels constitutes another challenge and is hardly discussed in [TSAD](#) method papers.

Given the lack of comprehensive discussion and comparison of anomaly label extraction techniques for continuous anomaly scores, this is addressed in our work on unsupervised strategies for [TSAD](#), more precisely in Section [13.3](#).

Concerning the evaluation of predicted anomaly labels, a variety of performance metrics and evaluation protocols exist. Standard (point-wise) evaluation metrics for anomaly detection are introduced in Section [3.4.2](#). Kim et al. [[198](#)] analyse evaluation protocols, especially those focusing on anomalous sequences, and demonstrate their tendency to overestimate a model's detection performance. Sehili and Zhang [[244](#)] build upon these studies and review various evaluation protocols. Notably, the authors also link the limitations of popular benchmark datasets with the need for more suitable and robust evaluation protocols for a fairer comparison of detection performance. They affirm that point-wise metrics remain a valid choice for unbalanced data but discuss their limitations when dealing with very long anomalous sequences (as for instance in *SWaT*). To tackle this problem, they propose an *event-wise* evaluation protocol and apply it to established [TSAD](#) models.

12.4 TSAD for Physics

[TSAD](#) based on [ML](#) is only starting to be applied to various problems in fundamental physics. This is because many physics phenomena are either time invariant or the precise timing information of an experimental observation is not available due to detector effects. However, there are a few examples and studies where [TSAD](#) methods are tested or even implemented to tackle physics-related problems.

Gravitational waves physics has gained significant momentum since their first detection in 2015 [[245](#)], and [TSAD](#) can be used to identify such signals. In this context, there is an explicit temporal aspect to the physics phenomenon because waves manifest as time-dependent fluctuations. Hence, the data recorded by gravitational wave interferometer are naturally represented as univariate time series and [TSAD](#) methods can be employed to analyse and distinguish gravitational events from the noisy background. Most of the standard approaches to gravitational wave detection still rely on supervised methods, but recently, unsupervised anomaly detection methods based on [ML](#) have been investigated [[246](#)]. A straightforward approach is implementing an [AE](#) based on a one-dimensional [CNN](#) for detecting gravitational events [[247](#)]. Other approaches entail an explicit time-frequency method that decomposes the time series into frequency space before applying a 2D [CNN](#) to extract the gravitational wave signal [[248](#)]. Additionally, the use of a [LSTM](#)-based [AE](#) has been investigated and compared

with other types of AE models, where LSTMs appear advantageous due to their inherent ability to model sequential data [249]. In follow-up studies, the LSTM-AE method has been completed with a low-dimensional embedding space, where different regions correspond to different types of simulated signal or background events [250]. Although the input time series are univariate, additional time series containing sensor and system information can be incorporated, effectively rendering the problem multidimensional. Furthermore, recent approaches consider time series from multiple data streams originating from different interferometers, aiming to detect coherence across detectors [251]. This is achieved using two CNN models; one targeting coincident signals across multiple detectors and another focusing on phase and amplitude correlations between the same detectors.

Differently, collisions at the LHC are considered to be permutation invariant (i.e. they have no explicit temporal order) on their own. However, in practice, temporal dependencies are added to the events through the detector interaction and additional effects in the detector readout. Therefore, there is a growing interest in automating or improving the efficiency of DQM tasks regarding various physics experiments, even though very few projects have been concretely implemented yet [153]. DQM is an essential task for the experiments to assure the quality of the data used in the subsequent physics analyses. This is especially relevant with the upcoming HL-LHC upgrade that is leading to a significant increase in the data-taking rate, and the time and resources required for DQM cannot scale proportionally. Therefore, it is crucial to study and implement more efficient and less human-centred protocols as soon as possible.

CMS has investigated various supervised and unsupervised approaches to DQM [191, 252, 253]. For instance, for online DQM of the CMS electromagnetic calorimeter, a CNN-based AE is proposed [154], where the anomaly detection model classifies 2D images of the calorimeter cells. The 2D images contain synthetic and real detector defects that the learning model tries to identify. To account for the temporal dependencies of anomalies in the detector, dedicated time corrections are implemented during post-processing. Furthermore, unsupervised anomaly detection techniques for multivariate time series have been studied and employed into the online and offline DQM system of the CMS hadronic calorimeter [254].

Similar approaches are investigated within ATLAS, but fewer have been implemented so far. A TSAD-based approach has been proposed for the ATLAS Data Flow and has been tested on a part of the TDAQ system [255]. This project applies a clustering algorithm for dimensionality reduction combined with an unsupervised TSAD algorithm to monitor the HLT system. Finally, the application of a LSTM-AE to DQM tasks for the LAr calorimeter has been proposed [78], and our work presented in Section 2.4 will build upon that.

Unsupervised Strategies for Multivariate TSAD

This work is adapted from a paper I prepared and submitted, which is currently under review. The preprint is available in Ref. [3].

In this chapter, the focus lies on unsupervised techniques for [TSAD](#), due to the previously mentioned challenges associated with supervised approaches, namely the lack of labelled training data (for more details see [Section 3.4](#)), and the end-to-end setup of a [TSAD](#) algorithm is discussed. A wide choice of data from various applications is included in these studies to avoid biasing them by some of the limitations of current benchmark datasets.

First, the public datasets ([Section 13.1](#)), the algorithms ([Section 13.2](#)) and experimental setup ([Section 13.4](#)) used for our studies are described. Finally, [Section 13.5](#) presents and discusses the results. Our main contributions are:

- exploring the use of the [iTransformer \[256\]](#) architecture for anomaly detection, and studying the influence of window, step and internal model size on the anomaly detection performance;
- analysing how to extract anomaly labels from a multidimensional anomaly score provided by an anomaly detection model, and what metric to use to evaluate the extracted labels;
- further investigating the impact of anomalies present in training data and how to mitigate this using alternative loss functions;
- comparing different transformer-based models for [TSAD](#) on a wide range of datasets.

13.1 Datasets

13.1.1 Public Data for TSAD

Despite the many issues with current benchmarks and ongoing efforts for more challenging datasets (see [Section 12.2](#)), many of the benchmark sets mentioned

before are still widely used in TSAD. First, we incorporate established benchmark datasets to enable comparison with previous models, though avoiding problematic subsets. Second, we prioritise real-world datasets or datasets with realistically injected anomalies. Finally, we select datasets that feature diverse anomaly types, ensuring a comprehensive evaluation across different scenarios. With this selection, we aim to have a broadly varied spectrum of application fields, anomaly types and data characteristics.

Namely, we use the SMD [231] dataset containing server machine data, and the two NASA datasets, MSL and SMAP [242], which are both based on sensor data collected on Mars. We add the new UCR, which was proposed by Wu and Keogh [243], to mitigate some of the issues of the datasets above. Next, we also include the frequently used SWaT [257] and WADI [258] datasets, both incorporating data from monitoring water treatment plants, and SWaT_1D [259], a univariate time series derived from SWaT. Additionally, Lai et al. [222] mention the Credit Card [260] and the GECCO [261] datasets as examples of real-world benchmarks. These datasets are selected for their diverse anomaly types: The Credit Card set features point anomalies corresponding to fraudulent credit card transactions, while the GECCO set shows collective anomalous patterns signalling an unsafe water quality. Finally, we introduce the IEEECIS [262–264], a dataset consisting of online transactions with a challenging combination of point and collective anomalies. The datasets are described in Table 13.1, and additional information can be found in Appendix B.1.

Tab. 13.1.: Public datasets used for TSAD. The values in brackets next to the anomaly rate indicate the number of anomalous sequences in the test data. The average, maximum and minimum anomaly lengths are measured on the labelled test data.

Dataset	Description	Dimensions	Train size	Test size	Anomaly rate (# anomalous seq.)	Avg. anom. length	Max. anom. length	Min. anom. length
Credit Card	Credit Card Transactions	29	200k	85k	0.13% (107)	1	2	1
GECCO	Water Quality	9	70k	70k	1.05% (22)	33	157	23
IEEEECIS	Online Payments	178 (use 30)	17k	1k	6.67% (12)	5	21	1
MSL	Sensors of Mars Rover	55	58k	74k	10.48% (36)	215	1140	10
SMAP	Soil Moisture on Mars	25	138k	436k	12.82% (68)	821	4217	30
SMD	Server Machine	38	105k	105k	5.14% (35)	154	837	2
SWaT	Water Treatment	51	495k	450k	12.13% (35)	1560	35900	101
SWaT_1D	Water Quality	1	3k	5k	12.88% (83)	8	211	1
UCR	Healthcare	1	1.6k	5.9k	1.88% (4)	59	111	10
WADI	Water Treatment	127 (use 30)	785k	173k	5.77% (14)	713	1741	88

13.1.2 Time Series Anomaly Types

In TSAD, one typically distinguishes between *point* and *collective* anomalies. While the taxonomy of time series anomalies may differ slightly depending on

the exact application, the fundamental definitions remain consistent. A visual characterisation of them is given in Figure 13.1 and further discussion can be found in [105, 213, 215, 218, 222, 265].

First, *point* anomalies describe anomalies that are highly-localised and correspond to one anomalous time stamp. We further distinguish anomalies corresponding to an extreme point of the time series, which are called *global* outliers, from *contextual* anomalies, that describe time stamps that only appear anomalous given the context of their neighbouring time stamps.

Second, *collective* anomalies are also called subsequence or group anomalies, as they encompass a longer, anomalous sequence of a time series, containing multiple time stamps that are anomalous. Often, each of the individual time stamps of such a collective anomaly appears normal, but together they create an anomalous pattern. Such anomalous sequences can be characterised by having either a different pattern shape, trend or seasonality than the previous part of the time series.

Since contextual and collective anomalies both depend on temporal context, they are occasionally treated as a single category, as done in [242].

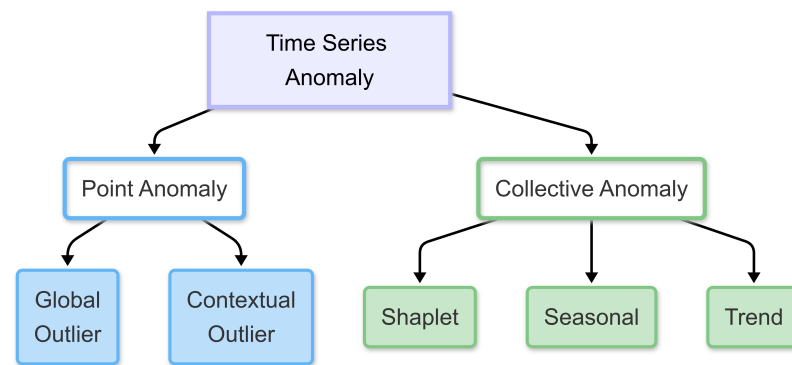


Fig. 13.1.: Taxonomy for anomalies in time series.

Table 13.1 describes the anomaly characteristics such as the type, rate, average, maximum and minimum length of anomalies for each dataset. The given numbers only concern the testing set and not the training set, as in most cases, the training data are either provided without labels or do not contain anomalies by construction.

13.2 Unsupervised Algorithms

TSAD models can be categorised by their underlying assumptions. Most are *reconstruction-based*, where the model reconstructs the input and anomalies are identified through the reconstruction error, typically measured by MSE. The

assumption is that normal data, since abundant and therefore well-learned, are easy to reconstruct, whereas anomalous instances entail higher reconstruction errors (see workflow in Figure 13.2).

An alternative is *forecasting-based* anomaly detection, where anomalies are assumed to be harder to predict, leading to larger prediction errors in a well-trained model. This approach might capture different anomaly types; for instance, contextual anomalies in rapidly changing time series may yield higher forecasting than reconstruction errors [105].

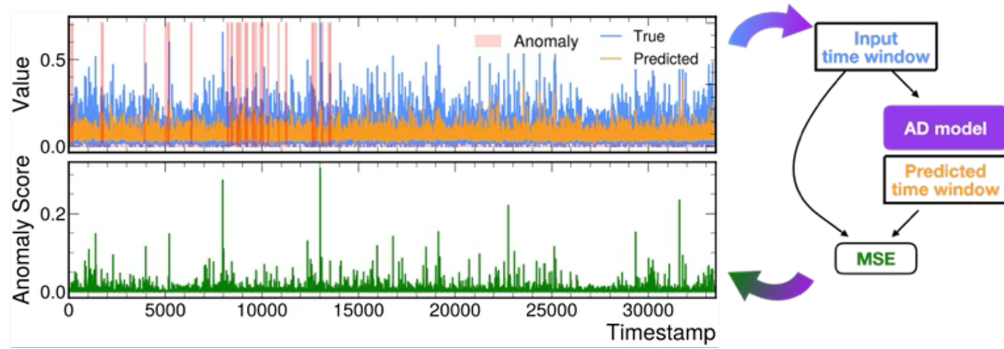


Fig. 13.2.: Schema of reconstruction-based TSAD. The illustration is limited to one dimension, but this naturally extends to multivariate time series, where the anomaly detection (AD) model processes multidimensional input windows.

We first introduce a deterministic baseline approach and a non-transformer model for comparison, followed by transformer-based model candidates for TSAD.

13.2.1 Baseline Approach

As detailed previously, we do not use supervised models, focusing on strategies that do not rely on labelled data for training. To provide a simple baseline algorithm that satisfies that requirement, we directly take the absolute value of the input time series as an anomaly score. This approach is described in [198] and recommended for better comparison against other established anomaly detection models. Different anomaly label extraction strategies (explained in Section 13.3) can be applied to those anomaly scores. The baseline, already achieving a high anomaly detection score, implies that the information in the unprocessed time series is already sufficient to find anomalies, and more complex learning models are unnecessary.

Another popular assumption for time series anomaly detection is that anomalies are harder to predict than normal instances, i.e. a well-trained forecasting model would have larger prediction errors for anomalies than normal data,

and its prediction error can hence be used as an anomaly score. Furthermore, *forecasting-based* anomaly detection might be sensitive to other types of anomalies than reconstruction-based. For instance, it might be harder for a forecasting-based model to predict contextual anomalous data in a rapidly changing time series than for a reconstruction-based model [105], leading to larger forecasting errors.

13.2.2 Deep Learning Baseline Approach

As a deep learning baseline approach, we choose the USAD [224] model. It tackles multivariate TSAD by combining an AE with adversarial training to improve the robustness and efficiency of the training. It is a reconstruction-based anomaly detection model, but instead of solely focusing on the reconstruction error (measured by the MSE), USAD splits the training into two phases.

In the first phase, the compression and reconstruction are learned with the MSE. The USAD uses a single encoder and two decoder models, combining them into two AE models AE_1, AE_2 with the same encoder. Both AEs are trained identically during this phase. In the second phase, adversarial training is introduced: the two AEs are depicted against each other, where data reconstructed by AE_1 is used to try to fool AE_2 , which tries to distinguish real data from reconstructed data. During testing, the anomaly score is defined as

$$l(y) = \alpha|y - AE_1(y)| + \beta|y - AE_2(AE_1(y))| \quad (13.1)$$

where $\alpha + \beta = 1$.

13.2.3 Transformer-Based Anomaly Detection

Transformers are known for dealing exceptionally well with sequential data such as language. Based on this observation, current research tries to determine how far this applies to other types of sequential data, including time series. Discussions concerning the usefulness of transformers for time series forecasting are still ongoing [266–269], as Zeng et al. [268] produced better results with linear models than with transformer-based models. However, unlike traditional sequential models such as RNNs and LSTMs, transformers are not susceptible to vanishing gradients or inefficiencies in computation. Furthermore, their self-attention mechanism enables them to capture complex, multidimensional dependencies in parallel, making them highly effective for modelling long-range relationships in sequential data. For these reasons, various transformer-based models for time series have emerged in the last few years, and we will focus on

two recent and interesting models while also including a vanilla Transformer model for comparison.

First, Tuli et al. [225], the authors of the **TranAD** model, tackle multivariate **TSAD** by introducing adversarial training and self-conditioning. The adversarial training procedure helps amplify the reconstruction error for subtle anomalies, whereas the self-conditioning increases the focus on anomalous subsequences.

Second, the AnomalyTransformer [238] relies on association discrepancy as an anomaly score. The discrepancy is defined as the KL divergence between the prior-association, based on temporal associations between time points, and the series-association, derived from the transformer's self-attention weights.

Next, the inverted Transformer (**iTransformer**) model presented by Liu et al. [256] was initially introduced for long-term time series forecasting. The authors argue that it is not the transformer itself that is not apt for forecasting multivariate time series, but the standard approach of data encoding. The differences between the standard transformer approach and the inverted version are visualised in Figure 13.3. Inverting the data and thereby switching the roles of time and variates forces the transformer model to apply its attention mechanism across the variates instead of the time dimension. This allows the model to learn multivariate connections while conserving the temporal dependencies. Further, adding a feed-forward layer after the attention layer encourages the model to learn a representation of the time series. It has been shown that this inverted embedding improves the results for long-term time series forecasting remarkably.

A naturally emerging research problem is to test this approach on anomaly detection. As a first step, we employ the iTransformer model for forecasting-based anomaly detection. This application requires minimal adaptations to the model, but substantial changes to the data loading process. Furthermore, we explore whether the inverted encoding also proves to be useful for reconstruction and reconstruction-based anomaly detection. This represents a novel use case for the model, necessitating a conceptual shift and further adaptation.

Finally, the (**vanilla**) **transformer** refers to the standard transformer architecture introduced by Vaswani et al. [98]. Just using the encoder of said model and plugging its outputs into a feed-forward NN can provide a simple reconstruction-based anomaly detection model. To not lose the temporal aspect of the data, positional encoding is used when embedding the data. Multiple transformers have already been studied for AD [105, 233, 234].

The vanilla Transformer model is introduced to study the effect of the inverted embedding used for the iTransformer, against the standard embedding

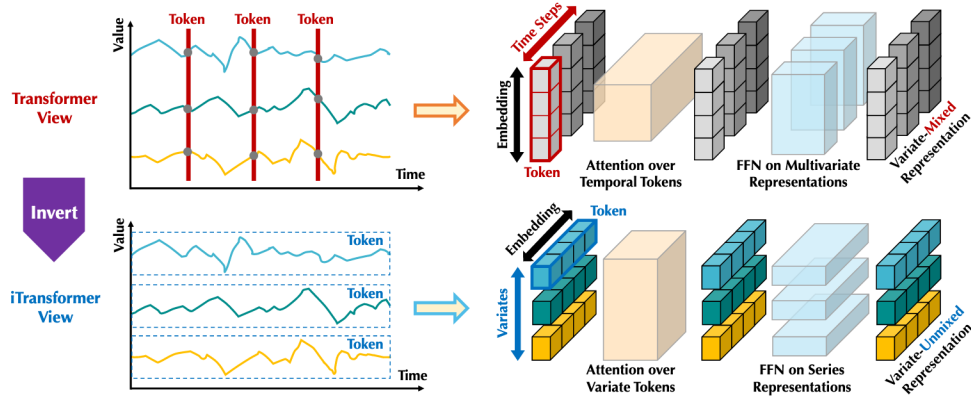


Fig. 13.3.: Comparison of the time series encoding for the (vanilla) transformer and the iTransformer. Figure from [256].

of the vanilla Transformer. It has been shown in the iTransformer paper [256] that for long-term time series forecasting, the inverted embedding shows a clear advantage in terms of prediction performance and flexibility over the standard approach. However, it is not evident whether this advantage holds in the case of anomaly detection, and we therefore compare the iTransformer to the vanilla Transformer to assess the impact of the inverted embedding, within the context of reconstruction-based TSAD.

13.3 Anomaly Label Extraction Strategies

The following paragraphs detail how to obtain binary anomaly labels starting from continuous anomaly scores. Initially, we focus on one-dimensional anomaly scores before extending our approach to the multidimensional case.

13.3.1 Peak Over Threshold

Peak-Over-Threshold (POT) is an approach based on **Extreme Value Theory (EVT)** [270] and commonly used to identify extreme points from data [271] without any strong assumptions on the data distribution. This is useful for determining whether an anomaly score derived with some anomaly detection algorithm should be labelled as an actual anomaly. Some statistical background for the POT method is provided in the following.

Theoretical Background

Let X be a random variable and $F(X) = P(X \leq x)$ its cumulative distribution. One can further introduce $\bar{F}(X) = 1 - F(X)$ as the ‘tail’ of the distribution and

$$F_t(X) = P(X - t \leq x | X > t) = \frac{F(t+x) - F(t)}{1 - F(t)} \quad (13.2)$$

as the excess distribution function over a threshold t . Finally, one also defines the extreme value distributions and which have the following form:

$$G_\gamma : x \mapsto \exp(-(1 + \gamma x)^{-1/\gamma}), \quad \gamma \in \mathbb{R}, \quad 1 + \gamma x > 0. \quad (13.3)$$

The so-called *Pickands–Balkema–De Haan theorem* [272, 273] then states that such excess distribution functions $F_t(X)$ follow a [Generalised Pareto Distribution \(GPD\)](#) distribution for a large class of cumulative distribution functions $F(X)$. The theorem is also referred to as the *second theorem* in [EVT](#).

Formally, the cumulative distribution function $F \in D_\gamma$ (denoting the extrema of the distribution of F converging in distribution to G_γ) if and only if a function σ exists, for all $x \in \mathbb{R}$ such that $1 + \gamma x > 0$:

$$\frac{\bar{F}(t + \sigma(t)x)}{F(t)} \xrightarrow[t \rightarrow \tau]{} (1 + \gamma x)^{-1/\gamma}. \quad (13.4)$$

where τ denotes the bound of the initial distribution (can be finite or infinite). This is equivalent to

$$\bar{F}_t(x) = P(X - t > x | X > t) \sim [t \rightarrow \tau] \left(1 + \frac{\gamma x}{\sigma(t)}\right)^{-1/\gamma}. \quad (13.5)$$

So instead of fitting an extreme value distribution to the extreme values of X , the [POT](#) approach focuses on fitting a [GPD](#) to $X - t$ [271].

In the next step, one has to fit the parameters γ, σ of the [GPD](#) to the excess distribution. This is often done with maximum likelihood estimation. Using the [GPD](#)’s fitted parameters $\hat{\gamma}, \hat{\sigma}$, one can estimate the quantile associated to the probability q with the following equation:

$$z_q \simeq t + \frac{\hat{\sigma}}{\hat{\gamma}} \left(\left(\frac{qn}{N_t} \right)^{-\hat{\gamma}} - 1 \right), \quad (13.6)$$

where t is the initial (high) threshold, n the number of data instances and N_t the number of peaks (i.e. data instances surpassing the threshold t).

Implementation

The POT method for streaming is introduced in [271]. It follows the following outline. Schematically, the method can be represented as shown in Figure 13.4.

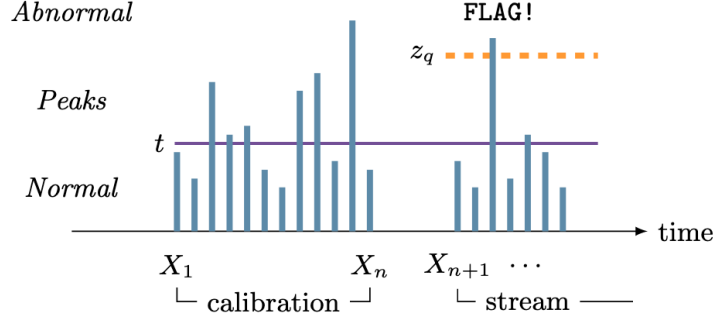


Fig. 13.4.: Schema of flagging anomalies with the POT method. Figure from [271].

Initialisation Step This step corresponds to the left part of Figure 13.4. Starting from the initially available data, one first needs to set an initial threshold t , from which one then derives a quantile z_q (i.e. $P(X > z_q) < q$) where q is the desired probability given as an input parameter. The goal is to choose t as high as possible to only have a few data instances that peak surpass the threshold. Identifying these data instances, declaring them as peaks and then fitting the GPD to them gives a first estimate of the GPD's parameters. Using Eq.(13.6), one can derive a quantile z_q which serves as an updated estimation of the threshold. This initialisation can be considered as a calibration of the method to the actual data. Note that for this step, the data does not need to be labelled or contain any additional information; the calibration simply analyses the statistical distribution of the initial data. The only remaining challenge is the choice of the initial threshold t . t needs to be sufficiently high such that peaks and the GPD fit are meaningful (low bias), but too few peaks can lead to a bad fit of the GPD (high variance). Finally, it is important that $t < z_q$ is satisfied. In practice, this can easily be done by setting t to an empirical quantile associated with a probability $p_t < (1 - q)$.

Anomaly Detection This step corresponds to the right part of Figure 13.4. If the underlying distribution of the data does not significantly change over time (stationary case), the POT method is now ready for identifying anomalies. For this, one simply goes through the new data instances and checks whether they pass the updated threshold z_q or not. If an instance $X_i > z_q$, it is declared an anomaly and the anomaly count is increased. Else, if $t < X_i \leq z_q$ it is considered

a peak and z_q is updated using this new set of peak instances. Finally, if X_i is neither an anomaly nor a peak, it is labelled as a normal case and no adjustments are made.

Variations of POT The previously described [POT](#) method can analogously be modified to lower-bound thresholds to identify low-valued anomalies. Differently, if working with non-stationary data (i.e. the distribution of X changes with time), further adjustments are needed. The main idea is to normalise each data instance X_i by subtracting a moving average M_i from it. This method is called the drifting SPOT [\[271\]](#).

13.3.2 Anomaly Labels

In practice, the [POT](#) method is often used in combination with anomaly detection models to derive anomaly labels from anomaly scores.

Anomaly Labels for Univariate Time Series

A possible, static labelling approach is studying the anomaly scores on a validation set with corresponding anomaly labels and determining the threshold that yields the best value of a chosen metric on this set. The same threshold is then used for the test set. The downside of this approach is the need for a labelled holdout set, in a setting where labelled data are already scarce [\[198\]](#).

If one has an approximate idea of the fraction f of anomalies present in the data, it can be used to declare the f percentile of the highest anomaly scores as anomalous to derive a threshold from there.

Finally, as described in Section [13.3.1](#), the [POT](#) [\[271\]](#) method allows to derive a threshold based on [EVT](#) [\[270\]](#) and is used to identify extreme points from data without any strong assumptions on the underlying data distribution.

Anomaly Labels for Multivariate Time Series

The challenge with multivariate time series is effectively managing the complexity introduced by the multidimensional nature of the data. Typically, models provide anomaly scores separately for every variate of the time series, i.e. a time series with N variates yields an N -dimensional anomaly score. However, for most applications, a single label per time stamp is more practical, regardless of which variate is anomalous. This raises two key questions: how should these scores be combined, and how does the choice of combination method

affect anomaly detection performance? We distinguish between *global* and *local* combination strategies and compare three such approaches.

Note that in the case of univariate time series, all methods produce identical anomaly labels.

- **global**: The anomaly scores are first averaged over all N variates to get a univariate series of anomaly scores to which the previously described anomaly label extraction strategies can be applied.
- **local (inclusive OR)**: The anomaly label extraction strategies are applied separately to each variate of the time series. Next, to end up with only one label per time stamp, we apply an *inclusive OR*-pooling function across the N variates, i.e. if a time stamp is anomalous in at least one of the N dimensions, the time stamp is anomalous in the final labels.
- **local (majority voting)**: Again, the anomaly label extraction strategies are applied separately to each variate of the time series and then we apply *majority voting* across the variates, i.e. if at least $N/2$ dimensions are anomalous at a certain time stamp, the time stamp is declared anomalous in the combined labels.

These methods are visually summarised in Figure 13.5.

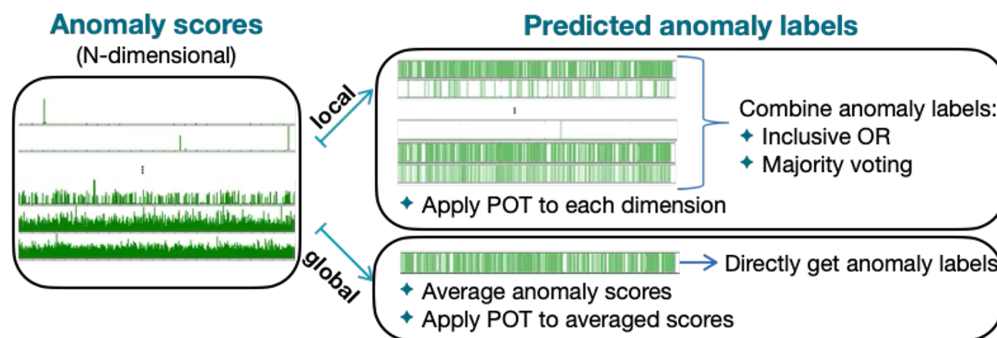


Fig. 13.5.: Illustration of different anomaly label extraction methods for multivariate time series. For univariate time series, all these methods coincide.

In prior work, the *global* method is the most prevalent approach for dealing with multidimensional anomaly labels [225, 239], though other options have been explored [242, 274]. It is also possible to explore alternative pooling functions for local anomaly labels, rather than limiting oneself to *inclusive OR* or *majority voting* as done in the context of this work. Finally, the anomaly score combination strategies presented here can be further enhanced by incorporating domain knowledge. For instance, a practitioner may assign higher weights to individual input variates if those variates are considered more reliable. Additionally, point-wise estimates of the reconstruction uncertainty can be leveraged to perform a weighted combination of anomaly scores.

13.4 Experimental Setup

13.4.1 Sliding Windows

Most models that are applied to time series operate on smaller, fixed-sized segments of said series. For this, the time series is sliced into equally sized windows of size W as shown in Figure 13.6. The overlap between consecutive windows is determined by the step size S . To keep the window size consistent, the last window is padded by repeating the last value.

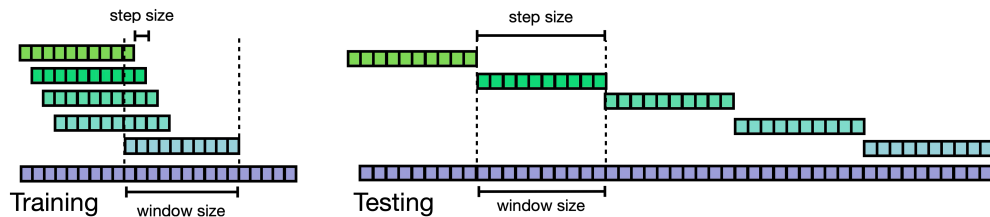


Fig. 13.6.: Sliding window protocol for training and testing.

During training, the windows are passed to the model as inputs. For forecasting-based models, it is beneficial to set $S = 1$, such that the model only has to predict the last time stamp per window. As S increases, the model has to predict multiple time stamps simultaneously, which typically decreases performance due to increased difficulty. The training loss is generally given by the [MSE](#) between the predicted and actual last time stamp of the window and then averaged over all windows. The same configuration is also used for testing, except here, the [MSE](#) is not averaged but retained as a multidimensional anomaly score.

In contrast, reconstruction-based models allow for more flexibility when adjusting the step size. During training, each window is passed through the model, which compresses and then reconstructs it. The loss is defined as the [MSE](#) between the input window and the reconstructed output. For each training epoch, the loss is averaged over all windows. For testing, the windows are chosen to be non-overlapping (i.e. $S = W$) and concatenating the [MSE](#) of each window gives the desired, multidimensional anomaly score.

13.4.2 Detection Performance Metrics

Once the predicted labels are obtained, the difficulty is how to evaluate and compare them against the true labels. Following the discussion from Section 3.4.2, we choose to focus mainly on the [MCC](#) score, as it weighs [TP](#), [TN](#), [FP](#) and [FN](#) data instances equivalently. We sometimes also include the F1 score who attributes more weight to [FPs](#).

13.4.3 Alternative Loss Functions

The **MSE** is the most commonly used loss function for anomaly detection. However, we observe that for some anomalies, **TSAD** models can learn the patterns sufficiently well to achieve low reconstruction or forecasting errors. This suggests that **MSE** may not be the optimal metric for calculating anomaly scores. Furthermore, in the case where anomalies are included in the training data, it might be beneficial to use a loss function that is less sensitive to outliers than the **MSE**. Therefore, we investigate the use of alternative loss functions such as the *Huber loss* and the *Soft Dynamic Time Warping (Soft-DTW)* loss [275].

The Huber loss [276] is a continuous combination of the **MSE** and the absolute value error, which makes it more robust against outliers deviating from the bulk part of the distribution by a large amount. It is given by

$$l_{\delta}(\hat{y}, y) = \begin{cases} \frac{1}{2}(y - \hat{y})^2, & \text{if } |y - \hat{y}| < \delta, \\ \delta \cdot (|y - \hat{y}| - \frac{1}{2}\delta), & \text{otherwise,} \end{cases} \quad (13.7)$$

where y denotes the input and $\hat{y}$ the predicted or reconstructed outputs.

Dynamic Time Warping (DTW) [277–279] is a similarity measure specifically designed for time series. It has been utilised in kernel or nearest neighbour methods to compare and align time series sequences effectively. However, due to its non-differentiability, it cannot be directly optimised within a deep learning model as a loss function. Cuturi and Blondel [275] introduced the **Soft-DTW**, a differentiable version of **DTW**, and used it as a loss function for time series forecasting.

Here, we test whether the **Soft-DTW** loss can be useful for **TSAD**. Since the **Soft-DTW** does not provide an element-wise loss value, it can only be used during training, and we again use the **MSE** as the anomaly score during testing.

13.4.4 Model Configurations

Whenever possible, we follow the model configuration as detailed in their respective papers. The baseline, introduced in Section 13.2.1, is not based on any learning model and does not need any specific configuration.

Since the **iTransformer** architecture has not yet been explored for anomaly detection, we take this opportunity to evaluate various configurations for both the reconstruction- and forecasting-based approaches. We focus on the relations between the window size W , the step size S and the internal model size M and aim to find the configuration that yields the best **MCC** scores, averaged

over five random seed repetitions, for each dataset. For the forecasting-based approach, we reduce the number of tested configurations to prioritise efficiency and robustness of the model. We also use this parameter search to compare the impact of the different anomaly label extraction strategies presented in Section 13.3.

For each parameter configuration, we compute the MCC scores using one of the three anomaly labelling methods and select the configuration with the highest score. If the highest MCC score across five repetitions, along with its standard deviation, overlaps with the scores from other configurations, the configuration with the smallest model size is preferred. If multiple optimal configurations remain, the one with the smallest step size is selected. This choice often aligns with the configuration that has a smaller standard deviation, indicating greater robustness. Further details of this study can be found in Appendix B.2.

13.5 Results

13.5.1 iTransformer Model Configurations

We perform a parameter search for the best configuration for the iTransformer model, once when the model is used for reconstruction-based anomaly detection (iTransformer-reco) and once for forecasting-based anomaly detection (iTransformer-fc). The optimal configuration choices for each dataset are presented in Appendix B.2.

For both approaches, reconstruction and forecasting, we observe that the best window size does not correlate with the average anomaly length. Naively, one would expect that anomalies covering entire time windows might be harder to detect, since the model might learn to reconstruct the anomalous window. However, this might be a simplistic assumption, especially given the collective nature of these longer anomalies.

For iTransformer-reco, a smaller internal model size is preferred, as the largest internal model size did not produce remarkable results on any of the datasets. This suggests that smaller model sizes successfully force compression of the input data, thus allowing the model to capture the average behaviour of the time series. Consequently, this prevents the model from learning to reconstruct unusual patterns, which might hint at anomalies.

Further, the step size is varied for the reconstruction strategy, where we observe a clear preference towards smaller step sizes. A small step size creates more input windows for training, thus allowing the model to train on additional data

while scanning the time series in more detail. Therefore, the choice of the step size directly impacts the robustness of the learning model, as also observed in our parameter search. However, we observed that comparable loss values and robustness can be achieved with larger step sizes, at the cost of longer training.

13.5.2 Studies on Anomaly Label Extraction

Next, we determine a preferred anomaly label extraction method for each dataset and parameter configuration. For `iTransformer-fc`, all configurations achieve the best results with the same anomaly label method for a given dataset. For `iTransformer-reco`, at least 80% of the configurations on most datasets agreed on the same anomaly label method, which was the one that gave the best anomaly detection results. The only exceptions to this were `SMAP` and `SWaT`. For `SMAP`, all three anomaly label methods gave very similar results for half of the configurations; on the other half, the *global* method dominated. For `SWaT`, the *local (majority voting)* and *global* methods achieved very similar `MCC` scores. Table 13.2 details which anomaly label extraction method worked best for each multivariate dataset. It is interesting to note that the anomaly label method barely depends on the configuration of the model, but only on the dataset.

Tab. 13.2.: Anomaly label extraction producing the best results for reconstruction- or forecasting-based `iTransformer` models on each dataset. The univariate datasets `SWaT_1D` and `UCR` are omitted because all anomaly label extraction methods yield the same results.

Dataset	Anomaly Label Extraction for <code>iTransformer-reco</code>	Anomaly Label Extraction for <code>iTransformer-fc</code>
Credit Card	local (incl. OR)	local (incl. OR)
GECCO	local (incl. OR)	local (incl. OR)
IEEECIS	local (incl. OR)	local (incl. OR)
MSL	local (maj. voting)	global
SMAP	local (maj. voting)	global
SMD	local (incl. OR)	local (incl. OR)
SWaT	local (maj. voting)	global
WADI	global	global

We observe a clear pattern: datasets with shorter anomalous sequences (average anomaly length < 100) work best with the *local (inclusive OR)*, whereas datasets with longer anomalous sequences prefer *local (majority voting)* or *global*. For larger anomalous sequences, the reconstruction-based `iTransformer` performs better with *local (majority voting)*, whereas the forecasting-based `iTransformer`

prefers the *global* approach. Surprisingly, SMD, which contains longer anomalous sequences, achieves the best MCC scores with *local (inclusive OR)*. Analysis of the SMD test data reveals 35 anomalous sequences, ten of which are long (≥ 100 consecutive anomalous timestamps), but many short, nearly point-like anomalies lower the average anomaly length. Furthermore, the point anomalies are only detected with the *local (inclusive OR)* method, indicating that this method is most sensitive to subtle, single-variate anomalies. This can be explained by the fact that point anomalies are very short and may only affect a single variate, whereas collective anomalies, by definition, involve several timestamps or dimensions.

It is also possible to check the accuracy of anomaly predictions for each of the anomaly label extraction methods as a function of the length of the anomalous sequence. This is shown in Figure 13.7, where the accuracies were averaged over all repetitions and multivariate datasets. As previously discussed, for short anomaly lengths, only the *local (inclusive OR)* method manages to accurately identify anomalies. However, Figure 13.7 solely focuses on the anomalous sequences and does not present any information about FPs.

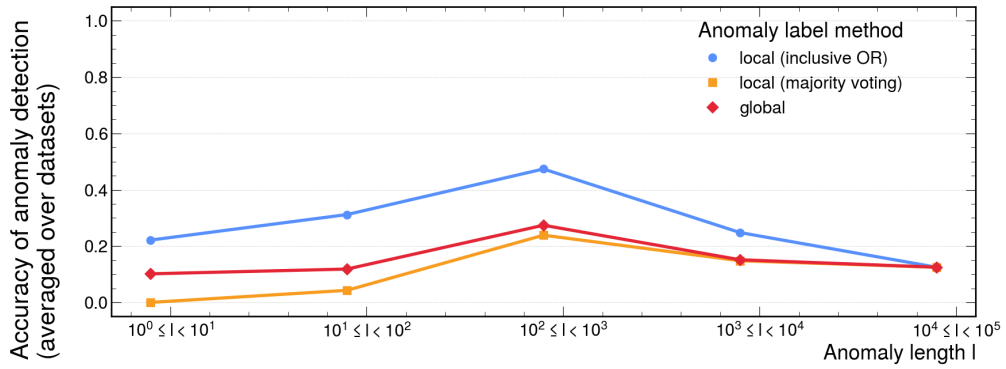


Fig. 13.7.: The accuracy of anomaly predictions for each anomaly label extraction methods, averaged over all multivariate datasets and repetitions, is shown for different categories of anomaly lengths l .

Figure 13.8 displays the false positive rate associated to the anomaly predictions for the same three anomaly label extraction methods averaged over the same datasets. The x axis corresponds to the length of non-anomalous sequences, and the y axis indicates the part of timestamps that have been falsely flagged as anomalous. These results indicate that while the *local (inclusive OR)* method achieves high anomaly detection accuracy, it does so at the expense of a higher false positive rate. Therefore, depending on the exact application, each anomaly label method offers distinct advantages and trade-offs.

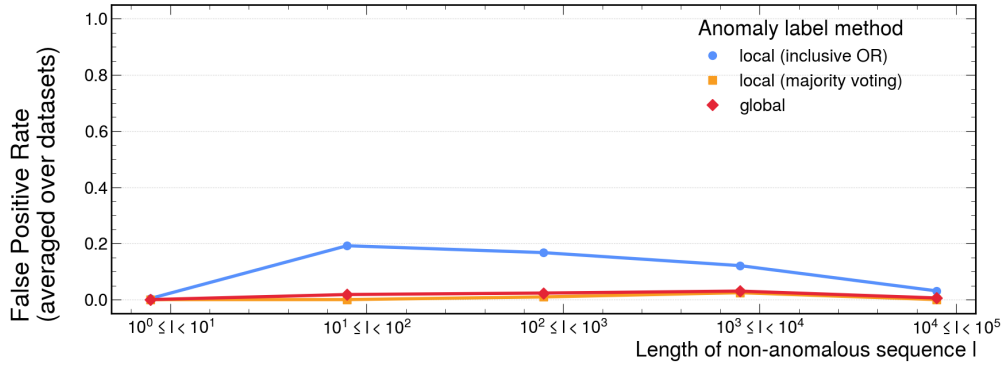


Fig. 13.8.: The false positive rate associated with anomaly predictions for each anomaly label extraction method, averaged over all multivariate datasets and repetitions, is shown for different lengths of non-anomalous sequences l .

13.5.3 Influences of Anomalies in Training Data

Given that the underlying assumption for both forecasting- and reconstruction-based TSAD relies on the rarity of anomalies in training data, it is crucial to investigate how the presence of anomalies during training actually impacts the anomaly detection performance. This is important as it is not possible to always guarantee the absence of anomalies in real-world applications. Concretely, we study the effect of anomalies in training data on the example of the reconstruction-based iTransformer because it is the most flexible and efficient model of our selection. First, we use the Credit Card and GECCO datasets, where anomaly labels are also available for the training data, thus allowing an explicit comparison of the anomaly detection performance when training with and without anomalies. Second, further change the loss function during training, to check whether the loss functions presented in Section 13.4.3 might mitigate this issue, as they may be less sensitive to outliers than the MSE. However, for evaluation, we always employ the MSE as an anomaly score, as a high sensitivity to outliers is preferred in this case.

Table 13.3 presents the anomaly rates in the training data and the MCC and F1 scores achieved by iTransformer-reco.

For both datasets, anomalies in training data degrade the model’s anomaly detection performance. This is expected, as the model may learn to reconstruct anomalies, thereby invalidating the assumption of anomalies having a higher reconstruction error. However, it is remarkable that such a low anomaly rate ($\leq 2\%$) already affects training and deteriorates the anomaly detection. Moreover, for both datasets, employing a loss function other than the MSE seems beneficial if the training set includes anomalies. Therefore, in cases where anomalous instances in training data cannot be avoided, alternative loss functions should be explored.

Tab. 13.3.: MCC and F1 scores (averaged over five random seeds) for different loss functions during training, with and without anomalies in training data.

Dataset	Anomaly Rate in Train Data	Loss Function	Best MCC	Best F1
Credit Card	0.00%	MSE	0.241 $\pm$ 0.019	0.157 $\pm$ 0.014
		Huber	0.220 $\pm$ 0.012	0.140 $\pm$ 0.008
		Soft-DTW	0.230 $\pm$ 0.021	0.148 $\pm$ 0.018
	0.19%	MSE	0.226 $\pm$ 0.020	0.145 $\pm$ 0.016
		Huber	0.234 $\pm$ 0.015	0.150 $\pm$ 0.011
		Soft-DTW	0.209 $\pm$ 0.012	0.132 $\pm$ 0.007
GECCO	0.00%	MSE	0.760 $\pm$ 0.004	0.761 $\pm$ 0.005
		Huber	0.767 $\pm$ 0.010	0.768 $\pm$ 0.010
		Soft-DTW	0.774 $\pm$ 0.018	0.774 $\pm$ 0.017
	1.43%	MSE	0.631 $\pm$ 0.039	0.624 $\pm$ 0.043
		Huber	0.597 $\pm$ 0.063	0.589 $\pm$ 0.068
		Soft-DTW	0.696 $\pm$ 0.024	0.695 $\pm$ 0.025

However, the lack of labelled training data in public datasets does not allow for a comprehensive comparison of alternative loss functions and their potential advantages when dealing with anomalies during training. This further emphasises the need for more and better labelled benchmark datasets for [TSAD](#).

For completeness, the use of alternative loss functions during training has also been tested on the remaining datasets, where we observed a consistent anomaly detection performance across all loss functions (see Appendix [B.3](#)).

13.5.4 Benchmarking Results for Transformer-Based TSAD

Next, we report the results of multiple transformer-based anomaly detection models for time series. We investigated early stopping strategies based on the model’s performance on a validation set, but noticed that the anomaly detection performance often deteriorated with longer training times. Consequently, and also for computational efficiency, we chose to train all models for only ten epochs, repeating the procedure for five random seeds to obtain an estimate of a model’s robustness. We compare the baseline from Section [13.2.1](#) and the USAD model from Section [13.2.2](#) against the two *iTransformer* implementations from Section [13.5.1](#), a reconstruction-based transformer, the *AnomalyTransformer* and the *TranAD* model.

The hyperparameter configurations and number of parameters of each model on each dataset are detailed in Appendix [B.4](#). Most of the models adjust the

number of trainable parameters according to the number of variates in the dataset. In general, it appears that a larger model (i.e. a higher number of parameters) does not guarantee a better anomaly detection performance: The `AnomalyTransformer` constantly has some of the highest parameter numbers but does not achieve noticeably higher scores.

Concerning the datasets, the same data are used for all models except for `SWaT`, where it was necessary to reduce the dataset size to a tenth of the original size, because all models except for `iTransformer-reco` struggled to process the full-sized one. This is expected as the inverted embedding of the `iTransformer` was conceived to mitigate such issues, and reconstruction-based models often allow for more flexibility in their implementation. For `TranAD`, it was further necessary to restrict the variates of the input to the first 30 variates for all datasets due to computational limitations.

The `MCC` and F1 scores averaged over five random seed repetitions on each dataset are presented in Table 13.4. Both scores differ little, and they are generally achieved with the same anomaly label extraction method, which is added in brackets. The only exception to this is the `USAD` model on the first three datasets, where the F1 score combined with the *local (incl. OR)* method is higher than with *local (maj. voting)*, which was the preferred method associated with the `MCC` score.

The baseline algorithm provides interesting insights into which datasets are nearly ‘trivial’ for anomaly detection, such as `SMD`, `SWaT_1D` and, to a lesser extent, `MSL`. However, for the remaining datasets, the baseline does not offer any additional information.

The `iTransformer` models dominate on all ten datasets based on the `MCC` or F1 score, but on four `IEEECIS`, `SMAP`, `SWAT_1D`, and `WADI`, the `vanilla Transformer` or the `USAD` model provide similar scores. The `iTransformer-reco` model achieves the best scores across the most datasets, but also demonstrates strong flexibility and efficiency. Since its self-attention is applied across variates instead of individual time stamps, the model scales linearly with the window size (and quadratically with the number of variates N), allowing greater flexibility in the choice of window sizes. This suggests that inverting the data embedding can lead to performance gains, highlighting the effectiveness of this simple yet impactful modification. Interestingly, the `iTransformer-fc` is the best model on the `Credit Card` data, implying that predictive models might offer some advantages in identifying point anomalies.

Concerning the univariate time series datasets, `SWaT_1D` and `UCR`, the reconstruction-based `iTransformer` and `vanilla Transformer` achieve nearly identical `MCC` scores. This is expected as the multivariate attention mechanism of the `iTransformer`

cannot be exploited on one-dimensional data, as there are no feature correlations to be learned. Furthermore, on SWaT_1D, even the baseline algorithm achieves high [MCC](#) and F1 scores, suggesting that complex transformer models offer little to no advantage here.

The `AnomalyTransformer` applies its own custom compression of anomaly scores, directly outputting a one-dimensional score based on association discrepancy, which is then converted into binary labels via the [POT](#) method. This appears to be competitive with the other anomaly label extraction methods detailed in this paper.

Next, the TranAD and the USAD model struggle on multiple datasets. On Credit Card and IEEEECIS, both models fail at reconstructing the input and return a constant value. This also happens when applying USAD to GECCO, whereas TranAD fails to reconstruct SWaT. Upon closer inspection of the GECCO and IEEEECIS datasets, it seems that there are some trend changes in both of them, which hinder both models from correctly reconstructing the time series. This leads to artificially high reconstruction errors in those regions of the time series, where such trend changes are very noticeable and therefore the [POT](#) method converts these high reconstruction errors into (falsely) positive anomaly labels. This also explains why the F1 score might be slightly higher on these datasets for USAD, as the F1 score does not attribute much weight to false negatives, but the [MCC](#) score does.

Tab. 13.4.: Best MCC and F1 scores (averaged over five random seeds) for forecasting-based, reconstruction-based iTransformer, vanilla Transformer, AnomalyTransformer, TranAD and USAD model on various datasets. The baseline is deterministic, so only a single score is reported. The letters in the brackets denote the anomaly label extraction method. For the univariate time series SWaT_1D and UCR, all anomaly label extraction methods yield the same results.

Data	Model	Best MCC	Best F1	Data	Model	Best MCC	Best F1
Credit Card	Baseline	0.000 (LMV)	0.002 (LIO)	SMD	Baseline	0.924 (LIO)	0.927 (LIO)
	USAD	0.000 $\pm$ 0.000 (LMV)	0.001 $\pm$ 0.001 (LIO)		USAD	0.326 $\pm$ 0.056 (LIO)	0.331 $\pm$ 0.062 (LIO)
	AnomalyTransformer	0.011 $\pm$ 0.018 (C)	0.009 $\pm$ 0.014 (C)		AnomalyTransformer	0.053 $\pm$ 0.107 (C)	0.028 $\pm$ 0.057 (C)
	TranAD	0.000 $\pm$ 0.000 (LIO)	0.000 $\pm$ 0.000 (LIO)		TranAD	0.726 $\pm$ 0.039 (LIO)	0.731 $\pm$ 0.031 (LIO)
	Transformer	0.254 $\pm$ 0.007 (LIO)	0.165 $\pm$ 0.006 (LIO)		Transformer	0.953 $\pm$ 0.005 (LIO)	0.956 $\pm$ 0.005 (LIO)
	iTransformer-fc	0.381 $\pm$ 0.001 (LIO)	0.303 $\pm$ 0.001 (LIO)		iTransformer-fc	0.913 $\pm$ 0.000 (LIO)	0.916 $\pm$ 0.000 (LIO)
	iTransformer-reco	0.241 $\pm$ 0.019 (LIO)	0.157 $\pm$ 0.014 (LIO)		iTransformer-reco	0.967 $\pm$ 0.010 (LIO)	0.968 $\pm$ 0.009 (LIO)
GECCO	Baseline	0.027 (G)	0.037 (G)	SWaT	Baseline	0.000 (ALL)	0.217 (ALL)
	USAD	0.000 $\pm$ 0.000 (LMV)	0.019 $\pm$ 0.005 (LIO)		USAD	0.838 $\pm$ 0.023 (G)	0.846 $\pm$ 0.024 (G)
	AnomalyTransformer	0.187 $\pm$ 0.066 (C)	0.166 $\pm$ 0.050 (C)		AnomalyTransformer	0.826 $\pm$ 0.024 (C)	0.840 $\pm$ 0.026 (C)
	TranAD	0.007 $\pm$ 0.013 (G)	0.024 $\pm$ 0.009 (G)		TranAD	0.000 $\pm$ 0.000 (ALL)	0.165 $\pm$ 0.000 (ALL)
	Transformer	0.752 $\pm$ 0.010 (LIO)	0.753 $\pm$ 0.010 (LIO)		Transformer	0.839 $\pm$ 0.011 (G)	0.858 $\pm$ 0.010 (G)
	iTransformer-fc	0.806 $\pm$ 0.029 (LIO)	0.803 $\pm$ 0.026 (LIO)		iTransformer-fc	0.824 $\pm$ 0.052 (G)	0.840 $\pm$ 0.051 (G)
	iTransformer-reco	0.760 $\pm$ 0.004 (LIO)	0.761 $\pm$ 0.005 (LIO)		iTransformer-reco	0.964 $\pm$ 0.004 (LMV)	0.967 $\pm$ 0.004 (LMV)
IEEECIS	Baseline	0.000 (LMV)	0.209 (LMV)	SWaT_1D	Baseline	0.804 (ALL)	0.807 (ALL)
	USAD	0.000 $\pm$ 0.000 (LMV)	0.116 $\pm$ 0.007 (LIO)		USAD	0.807 $\pm$ 0.002 (ALL)	0.812 $\pm$ 0.003 (ALL)
	AnomalyTransformer	0.456 $\pm$ 0.109 (C)	0.490 $\pm$ 0.097 (C)		AnomalyTransformer	0.724 $\pm$ 0.059 (C)	0.733 $\pm$ 0.063 (C)
	TranAD	0.000 $\pm$ 0.000 (LMV)	0.083 $\pm$ 0.003 (LIO)		TranAD	0.805 $\pm$ 0.006 (ALL)	0.811 $\pm$ 0.004 (ALL)
	Transformer	0.612 $\pm$ 0.040 (LIO)	0.624 $\pm$ 0.042 (LIO)		Transformer	0.806 $\pm$ 0.002 (ALL)	0.814 $\pm$ 0.002 (ALL)
	iTransformer-fc	0.456 $\pm$ 0.008 (LIO)	0.455 $\pm$ 0.009 (LIO)		iTransformer-fc	0.737 $\pm$ 0.050 (ALL)	0.744 $\pm$ 0.058 (ALL)
	iTransformer-reco	0.629 $\pm$ 0.032 (LIO)	0.649 $\pm$ 0.032 (LIO)		iTransformer-reco	0.805 $\pm$ 0.006 (ALL)	0.815 $\pm$ 0.004 (ALL)
MSL	Baseline	0.430 (LIO)	0.386 (LIO)	UCR	Baseline	0.000 (ALL)	0.000 (ALL)
	USAD	0.626 $\pm$ 0.034 (G)	0.638 $\pm$ 0.033 (G)		USAD	0.105 $\pm$ 0.234 (ALL)	0.106 $\pm$ 0.237 (ALL)
	AnomalyTransformer	0.754 $\pm$ 0.031 (C)	0.779 $\pm$ 0.029 (C)		AnomalyTransformer	0.163 $\pm$ 0.236 (C)	0.170 $\pm$ 0.233 (C)
	TranAD	0.154 $\pm$ 0.016 (LIO)	0.162 $\pm$ 0.008 (LIO)		TranAD	0.540 $\pm$ 0.306 (ALL)	0.509 $\pm$ 0.285 (ALL)
	Transformer	0.886 $\pm$ 0.016 (LMV)	0.891 $\pm$ 0.016 (LMV)		Transformer	0.757 $\pm$ 0.019 (ALL)	0.732 $\pm$ 0.023 (ALL)
	iTransformer-fc	0.866 $\pm$ 0.002 (G)	0.879 $\pm$ 0.001 (G)		iTransformer-fc	0.305 $\pm$ 0.274 (ALL)	0.310 $\pm$ 0.270 (ALL)
	iTransformer-reco	0.935 $\pm$ 0.011 (LMV)	0.940 $\pm$ 0.011 (LMV)		iTransformer-reco	0.775 $\pm$ 0.010 (ALL)	0.755 $\pm$ 0.011 (ALL)
SMAP	Baseline	0.000 (LMV)	0.067 (LIO)	WADI	Baseline	0.153 (LIO)	0.156 (LIO)
	USAD	0.836 $\pm$ 0.014 (G)	0.855 $\pm$ 0.013 (G)		USAD	0.411 $\pm$ 0.080 (G)	0.409 $\pm$ 0.077 (G)
	AnomalyTransformer	0.779 $\pm$ 0.084 (C)	0.787 $\pm$ 0.093 (C)		AnomalyTransformer	0.658 $\pm$ 0.106 (C)	0.666 $\pm$ 0.113 (C)
	TranAD	0.481 $\pm$ 0.160 (LIO)	0.475 $\pm$ 0.192 (LIO)		TranAD	0.530 $\pm$ 0.176 (G)	0.535 $\pm$ 0.188 (G)
	Transformer	0.557 $\pm$ 0.023 (LIO)	0.549 $\pm$ 0.025 (LIO)		Transformer	0.804 $\pm$ 0.037 (G)	0.815 $\pm$ 0.034 (G)
	iTransformer-fc	0.671 $\pm$ 0.002 (G)	0.690 $\pm$ 0.001 (G)		iTransformer-fc	0.676 $\pm$ 0.056 (G)	0.674 $\pm$ 0.066 (G)
	iTransformer-reco	0.826 $\pm$ 0.091 (LMV)	0.835 $\pm$ 0.095 (LMV)		iTransformer-reco	0.825 $\pm$ 0.024 (G)	0.835 $\pm$ 0.023 (G)

ALL: all equal, C: custom, LIO: local (incl. OR), LMV: local (maj. voting), G: global.

13.6 Summary and Outlook

We investigated each stage of the end-to-end [TSAD](#) algorithm, including the generation of anomaly scores from anomaly detection models, the process of combining and converting these scores into anomaly labels, and the evaluation of the resulting labels using appropriate metrics.

We explored the [iTransformer](#) [256] as a robust model for anomaly detection based on either reconstruction or forecasting, and analysed the impact of key parameters. Especially, the reconstruction-based [iTransformer](#) proved to be flexible and efficient on datasets with various dimensions and anomaly types. Additionally, we investigated methods for deriving anomaly labels from multidimensional anomaly scores generated by detection models and derived recommendations on how to find the best method given a dataset.

We further studied the impact of anomalous data in training on the anomaly detection performance and observed a clear deterioration even with very low anomaly rates. It was possible to slightly mitigate this effect by employing loss functions different from the [MSE](#). However, due to the lack of labelled training data, it was only possible to conduct these studies on two datasets. For a more exhaustive evaluation of the effects of anomalies during training, more such datasets are needed, especially with higher diversity of anomalies, or ways to control the rate of anomalies present in training data.

Finally, we compared various transformer-based models for [TSAD](#) across a diverse set of datasets, ensuring a broad representation of applications. We concluded that the [iTransformer](#) model demonstrates a strong suitability for [TSAD](#), effectively distinguishing anomalous patterns from normal temporal behaviour. This highlights the performance impact of the inverted embedding and raises questions about the necessity of applying more complex architectural modifications to transformers.

Synthetic Data Generation with `Lorenzetti` for TSAD in HEP Calorimeters

14.1 Introduction

Many datasets used for [TSAD](#) suffer from unreliable or scarce labels, as discussed in the previous chapter. To address the lack and unreliability of labelled data, we produce synthetic data with the `Lorenzetti Shower Simulator` [212], a general-purpose framework simulating a high-energy experiment, where we introduce artificial anomalies in the calorimeter.

The timely detection of detector issues is an important challenge for [DQM](#) at [HEP](#) experiments and requires substantial human input, as mentioned in Section 2.4. The application of [TSAD](#) techniques to this problem might relieve some of this workload.

For this reason, we implement artificial defects in the `Lorenzetti` simulation framework, which allow us to systematically evaluate the effectiveness and sensitivity of various anomaly detection methods, including transformer-based and other deep learning models discussed in Chapter 13. The approach described in the following is generic and can be adapted to various detector architectures and their potential defects.

14.2 Lorenzetti Showers Simulation

`Lorenzetti Showers` is an integrated software framework for simulating general-purpose calorimeters in [HEP](#) [212]. It has been developed by a collaboration among research groups from multiple Brazilian universities that maintains and regularly updates the software. Although the simulated calorimeter is of a general purpose, its cell granularity is based on the ATLAS technical design. However, this can be adapted. For instance, there are ongoing studies for modelling calorimeters for future particle detectors with `Lorenzetti`. The simulation output provides complete calorimeter information, including pileup and crosstalk effects, but no tracking simulation.

The simulation is based on `PYTHIA8` [28, 29] and `Geant4` [34], which are a popular [MC](#) event generator and detector simulator, respectively. The simulation chain is schematically represented in Figure 14.2 and consists of the following steps:

1. **Event Generation (EVT):** This step includes the physics event generation with the `PYTHIA8` particle generator. For our studies, we simulate proton-proton collisions producing [QCD](#) jets.

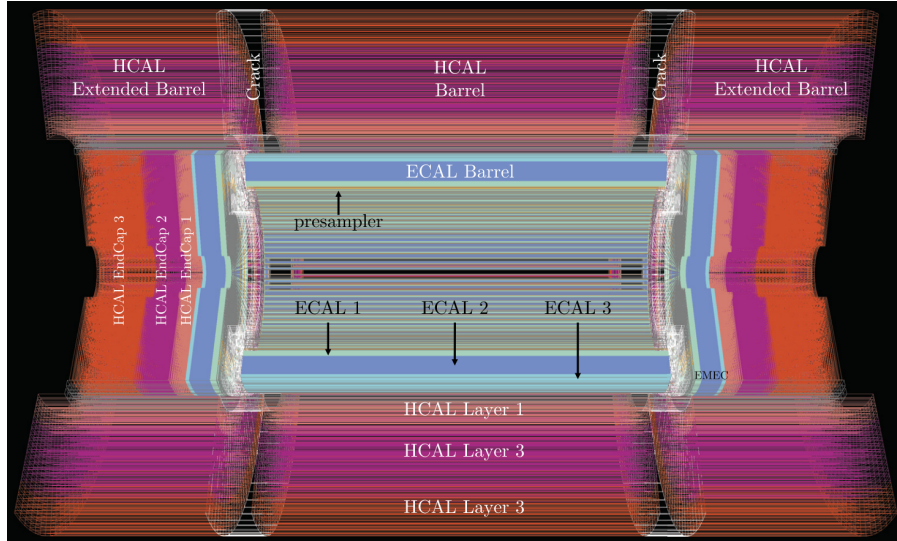


Fig. 14.1.: Cross-sectional view of the general-purpose calorimeter modelled by the Lorenzetti Showers framework. Figure from [212].

2. **Detector Simulation (HIT):** Next, the interaction of the modelled physics events with the calorimeter is simulated using Geant4.
3. **Pileup Merge (MB):** Steps 1-2 can be repeated once for signal particles (QCD jets) and once for background (pileup) events. This step is optional; the simulation can also be realised without pileup.
4. **Digitisation (ESD):** In this step, the electronic signals induced by the energy deposits in the calorimeter cells are modelled and processed. After this, the data are in the ESD format introduced in Section 2.2.4.
5. **Reconstruction (AOD):** Finally, energy clusters and various high-level variables (such as shower shape variables) are constructed from the previously simulated cell signals. After this, the data are in the AOD format detailed in Section 2.2.4.
6. **Analysis (NTUPLE):** It is further possible to only select a subset of reconstructed variables and transfer them into NTuples.

The complete simulation framework has a couple of limitations that are due to efficiency reasons or previous implementation choices. First, in the HIT step, the default setup discards all cells not associated with the seed of a simulated particle and thereby drastically reduces the amount of saved information. This implies that these cells are not accessible in later simulation stages and their values cannot be modified. Second, initially, the simulation primarily focused on the simulation of truth-level events, and there are still remnants of this in the current software implementation. For instance, the AOD step partially involves

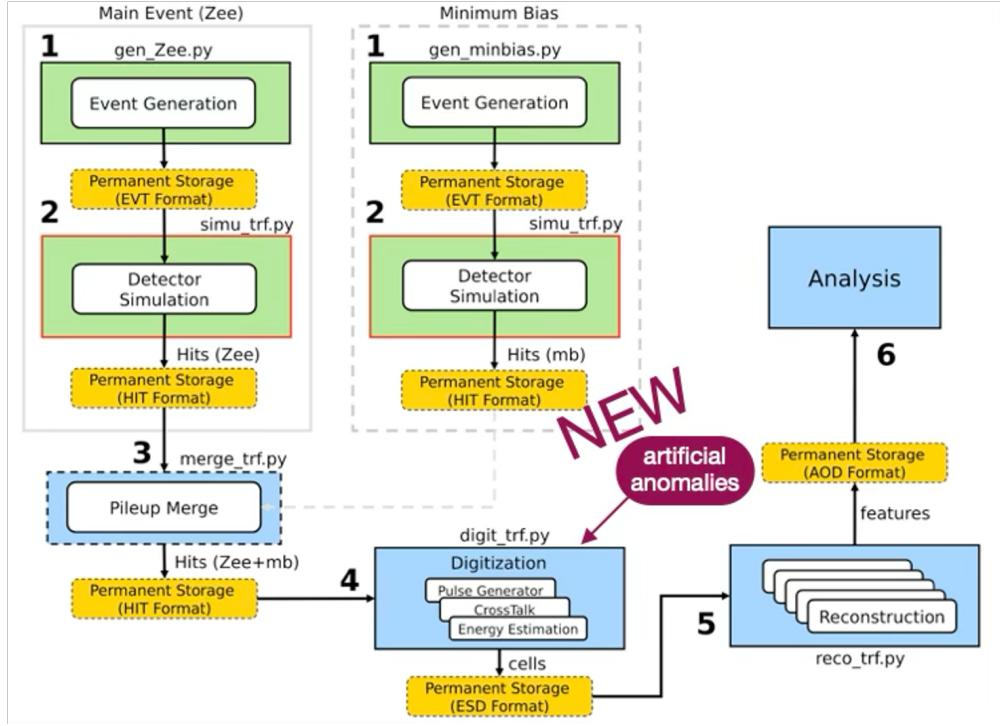


Fig. 14.2.: Schematic representation of the Lorenzetti Showers framework, including the new idea of artificially inserting anomalies related to detector defects during the digitisation (ESD) step. Figure adapted from [212].

truth-level information, which is unusual since the reconstruction is typically based on detector-level information.

To summarise, the limitations described above introduce differences between standard **MC** simulations within ATLAS and **MC** simulations produced with Lorenzetti that must be taken into account. In addition, the exact reconstruction techniques used in Lorenzetti may differ from the standard ATLAS object reconstruction detailed in Section 2.2.5. Jets, for instance, are only built up to the cluster level with an algorithm that is simpler than the topocluster method.

14.3 Dataset with Synthetic Detector Defects

The goal of this project is to introduce artificial, time-dependent detector defects into the simulation and to re-identify them using some of the **TSAD** approaches discussed in Chapter 13, to quantify their anomaly detection performance.

The synthetic anomalies studied in this project are restricted to detector defects and are introduced during the ESD step (see Item 4). Concretely, two types of synthetic detector issues are implemented, namely anomalies due to inactive

calorimeter modules and anomalies related to increased noise in the cells recording a physics signal. We carefully select varying detector regions and identify cells within those regions that will be affected by the artificial anomalies.

Dead Modules The first case focuses on ‘dead’ (i.e. inactive) detector modules. Here, the electronic signals corresponding to the deposited energy in the cells is not transmitted for entire modules of the calorimeter. This is a rather extreme case of a detector anomaly and is quite reliably identified during **DQM** tasks. However, it constitutes a good use case for validating the usefulness of **TSAD** algorithms on an easy and well-understood example. This is straightforward to implement; the electronic pulse is set to 0 for all cells that belong to the concerned detector regions.

In Figure 14.3, a visual example of such a defect, affecting 10 sequential collision events, is given. It is visible that three energy signals in the right-hand plot disappeared because the signal was not passed on by the electronic processing steps. In the example presented here, only some detector regions on the right-hand side of the detector (i.e. $\eta > 0$) are concerned.

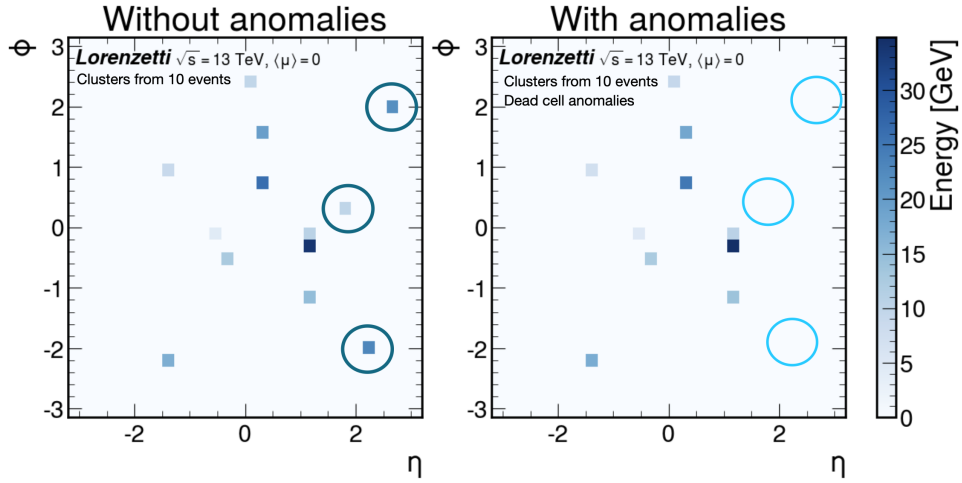


Fig. 14.3.: Phase (η, ϕ) space shows energy values for simulated detector cells after the digitisation (ESD) step and aggregated over 10 events. The left plot contains the default simulation without defects, whereas the right plot shows anomalies related to inactive detector modules (missing deposits in cyan circles).

Increased Noise In the second case, the electronic signal from the calorimeter cells is distorted more strongly than this is usually done by the default noise. Initially, the noise is drawn from a Gaussian distribution with a default mean and standard deviation. To disturb this, the standard deviation for specific calorimeter cells is multiplied by an adjustable noise factor, giving rise to modified detector signals.

Due to the previously mentioned limitations of `Lorenzetti`, this approach is limited to cells already containing a physics signal. It is currently not possible to introduce noise in detector parts that are uncorrelated with a physics signal. Future work is aiming to mitigate these limitations.

14.4 Data Preparation and Model Training

Initially, 100k jet events are simulated with `Lorenzetti`. The first 64k are set aside and used for training, whereas anomalies are injected into the last 35k events.

The events are ordered according to their event number. Nine features are kept, consisting of the cluster four-vector and five shower shapes (high-level variables that can be built at reconstruction level). The shower shapes and their definitions are given in Table 14.1.

Concretely, we include the cluster energy e , the transverse cluster energy et , the cluster pseudorapidity η and the cluster azimuthal angle ϕ and the four shower shapes r_η, r_ϕ, r_{had} and E_{ratio} .

Dead Modules Figure 14.4 visualises the feature distributions associated with inactive detector modules. Several features distributions are noticeably affected by the simulated detector issues. Since the dead modules are chosen to be on the left and right side of the detector and not in the barrel region, the cluster η is visibly distorted. This also translates to modified shower shapes r_η and r_ϕ .

Increased Noise Figure 14.5 shows the distributions of the features on the same test data for various anomaly rates without pileup (i.e. $\langle\mu\rangle = 0$). The only differences between the four test sets are the injected anomalies and their frequency. The anomalies are all associated with a noise factor of 20, meaning that the standard deviation of the default noise has been increased by this factor. The noise was increased in selected groups of cells for hundreds of sequential events though very often this disturbance only affected few events due to the aforementioned limitations of the simulation.

Similarly, Figure 14.6 displays the distributions of the features on the same test data for various anomaly rates with pileup ($\langle\mu\rangle = 40$). All other simulation parameters are identical for the two figures except for the presence or absence of pileup, respectively. The distribution plots are constrained to the 99-percentile of all four distributions.

The presence of pileup seems to slightly shift the cluster energies towards

lower values, and r_η and r_ϕ appear shifted to the left. The energy shift is probably due to corrections from the optimal filtering method, which aims to minimise electronic noise and is used as the default energy estimation method in Lorenzetti [212, 280].

Interestingly, the features where the normal and anomalous distributions differ the most are the two shower shape variables r_η, r_ϕ , independently of the pileup conditions. For Figure 14.5, the cluster energy also shows some slight differences.

Tab. 14.1.: List of shower shape variables, variables denoted with * are used as input for the TSAD methods.

Variable	Description
e_{233}	Uncalibrated energy (sum of cells) of the middle sampling in a rectangle of size 3×3 (in cell units $\eta \times \phi$).
e_{237}	Uncalibrated energy (sum of cells) of the middle sampling in a rectangle of size 3×7 (in cell units $\eta \times \phi$).
e_{277}	Uncalibrated energy (sum of cells) of the middle sampling in a rectangle of size 7×7 (in cell units $\eta \times \phi$).
$E_{T,had}$	E_T leakage into the hadronic calorimeter except for energy in the TileGap3.
E_{maxs1}	Energy of strip with maximal energy deposit.
E_{2tsts1}	Energy of strip with second maximal energy deposit, calculated by summing three strips.
r_η^*	Energy ratio using the second layer of the electromagnetic calorimeter in 3×7 cells over the energy in 7×7 cells centred at the cluster position: $r_\eta = e_{237}/e_{277}$.
r_ϕ^*	Energy ratio using the second layer of the electromagnetic calorimeter in 3×3 cells over the energy in 3×7 cells centred at the cluster position: $r_\phi = e_{233}/e_{237}$.
r_{had}^*	Ratio of E_T in the hadronic calorimeter to E_T of the electromagnetic cluster (for $0.8 < \eta < 1.37$): $r_{had} = E_{T,had}/E_T$.
E_{ratio}^*	Ratio of the energy difference between the maximum energy deposit E_{maxs1} and the energy deposit in a secondary maximum in the cluster E_{2tsts1} to the sum of these energies: $E_{ratio} = (E_{maxs1} - E_{2tsts1})/(E_{maxs1} + E_{2tsts1})$.
$w_{\eta 2}^*$	The lateral width is calculated with a window of 3×5 cells using the energy-weighted sum over all cells, which depends on the particle impact point inside the cell: $w_{\eta 2} = \sqrt{\sum_i E_i \eta^2 / (\sum_i E_i) - ((\sum_i E_i \eta) / (\sum_i E_i))^2}$, where E_i is the energy of the i -th cell.

To handle the varying number of clusters per event, the entries are aggregated for each event to reduce them to a fixed size. For this, the mean and the standard deviation of each feature are taken, increasing the number of input dimensions for the **TSAD** models to 18. Then, all train features are normalised such that their values are contained within the 0 and 1, and the same scaling is applied to the test set.

We select the unsupervised baseline, the reconstruction-based **iTransformer**, the **TranAD** and the **USAD** model – all of these models are presented in Chapter 13 – and apply them to the synthetic datasets produced with **Lorenzetti**. They are combined with the unsupervised strategies – also described in Chapter 13 – to determine the predicted binary anomaly labels and then evaluate the anomaly detection performance using the **MCC** score. The training duration is ten epochs and it is repeated five times, each time a different slice of the training set is used for validation, and the **MSE** constitutes the loss function. The sliding window is 10, for all models, and the step size 1 for the **iTransformer** and **TranAD** models, and 5 for the **USAD** model.

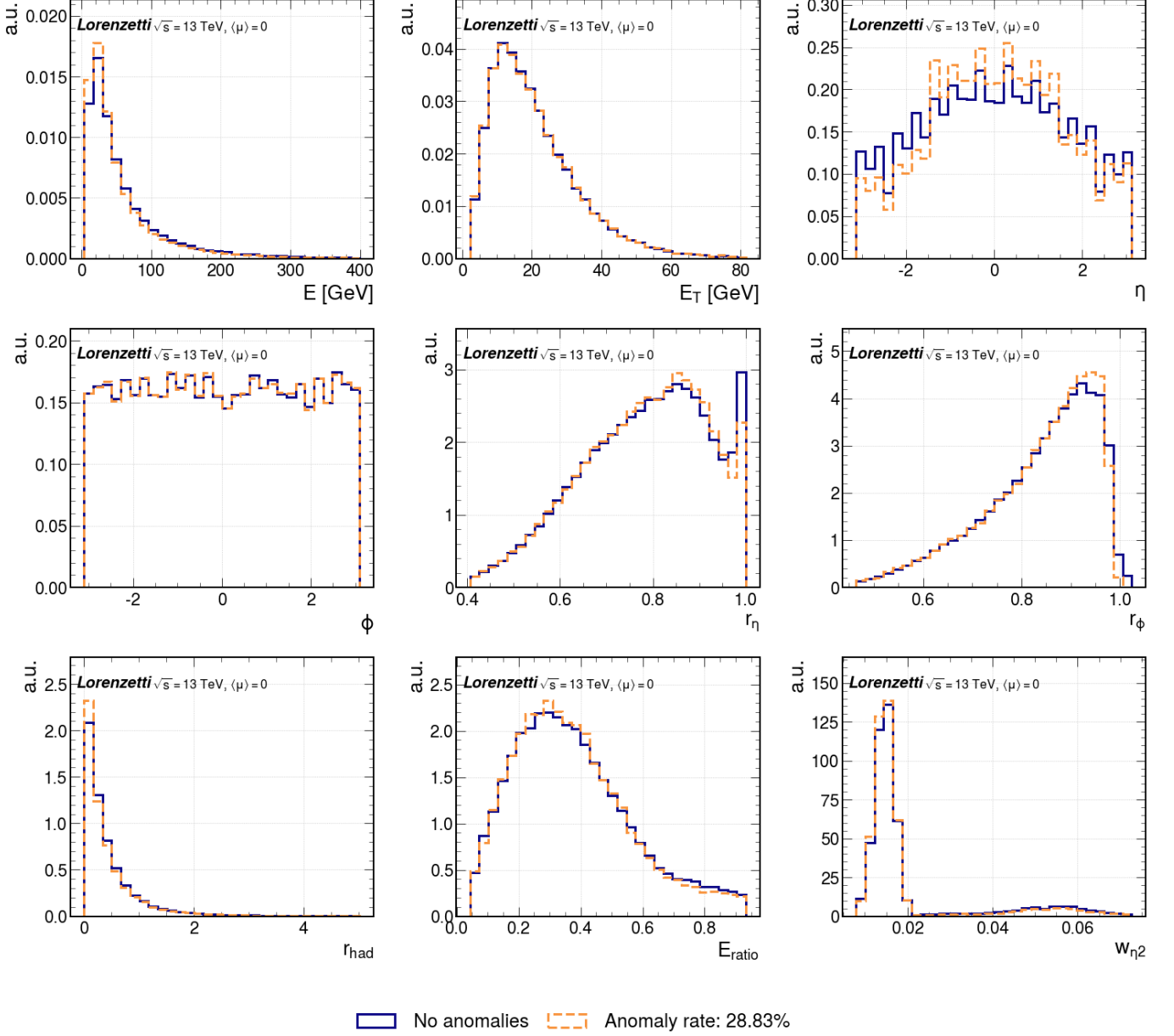


Fig. 14.4.: Distribution plots with dead detector modules and without pileup compared to normal feature distributions.

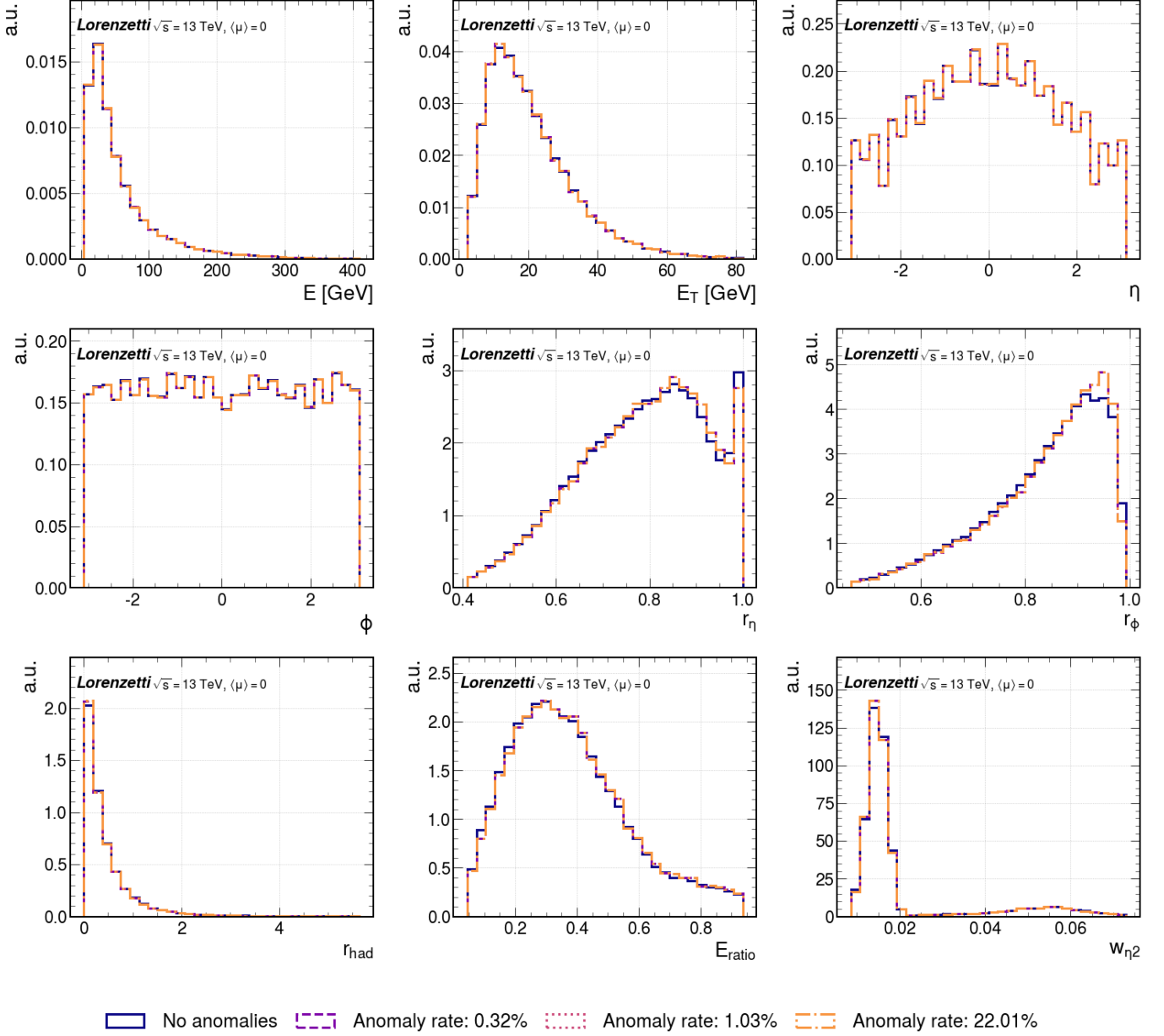


Fig. 14.5.: Distribution plots with increased noise, using noise factor 20, and without pileup for various anomaly rates.

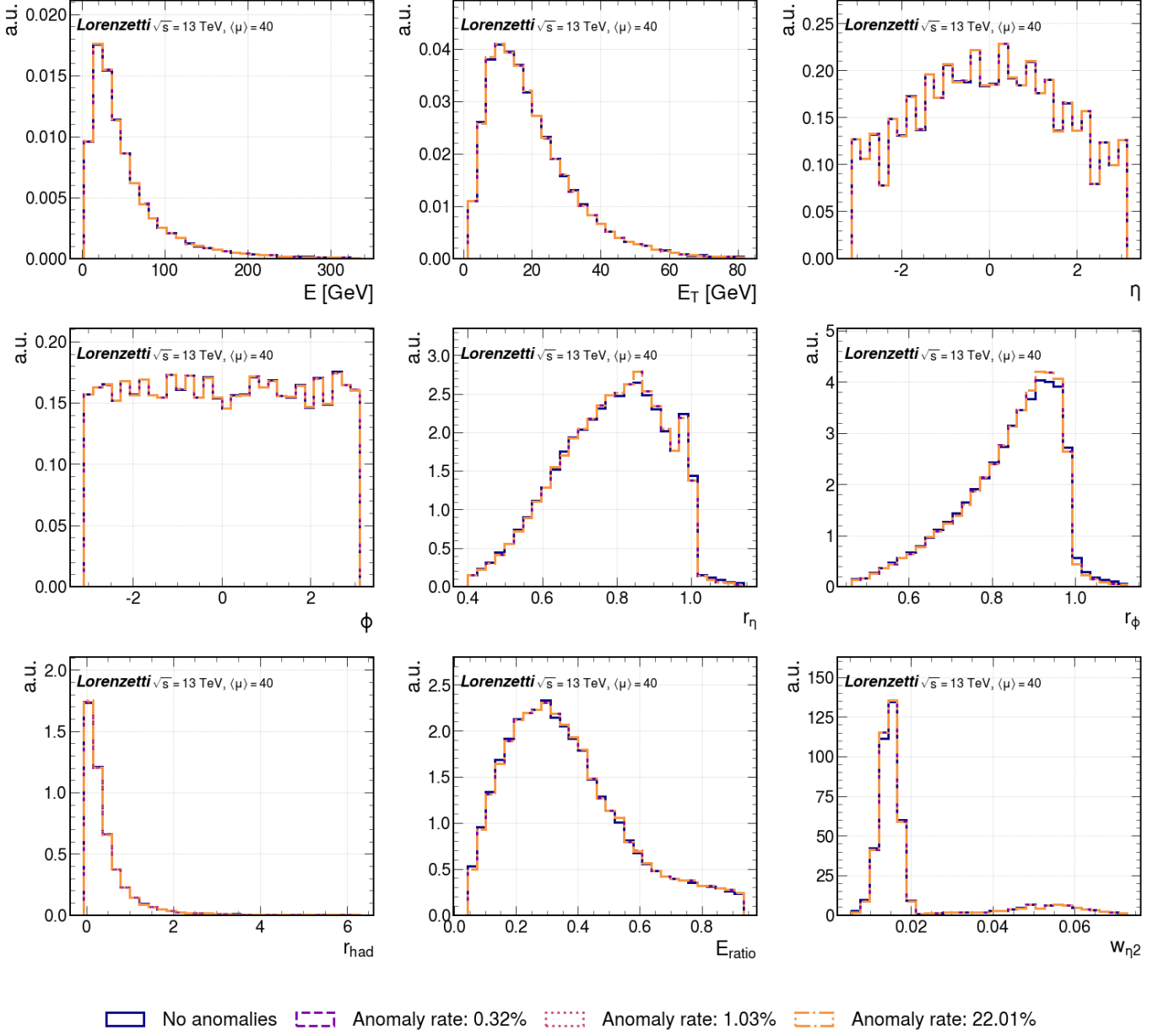


Fig. 14.6.: Distribution plots with increased noise, using noise factor 20, and with pileup $\langle \mu \rangle = 40$ for various anomaly rates.

14.5 Results

Table 14.2 reports the MCC scores achieved with the TSAD models on various test sets containing different artificial defects and varying pileup conditions. For each pileup condition (i.e. $\langle\mu\rangle = 0, 40$), the models are trained on a training set, subjected to the same pileup conditions. All of the models struggle extremely to identify the synthetic anomalies, especially for low anomaly rates.

Additional studies with varying noise factors (between 5, 10, 50) can be found in Appendix B.5. These results are not reported here, as the anomaly detection failed in most cases, probably due to the anomalies being extremely rare and subtle.

The anomalies associated with inactive modules are rather easy to identify for the TSAD models, which is expected since they affect large phase space regions. Compared to other test data, all DL models seem to learn and correctly detect part of the anomalies, while the baseline fails at doing so. The MCC scores are quite high despite all models tagging some FPs.

Figure 14.7 displays the time series inputs, the outputs reconstructed by the TranAD model, averaged anomaly score and predicted anomaly labels using the *local (majority voting)* method. Despite correctly identifying some anomalies within the larger anomalous sequences, the TranAD model still mistakenly tags many normal events (FP).

Tab. 14.2.: MCC scores (averaged over five random seeds) for reconstruction-based iTransformer, TranAD and USAD models on test datasets generated with Lorenzetti, under varying pileup conditions, anomaly types, and anomaly rates. The baseline is deterministic, so only a single MCC score is reported.

Anomaly Type	Pileup $\langle\mu\rangle$	Anomaly Rate [%]	Model	Best MCC	Anomaly label method
Dead Modules	0	28.83	Baseline	0.006	local (incl. OR)
			iTransformer	0.571 ± 0.013	local (maj. voting)
			TranAD	0.775 ± 0.008	local (maj. voting)
			USAD	0.642 ± 0.017	local (maj. voting)
Increased Noise ($f_N = 20$)	0	0.32	Baseline	0.019	local (incl. OR)
			iTransformer	0.000 ± 0.000	local (maj. voting)
			TranAD	0.014 ± 0.008	local (incl. OR)
			USAD	0.000 ± 0.000	local (maj. voting)
Increased Noise ($f_N = 20$)	0	1.03	Baseline	0.007	local (incl. OR)
			iTransformer	0.000 ± 0.000	local (maj. voting)
			TranAD	0.011 ± 0.018	local (incl. OR)
			USAD	0.003 ± 0.009	local (incl. OR)
Increased Noise ($f_N = 20$)	0	22.01	Baseline	0.316	global
			iTransformer	0.041 ± 0.008	global
			TranAD	0.127 ± 0.004	local (maj. voting)
			USAD	0.186 ± 0.026	local (maj. voting)
Increased Noise ($f_N = 20$)	40	0.32	Baseline	0.019	local (incl. OR)
			iTransformer	0.000 ± 0.000	local (maj. voting)
			TranAD	0.003 ± 0.004	local (incl. OR)
			USAD	0.001 ± 0.003	local (incl. OR)
Increased Noise ($f_N = 20$)	40	1.03	Baseline	0.000	local (maj. voting)
			iTransformer	0.000 ± 0.000	local (maj. voting)
			TranAD	-0.002 ± 0.000	all equal
			USAD	0.000 ± 0.000	local (maj. voting)
Increased Noise ($f_N = 20$)	40	22.01	Baseline	0.287	global
			iTransformer	0.036 ± 0.008	global
			TranAD	0.088 ± 0.021	local (maj. voting)
			USAD	0.114 ± 0.064	local (maj. voting)

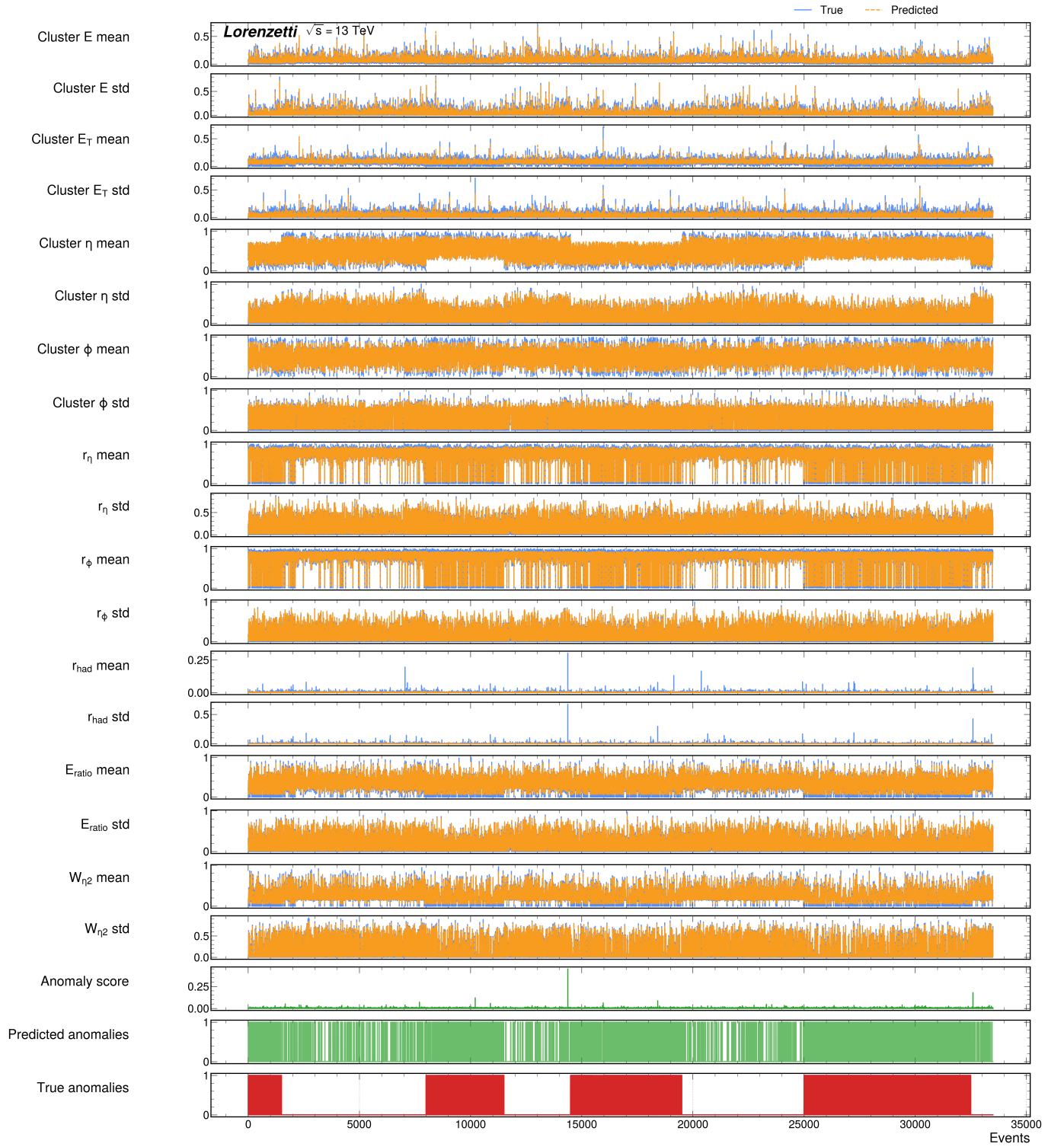
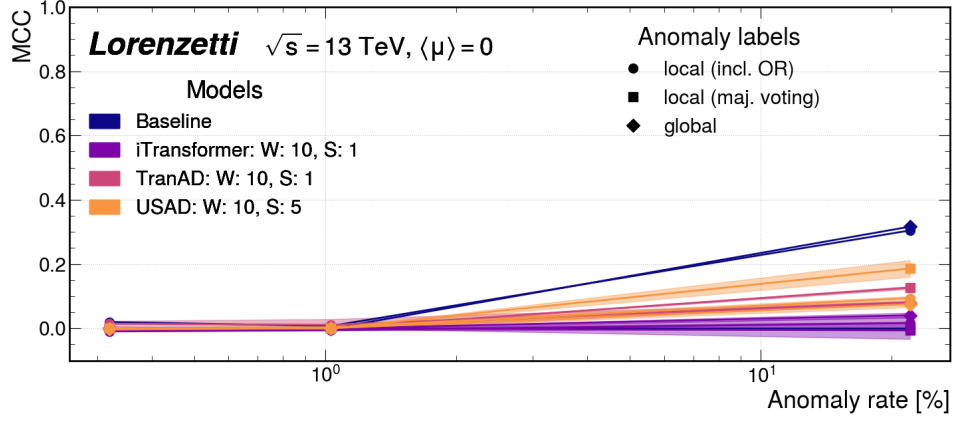
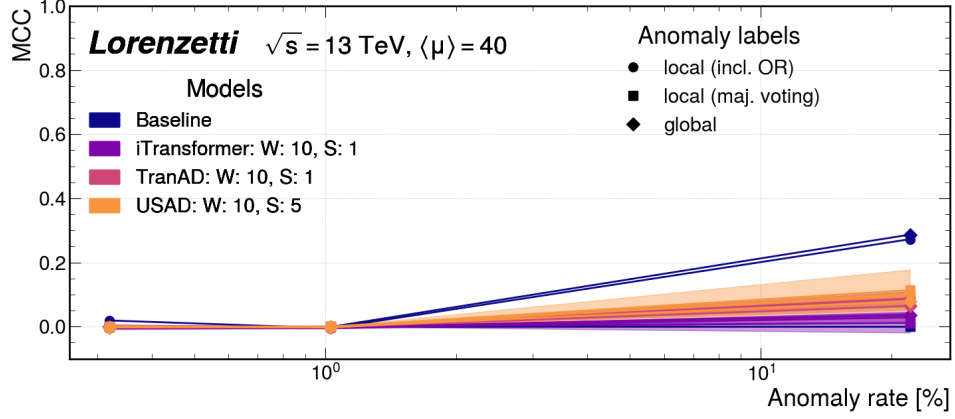


Fig. 14.7.: Applying TranAD to test data produced with *Lorenzetti*, which contains artificial anomalies associated with inactive detector modules. The plot shows the input (blue) and reconstructed (orange) multivariate time series, predicted anomaly score (dark green) and anomaly labels obtained with *local (maj. voting)* (light green) compared to the true labels (red).

Figure 14.8 displays the averaged MCC scores for each TSAD model as a function of the anomaly rate with and without pileup when dealing with increased noise.



(a) Increased noise without pileup ($\langle \mu \rangle = 0$).



(b) Increased noise with pileup characterised by $\langle \mu \rangle = 40$.

Fig. 14.8.: MCC scores (averaged over five random seeds) as a function of the anomaly rate in the test dataset without pileup (Figure 14.8a) and with pileup (Figure 14.8b) for three anomaly label methods.

Figure 14.8a presents the case without pileup. Only for the highest anomaly rate (i.e. 22.01%), the baseline manages to correctly identify some anomalies. Similarly, the USAD model does not fail completely, as the anomaly rate increases, and correctly tags some of the anomalous events. Concerning the anomaly label methods, it seems that *local (majority voting)* sets an extreme threshold such that all the predicted labels are always 0, which leads to a MCC score of 0. The other anomaly label methods have more variation and provide thresholds of various degrees of usefulness.

Figure 14.8b represents the same information but for the test data containing pileup characterised by $\langle \mu \rangle = 40$. The tendencies are very similar to the case without pileup. The only difference lies in the fact that multiple models have higher standard variations on the MCC scores in the presence of pileup.

The second type of anomalies, which are associated with increased noise, appears to pose a greater challenge for the tested [TSAD](#) methods. This can be attributed to the fact that these anomalies always occur alongside a real physics signal, effectively breaking the underlying assumption that anomalies originate from a distribution distinct from the majority of the data (i.e. the ‘normal’ data).

14.6 Summary and Outlook

This chapter describes the injection of artificial detector defects into the general-purpose calorimeter simulation framework [Lorenzetti](#). Two types of defects were studied: The case where increased noise is convoluted with a physics signal and affects the electric signal induced by the energy deposit, and the case where some detector modules are inactive, implying that such cells do not transmit any electronic signal regardless of the physics event.

Next, two training datasets were produced, consisting of 65k events each, one with ($\langle\mu\rangle = 40$) and one without pileup ($\langle\mu\rangle = 0$). Associated with each training set, multiple test sets were generated, where various synthetic anomalies had been injected. The anomaly rates and types were varied among the test sets.

Finally, four [TSAD](#) algorithms, introduced in Chapter 13, namely the unsupervised baseline, the reconstruction-based [iTransformer](#), [TranAD](#) and [USAD](#), were applied to the produced data.

The models correctly identify anomalies associated to inactive detector modules. Since these are very obvious anomalies, this was expected and serves as a validation that [TSAD](#) models can automatically detect such defects.

Regarding anomalies associated with increased noise, all models struggled to correctly identify them. However, for higher anomaly rates associated with increased noise, some anomalies were detected, despite the convolution of noise and physics signal. It appears that limitations of the simulation framework – namely the impossibility of implementing noisy clusters uncorrelated from physics signals – prevent the simulation of realistic coherent noise patterns.

This project was presented as a poster and a flash talk at the [EuCAIF](#) conference 2025 and is included in the conference proceedings.

For future projects, it could be interesting to further study the discriminative power of the shower shape variables. Since these variables presented a slightly modified distribution in the presence of anomalies, this might indicate that they offer additional information to identify detector defects.

Furthermore, the inclusion of anomalies in the simulation framework and the

produced datasets containing anomalies can be used for various other studies, where Lorenzetti data are employed.

TSAD for Data Quality Monitoring in the ATLAS LAr Calorimeter

15.1 Introduction

This project has been realised in collaboration with the authors of [78] and details the application of the unsupervised strategies to ATLAS DQM processes. Concretely, we apply the reconstruction-based iTransformer, introduced in Chapter 13, to real data from the ATLAS LAr calorimeter and aim to identify various detector issues such as noise bursts in an automated and potentially more efficient way than current DQM techniques.

15.2 Data Preparation and Model Training

The focus lies on evaluating the quality of the signals provided by the electromagnetic LAr calorimeters of the ATLAS detector.

As described in Section 2.2.2, the FEBs receive and modify the electronic signal provided by the LAr calorimeter. A typical pulse shape of a signal is shown in Figure 2.7a. After the FEBs, the signal is bipolar-shaped and integrates to zero over its duration. We can distinguish the quality of the pulse shape and timing of the pulse τ_{cell} . The quality factor Q_{cell} measures how well the pulse shape matches the expected LAr pulse shape. It is defined as:

$$Q_{cell} = \sum_{i=1}^5 [E_{cell}(g_i - g'_i \tau_{cell}) - (s_i - P)]^2, \quad \text{with} \quad E_{cell} = \sum_{i=1}^5 a_i (s_i - P), \quad (15.1)$$

where s_i are the sampled cell signals, P the signal pedestal value, a_i filtering weights chosen to minimise noise and pileup effects, g_i the expected pulse shape and g'_i its time derivative. The quality factor has a saturation value of $Q_{cell} = 65535$ [62, 78, 80].

The signals are grouped together to form topoclusters as described in Section 2.3.1. For four distinct detector parts (two parts of the EMB and two of

the [EMEC](#)), the timing and quality factors of topoclusters are evaluated, and their mean and standard deviation are computed. Therefore, the final dataset contains $4 \cdot 2 \cdot 2 = 16$ features. The features related to timing information are standardised such that their mean is 0 and standard deviation 1, whereas the quality factor features are scaled by being divided by the saturation value. More details concerning the data preprocessing are given in [\[78\]](#), as the project was done in collaboration with the authors of this note.

The training dataset contains almost 113 million events, roughly 40 million events from the main (pp) stream, 8 million events from the hard probes stream and 65 million events from the [UPC](#) stream. For the pp collisions, a run recorded in 2018 at $\sqrt{s} = 13$ TeV was selected. The two streams containing heavy-ion collisions (hard probes and [UPC](#) stream) were collected in 2023 at $\sqrt{s_{NN}} = 5.36$ TeV with a bunch spacing of 50 ns. For testing, the trained model is applied to either a different data stream for validation purposes or to a selection of [LBs](#), where known detector issues occur. In total, the test data amounts to approximately 85k events.

Following the sliding windows protocol described in [Chapter 13](#), the input time series are split into overlapping (non-overlapping) windows for training (testing). As done in [\[78\]](#), the input window size is set to 2000 events with a step size of 1000 for training and for testing, the window size and step size are both set to 2000. Based on our studies regarding the robustness and flexibility of various [TSAD](#) methods, we choose the reconstruction-based iTransformer [\[256\]](#) model (iTransformer-reco) for this application to ATLAS [DQM](#) data. Large input windows are difficult to handle for many of the models discussed in [Chapter 13](#) and for some, this increases the training time remarkably while the reconstruction performance deteriorates. Therefore, iTransformer-reco is the natural choice for testing a transformer-based model on ATLAS [DQM](#) data.

15.3 Results

We apply the reconstruction-based iTransformer model detailed in Chapter 13 to the [DQM](#) data described in Section 15.2 to data-taking runs containing different types of detector issues.

15.3.1 Cosmic Calo Stream

First, the data are collected during empty bunch crossings, i.e. the only type of signal comes from cosmic rays or detector defects (*Cosmic Calo* stream). This setting can be used for validating the [TSAD](#) algorithm, to verify whether it can recognise that the detector conditions are different (thus anomalous) compared to the training conditions.

Figure 15.1 visualises the input (blue) and reconstructed (orange) time series for each of the features. The last three subplots further show the averaged anomaly score, the anomaly labels derived with the *local (majority voting)* method and the anomaly labels obtained with the standard approach (red). Clearly, iTransformer-reco notices that the Cosmic Calo stream events differ from the training data, and flags them correctly.

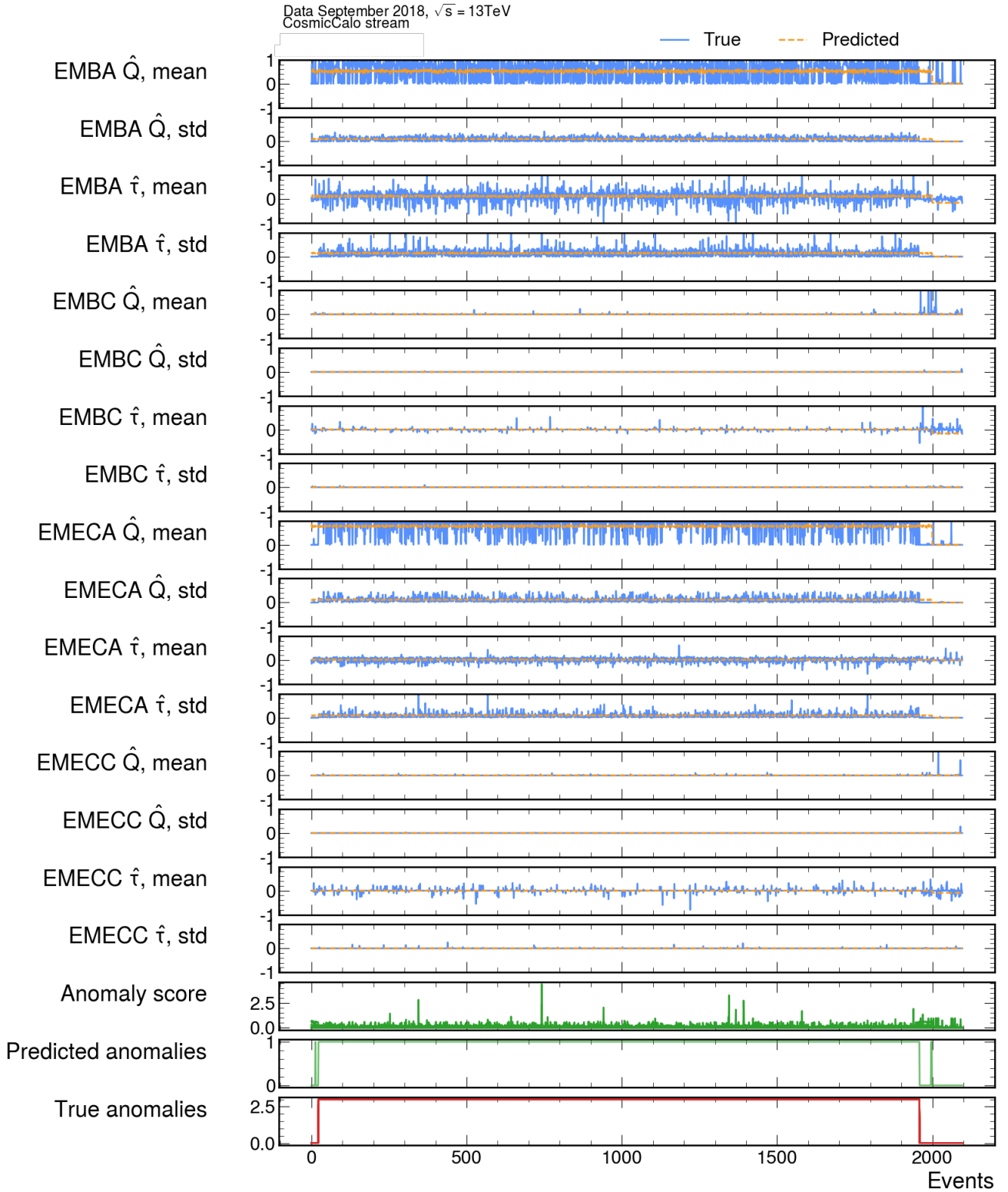


Fig. 15.1.: Applying iTransformer-reco to a run from the Cosmic Calo stream, which is different from the streams used for training. The plot shows the input (blue) and reconstructed (orange) multivariate time series, predicted anomaly score (dark green) and anomaly labels obtained with *local (maj. voting)* (light green) compared to labels from the standard algorithm (red).

15.3.2 Main Stream

Here, the data are collected from pp collisions during standard conditions of the LHC.

Figure 15.2 shows the input time series (blue) and the reconstructed output (orange). Additionally, in the last three subplots, the reconstruction error (anomaly score), the predicted anomaly labels using again the *local (majority voting)* method (green) and the anomaly labels from the standard algorithm (red). Visibly, the issue with a HV trip is identified by both the `iTransformer-reco` and the standard approach.

Differently, in Figure 15.3, the TSAD model identifies a detector issue shortly after event 10'000 that has not been flagged by the standard algorithm. However, this sequence visibly differs from the rest of the time series and should be labelled anomalous. Further investigation indicated that this anomalous sequence is probably due to an issue with the vacuum pump of the LAr calorimeter and had been missed by the human shifters. This anomaly was also identified by the LSTM-AE presented in [78] and constitutes a prime example of how TSAD methods could supplement the current ATLAS DQM.

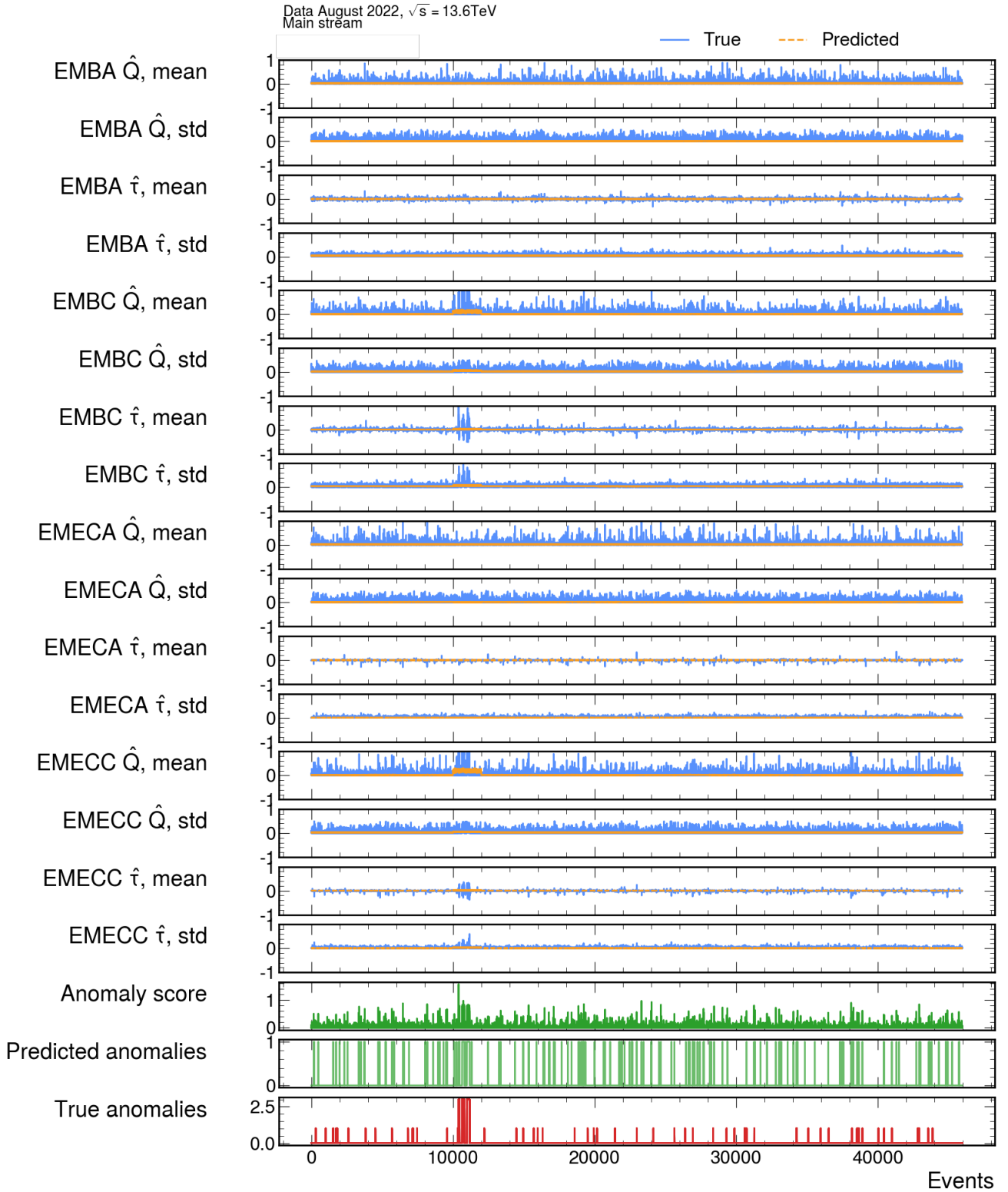


Fig. 15.2.: Applying iTransformer-reco to a run from the main stream with a HV issue shortly after 10'000 events. The plot shows the input (blue) and reconstructed (orange) multivariate time series, predicted anomaly score (dark green) and anomaly labels obtained with *global* (light green) compared to labels from the standard algorithm (red).

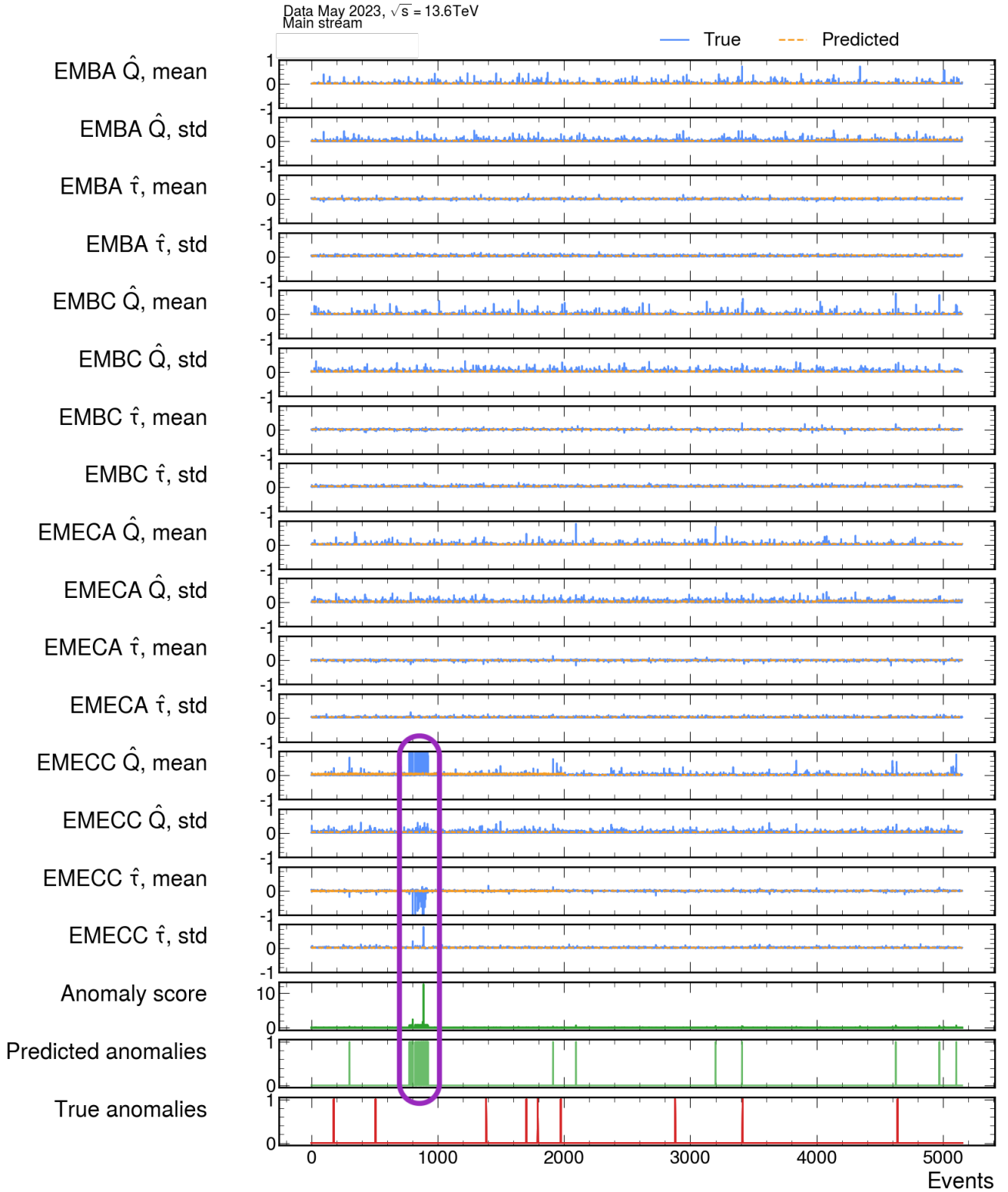


Fig. 15.3.: Applying iTransformer-reco to a run from the main stream with an issue shortly before 1'000 events. This issue is probably due to a malfunction in the vacuum pump and has not been detected by the standard algorithm. The plot shows the input (blue) and reconstructed (orange) multivariate time series, predicted anomaly score (dark green) and anomaly labels obtained with *global* (light green) compared to labels from the standard algorithm (red).

15.3.3 Hard Probes Stream

Finally, the data are collected during heavy-ion collisions. The training data consists partially of heavy-ion streams containing hard probes and UPC data, so these events should be familiar to the model. Indeed, as visible in Figure 15.4, the model is able to reconstruct most of the time series correctly. It identifies the three anomalous events, but it also adds some FPs.

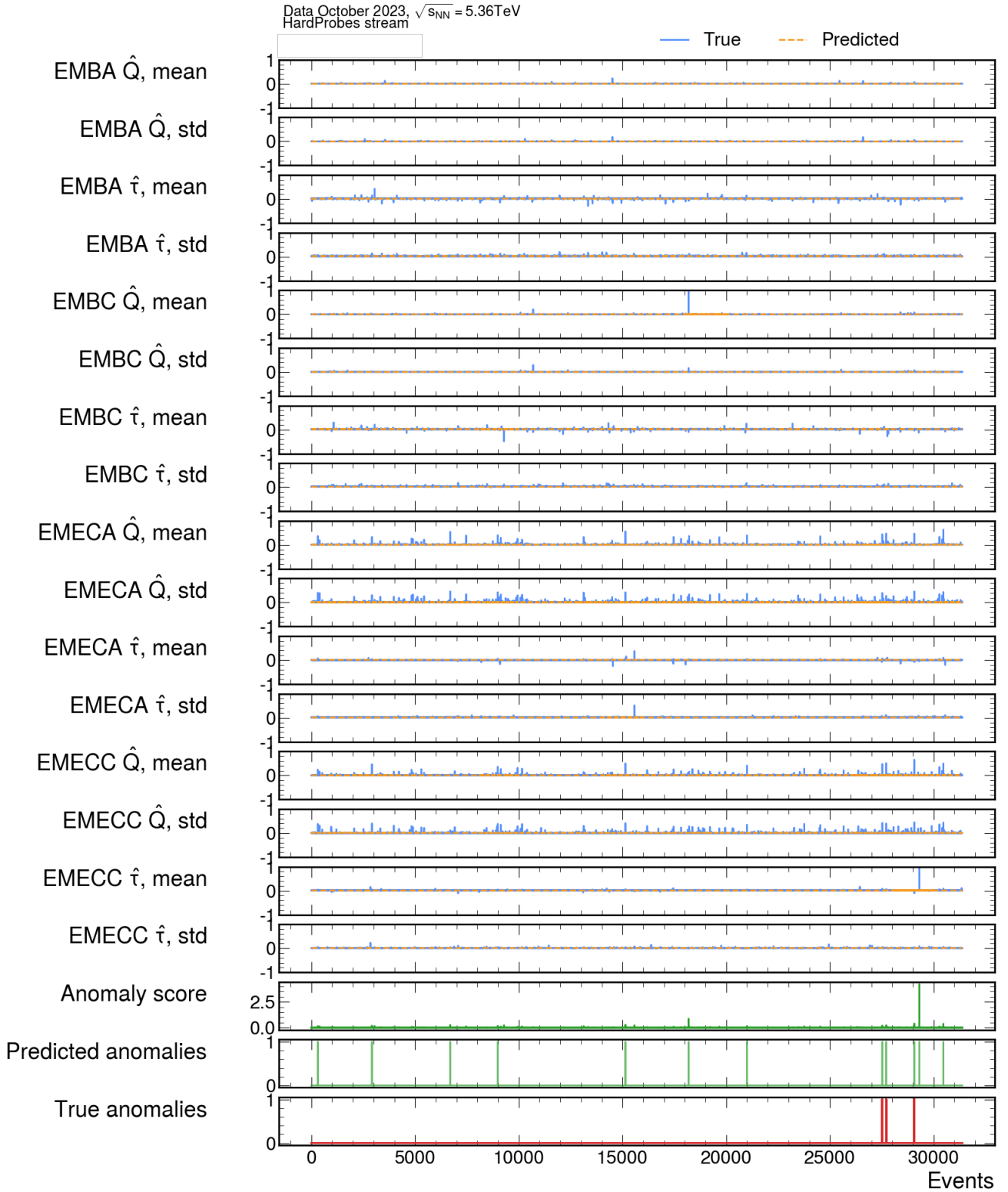


Fig. 15.4.: Applying iTransformer-reco to a run from the hard probes stream. The plot shows the input (blue) and reconstructed (orange) multivariate time series, predicted anomaly score (dark green) and anomaly labels obtained with *global* (light green) compared to labels from the standard algorithm (red).

15.4 Summary and Outlook

This chapter details the application of a transformer-based model for [TSAD](#), previously introduced in Chapter 13, to [DQM](#) of the ATLAS [LAr](#) calorimeter. The goal was to identify detector defects or other issues within these data to evaluate the usefulness of [TSAD](#) for detector monitoring in ATLAS.

The `iTransformer-reco` model was trained on real, experimental data from proton-proton and heavy-ion collision events and was tested on various runs containing specific detector issues. The data were represented as multivariate time series corresponding to calorimeter cluster quality factors and timing, and the training data were explicitly chosen to be heterogeneous, intending to increase the model's robustness to different data-taking conditions. The trained model was able to pick up on the various types of detector issues in different types of data streams. It also successfully identified that the Cosmic Calo stream differed from the learned training conditions and flagged it as anomalous. Most interestingly, the model managed to detect an issue related to the vacuum pump that had gone unnoticed by the standard approach. A different [TSAD](#) model also identified this anomalous sequence, which hints at the advantage of including temporal information in [DQM](#).

Furthermore, this work has been presented at the 'ATLAS [ML](#) Forum' in December 2024 and the 'ATLAS [ML](#) Workshop 2025'.

In conclusion, [TSAD](#) models based on [LSTMs](#) or transformers offer a robust and automated approach for detecting known and new detector issues, which could potentially support human shifters in [DQM](#) and improve the efficiency of the monitoring process and the resulting data quality.

Conclusion

This thesis investigated various applications and use cases of ML to HEP data. Given the large amounts of data available in HEP and the even larger amounts expected to be produced during the running of the HL-LHC, new and more efficient algorithms for data taking and processing are urgently needed. ML appears to be the ideal tool to tackle these challenges and is being intensely explored within the HEP community. As the quote from Freeman Dyson in the introduction suggests, breakthroughs are more often due to new technologies and tools than to new theoretical concepts. This is why so much effort is invested in exploiting the full potential of ML for HEP.

The first project, presented in Part II, described a deep sets-based approach to jet calibration. Based on the dijet *in situ* asymmetry, the ML model aims to improve the JER in data directly. Two approaches were identified to optimise the JER: one completely independent of truth-level information and one partially dependent on it. However, both methods show a strong hyperparameter dependency, which needs to be resolved empirically. Nonetheless, this provides a first study for integrating resolution optimisation into a future, ML-based calibration. For further improvements of the method, it would be crucial to apply the model to other event topologies than dijets. Finally, this work has been presented at the ‘ATLAS Hadronic Calibration Workshop’ in 2023 and 2024, and was detailed in an internal ATLAS note [207].

The second project, detailed in Part III, explored the use of TSAD methods for different applications. This is particularly interesting as we found a natural connection between HEP research and more industry-focused research with this topic. Due to this connection, both ATLAS and IBM Research have a shared interest in this problem for its potential applications to DQM or fraud detection, respectively. This shared research interest allowed us to fully exploit the industry collaboration set up by SMARTHEP, the European Training Network that this thesis is part of.

This project was split into several smaller studies: First, Chapter 13 investigated and benchmarked unsupervised strategies for anomaly detection in multivariate time series. This included the evaluation of methods to derive binary anomaly labels based on multivariate anomaly scores, where it was observed that the optimal choice strongly depends on the type and length of anomalies present in the data. It was further shown that anomalies in the training data deteriorate

the anomaly detection performance, and an attempt was made to mitigate this through the use of alternative loss functions.

Second, to have perfect control over the anomaly detection labels, in Chapter 14, QCD jet events were simulated and enhanced by the injection of synthetic anomalies. These anomalies mimicked detector issues related to inactive detector modules or large amounts of noise convoluted with actual physics events. Four unsupervised TSAD methods, introduced in Chapter 13, were applied to the simulated data to identify the injected anomalies. The models successfully identified anomalies associated with the inactive detector modules but struggled with those related to increased levels of noise correlated with a physics signal. This highlights the need for further studies simulating more realistic detector defects, in which random noise appears uncorrelated with physics signals. The inclusion of anomalies in the Lorenzetti framework also enables broader investigations into detector-related issues for future studies. These studies were presented at the ‘EuCAIF Conference 2025’ and will appear in the conference proceedings.

Finally, Chapter 15 applied a transformer-based TSAD model to ATLAS DQM tasks. This was evaluated on real data from the ATLAS LAr calorimeter. The employed TSAD model successfully identified several detector issues in different data streams and even managed to detect a defect that was missed by the standard algorithm.

To conclude, ML-based jet calibration and TSAD models for DQM have demonstrated potential for more efficient usage of the available resources in HEP. However, in both cases, their practical implementation and integration within the ATLAS framework remain challenging and require additional effort from the collaboration.

Résumé Substantiel

La thèse présentée s'inscrit dans le cadre des activités de l'expérience ATLAS, l'un des deux détecteurs polyvalents installés sur le Grand Collisionneur de Hadrons ([Large Hadron Collider \(LHC\)](#)) au CERN. ATLAS constitue l'un des instruments les plus complexes jamais construits par l'humanité pour explorer les constituants fondamentaux de la matière. Il est conçu pour enregistrer les collisions proton-proton ainsi que les collisions de noyaux lourds à des fréquences pouvant atteindre jusqu'à 40 MHz, générant ainsi un volume de données sans précédent dans l'histoire de la physique des particules expérimentale. Chaque interaction détectée par le système engendre une multitude de signaux électroniques qu'il faut ensuite reconstruire, calibrer et interpréter avec la plus grande précision possible.

L'exploitation scientifique de ces données est doublement cruciale : elle permet d'une part d'affiner la précision des mesures relatives au Modèle Standard, en testant ses prédictions à des échelles d'énergie inégalées, et d'autre part d'ouvrir la voie à la recherche de phénomènes nouveaux au-delà de ce cadre théorique. Cependant, atteindre ces objectifs scientifiques ambitieux implique de relever des défis techniques considérables, tant sur le plan instrumental que méthodologique. Parmi ceux-ci, le développement d'algorithmes de reconstruction et d'analyse capables de traiter efficacement des volumes massifs de données, tout en préservant la précision et la fiabilité des résultats, occupe une place centrale.

C'est précisément dans ce contexte que s'inscrit la présente recherche, qui explore l'apport des méthodes d'apprentissage automatique ([Machine Learning \(ML\)](#)) à deux aspects essentiels du traitement des données au sein d'ATLAS : l'étalonnage de l'énergie des jets et le contrôle de la qualité des données du calorimètre. Ces deux thématiques, bien que distinctes dans leurs finalités, partagent des problématiques communes : la nécessité d'extraire une information physique fiable à partir de signaux complexes et bruités, et l'exigence d'automatiser certaines étapes de traitement afin de répondre à la cadence toujours croissante des prises de données du LHC.

L'Étalonnage des Jets

La première problématique abordée concerne l'étalonnage des jets, ces gerbes de particules issues de l'hadronisation des quarks et des gluons produits lors des

collisions. Dans l'expérience ATLAS, les jets jouent un rôle fondamental dans la plupart des analyses de physique, qu'il s'agisse de mesures précises de sections efficaces, d'études de processus de production du boson de Higgs ou encore de la recherche de nouvelles particules massives. L'étalonnage des jets est donc une étape critique, car il conditionne directement la précision des mesures et, par conséquent, la sensibilité globale de l'expérience aux nouveaux phénomènes physiques.

Traditionnellement, l'étalonnage repose sur une chaîne complexe de procédures successives, où chaque étape vise à corriger un biais particulier : effets de la non-linéarité de la réponse du calorimètre, dépendances géométriques, pertes d'énergie dans les zones inactives, ou encore différences entre les particules hadroniques et électromagnétiques. Chaque correction est généralement dérivée à partir de simulations détaillées, calibrées elles-mêmes à l'aide de données expérimentales. Cette approche, bien qu'extrêmement rigoureuse, reste lourde à mettre en œuvre et difficile à adapter à de nouveaux contextes, notamment dans la perspective du [High-Luminosity Large Hadron Collider \(HL-LHC\)](#) où le volume de données augmentera de manière exponentielle.

L'originalité du travail présenté dans cette thèse réside dans la proposition d'une approche unifiée de l'étalonnage, exploitant la flexibilité et la puissance d'outils issus de l'apprentissage automatique, en particulier les réseaux de neurones. L'idée est de fusionner plusieurs étapes de la chaîne d'étalonnage traditionnelle en un seul modèle global, entraîné de manière à optimiser directement la résolution de l'énergie des jets. Le cœur de la méthode repose sur l'utilisation de l'asymétrie des dijets comme observable de substitution pour estimer la résolution, permettant ainsi d'éviter le recours systématique à des informations de vérité simulées, souvent coûteuses à produire et sujettes à des incertitudes liées aux modèles de simulation.

Deux variantes principales ont été explorées : l'une totalement indépendante des données de vérité simulées, et l'autre partiellement dépendante, offrant ainsi un compromis entre réalisme et contrôle méthodologique. Les résultats obtenus ont montré une sensibilité notable aux hyperparamètres du modèle et à la nature des jeux de données utilisés pour l'entraînement. Toutefois, ils ont également mis en évidence le potentiel de cette approche pour améliorer la précision de la calibration des jets, ouvrant ainsi la voie à une intégration plus flexible et potentiellement plus efficace des outils d'apprentissage automatique dans la chaîne officielle d'étalonnage d'ATLAS.

Ces travaux ont donné lieu à plusieurs présentations dans le cadre de la collaboration ATLAS, notamment lors des ATLAS Hadronic Calibration Workshops de 2023 et 2024, ainsi qu'à la rédaction d'une note technique interne [207].

Détection d'Anomalies dans des Séries Temporelles

La seconde problématique étudiée dans cette thèse concerne le contrôle qualité des données issues du calorimètre à argon liquide ([Liquid Argon \(LAr\)](#)) d'ATLAS. Dans une expérience de l'ampleur d'ATLAS, qui comporte plusieurs millions de canaux électroniques, la présence d'anomalies ou de défauts instrumentaux est inévitable. Ces anomalies peuvent avoir diverses origines : défaillances de modules électroniques, instabilités de gain, interférences parasites ou encore bruits transitoires dans les signaux. Une détection rapide et fiable de ces défauts est essentielle pour préserver la qualité des analyses de physique, car des données défectueuses peuvent introduire des biais systématiques significatifs.

Le principal défi réside dans la rareté des données étiquetées, c'est-à-dire des exemples de comportements anormaux connus, ainsi que dans la grande diversité des types d'anomalies possibles, souvent inconnues à l'avance. Pour répondre à cette difficulté, la thèse explore des approches non supervisées de détection d'anomalies dans des séries temporelles multivariées, en s'appuyant sur des architectures modernes de type transformeur, que l'on suppose capables de modéliser des dépendances temporelles et des corrélations complexes entre variables.

L'étude s'est déroulée en plusieurs étapes complémentaires.

Dans un premier temps, différentes méthodes non supervisées ont été évaluées sur des ensembles de données publiques, afin de caractériser leurs performances et leurs limites dans des conditions contrôlées. Une attention particulière a été portée à la conversion des scores continus de détection en étiquettes binaires exploitables, étape souvent sous-estimée mais cruciale pour l'interprétation des résultats.

Les analyses ont montré que l'efficacité de la détection dépend fortement de la nature, de l'amplitude et de la durée des anomalies. De plus, la présence d'anomalies dans les données d'entraînement peut dégrader les performances des modèles, phénomène qui a conduit à l'exploration de fonctions de perte robustes et de stratégies d'entraînement alternatives pour atténuer cet effet.

Dans un second temps, le cadre de simulation *Lorenzetti* a été utilisé pour introduire artificiellement des anomalies dans des événements simulés de jets. Ce dispositif permet de reproduire de manière contrôlée divers types de défauts instrumentaux, tels que des modules inactifs ou du bruit électronique anormal superposés aux signaux physiques. L'intérêt de cette approche réside dans la possibilité d'obtenir des étiquettes parfaitement fiables, indispensables pour quantifier la sensibilité et la robustesse des méthodes proposées.

Les résultats ont montré que les modèles testés détectaient efficacement les anomalies associées à des modules inactifs, mais rencontraient davantage de

difficultés face aux anomalies de type bruit, plus subtiles et diffuses. Ces observations ouvrent des perspectives d'amélioration, notamment en simulant des défauts de détecteur plus réalistes.

Enfin, les mêmes méthodes ont été appliquées à des données réelles issues du calorimètre à argon liquide d'ATLAS. Le modèle du type transformeur s'est révélé capable d'identifier des défauts passés inaperçus des algorithmes de détection standards, démontrant ainsi leur potentiel pour renforcer la fiabilité du système de [Data Quality Monitoring \(DQM\)](#). Ces résultats ont été présentés lors de plusieurs conférences, dont la conférence [European Coalition for AI in Fundamental Physics \(EuCAIF\)](#) 2025 [5, 6], et ont suscité un intérêt marqué au sein de la collaboration.

Dimension Interdisciplinaire

Au-delà des aspects purement techniques, cette recherche s'inscrit dans un cadre résolument interdisciplinaire. En effet, la détection d'anomalies dans les séries temporelles multivariées constitue un problème commun à de nombreux domaines scientifiques et industriels, allant de la surveillance de réseaux électriques à la détection de fraudes bancaires. Dans ce contexte, la collaboration établie avec IBM Research, dans le cadre du projet européen SMARTHEP, illustre parfaitement les bénéfices d'une approche croisée entre physique des hautes énergies et apprentissage automatique appliquée. Cette synergie permet d'enrichir mutuellement les outils et les perspectives : les techniques développées pour la physique trouvent des applications dans le monde industriel, tandis que les méthodes industrielles contribuent à l'innovation au sein des expériences de physique fondamentale.

Conclusion

En conclusion, cette thèse met en évidence le rôle croissant et désormais incontournable de l'apprentissage automatique dans l'écosystème des grandes expériences de physique des particules. Les méthodes développées pour l'étalonnage des jets et pour le contrôle qualité des données démontrent leur potentiel concret pour améliorer à la fois la précision des mesures et la robustesse des analyses. Elles constituent une première étape vers une intégration plus large et systématique des outils d'apprentissage automatique dans les chaînes de reconstruction et d'analyse du [LHC](#). Cependant, leur déploiement opérationnel à grande échelle demeure un défi, nécessitant des efforts concertés de validation, d'interprétabilité et d'adaptation aux contraintes spécifiques de l'infrastructure d'ATLAS.

Appendices

Appendix on Jet Calibration

A.1 Jet Hierarchy

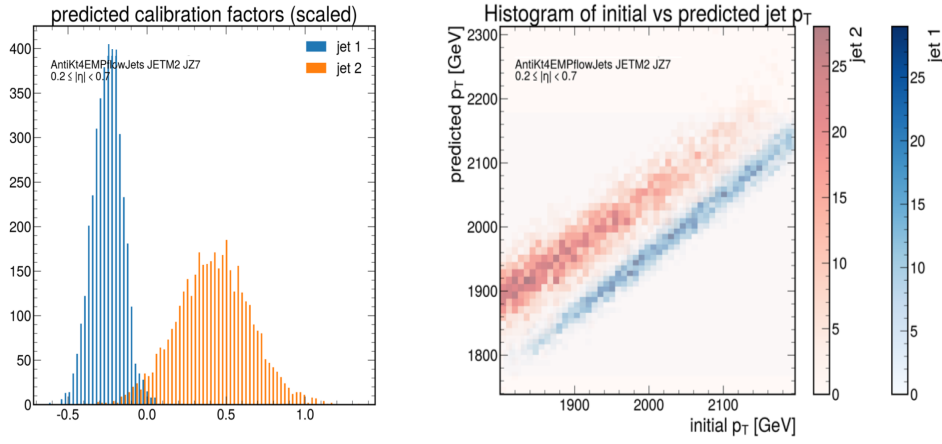
In general, the goal is to find a calibration model to predict the calibration factors for any jet separately from any other jet present in the collision event that produced it. Therefore, we want the predictions for the two leading jets of an event to be independent of which one is the leading jet, i.e. independent of the jet hierarchy.

However, it has been observed that as soon as **GSC** variables are also given as inputs to the model, it predicts very different calibration factors for each of the two leading jets per event. The model is acting as a (very good) classifier for the jet hierarchy per collision event.

The following results are found with a slightly different learning model (no time-distributed layer) using the alternative asymmetry-based term from Eq.(8.12) and the bin-wise constraint term from Eq.(8.10). This constraint is not used in other studies because its implementation is more complicated and the results are similar to those obtained with the event-wise constraint from Eq.(8.6) (the weighting factor f_r changes, of course). The training is done over 30 epochs using roughly 35k dijet events. For testing, we use only 4k dijet events, which have not been used previously.

The following plots in Figure A.1 further emphasise the problem. In Figure A.1a, it is obvious that the model predicts very different calibration factors for the two leading jets. This also translates to predicted p_T values that differ a lot for the two jets, as can be seen in Figure A.1b.

When repeating the training with the same training configuration but without the **GSC** variables, the model does not show such a remarkable distinction of the leading jets any more. If the number of training epochs is increased, one again observes the distinction of the two leading jets. Therefore, it seems that including the **GSC** variables as inputs allows the model to distinguish the two leading jets faster, but the same can be achieved with a longer training duration.



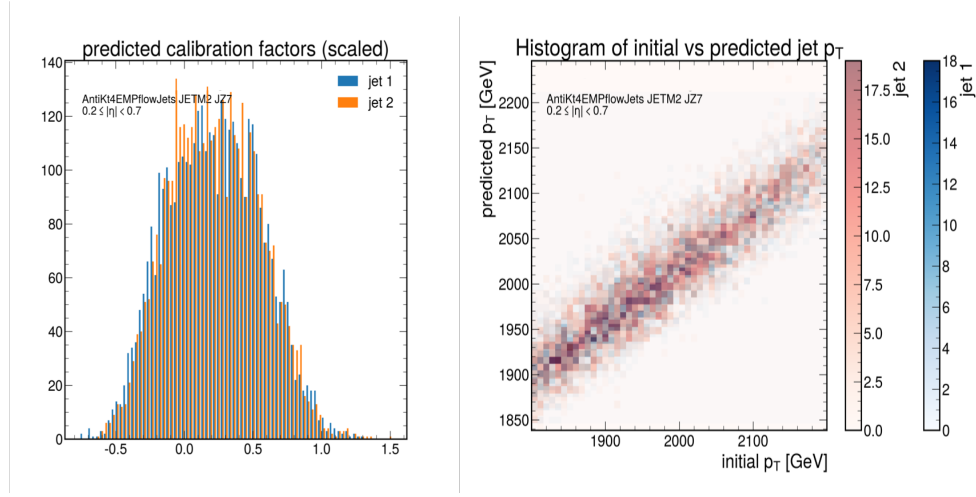
(a) Histogram of standardised calibration factors predicted by the model. (b) Histogram of initial and predicted p_T values for the two leading jets.

Fig. A.1.: Results with loss function from Eq.(8.12) with $f_t = 0, f_a = f_r = 1$ and additional GSC variables as input.

Solution It seems that the behaviour is caused by the use of BatchNorm1d layers in all dense NNs used in the ML model. Since the inputs are always either leading or sub-leading jets (for all events in a batch), computing the batch norm allows the model to learn something about the average p_T per mean, which is different for leading or sub-leading jets. As soon as the batch normalisation layers are ignored, this behaviour is no longer observed. In addition, omitting batch normalisation also guarantees more deterministic predictions, i.e. for the same inputs the model always predicts the same calibration factors.

However, using `model.eval()` [281] mode for evaluation and testing allows to make the predictions deterministic and ensures the equal treatment of the two leading jets. This has been experimentally verified: switching the two leading jets now gives the same results. In that case, the batch normalisation layer can be introduced again, but there are still some doubts on whether this operation adds any unwanted information to the training process as it keeps track of the batch means of certain features.

Finally, to completely avoid this issue and ensure that it will not affect the learning process in any way, the jets are shuffled randomly. In the loss function, where the event information is needed to compute the asymmetry term, the jets are re-ordered using the event number. This solves the issue and the model no longer shows any noticeable distinction between the two leading jets of an event, as shown in Figure A.2.



(a) Histogram of standardised calibration factors predicted by the model. (b) Histogram of initial and predicted p_T values for the two leading jets.

Fig. A.2.: Results with loss function from Eq.(8.12) with $f_t = 0, f_a = f_r = 1$ and additional GSC variables as input after fixing the jet hierarchy issue.

A.2 Feature Dependence

To study the importance of certain input features for the minimisation of the asymmetry-based loss function (based on Eq.(8.12) with $f_t = f_r = 0, f_a = 1$), the performance of the model with and without them has been studied. To quantify the model's performance, the JER plots are compared. The model performs quite badly without p_T as input (i.e. no jet p_T and no jet constituents p_T) for 30 epochs, which is slightly surprising as p_x, p_y and E are still among the input features and they are highly correlated with p_T . However, when doubling the number of epochs (60 epochs) in run 5, the model reproduces similar results, as it did with all the input features. This indicates that the model can extract the missing p_T information from p_x, p_y and E but needs more training time for this. Extending these studies, also the jet energy (jet E) and jet p_y (jet p_y and constituents for p_y) are omitted separately and then in combination. The configurations and observations made for every run are presented in Table A.1.

Tab. A.1.: Configurations and observations when studying feature dependence.

Runs	Epochs	Input p_T (Yes/No)	Input E (Yes/No)	Input p_y (Yes/No)	Observations
1	30	Y	Y	Y	good JER, deviated JES
2	30	N	Y	Y	bad JER but training loss still decreasing
3	30	N	Y	Y	good JER, deviated JES (like run 1)
4	30	N	N	Y	JER like initial JER or worse
5	60	N	N	Y	good JER (like run 1)
6	30	N	Y	N	good JER (worse than run 1, but better than initial JER)
7	60	N	Y	N	good JER (like run 1)
8	30	N	N	N	JER looks like initial JER (no improvement)

Permutation Feature Importance Further, a short permutation feature importance study has been performed. For this, the entries of one feature at a time are randomly shuffled and passed to the model, which has been trained with all features. The more the modification of a certain feature causes the loss to increase (decrease), the more (less) important is the feature for the model's performance. The results of this study are shown in Figure A.3. As expected, the jet transverse momentum is by far the most important input feature. It is followed by the energy (which is strongly correlated with the momentum) and EM0, which corresponds to the energy fraction measured in the 0th layer of the electromagnetic LAr calorimeter. Next is the GSC variable Ntrk1000 which describes the number of tracks with $p_T > 1$ GeV ghost-associated with the jet.

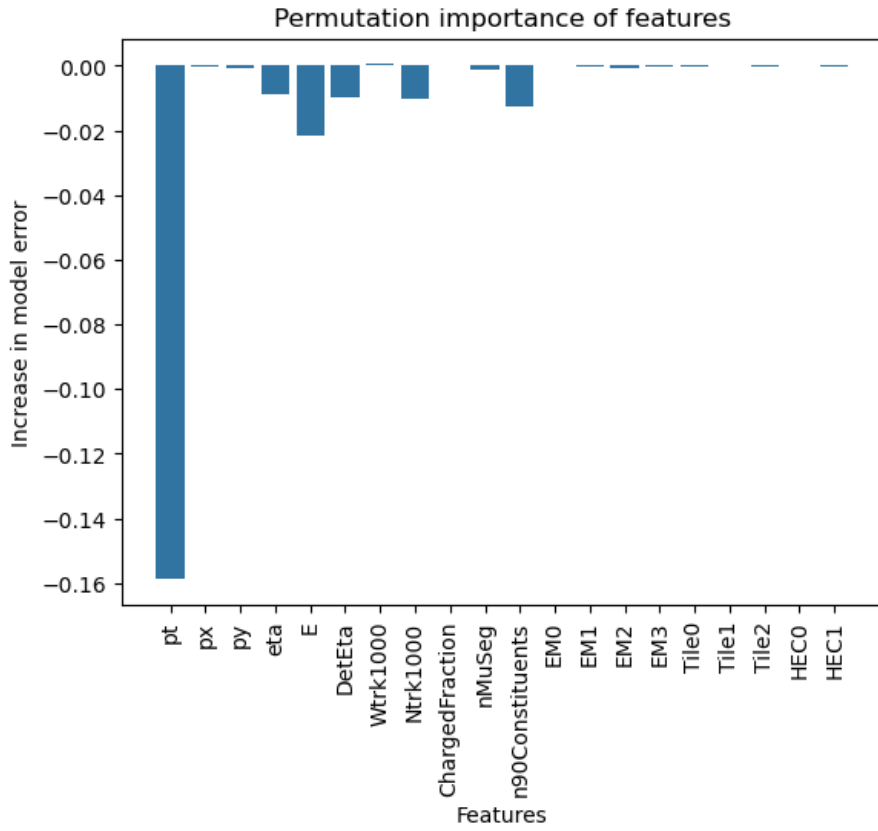


Fig. A.3.: Permutation feature importance.

A.3 Additional Results

A.3.1 Truth-Independent Training: Results for JETM2 JZ2

Figure A.4 displays the evolution of the loss function and its components for each hyperparameter configuration using JETM2 JZ2 samples.

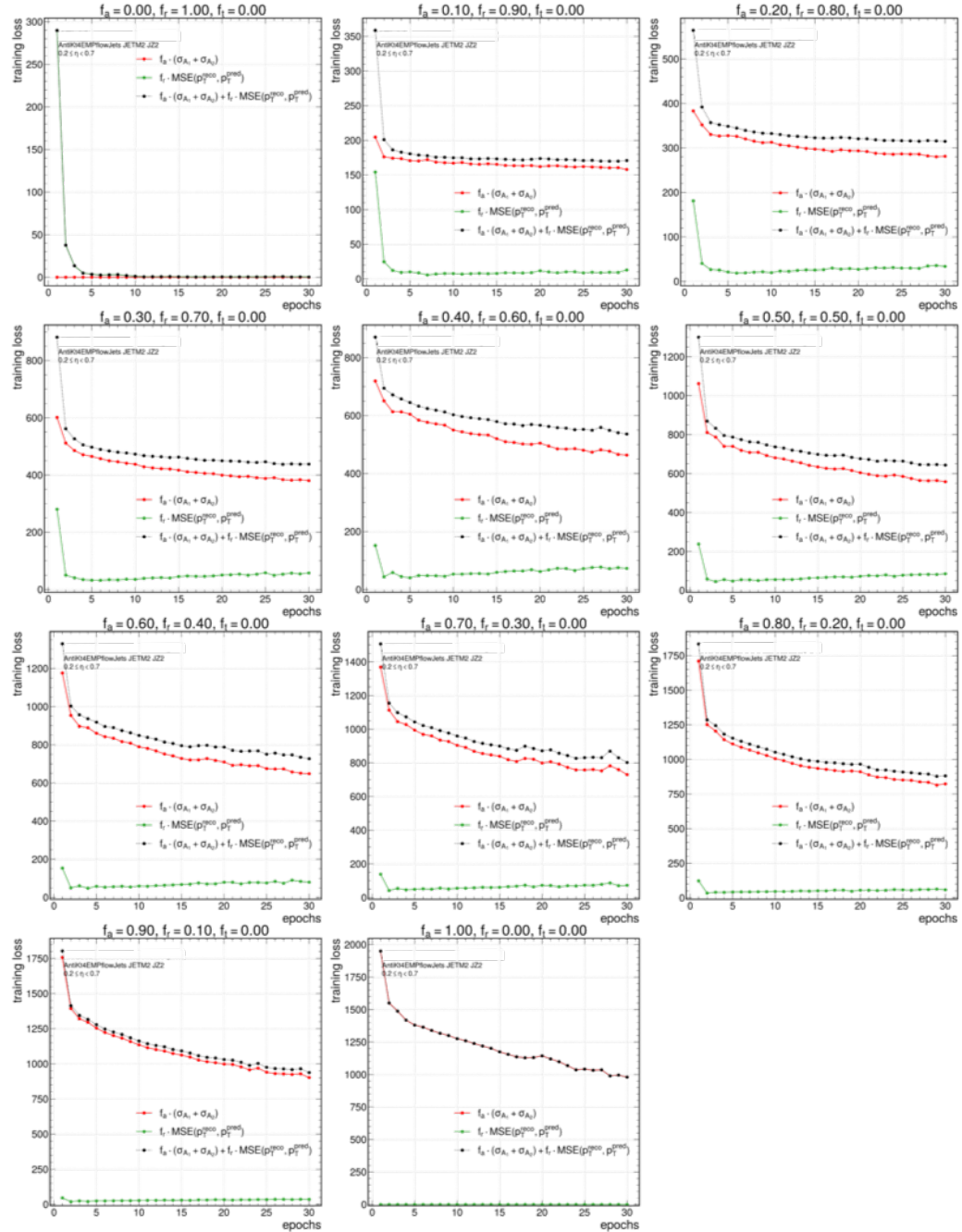


Fig. A.4.: Training loss functions and their components for each combination of hyperparameters f_a and $f_r = 1 - f_a$.

A.3.2 Partially Truth-Based Training: Results for JETM2 JZ2

Figure A.5 displays the evolution of the loss function and its components for each hyperparameter configuration using JETM2 JZ2 samples.

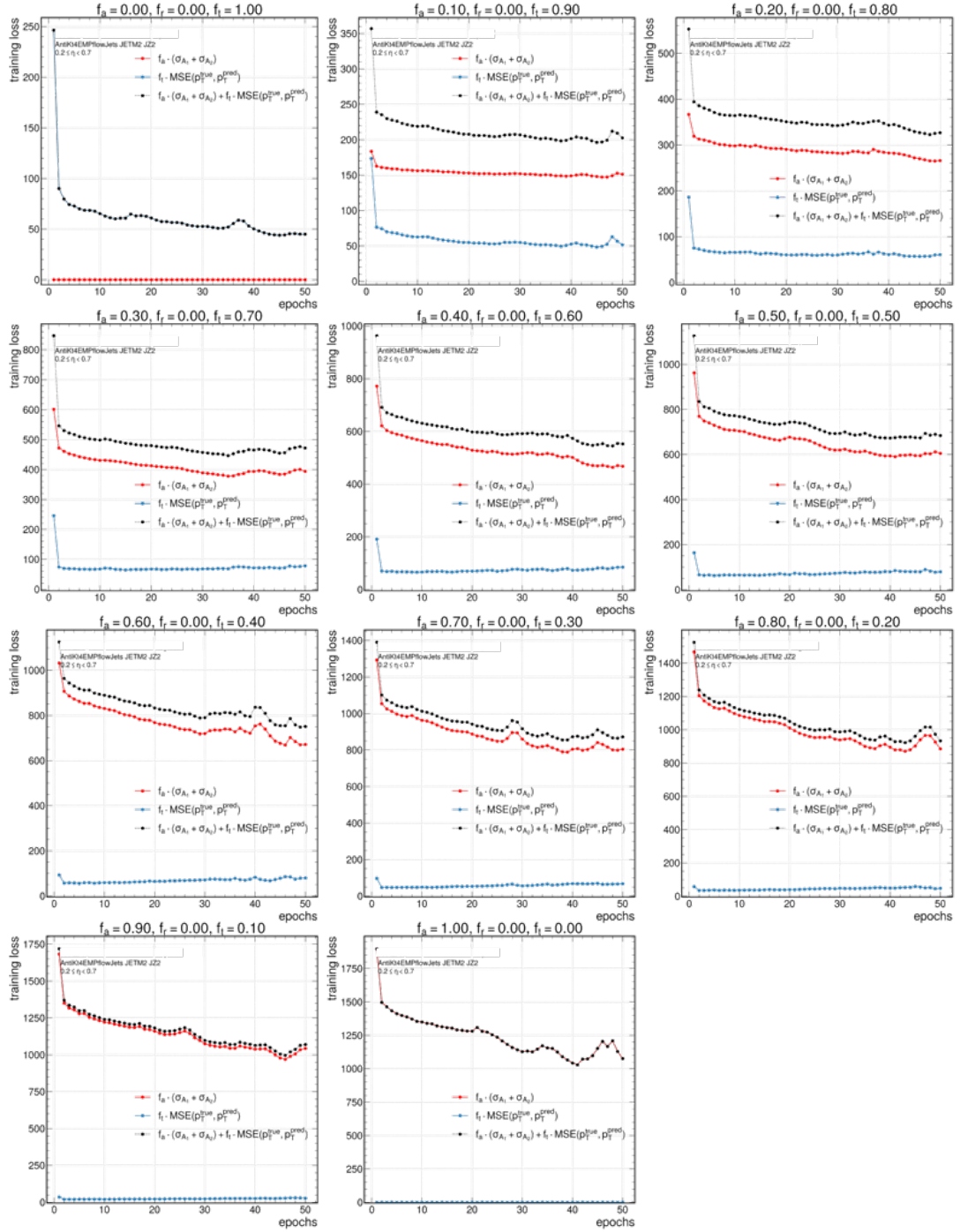
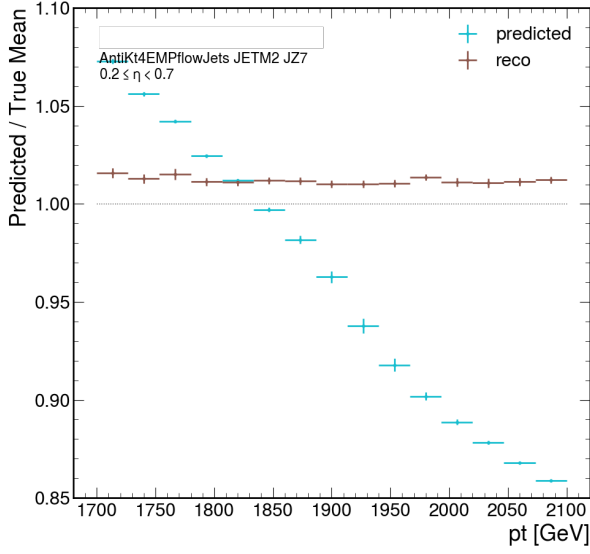


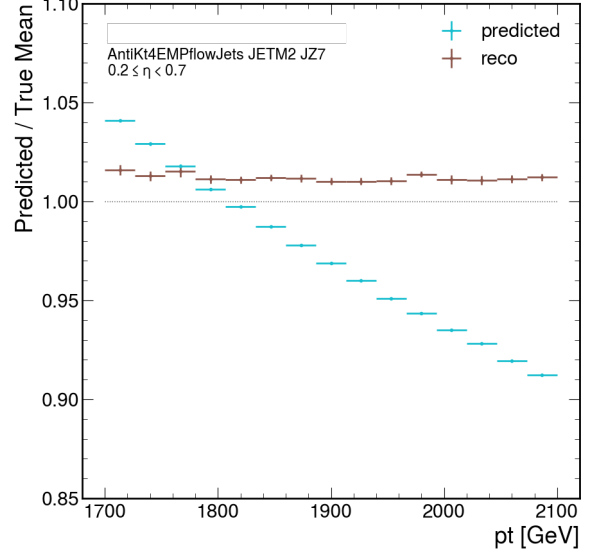
Fig. A.5.: Training loss functions and their components for each combination of hyperparameters f_a and $f_t = 1 - f_a$.

A.4 Iterative Approach: Additional Results

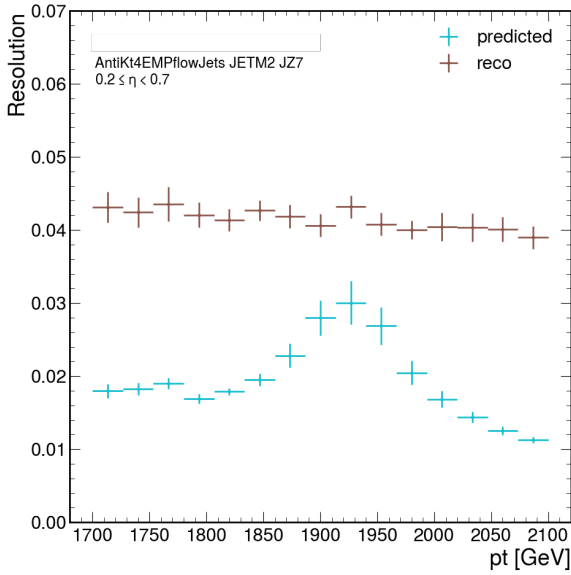
The following figures show the results obtained with the iterative approach presented in Section 9.4. For this, the partially supervised approach is used, with the loss function from Eq.(9.2). Figure A.6 and Figure A.7 show the JES and JER for the hyperparameters $f_a = f_t = 0.5$ and the JETM2 JZ7 or JZ2 samples, respectively. Similarly, Figure A.8 and Figure A.9 show the JES and JER for the hyperparameters $f_a = 0.9, f_t = 0.1$ and the JETM2 JZ7 or JZ2 samples, respectively.



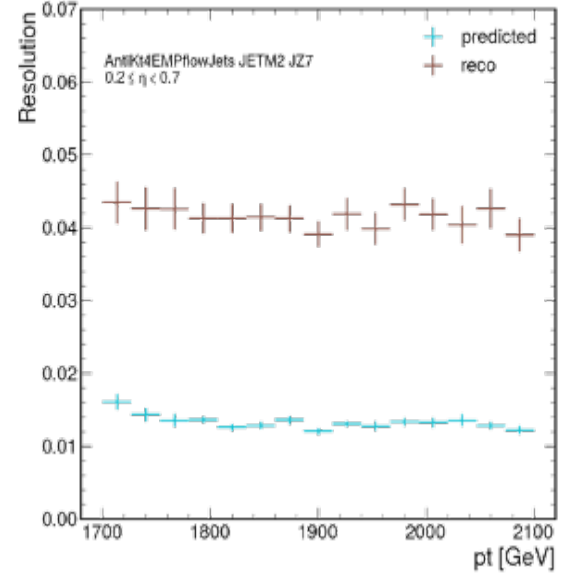
(a) Mean response per bin of ratio of predicted (cyan) or reco (brown) over true p_T after training for 50 epochs.



(b) Mean response per bin of ratio of predicted (cyan) or reco (brown) over true p_T after re-training for 20 epochs.

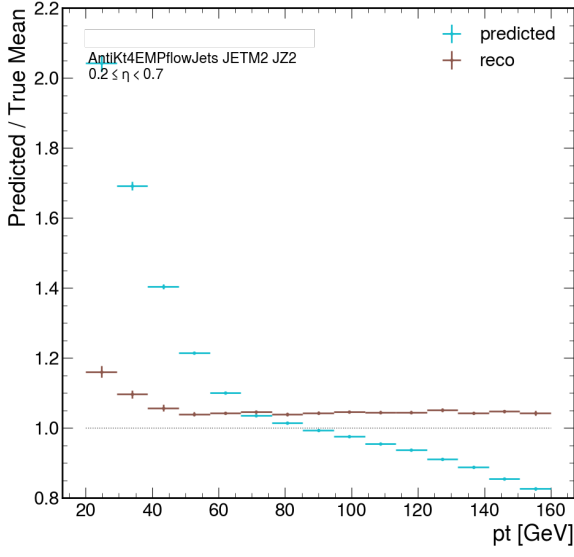


(c) Resolution per bin for predicted (cyan) and reco (brown) jet p_T after training for 50 epochs.

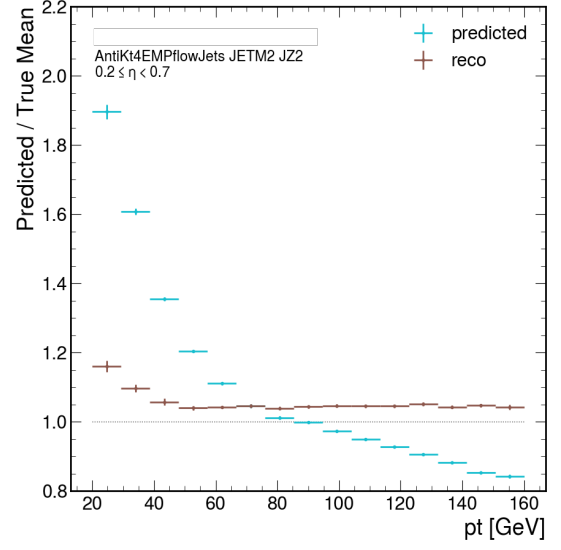


(d) Resolution per bin for predicted (cyan) and reco (brown) jet p_T after re-training for 20 epochs.

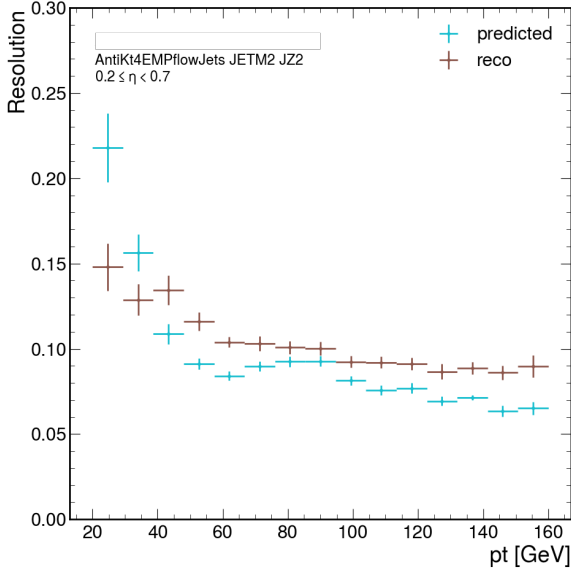
Fig. A.6.: Iterative approach with two iterations using JETM2 JZ7 samples with $f_a = f_t = 0.5$. The left two plots visualise the JES and JER after an initial training, respectively. The right plots display the JES and JER after an iteration with 20 epochs.



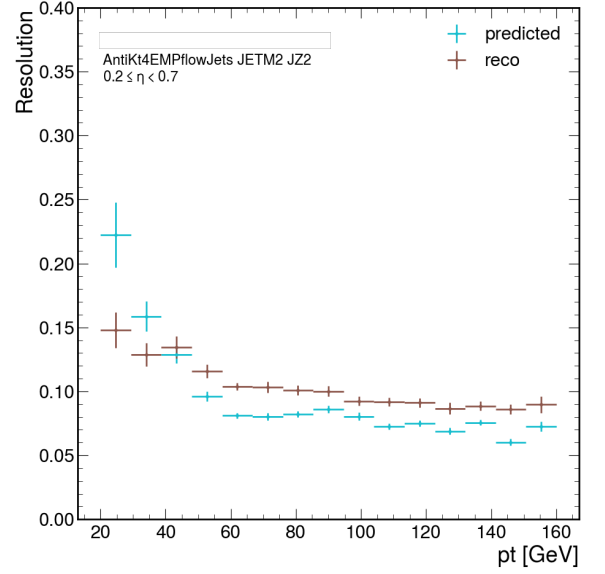
(a) Mean response per bin of ratio of predicted (cyan) or reco (brown) over true p_T after training for 50 epochs.



(b) Mean response per bin of ratio of predicted (cyan) or reco (brown) over true p_T after re-training for 20 epochs.

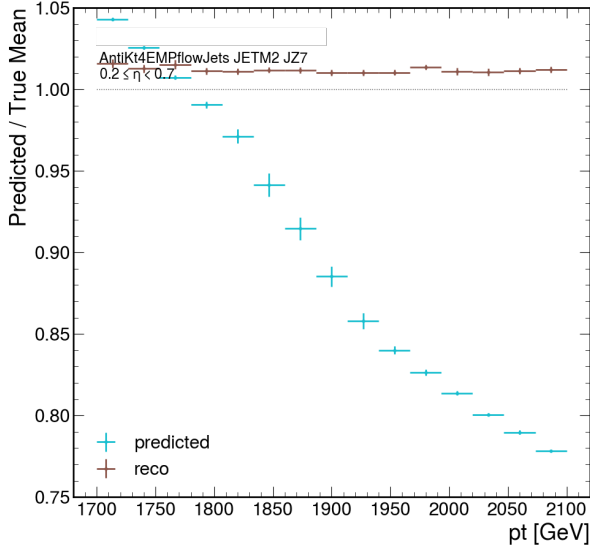


(c) Resolution per bin for predicted (cyan) and reco (brown) jet p_T after training for 50 epochs.

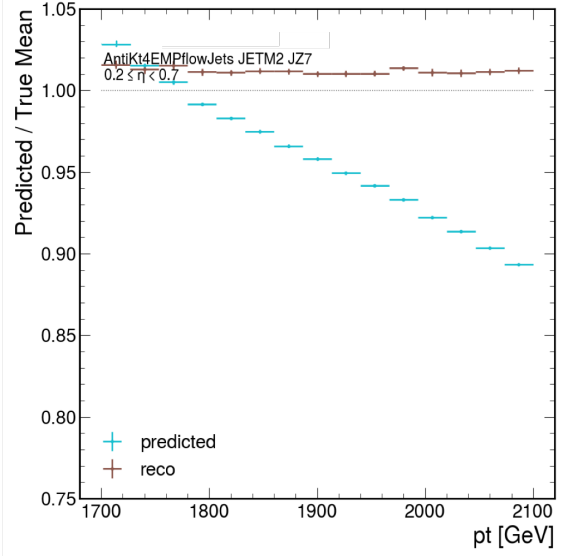


(d) Resolution per bin for predicted (cyan) and reco (brown) jet p_T after re-training for 20 epochs.

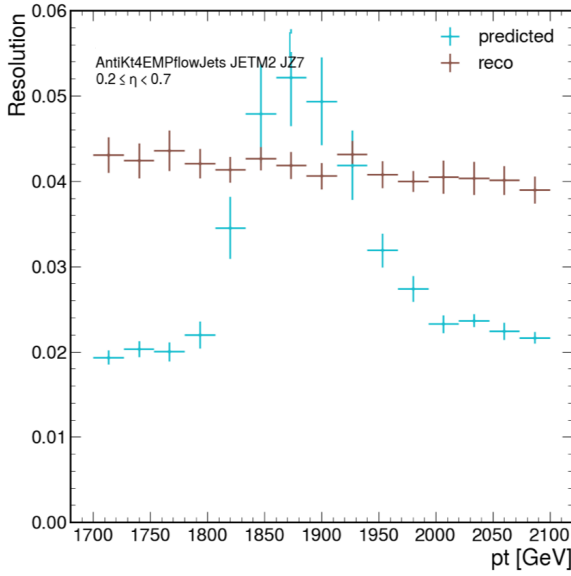
Fig. A.7.: Iterative approach with two iterations using JETM2 JZ2 samples with $f_a = f_t = 0.5$. The left two plots visualise the JES and JER after an initial training, respectively. The right plots display the JES and JER after an iteration with 20 epochs.



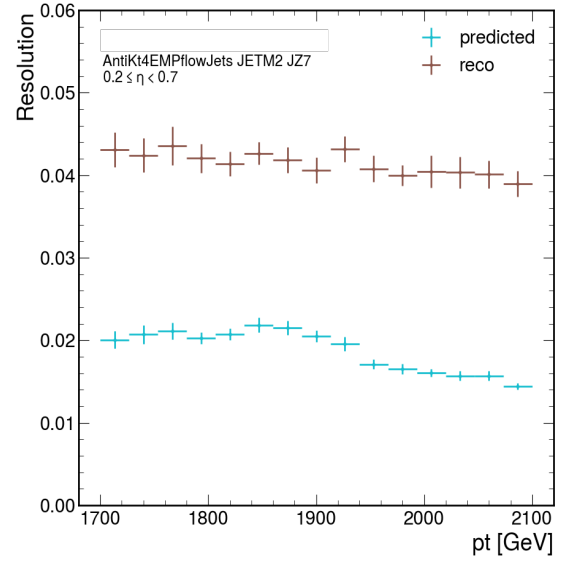
(a) Mean response per bin of ratio of predicted (cyan) or reco (brown) over true p_T after training for 50 epochs.



(b) Mean response per bin of ratio of predicted (cyan) or reco (brown) over true p_T after re-training for 20 epochs.

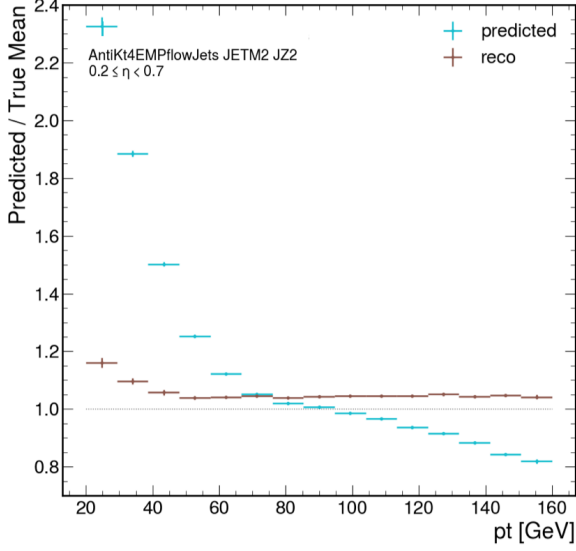


(c) Resolution per bin for predicted (cyan) and reco (brown) jet p_T after training for 50 epochs.

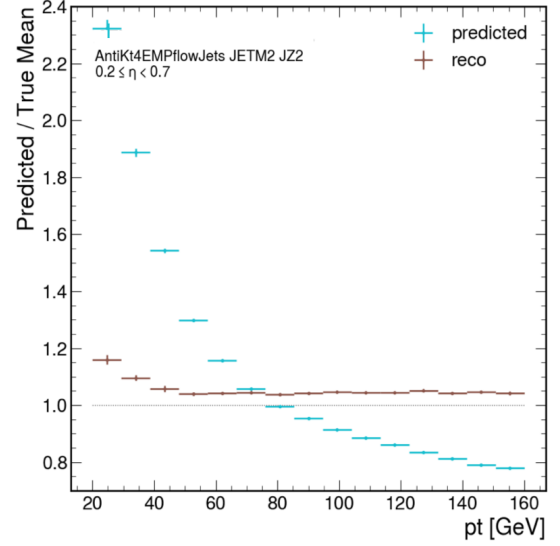


(d) Resolution per bin for predicted (cyan) and reco (brown) jet p_T after re-training for 20 epochs.

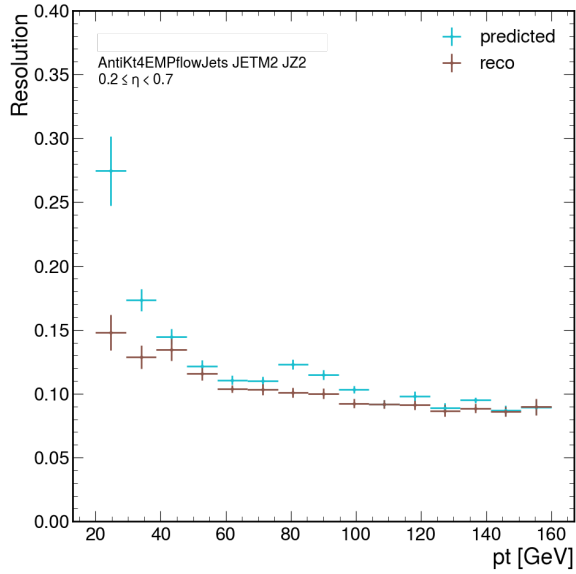
Fig. A.8.: Iterative approach with two iterations using JETM2 JZ7 samples with $f_a = 0.9$, $f_t = 0.1$. The left two plots visualise the JES and JER after an initial training, respectively. The right plots display the JES and JER after an iteration with 20 epochs.



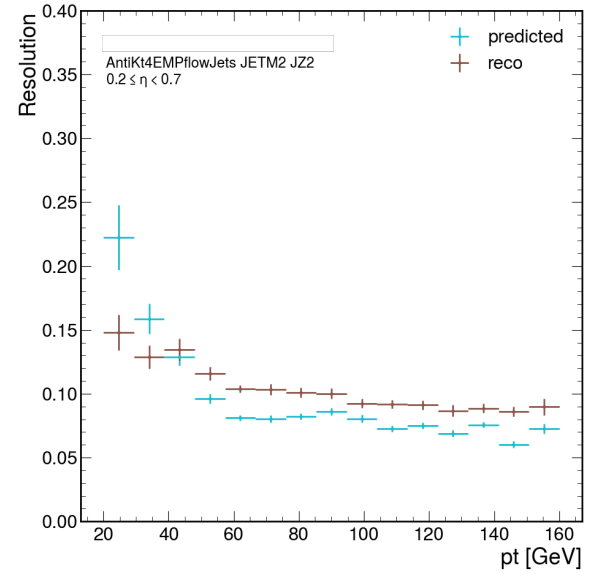
(a) Mean response per bin of ratio of predicted (cyan) or **reco** (brown) over true p_T after training for 50 epochs.



(b) Mean response per bin of ratio of predicted (cyan) or **reco** (brown) over true p_T after re-training for 20 epochs.



(c) Resolution per bin for predicted (cyan) and **reco** (brown) jet p_T after training for 50 epochs.



(d) Resolution per bin for predicted (cyan) and **reco** (brown) jet p_T after re-training for 20 epochs.

Fig. A.9.: Iterative approach with two iterations using JETM2 JZ2 samples with $f_a = 0.9$, $f_t = 0.1$. The left two plots visualise the **JES** and **JER** after an initial training, respectively. The right plots display the **JES** and **JER** after an iteration with 20 epochs.

Appendix on TSAD

B.1 Descriptions of Datasets for TSAD

The following list offers some more details on each dataset used in this paper:

1. **Credit Card data:** The dataset describes credit card transactions by European cardholders in September 2013. It has been prepared by a research collaboration of Worldline and the Machine Learning Group of Université Libre de Bruxelles and is now accessible in the form of a Kaggle challenge [260]. For confidentiality reasons, the initial features have been anonymised by applying a PCA transformation.
2. **GECCO data:** This set contains real-world data collected for ‘*GECCO 2018 Industrial Challenge*’ [261]. The time series has nine variates which describe the temperature, the Chlorine dioxide concentration, the pH value of the water, etc.
3. **IEEECIS data:** The dataset consists of e-commerce data provided by Vesta Corporation for the IEEE-CIS fraud detection kaggle challenge [262]. The fraud-dataset-benchmark repository [263] provides a cleaned-up version of this dataset with a reduced number of features and including a user ID. From there, we create multivariate user time series tracking the transactions of one individual user over time.
4. **Mars Science Laboratory (MSL) data:** The set consists of real-world data collected by sensors of NASA’s Mars Rover [242]. We only kept the sequences (A-4, C-2, T-1) that were non-trivial, as mentioned in [221].
5. **Soil Moisture Active Passive (SMAP) data:** A real-world dataset that has been collected by the NASA satellite SMAP [242].
6. **Server Machine Dataset (SMD):** The dataset contains real-world data from an internet company, which has been collected over five weeks, and incident reports have been labelled by experts [231]. It contains information like the resources used by individual machines in a cluster. Based on comments from [225, 243], we only include four individual time series with less trivial anomalies (namely sequences ‘machine-1-1’, ‘machine-2-1’, ‘machine-3-2’, ‘machine-3-7’).

7. Secure Water Treatment (SWaT) data: It contains real data collected by a water treatment plant over 11 days of continuous operation: seven under normal operation and four days with attack scenarios [257]. We use the updated version, which excludes the first 30 minutes during which the water tank is filled. Since some algorithms struggle with the full-sized dataset, we use a second version of the set that has been down sampled to one tenth of its initial volume.
8. SWaT_1D data: A univariate time series based on SWaT data, which has been prepared and provided for an anomaly detection tutorial [259].
9. Hexagon ML/UCR Time Series Anomaly Archive (UCR) data: The dataset contains multiple one-dimensional time series that are based on real data. We focus on time series where anomalies from natural sources (the InternalBleeding and ECG datasets) are carefully introduced in those time series [243].
10. Water Distribution (WADI) data: Similarly to SWaT, WADI contains data from 123 sensors and actuators of a real water plant with 14 days of normal operation and two days with various attack scenarios [258]. For efficiency reasons, the dataset has been downsampled to one-tenth of its initial volume.

B.2 iTransformer Model Configurations

To determine the best configuration for the iTransformer model in the context of reconstruction-based and forecasting-based anomaly detection, we relate the window size W to the step size S and to the internal model size M . For all other parameters, such as batch size (fixed at 32), learning rate (fixed at 10^{-4}) and training epochs (fixed at 10), we stick to the default values given in the iTransformer publication [256]. For the other parameters, we use this parameter search as an opportunity to investigate the relations between a model's internal size and the window and step size that determine how the model reads the data.

We assume that for a specific dataset, the users have some domain knowledge and can estimate the average anomaly length a and the minimal anomaly length b . This estimation can be based on a small, labelled test set, as that is needed for the evaluation of the anomaly detection method any ways. For our case, we use the values provided in Table 13.1.

Next, for every dataset, we determine a big and a small window size W_+ and W_- , respectively. First, W_+ is chosen as the multiple of 50 that is greater than the average anomaly length of the dataset. Second, W_- is set to the maximum value between $\lceil a/2 \rceil$, where the square brackets denote rounding half up to the nearest integer, and 10. The lower bound of 10 is imposed because smaller time windows start to undermine the sequential nature of the data. Finally, we also use $W_{default} = 96$, which is based on the iTransformer paper. For reconstruction-based anomaly detection, there are again two options $S_+ = \lceil W/2 \rceil$ and $S_- = \lceil W/10 \rceil$ for the step size. For forecasting-based anomaly detection, we set $S = 1$ as typically done for forecasting-based models [105]. We further fix $M = 2$ for efficiency reasons.

Last, the model internal size M (denoted as d_{model} in the iTransformer paper) can also take two different values, either it is set equal to the window size, i.e. $M_+ = W$, or to a fifth of the window size, i.e. $M_- = \lceil W/5 \rceil$.

This approach leaves us with twelve possible iTransformer model configurations for reconstruction-based anomaly detection and six configurations for forecasting-based anomaly detection, and this for each dataset. Each configuration is repeated with five different random seeds to gauge the model's robustness. We also use this parameter search to compare the impact of the different anomaly label extraction strategies presented in Section 13.3.

For each parameter configuration and each of the three anomaly labelling methods, the MCC score is computed, and the configuration with the highest MCC score is selected. If the best MCC score over five repetitions plus or minus

its standard deviation overlaps with the [MCC](#) score from other configurations, the configuration with the smallest model size is chosen. Next, if there are still multiple optimal configurations remaining, the one with the smallest step size is taken. This typically also coincides with the configuration with the smaller standard deviation, i.e. the more robust choice.

The final iTransformer configuration choices on each dataset are given below: Table [B.1](#) lists the parameter configurations for reconstruction-based anomaly detection and Table [B.2](#) for forecasting-based anomaly detection.

Tab. B.1.: Best iTransformer configurations for reconstruction-based anomaly detection with window size W , step size S and internal model size M for each dataset. For the univariate time series SWaT_1D and UCR, all anomaly label extraction methods yield the same results.

Dataset	Best MCC Score	W	S	M	Anomaly Label Method
Credit Card	0.241 ± 0.019	10	1	2	local (incl. OR)
GECCO	0.760 ± 0.004	12	1	2	local (incl. OR)
IEEEECIS	0.629 ± 0.032	10	1	10	local (incl. OR)
MSL	0.935 ± 0.011	96	10	96	local (maj. voting)
SMAP	0.826 ± 0.091	15	2	15	local (maj. voting)
SMD	0.967 ± 0.010	350	35	70	local (incl. OR)
SWaT	0.964 ± 0.004	51	5	10	local (maj. voting)
SWaT_1D	0.805 ± 0.006	10	5	2	-
UCR	0.775 ± 0.010	10	5	2	-
WADI	0.825 ± 0.024	44	4	44	global

Tab. B.2.: Best iTransformer configurations for forecasting-based anomaly detection with window size W for each dataset, while the step size $S = 1$ and internal model size $M = 2$ are fixed. For the univariate time series SWaT_1D and UCR, all anomaly label extraction methods yield the same results.

Dataset	Best MCC Score	W	Anomaly Label Method
Credit Card	0.381 ± 0.001	96	local (incl. OR)
GECCO	0.806 ± 0.029	12	local (incl. OR)
IEEEECIS	0.456 ± 0.008	10	local (incl. OR)
MSL	0.866 ± 0.001	10	global
SMAP	0.671 ± 0.002	15	global
SMD	0.913 ± 0.000	10	local (incl. OR)
SWaT	0.824 ± 0.052	51	global
SWaT_1D	0.737 ± 0.050	50	-
UCR	0.305 ± 0.274	96	-
WADI	0.676 ± 0.056	44	global

B.3 Results with Alternative Loss Functions

We study the influence of changing the loss function on the example of the reconstruction-based iTransformer. Namely, we compare the anomaly detection performance using three different training losses: MSE, Huber, and Soft-DTW. Regardless of the training loss, during testing, the anomaly score is always computed with the MSE. This choice is motivated by two reasons. First, models are trained with the Huber loss and evaluated with the MSE, because the MSE is more sensitive to outliers, which should be avoided during training but not during testing. Second, the Soft-DTW loss does not produce an element-wise loss that could serve as an anomaly score. For each dataset, the iTransformer model uses the optimal parameter configuration determined previously and is trained over ten epochs.

The results when training with each of the three loss functions are presented in Table B.3.

Tab. B.3.: MCC scores (averaged over five random seeds) for reconstruction-based iTransformer using either the mean-squared error (MSE), Huber loss or the soft dynamic time warping (Soft-DTW) loss during training and the MSE during testing. For the univariate time series SWaT_1D and UCR, all anomaly label extraction methods yield the same results.

Dataset	Loss Function	Best MCC	Anomaly Label Method
Credit Card	MSE	0.241 $\pm$ 0.019	local (incl. OR)
	Huber	0.220 $\pm$ 0.012	local (incl. OR)
	Soft-DTW	0.230 $\pm$ 0.021	local (incl. OR)
GECCO	MSE	0.760 $\pm$ 0.004	local (incl. OR)
	Huber	0.767 $\pm$ 0.010	local (incl. OR)
	Soft-DTW	0.774 $\pm$ 0.018	local (incl. OR)
IEEEECIS	MSE	0.629 $\pm$ 0.032	local (incl. OR)
	Huber	0.532 $\pm$ 0.129	local (incl. OR)
	Soft-DTW	0.567 $\pm$ 0.071	local (incl. OR)
MSL	MSE	0.935 $\pm$ 0.011	local (maj. voting)
	Huber	0.938 $\pm$ 0.006	local (maj. voting)
	Soft-DTW	0.933 $\pm$ 0.009	local (maj. voting)
SMAP	MSE	0.826 $\pm$ 0.091	local (maj. voting)
	Huber	0.777 $\pm$ 0.066	local (maj. voting)
	Soft-DTW	0.714 $\pm$ 0.073	local (incl. OR)
SMD	MSE	0.967 $\pm$ 0.010	local (incl. OR)
	Huber	0.956 $\pm$ 0.019	local (incl. OR)
	Soft-DTW	0.955 $\pm$ 0.020	local (incl. OR)
SWaT	MSE	0.964 $\pm$ 0.004	local (maj. voting)
	Huber	0.967 $\pm$ 0.006	local (maj. voting)
	Soft-DTW	0.963 $\pm$ 0.020	local (maj. voting)
SWaT_1D	MSE	0.810 $\pm$ 0.000	all equal
	Huber	0.806 $\pm$ 0.002	all equal
	Soft-DTW	0.806 $\pm$ 0.005	all equal
UCR	MSE	0.775 $\pm$ 0.010	all equal
	Huber	0.716 $\pm$ 0.115	all equal
	Soft-DTW	0.758 $\pm$ 0.041	all equal
WADI	MSE	0.825 $\pm$ 0.024	global
	Huber	0.847 $\pm$ 0.004	global
	Soft-DTW	0.774 $\pm$ 0.051	global

B.4 Model Configurations

For TranAD, we use the optimal configuration established in the corresponding publication (window size $W = 10$, step size $S = 1$, internal model size $M = 2 \cdot N$, where N is the number of variates, and we proceed similarly for USAD (window size $W = 10$, step size $S = 5$, latent dimension = 5) [224] and the AnomalyTransformer (window size $W = 10$, step size $S = 1$, latent dimension = 512), where we had to reduce the window size for more efficient memory usage. For the vanilla Transformer, we use the optimal parameter configuration from the reconstruction-based iTransformer in Section 13.5.1. The only difference between the two models lies in the embedding, which is either *normal* or *inverted*.

The total numbers of trainable parameters per model and dataset are listed in Table B.4. They often depend on the number of variates N included per dataset and differ remarkably across models.

Tab. B.4.: Number of variates N and trainable parameters for forecasting-based, reconstruction-based iTransformer, vanilla Transformer, AnomalyTransformer, TranAD and USAD model on various datasets.

Data	Model	N	# Trainable Parameters	Data	Model	N	# Trainable Parameters
Credit Card	USAD	29	16K	SMD	USAD	38	20K
	AnomalyTransformer	29	4.8M		AnomalyTransformer	38	4.9M
	TranAD	29	76K		TranAD	30	81K
	Transformer	29	4K		Transformer	38	1.6M
	iTransformer-fc	29	3K		iTransformer-fc	38	3K
	iTransformer-reco	29	111K		iTransformer-reco	38	1.6M
GECCO	USAD	9	6K	SWaT	USAD	51	27K
	AnomalyTransformer	9	4.8M		AnomalyTransformer	51	4.9M
	TranAD	9	9K		TranAD	30	81K
	Transformer	9	3K		Transformer	51	58K
	iTransformer-fc	9	2K		iTransformer-fc	51	4K
	iTransformer-reco	9	3K		iTransformer-reco	51	56K
IEEECGIS	USAD	30	16k	SWaT_1D	USAD	1	2K
	AnomalyTransformer	30	4.8M		AnomalyTransformer	1	4.8M
	TranAD	30	81K		TranAD	1	400
	Transformer	30	39K		Transformer	1	3K
	iTransformer-fc	30	3K		iTransformer-fc	1	1K
	iTransformer-reco	30	37K		iTransformer-reco	1	12K
MSL	USAD	55	29K	UCR	USAD	1	2K
	AnomalyTransformer	55	4.9M		AnomalyTransformer	1	4.8M
	TranAD	30	81K		TranAD	1	400
	Transformer	55	4.2M		Transformer	1	3K
	iTransformer-fc	55	4K		iTransformer-fc	1	2K
	iTransformer-reco	55	28M		iTransformer-reco	1	205K
SMAP	USAD	25	14K	WADI	USAD	30	16K
	AnomalyTransformer	25	4.8M		AnomalyTransformer	30	4.8M
	TranAD	25	57K		TranAD	30	81K
	Transformer	25	74K		Transformer	30	527K
	iTransformer-fc	25	2K		iTransformer-fc	30	3K
	iTransformer-reco	25	64K		iTransformer-reco	30	2.3M

B.5 Additional Results for TSAD on Lorenzetti Data

Table B.5 presents a complete overview over all Lorenzetti test datasets with varying anomaly types and rates, and how TSAD methods performed on these. In these datasets, no pileup was added (i.e. $\langle\mu\rangle = 0$).

Tab. B.5.: MCC scores (averaged over five random seeds) for reconstruction-based iTransformer, TranAD and USAD model on datasets generated with Lorenzetti, without pileup and for varying anomaly types and anomaly rates. The baseline is deterministic, so only a single MCC score is reported.

Anomaly Type	Pileup $\langle\mu\rangle$	Anomaly Rate [%]	Model	Best MCC Score	Anomaly Label Method
Increased Noise ($f_N = 5$)	0	0.32	Baseline	0.000	local (maj. voting)
			iTransformer	0.000 ± 0.000	global/local (maj. voting)
			TranAD	0.012 ± 0.008	local (maj. voting)
			USAD	0.001 ± 0.009	local (incl. OR)
Increased Noise ($f_N = 10$)	0	0.32	Baseline	0.000	local (maj. voting)
			iTransformer	0.000 ± 0.000	local (maj. voting)
			TranAD	0.009 ± 0.005	local (incl. OR)
			USAD	0.004 ± 0.008	local (incl. OR)
Increased Noise ($f_N = 20$)	0	0.32	Baseline	0.019	local (incl. OR)
			iTransformer	0.000 ± 0.000	local (maj. voting)
			TranAD	0.014 ± 0.008	local (incl. OR)
			USAD	0.000 ± 0.000	local (maj. voting)
Increased Noise ($f_N = 20$)	0	1.03	Baseline	0.007	local (incl. OR)
			iTransformer	0.000 ± 0.000	local (maj. voting)
			TranAD	0.011 ± 0.018	local (incl. OR)
			USAD	0.003 ± 0.009	local (incl. OR)
Increased Noise ($f_N = 20$)	0	22.01	Baseline	0.316	global
			iTransformer	0.041 ± 0.008	global
			TranAD	0.127 ± 0.004	local (maj. voting)
			USAD	0.186 ± 0.026	local (maj. voting)
Increased Noise ($f_N = 50$)	0	1.03	Baseline	0.043	global
			iTransformer	0.042 ± 0.008	global
			TranAD	0.076 ± 0.000	global
			USAD	0.049 ± 0.016	global

Table B.6 lists the same information but for test datasets with pileup characterised by $\langle\mu\rangle = 40$. The TSAD models in this case were trained on a different training set that was subjected to the same pileup conditions.

Tab. B.6.: MCC scores (averaged over five random seeds) for reconstruction-based iTransformer, TranAD and USAD model on datasets generated with Lorenzetti, without pileup and for varying anomaly types and anomaly rates. The baseline is deterministic, so only a single MCC score is reported.

Anomaly Type	Pileup $\langle\mu\rangle$	Anomaly Rate [%]	Model	Best MCC Score	Anomaly Label Method
Increased Noise ($f_N = 5$)	40	0.32	Baseline	0.019	local (incl. OR)
			iTransformer	0.004 ± 0.013	global
			TranAD	0.007 ± 0.007	local (incl. OR)
			USAD	0.001 ± 0.003	local (incl. OR)
Increased Noise ($f_N = 10$)	40	0.32	Baseline	0.019	local (incl. OR)
			iTransformer	0.011 ± 0.020	global
			TranAD	0.008 ± 0.003	local (incl. OR)
			USAD	0.001 ± 0.009	global
Increased Noise ($f_N = 20$)	40	0.32	Baseline	0.019	local (incl. OR)
			iTransformer	0.000 ± 0.000	local (maj. voting)
			TranAD	0.003 ± 0.004	local (incl. OR)
			USAD	0.001 ± 0.003	local (incl. OR)
Increased Noise ($f_N = 20$)	40	1.03	Baseline	0.000	local (maj. voting)
			iTransformer	0.000 ± 0.000	local (maj. voting)
			TranAD	-0.002 ± 0.000	all equal
			USAD	0.000 ± 0.000	local (maj. voting)
Increased Noise ($f_N = 20$)	40	22.01	Baseline	0.287	global
			iTransformer	0.036 ± 0.008	global
			TranAD	0.088 ± 0.021	local (maj. voting)
			USAD	0.114 ± 0.064	local (maj. voting)
Increased Noise ($f_N = 50$)	40	1.03	Baseline	0.019	local (incl. OR)
			iTransformer	0.000 ± 0.000	local (maj. voting)
			TranAD	0.002 ± 0.000	local (incl. OR)
			USAD	0.005 ± 0.007	local (incl. OR)

Bibliography

- [1]Ilyas Toumlilt. “Colony: A Hybrid Consistency System for Highly-Available Collaborative Edge Computing”. PhD thesis. Sorbonne Université, 2021 (cit. on p. vii).
- [2]Julian Schuhmacher, Laura Boggia, Vasilis Belis, et al. “Unravelling Physics beyond the Standard Model with Classical and Quantum Anomaly Detection”. In: *Machine Learning: Science and Technology* 4.4 (Nov. 16, 2023), p. 045031 (cit. on p. xxv).
- [3]Laura Boggia, Rafael Teixeira de Lima, and Bogdan Malaescu. *Benchmarking Unsupervised Strategies for Anomaly Detection in Multivariate Time Series*. June 25, 2025. arXiv: [2506.20574](https://arxiv.org/abs/2506.20574) [cs]. URL: <http://arxiv.org/abs/2506.20574> (visited on June 26, 2025). Pre-published (cit. on pp. xxv, 115).
- [4]Laura Boggia, Carlos Cocha, Fotis Giasemis, et al. *Review of Machine Learning for Real-Time Analysis at the Large Hadron Collider Experiments ALICE, ATLAS, CMS and LHCb*. June 17, 2025. arXiv: [2506.14578](https://arxiv.org/abs/2506.14578) [hep-ex]. URL: <http://arxiv.org/abs/2506.14578> (visited on June 18, 2025). Pre-published (cit. on p. xxv).
- [5]Laura Boggia. “Synthetic Data Generation with Lorenzetti for Time Series Anomaly Detection in High-Energy Physics Calorimeters”. Flash talk. EuCaif Conference 2025 (Cagliari). June 17, 2025 (cit. on pp. xxv, 170).
- [6]Laura Boggia and Bogdan Malaescu. *Synthetic Data Generation with Lorenzetti for Time Series Anomaly Detection in High-Energy Physics Calorimeters*. Sept. 9, 2025. arXiv: [2509.07451](https://arxiv.org/abs/2509.07451) [hep-ex]. URL: <http://arxiv.org/abs/2509.07451> (visited on Sept. 11, 2025). Pre-published (cit. on pp. xxv, 170).
- [7]Laura Boggia. “Unsupervised Anomaly Detection in Multivariate Time Series Using Public Benchmarks and Synthetic Data from Lorenzetti”. Workshop. UQ4ML, COMETA Workshop on Uncertainty Quantification for Machine Learning (Saclay). Sept. 17, 2025 (cit. on p. xxv).
- [8]Laura Boggia. “Time Series Anomaly Detection for High Energy Physics and Fraud Detection”. SMARTHEP Meets Industry (CERN). May 23, 2025 (cit. on p. xxvi).
- [9]ATLAS Collaboration. “Observation of a New Particle in the Search for the Standard Model Higgs Boson with the ATLAS Detector at the LHC”. In: *Physics Letters B* 716.1 (Sept. 2012), pp. 1–29. arXiv: [1207.7214](https://arxiv.org/abs/1207.7214) [hep-ex] (cit. on pp. 1, 5, 17, 51).
- [10]CMS Collaboration. “Observation of a New Boson at a Mass of 125 GeV with the CMS Experiment at the LHC”. In: *Physics Letters B* 716.1 (Sept. 2012), pp. 30–61. arXiv: [1207.7235](https://arxiv.org/abs/1207.7235) [hep-ex] (cit. on pp. 1, 5, 17, 51).
- [11]X. Fan, T. G. Myers, B. A. D. Sukra, and G. Gabrielse. “Measurement of the Electron Magnetic Moment”. In: *Physical Review Letters* 130.7 (Feb. 13, 2023), p. 071801 (cit. on p. 4).

- [12]Cush. *File:Standard Model of Elementary Particles.Svg*. Wikipedia. Sept. 17, 2019.
URL: https://commons.wikimedia.org/wiki/File:Standard_Model_of_Elementary_Particles.svg (visited on May 30, 2025) (cit. on p. 5).
- [13]Mark Thomson. *Modern Particle Physics*. Cambridge University Press, Sept. 2013 (cit. on pp. 6–9).
- [14]Franz Gross, Eberhard Klempt, Stanley J. Brodsky, et al. “50 Years of Quantum Chromodynamics”. In: *The European Physical Journal C* 83.12 (Dec. 12, 2023), p. 1125. arXiv: [2212.11107](https://arxiv.org/abs/2212.11107) [hep-ph] (cit. on pp. 7, 8, 15).
- [15]Gavin P. Salam. *Elements of QCD for Hadron Colliders*. Jan. 10, 2011. arXiv: [1011.5131](https://arxiv.org/abs/1011.5131) [hep-ex, physics:hep-ph]. URL: <http://arxiv.org/abs/1011.5131> (visited on Dec. 7, 2023). Pre-published (cit. on p. 7).
- [16]Peter Skands. “Introduction to QCD”. In: *Searching for New Physics at Small and Large Scales*. Nov. 2013, pp. 341–420. arXiv: [1207.2389](https://arxiv.org/abs/1207.2389) [hep-ph, physics:hep-th] (cit. on p. 7).
- [17]Michelangelo L. Mangano. “Introduction to QCD”. In: *1998 European School of High-Energy Physics*. 1998, pp. 53–97 (cit. on p. 7).
- [18]Particle Data Group, R L Workman, V D Burkert, et al. “Review of Particle Physics”. In: *Progress of Theoretical and Experimental Physics* 2022.8 (Aug. 8, 2022), p. 083C01 (cit. on p. 8).
- [19]B. Andersson, G. Gustafson, G. Ingelman, and T. Sjöstrand. “Parton Fragmentation and String Dynamics”. In: *Physics Reports* 97.2 (July 1, 1983), pp. 31–145 (cit. on pp. 8, 15).
- [20]Paul Langacker. *The Standard Model and Beyond*. 2nd ed. Boca Raton: CRC Press, June 26, 2017. 650 pp. (cit. on p. 9).
- [21]Simone Marzani. “Jets at Colliders”. In: *Instrumentation and Techniques in High Energy Physics*. WORLD SCIENTIFIC, Aug. 27, 2024, pp. 179–211 (cit. on pp. 10, 11).
- [22]Dilia Maria Portillo Quintero. “Introduction Jet Inputs and MC Calibration”. Hadronic Calibration Workshop (Ottawa). Aug. 26, 2024 (cit. on p. 11).
- [23]Matteo Cacciari, Gavin P. Salam, and Gregory Soyez. “The Anti-Kt Jet Clustering Algorithm”. In: *Journal of High Energy Physics* 2008.04 (Apr. 2008), p. 063 (cit. on pp. 10, 12, 34).
- [24]Matteo Cacciari, Gavin P. Salam, and Gregory Soyez. “FastJet User Manual”. In: *The European Physical Journal C* 72.3 (Mar. 2012), p. 1896. arXiv: [1111.6097](https://arxiv.org/abs/1111.6097) [hep-ph] (cit. on p. 11).
- [25]Ryan Atkin. “Review of Jet Reconstruction Algorithms”. In: *Journal of Physics: Conference Series* 645 (Oct. 15, 2015), p. 012008 (cit. on pp. 11, 12).
- [26]ATLAS Collaboration. *Standard Model Summary Plots February 2022*. ATL-PHYS-PUB-2022-009. Geneva: CERN, 2022 (cit. on p. 13).
- [27]Andy Buckley, Jonathan Butterworth, Stefan Gieseke, et al. “General-Purpose Event Generators for LHC Physics”. In: *Physics Reports* 504.5 (July 2011), pp. 145–233. arXiv: [1101.2599](https://arxiv.org/abs/1101.2599) [hep-ph] (cit. on pp. 13, 15).

- [28]Torbjörn Sjöstrand. “The Pythia Event Generator: Past, Present and Future”. In: *Computer Physics Communications* 246 (Jan. 1, 2020), p. 106910 (cit. on pp. 13, 138).
- [29]Xiaoxiao Ma, Jia Wu, Shan Xue, et al. “A Comprehensive Survey on Graph Anomaly Detection with Deep Learning”. In: *IEEE Transactions on Knowledge and Data Engineering* 35.12 (Dec. 1, 2023), pp. 12012–12038. arXiv: 2106.07178 [cs] (cit. on pp. 13, 138).
- [30]Christian Bierlich, Smita Chakraborty, Nishita Desai, et al. *A Comprehensive Guide to the Physics and Usage of PYTHIA 8.3*. Mar. 22, 2022. arXiv: 2203.11601 [hep-ph]. URL: <http://arxiv.org/abs/2203.11601> (visited on June 13, 2025). Pre-published (cit. on p. 14).
- [31]Stefan Hoeche. “Introduction to Parton-Shower Event Generators”. In: (Nov. 14, 2014) (cit. on pp. 14, 15).
- [32]Bryan Webber. “Parton Shower Monte Carlo Event Generators”. In: *Scholarpedia* 6.12 (Dec. 8, 2011), p. 10662 (cit. on pp. 14, 15).
- [33]Bo Andersson. “The Lund Model”. In: *Nuclear Physics A* 461.1 (Jan. 5, 1987), pp. 513–520 (cit. on p. 15).
- [34]S. Agostinelli, J. Allison, K. Amako, et al. “Geant4—a Simulation Toolkit”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 506.3 (July 1, 2003), pp. 250–303 (cit. on pp. 16, 138).
- [35]CERN. *The History of CERN | Timeline*. Web.Cern.Ch. 2025. URL: <https://timeline.web.cern.ch/timeline-header/89#366> (visited on Apr. 12, 2025) (cit. on p. 17).
- [36]G. Baur, G. Boero, A. Brauksiepe, et al. “Production of Antihydrogen”. In: *Physics Letters B* 368.3 (Feb. 1, 1996), pp. 251–258 (cit. on p. 17).
- [37]G. B. Andresen, M. D. Ashkezari, M. Baquero-Ruiz, et al. “Trapped Antihydrogen”. In: *Nature* 468.7324 (Dec. 2010), pp. 673–676 (cit. on p. 17).
- [38]E. K. Anderson, C. J. Baker, W. Bertsche, et al. “Observation of the Effect of Gravity on the Motion of Antimatter”. In: *Nature* 621.7980 (Sept. 2023), pp. 716–722 (cit. on p. 17).
- [39]Tim Berners-Lee. “Information Management: A Proposal”. In: (Mar. 1989) (cit. on p. 17).
- [40]CERN. *Worldwide LHC Computing Grid*. 2025. URL: <https://wlcg-public.web.cern.ch/> (visited on Apr. 12, 2025) (cit. on p. 17).
- [41]WLCG Collaboration. “The Worldwide LHC Computing Grid Input to the ESPP24-26”. In: (Mar. 17, 2025) (cit. on p. 17).
- [42]Manuela Cirilli, Marzena Lapka, and Helen Dixon-Altaber. *Impact of CERN Technologies: From Fundamental Research to Our Everyday Lives*. CERN, 2023 (cit. on p. 17).

- [43]Esma Mobs. *The CERN Accelerator Complex in 2019. Complexe Des Accélérateurs Du CERN En 2019*. CERN. 2019. URL: <https://cds.cern.ch/record/2684277> (cit. on p. 18).
- [44]ALICE Collaboration. “The ALICE Experiment at the CERN LHC”. In: *Journal of Instrumentation* 3.08 (Aug. 2008), S08002 (cit. on p. 18).
- [45]ATLAS Collaboration. “The ATLAS Experiment at the CERN Large Hadron Collider”. In: *Journal of Instrumentation* 3.08 (Aug. 2008), S08003 (cit. on pp. 18, 21, 24, 26, 29).
- [46]CMS Collaboration. “The CMS Experiment at the CERN LHC”. In: *Journal of Instrumentation* 3.08 (Aug. 2008), S08004 (cit. on p. 18).
- [47]LHCb Collaboration. “The LHCb Detector at the LHC”. In: *Journal of Instrumentation* 3.08 (Aug. 2008), S08005 (cit. on p. 18).
- [48]ATLAS Collaboration. “The ATLAS Experiment at the CERN Large Hadron Collider: A Description of the Detector Configuration for Run 3”. In: *Journal of Instrumentation* 19.05 (May 1, 2024), P05063. arXiv: 2305.16623 [physics] (cit. on pp. 19, 21–23, 29).
- [49]Lyndon Evans and Philip Bryant. “LHC Machine”. In: *Journal of Instrumentation* 3.08 (Aug. 2008), S08001 (cit. on p. 19).
- [50]Lyndon Evans. *The Large Hadron Collider - A Marvel of Technology*. 2nd ed. EPFL Press English Imprint, Nov. 13, 2018 (cit. on pp. 19, 21, 22, 24, 26, 50).
- [51]Ana Lopes and Melissa Loyse Perrey. *FAQ-LHC The Guide*. CERN, 2022 (cit. on p. 19).
- [52]ATLAS Collaboration. *Total Integrated Luminosity in Run 3 (13.6 TeV p-p Data Only)*. Public ATLAS Luminosity Results for Run-3 of the LHC. 2025. URL: <https://twiki.cern.ch/twiki/bin/view/AtlasPublic/LuminosityPublicResultsRun3> (visited on May 21, 2025) (cit. on p. 19).
- [53]ATLAS Collaboration. *Interactions per Crossing 2011-2024*. Public ATLAS Luminosity Results for Run-3 of the LHC. 2024. URL: <https://twiki.cern.ch/twiki/bin/view/AtlasPublic/LuminosityPublicResultsRun3> (visited on May 20, 2025) (cit. on p. 20).
- [54]ATLAS Collaboration. “Determination of Jet Calibration and Energy Resolution in Proton-Proton Collisions at $\sqrt{s} = 8$ TeV Using the ATLAS Detector”. In: *The European Physical Journal C* 80.12 (Dec. 2020), p. 1104. arXiv: 1910.04482 [hep-ex] (cit. on pp. 20, 35, 39, 88).
- [55]ATLAS Collaboration. “Jet Reconstruction and Performance Using Particle Flow with the ATLAS Detector”. In: *The European Physical Journal C* 77.7 (July 2017), p. 466 (cit. on pp. 20, 32, 33, 70).
- [56]ATLAS Collaboration. *ATLAS HL-LHC Computing Conceptual Design Report*. Geneva: CERN, 2020 (cit. on pp. 20, 51).
- [57]O. Brüning and L. Rossi. “Chapter 1: High-Luminosity Large Hadron Collider”. In: *CERN Yellow Reports: Monographs* 10 (Dec. 17, 2020), pp. 1–1 (cit. on pp. 20, 51).

- [58]Mike Lamont. *Updated Schedule for CERN's Accelerators*. CERN. Oct. 4, 2024. URL: <https://home.cern/news/opinion/accelerators/updated-schedule-cerns-accelerators> (visited on May 15, 2025) (cit. on pp. 20, 51).
- [59]Riccardo Maria Bianchi. *ATLAS Experiment Schematic or Layout Illustration*. CERN. 2022. URL: <https://cds.cern.ch/record/2837191> (cit. on p. 21).
- [60]ATLAS Collaboration. *Jet Energy Scale and Resolution Measured in Proton-Proton Collisions at $\sqrt{s} = 13$ TeV with the ATLAS Detector*. Aug. 27, 2021. arXiv: 2007.02645 [hep-ex]. URL: <http://arxiv.org/abs/2007.02645> (visited on Jan. 18, 2023). Pre-published (cit. on pp. 24, 35, 71).
- [61]ATLAS Collaboration. *LArCaloPublicStableBeam2022 < AtlasPublic < TWiki*. ATLAS Experiment - Public Results. Aug. 24, 2023. URL: <https://twiki.cern.ch/twiki/bin/view/AtlasPublic/LArCaloPublicStableBeam2022> (visited on May 27, 2025) (cit. on pp. 25, 26).
- [62]M (CERN) Aleksa, W (Pittsburgh) Cleland, Y (Tokyo) Enari, et al. *ATLAS Liquid Argon Calorimeter Phase-I Upgrade: Technical Design Report*. 2013 (cit. on pp. 26, 41, 154).
- [63]Olga Chuchuk. “Data Access Optimisation at CERN and in the Worldwide LHC Computing Grid (WLCG)”. PhD thesis. Université Côte d’Azur, Feb. 19, 2024 (cit. on p. 30).
- [64]Sascha Mehlhase. *ATLAS Detector Slice (and Particle Visualisations)*. CERN. 2021. URL: <https://cds.cern.ch/record/2770815> (cit. on p. 30).
- [65]ATLAS Collaboration. *Track Reconstruction Performance of the ATLAS Inner Detector at $\sqrt{s} = 13$ TeV*. Geneva: CERN, 2015 (cit. on p. 30).
- [66]ATLAS Collaboration. *Vertex Reconstruction Performance of the ATLAS Detector at $\sqrt{s} = 13$ TeV*. Geneva: CERN, 2015 (cit. on p. 31).
- [67]ATLAS Collaboration. *Electron Efficiency Measurements with the ATLAS Detector Using the 2015 LHC Proton-Proton Collision Data*. Geneva: CERN, 2016 (cit. on p. 31).
- [68]ATLAS Collaboration. “Measurement of the Muon Reconstruction Performance of the ATLAS Detector Using 2011 and 2012 LHC Proton–Proton Collision Data”. In: *The European Physical Journal C* 74.11 (Nov. 2014), p. 3130 (cit. on p. 32).
- [69]ATLAS Collaboration. “Performance of the ATLAS Muon Triggers in Run 2”. In: *JINST* 15.CERN-EP-2020-031 (2020), P09015. arXiv: 2004.13447 (cit. on p. 32).
- [70]ATLAS Collaboration. “Improving Topological Cluster Reconstruction Using Calorimeter Cell Timing in ATLAS”. In: *The European Physical Journal C* 84.5 (May 2024), p. 455 (cit. on p. 33).
- [71]ATLAS Collaboration. *Improving Jet Substructure Performance in ATLAS Using Track-CaloClusters*. ATL-PHYS-PUB-2017-015. Geneva: CERN, 2017 (cit. on p. 33).
- [72]ATLAS Collaboration. “Optimisation of Large-Radius Jet Reconstruction for the ATLAS Detector in 13 TeV Proton–Proton Collisions”. In: *The European Physical Journal C* 81.4 (Apr. 2021), p. 334 (cit. on pp. 33, 34).

- [73]ATLAS Collaboration. *R=0.4 Jets Input Comparison and Monte Carlo Calibration with the ATLAS Detector*. Geneva: CERN, 2022 (cit. on p. 34).
- [74]Peter Berta, Martin Spousta, David W. Miller, and Rupert Leitner. “Particle-Level Pileup Subtraction for Jets and Jet Shapes”. In: *Journal of High Energy Physics* 2014.6 (June 16, 2014), p. 92 (cit. on p. 34).
- [75]Matteo Cacciari, Gavin P. Salam, and Gregory Soyez. “SoftKiller, a Particle-Level Pileup Removal Method”. In: *The European Physical Journal C* 75.2 (Feb. 6, 2015), p. 59 (cit. on p. 34).
- [76]ATLAS Collaboration. “Jet Energy Scale Measurements and Their Systematic Uncertainties in Proton-Proton Collisions at $\sqrt{s} = 13$ TeV with the ATLAS Detector”. In: *Physical Review D* 96.7 (Oct. 2017), p. 072002 (cit. on p. 35).
- [77]ATLAS Collaboration. *New Techniques for Jet Calibration with the ATLAS Detector*. Mar. 30, 2023. arXiv: 2303.17312 [hep-ex]. URL: <http://arxiv.org/abs/2303.17312> (visited on Apr. 11, 2023). Pre-published (cit. on pp. 35–37, 69).
- [78]ATLAS Collaboration. *Autoencoder-Based Time Series Anomaly Detection for ATLAS Liquid Argon Calorimeter Data Quality Monitoring*. Geneva: CERN, 2024 (cit. on pp. 39–41, 59, 60, 64, 114, 154, 155, 158).
- [79]ATLAS Collaboration. “ATLAS Data Quality Operations and Performance for 2015–2018 Data-Taking”. In: *Journal of Instrumentation* 15.04 (Apr. 2020), P04003 (cit. on pp. 39, 40).
- [80]ATLAS Collaboration. “Monitoring and Data Quality Assessment of the ATLAS Liquid Argon Calorimeter”. In: *Journal of Instrumentation* 9.07 (July 23, 2014), P07024–P07024 (cit. on pp. 39, 41–43, 154).
- [81]Martin Krivos. “Hard Probes of Heavy Ion Collisions with ATLAS”. In: *Nuclear and Particle Physics Proceedings*. QCD 22 Is the 25th International Conference on Quantum Chromodynamics 324–329 (Mar. 1, 2023), pp. 30–35 (cit. on p. 39).
- [82]*Charged-Hadron Yield Measurements in Photo-Nuclear Collisions Using 5.02 TeV Pb+Pb Data with ATLAS*. ATLAS-CONF-2023-059. Geneva: CERN, 2023 (cit. on p. 40).
- [83]ATLAS Collaboration. *LAr Data Quality Inefficiency in Run 2*. LArCaloPublicResults2015. 2023. URL: https://twiki.cern.ch/twiki/bin/view/AtlasPublic/LArCaloPublicResults2015#LAr_Mini_Noise_Bursts_Plots_for (visited on June 27, 2025) (cit. on p. 41).
- [84]Javier M. Duarte and Dylan S. Rankin. “Machine Learning for Analysis and Instrumentation in High Energy Physics”. In: *Instrumentation and Techniques in High Energy Physics*. WORLD SCIENTIFIC, Aug. 27, 2024, pp. 125–177 (cit. on pp. 44, 51, 56).
- [85]Arthur Lee Samuel. “Some Studies in Machine Learning Using the Game of Checkers”. In: *IBM Journal of Research and Development* 3.3 (July 1959), pp. 210–229 (cit. on p. 44).
- [86]Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. Adaptive Computation and Machine Learning Series. Cambridge, MA: MIT Press, 2012. 1067 pp. (cit. on pp. 44–46).

- [87]Adrien-Marie Legendre. *Nouvelles Méthodes Pour La Détermination Des Orbites Des Comètes*, Par A.-M. Legendre. 1805 (cit. on pp. 44, 45).
- [88]T. Cover and P. Hart. “Nearest Neighbor Pattern Classification”. In: *IEEE Transactions on Information Theory* 13.1 (1967), pp. 21–27 (cit. on pp. 44, 111).
- [89]Harris Drucker and Corinna Cortes. “Boosting Decision Trees”. In: *Advances in Neural Information Processing Systems*. Ed. by D. Touretzky, M. C. Mozer, and M. Hasselmo. Vol. 8. MIT Press, 1995 (cit. on p. 45).
- [90]Corinna Cortes and Vladimir Vapnik. “Support-Vector Networks”. In: *Machine Learning* 20.3 (Sept. 1, 1995), pp. 273–297 (cit. on p. 45).
- [91]F. Rosenblatt. “The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain”. In: *Psychological Review* 65.6 (1958), pp. 386–408 (cit. on p. 45).
- [92]David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. “Learning Representations by Back-Propagating Errors”. In: *Nature* 323.6088 (Oct. 1986), pp. 533–536 (cit. on pp. 45, 55, 60, 111).
- [93]Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. “Gradient-Based Learning Applied to Document Recognition”. In: *Proceedings of the IEEE* 86.11 (Nov. 1998), pp. 2278–2324 (cit. on p. 45).
- [94]Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016 (cit. on pp. 45–50, 55, 60, 66).
- [95]Andrew Ng and Tengyu Ma. “CS229 Lecture Notes”. In: *CS229 Lecture notes* 1.1 (2023) (cit. on pp. 45, 46, 48–50, 55, 66).
- [96]Murray Campbell, A. Joseph Hoane, and Feng-hsiung Hsu. “Deep Blue”. In: *Artificial Intelligence* 134.1 (Jan. 1, 2002), pp. 57–83 (cit. on p. 46).
- [97]David Silver, Aja Huang, Chris J. Maddison, et al. “Mastering the Game of Go with Deep Neural Networks and Tree Search”. In: *Nature* 529.7587 (Jan. 2016), pp. 484–489 (cit. on pp. 46, 47).
- [98]Ashish Vaswani, Noam Shazeer, Niki Parmar, et al. “Attention Is All You Need”. In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc., 2017 (cit. on pp. 46, 61, 120).
- [99]OpenAI. *Introducing ChatGPT*. Nov. 30, 2022. URL: <https://openai.com/index/chatgpt/> (visited on Mar. 19, 2025) (cit. on pp. 46, 52).
- [100]John Atkinson-Abutridy. *Large Language Models: Concepts, Techniques and Applications*. CRC Press, Oct. 17, 2024. 184 pp. (cit. on pp. 46, 66).
- [101]Yoshua Bengio, Sören Mindermann, Daniel Privitera, et al. *International AI Safety Report*. Jan. 29, 2025. arXiv: 2501.17805 [cs]. URL: <http://arxiv.org/abs/2501.17805> (visited on Mar. 28, 2025). Pre-published (cit. on p. 46).
- [102]Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. New York: Springer, 2006. 738 pp. (cit. on pp. 46, 48, 49).
- [103]Xavier Vasques. “Concepts, Libraries, and Essential Tools in Machine Learning and Deep Learning”. In: *Machine Learning Theory and Applications*. John Wiley & Sons, Ltd, Jan. 12, 2024, pp. 1–33 (cit. on p. 46).

- [104] Andrzej Maćkiewicz and Waldemar Ratajczak. “Principal Components Analysis (PCA)”. In: *Computers & Geosciences* 19.3 (Mar. 1, 1993), pp. 303–342 (cit. on p. 47).
- [105] Zahra Zamanzadeh Darban, Geoffrey I. Webb, Shirui Pan, Charu Aggarwal, and Mahsa Salehi. “Deep Learning for Time Series Anomaly Detection: A Survey”. In: *ACM Computing Surveys* (Aug. 30, 2024), p. 3691338 (cit. on pp. 47, 63, 109, 111, 112, 117–120, 185).
- [106] Simon Hirlaender, Lukas Lamminger, Sabrina Pochaba, et al. “Towards few-shot reinforcement learning in particle accelerator control”. In: *15th International Particle Accelerator Conference, Nashville, Tennessee : May 19-24, 2024, Nashville, Tennessee, USA : proceedings. Ed.: F. Pilat. 15th International Particle Accelerator Conference (IPAC 2024), Nashville, TN, USA, 19.05.2024 – 24.05.2024. 2024*, p. 1804 (cit. on p. 47).
- [107] Jan Kaiser, Chenran Xu, Annika Eichler, et al. “Reinforcement Learning-Trained Optimisers and Bayesian Optimisation for Online Particle Accelerator Tuning”. In: *Scientific Reports* 14.1 (July 8, 2024), p. 15733 (cit. on p. 47).
- [108] Olivia Jullian Parra, Julián García Pardiñas, Lorenzo Del Pianta Pérez, et al. *Human-in-the-Loop Reinforcement Learning for Data Quality Monitoring in Particle Physics Experiments*. May 24, 2024. arXiv: 2405.15508 [hep-ex]. URL: <http://arxiv.org/abs/2405.15508> (visited on May 27, 2024). Pre-published (cit. on pp. 47, 64).
- [109] DeepSeek-AI, Daya Guo, Dejian Yang, et al. *DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning*. Jan. 22, 2025. arXiv: 2501.12948 [cs]. URL: <http://arxiv.org/abs/2501.12948> (visited on Mar. 14, 2025). Pre-published (cit. on p. 48).
- [110] Alexander Radovic, Mike Williams, David Rousseau, et al. “Machine Learning at the Energy and Intensity Frontiers of Particle Physics”. In: *Nature* 560.7716 (Aug. 2018), pp. 41–48 (cit. on pp. 50, 51).
- [111] Luca Clissa, Mario Lassnig, and Lorenzo Rinaldi. “How Big Is Big Data? A Comprehensive Survey of Data Production, Storage, and Streaming in Science and Industry”. In: *Frontiers in Big Data* 6 (Oct. 19, 2023) (cit. on p. 50).
- [112] Antonella Del Rosso. *How CERN IT Keeps up with the Data Deluge*. CERN. Sept. 4, 2024. URL: <https://home.cern/news/news/computing/how-cern-it-keeps-data-deluge> (visited on Mar. 19, 2025) (cit. on p. 50).
- [113] B. Denby. “Neural Networks and Cellular Automata in Experimental High Energy Physics”. In: *Computer Physics Communications* 49.3 (June 1, 1988), pp. 429–448 (cit. on p. 51).
- [114] Bruce H. Denby, W. Badgett, S. Bianchin, et al. “Neural Networks at the Tevatron”. In: *6th International Conference on Computing in High-Energy and Nuclear Physics*. Oct. 1992 (cit. on p. 51).
- [115] Leif Lönnblad, Carsten Peterson, and Thorsteinn Rögnvaldsson. “Using Neural Networks to Identify Jets”. In: *Nuclear Physics B* 349.3 (Feb. 11, 1991), pp. 675–702 (cit. on p. 51).

- [116]Leif Lönnblad, Carsten Peterson, and Thorsteinn Rögnvaldsson. “Pattern Recognition in High Energy Physics with Artificial Neural Networks — JETNET 2.0”. In: *Computer Physics Communications* 70.1 (May 1, 1992), pp. 167–182 (cit. on p. 51).
- [117]Carsten Peterson, Thorsteinn Rögnvaldsson, and Leif Lönnblad. “JETNET 3.0—A Versatile Artificial Neural Network Package”. In: *Computer Physics Communications* 81.1 (June 1, 1994), pp. 185–220 (cit. on p. 51).
- [118]ALEPH Collaboration. “Determination of $|V_{ub}|$ from the Measurement of the Inclusive Charmless Semileptonic Branching Ratio of b Hadrons”. In: *The European Physical Journal C* 6.4 (Jan. 1999), pp. 555–574 (cit. on p. 51).
- [119]Philippe Rosnet and Pierre Henrard. “Use of Neural Network to Search for Rare B Decays in ALEPH at LEP”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*. New Computing Techniques in Physics Research V 389.1 (Apr. 11, 1997), pp. 197–198 (cit. on p. 51).
- [120]B. P. Kersevan. “Neural Network Applications for the Analysis of LEP Data”. In: *Acta Phys. Polon. B* 34 (2003). Ed. by M. Praszalowicz, pp. 6069–6084 (cit. on p. 51).
- [121]Byron P. Roe, Hai-Jun Yang, Ji Zhu, et al. “Boosted Decision Trees as an Alternative to Artificial Neural Networks for Particle Identification”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 543.2 (May 11, 2005), pp. 577–584 (cit. on p. 51).
- [122]MiniBooNE Collaboration. “A Search for Electron Neutrino Appearance at the $\Delta M^2 \sim 1 \text{ eV}^2$ Scale”. In: *Physical Review Letters* 98.23 (June 8, 2007), p. 231801. arXiv: 0704.1500 [hep-ex] (cit. on p. 51).
- [123]DØ Collaboration, V. M. Abazov, B. Abbott, et al. “Evidence for Production of Single Top Quarks and First Direct Measurement of $|V_{tb}|$ ”. In: *Physical Review Letters* 98.18 (May 1, 2007), p. 181802 (cit. on p. 51).
- [124]CDF Collaboration, T. Aaltonen, J. Adelman, et al. “Observation of Electroweak Single Top-Quark Production”. In: *Physical Review Letters* 103.9 (Aug. 24, 2009), p. 092002 (cit. on p. 51).
- [125]BABAR Collaboration. *Measurement of CP-Violating Asymmetries in the $B^0 \rightarrow K+K-K^0$ Dalitz Plot*. July 31, 2006. arXiv: hep-ex/0607112. URL: <http://arxiv.org/abs/hep-ex/0607112> (visited on Mar. 20, 2025). Pre-published (cit. on p. 51).
- [126]HEP ML Community. *A Living Review of Machine Learning for Particle Physics*. URL: <https://iml-wg.github.io/HEPML-LivingReview/> (cit. on p. 51).
- [127]Markus Atkinson, Javier Duarte, Philip Harris, et al. “Opportunities for Accelerated Machine Learning Inference in Fundamental Physics”. In: (2019) (cit. on p. 51).
- [128]Kim Albertsson, Piero Altò, Dustin Anderson, et al. *Machine Learning in High Energy Physics Community White Paper*. May 16, 2019. arXiv: 1807.02876 [hep-ex, physics:physics, stat]. URL: <http://arxiv.org/abs/1807.02876> (visited on Apr. 10, 2024). Pre-published (cit. on p. 51).

- [129]Tilman Plehn, Anja Butter, Barry Dillon, et al. *Modern Machine Learning for LHC Physicists*. Apr. 24, 2025. arXiv: [2211.01421](https://arxiv.org/abs/2211.01421) [hep-ph]. URL: <http://arxiv.org/abs/2211.01421> (visited on June 6, 2025). Pre-published (cit. on p. 51).
- [130]Philip Harris, Erik Katsavounidis, William Patrick McCormack, et al. *Physics Community Needs, Tools, and Resources for Machine Learning*. Mar. 30, 2022. arXiv: [2203.16255](https://arxiv.org/abs/2203.16255) [cs]. URL: <http://arxiv.org/abs/2203.16255> (visited on May 16, 2025). Pre-published (cit. on p. 52).
- [131]EuCAIF. *EuCAIF*. European Coalition for AI in Fundamental Physics. 2025. URL: <https://eucaif.org/> (visited on May 14, 2025) (cit. on p. 52).
- [132]Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep Learning”. In: *Nature* 521.7553 (May 2015), pp. 436–444 (cit. on p. 52).
- [133]John Jumper, Richard Evans, Alexander Pritzel, et al. “Highly Accurate Protein Structure Prediction with AlphaFold”. In: *Nature* 596.7873 (Aug. 2021), pp. 583–589 (cit. on p. 52).
- [134]Izaak Neutelings. *Neural Networks – TikZ.Net*. Apr. 12, 2024. URL: https://tikz.net/neural_networks/ (visited on Mar. 21, 2025) (cit. on pp. 53, 58, 60).
- [135]Kunihiko Fukushima. “Visual Feature Extraction by a Multilayered Network of Analog Threshold Elements”. In: *IEEE Transactions on Systems Science and Cybernetics* 5.4 (Oct. 1969), pp. 322–333 (cit. on p. 54).
- [136]Vinod Nair and Geoffrey E. Hinton. “Rectified Linear Units Improve Restricted Boltzmann Machines”. In: *Proceedings of the 27th International Conference on International Conference on Machine Learning*. ICML’10. Madison, WI, USA: Omnipress, June 21, 2010, pp. 807–814 (cit. on p. 54).
- [137]Kurt Hornik, Maxwell Stinchcombe, and Halbert White. “Multilayer Feedforward Networks Are Universal Approximators”. In: *Neural Networks* 2.5 (Jan. 1989), pp. 359–366 (cit. on pp. 54, 57, 76).
- [138]G. Cybenko. “Approximation by Superpositions of a Sigmoidal Function”. In: *Mathematics of Control, Signals and Systems* 2.4 (Dec. 1, 1989), pp. 303–314 (cit. on p. 54).
- [139]Augustin-Louis Cauchy. *Oeuvres Complètes d’Augustin Cauchy*. Série 1, tome 10 vols. Académie des sciences, 1882–1974 (cit. on p. 55).
- [140]Nadav Cohen. *Understanding Optimization in Deep Learning by Analyzing Trajectories of Gradient Descent*. Off the Convex Path. Nov. 7, 2018. URL: <http://offconvex.github.io/2018/11/07/optimization-beyond-landscape/> (visited on May 20, 2025) (cit. on p. 56).
- [141]Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. Jan. 30, 2017. arXiv: [1412.6980](https://arxiv.org/abs/1412.6980) [cs]. URL: <http://arxiv.org/abs/1412.6980> (visited on Oct. 5, 2023). Pre-published (cit. on p. 55).
- [142]Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, et al. *Deep Sets*. Apr. 14, 2018. arXiv: [1703.06114](https://arxiv.org/abs/1703.06114) [cs, stat]. URL: <http://arxiv.org/abs/1703.06114> (visited on Apr. 3, 2023). Pre-published (cit. on pp. 56, 57).

- [143]Patrick T. Komiske, Eric M. Metodiev, and Jesse Thaler. “Energy Flow Networks: Deep Sets for Particle Jets”. In: *Journal of High Energy Physics* 2019.1 (Jan. 2019), p. 121. arXiv: [1810.05165](https://arxiv.org/abs/1810.05165) [[hep-ex](#), [physics:hep-ph](#), [stat](#)] (cit. on pp. 57, 76, 77).
- [144]ATLAS Collaboration. *Deep Sets based Neural Networks for Impact Parameter Flavour Tagging in ATLAS*. ATL-PHYS-PUB-2020-014. ATL-COM-PHYS-2020-244, May 25, 2020 (cit. on p. 57).
- [145]Nicole Michelle Hartman, Rafael Teixeira De Lima, and Michael Kagan. “Deep Sets for Flavor Tagging on the ATLAS Experiment”. In: *Proceedings of the 2020 Connecting the Dots Workshop*. Connecting the Dots Workshop. Princeton, USA: ATL-COM-PHYS-2020-427, June 17, 2020 (cit. on p. 57).
- [146]ATLAS Collaboration. “Fast B-Tagging at the High-Level Trigger of the ATLAS Experiment in LHC Run 3.” In: *JINST* 18.CERN-EP-2023-111 (11 2023), P11006. arXiv: [2306.09738](https://arxiv.org/abs/2306.09738) (cit. on p. 57).
- [147]Claire Prouve, Niklas Nolte, and Christoph Hasse. *Fast Inclusive Flavour Tagging at LHCb*. Apr. 24, 2024. arXiv: [2404.14145](https://arxiv.org/abs/2404.14145) [[hep-ex](#)]. URL: <http://arxiv.org/abs/2404.14145> (visited on June 5, 2025). Pre-published (cit. on p. 58).
- [148]Yann LeCun and Francoise Soulie Fogelman. “Modeles Connexionnistes de l’apprentissage”. In: *Intellectica, special issue apprentissage et machine* 2 (Jan. 1, 1987) (cit. on pp. 58, 111).
- [149]H. Bourlard and Y. Kamp. “Auto-Association by Multilayer Perceptrons and Singular Value Decomposition”. In: *Biological Cybernetics* 59.4 (Sept. 1, 1988), pp. 291–294 (cit. on pp. 58, 111).
- [150]Richard Zemel and Geoffrey E Hinton. “Developing Population Codes by Minimizing Description Length”. In: *Advances in Neural Information Processing Systems*. Vol. 6. Morgan-Kaufmann, 1993 (cit. on pp. 58, 111).
- [151]S. Kullback and R. A. Leibler. “On Information and Sufficiency”. In: *The Annals of Mathematical Statistics* 22.1 (Mar. 1, 1951), pp. 79–86 (cit. on p. 59).
- [152]Diederik P Kingma and Max Welling. “Auto-Encoding Variational Bayes”. In: *arXiv e-prints* (Dec. 2013), arXiv:1312.6114 (cit. on pp. 59, 111).
- [153]Vasilis Belis, Patrick Odagiu, and Thea Klæboe Årrestad. *Machine Learning for Anomaly Detection in Particle Physics*. Dec. 20, 2023. arXiv: [2312.14190](https://arxiv.org/abs/2312.14190) [[hep-ex](#), [physics:physics](#), [physics:quant-ph](#)]. URL: <http://arxiv.org/abs/2312.14190> (visited on Jan. 9, 2024). Pre-published (cit. on pp. 59, 63, 109, 114).
- [154]CMS ECAL Collaboration. “Autoencoder-Based Anomaly Detection System for Online Data Quality Monitoring of the CMS Electromagnetic Calorimeter”. In: *Computing and Software for Big Science* 8.1 (Dec. 2024), p. 11 (cit. on pp. 59, 64, 114).
- [155]Abhirami Harilal, Kyungmin Park, and Manfred Paulini. *Anomaly Detection Based on Machine Learning for the CMS Electromagnetic Calorimeter Online Data Quality Monitoring*. July 25, 2024. arXiv: [2407.20278](https://arxiv.org/abs/2407.20278) [[hep-ex](#), [physics:physics](#)]. URL: <http://arxiv.org/abs/2407.20278> (visited on Sept. 17, 2024). Pre-published (cit. on p. 59).

- [156] Florencia Canelli, Annapaola de Cosa, Luc Le Pottier, et al. “Autoencoders for Semivisible Jet Detection”. In: *Journal of High Energy Physics* 2022.2 (Feb. 2022), p. 74. arXiv: [2112.02864 \[hep-ex, physics:hep-ph\]](#) (cit. on pp. 59, 64).
- [157] Florian Eble and CMS Collaboration. “Unsupervised Tagging of Semivisible Jets with Normalized Autoencoders in CMS”. In: *Proceedings of The European Physical Society Conference on High Energy Physics — PoS(EPS-HEP2023)*. The European Physical Society Conference on High Energy Physics. Hamburg, Germany: Sissa Medialab, Jan. 24, 2024, p. 491 (cit. on pp. 59, 64).
- [158] CMS Collaboration. “Wasserstein Normalized Autoencoder”. In: (Sept. 4, 2024) (cit. on pp. 59, 64).
- [159] ATLAS Collaboration. *Search for New Physics in Final States with Semi-Visible Jets or Anomalous Signatures Using the ATLAS Detector*. May 2, 2025. arXiv: [2505.01634 \[hep-ex\]](#). URL: <http://arxiv.org/abs/2505.01634> (visited on June 6, 2025). Pre-published (cit. on pp. 59, 64).
- [160] Barry M. Dillon, Tilman Plehn, Christof Sauer, and Peter Sorrenson. “Better Latent Spaces for Better Autoencoders”. In: *SciPost Physics* 11.3 (Sept. 17, 2021), p. 061. arXiv: [2104.08291 \[hep-ph\]](#) (cit. on p. 59).
- [161] Barry M. Dillon, Luigi Favaro, Tilman Plehn, Peter Sorrenson, and Michael Krämer. *A Normalized Autoencoder for LHC Triggers*. June 22, 2023. arXiv: [2206.14225 \[hep-ph\]](#). URL: <http://arxiv.org/abs/2206.14225> (visited on Feb. 8, 2024). Pre-published (cit. on p. 59).
- [162] Gabriel Matos, Elena Busch, Ki Ryeong Park, and Julia Gonski. *Semi-Supervised Permutation Invariant Particle-Level Anomaly Detection*. Aug. 30, 2024. arXiv: [2408.17409 \[hep-ph\]](#). URL: <http://arxiv.org/abs/2408.17409> (visited on June 6, 2025). Pre-published (cit. on p. 59).
- [163] Eric M. Metodiev, Benjamin Nachman, and Jesse Thaler. “Classification without Labels: Learning from Mixed Samples in High Energy Physics”. In: *Journal of High Energy Physics* 2017.10 (Oct. 25, 2017), p. 174 (cit. on pp. 59, 63, 64).
- [164] CMS Collaboration. *Real-Time Anomaly Detection at the L1 Trigger of CMS Experiment*. Nov. 29, 2024. arXiv: [2411.19506 \[hep-ex\]](#). URL: <http://arxiv.org/abs/2411.19506> (visited on Mar. 31, 2025). Pre-published (cit. on pp. 59, 63).
- [165] CMS Collaboration. *2024 Data Collected with AXOL1TL Anomaly Detection at the CMS Level-1 Trigger*. 2024 (cit. on p. 59).
- [166] LHCb Collaboration. “Anomaly Detection for a New Trigger Line for Showers in the Muon Detectors”. In: (June 4, 2024) (cit. on p. 59).
- [167] Fritjof Bengtsson Folkesson, Caterina Doglioni, Per Alexander Ekman, et al. “Baler - Machine Learning Based Compression of Scientific Data”. In: *EPJ Web of Conferences* 295 (2024). Ed. by R. De Vita, X. Espinal, P. Laycock, and O. Shadura, p. 09023 (cit. on p. 59).
- [168] Warren S. McCulloch and Walter Pitts. “A Logical Calculus of the Ideas Immanent in Nervous Activity”. In: *The bulletin of mathematical biophysics* 5.4 (Dec. 1, 1943), pp. 115–133 (cit. on pp. 60, 111).
- [169] Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-Term Memory”. In: *Neural Comput.* 9.8 (Nov. 1997), pp. 1735–1780 (cit. on pp. 60, 111).

- [170]Y. Bengio, Patrice Simard, and Paolo Frasconi. “Learning Long-Term Dependencies with Gradient Descent Is Difficult”. In: *IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council* 5 (Feb. 1, 1994), pp. 157–66 (cit. on p. 60).
- [171]Rafael Teixeira de Lima. *Sequence-Based Machine Learning Models in Jet Physics*. Feb. 9, 2021. arXiv: [2102.06128](https://arxiv.org/abs/2102.06128) [hep-ex, physics:physics]. URL: <http://arxiv.org/abs/2102.06128> (visited on June 1, 2023). Pre-published (cit. on p. 60).
- [172]Kiichi Goto, Taikan Suehara, Tamaki Yoshioka, et al. “Development of a Vertex Finding Algorithm Using Recurrent Neural Network”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 1047 (Feb. 2023), p. 167836. arXiv: [2101.11906](https://arxiv.org/abs/2101.11906) [physics] (cit. on p. 60).
- [173]Maciej Wielgosz, Andrzej Skoczeń, and Matej Mertik. “Using LSTM Recurrent Neural Networks for Monitoring the LHC Superconducting Magnets. Using LSTM Recurrent Neural Networks for Detecting Anomalous Behavior of LHC Superconducting Magnets”. In: *Nucl. Instrum. Methods Phys. Res., A* 867 (2017), pp. 40–50 (cit. on p. 60).
- [174]M Barisits, T Beermann, V Garonne, et al. “The ATLAS Data Management System Rucio: Supporting LHC Run-2 and Beyond”. In: *Journal of Physics: Conference Series* 1085 (Sept. 2018), p. 032030 (cit. on p. 60).
- [175]J. Erdmann. “A Tagger for Strange Jets Based on Tracking Information Using Long Short-Term Memory”. In: *Journal of Instrumentation* 15.01 (Jan. 2020), P01021 (cit. on p. 61).
- [176]Shannon Egan, Wojciech Fedorko, Alison Lister, Jannicke Pearkes, and Colin Gay. *Long Short-Term Memory (LSTM) Networks with Jet Constituents for Boosted Top Tagging at the LHC*. Nov. 24, 2017. arXiv: [1711.09059](https://arxiv.org/abs/1711.09059) [hep-ex]. URL: <http://arxiv.org/abs/1711.09059> (visited on June 6, 2025). Pre-published (cit. on p. 61).
- [177]ATLAS Collaboration. *Identification of hadronic tau lepton decays using neural networks in the ATLAS experiment*. ATL-PHYS-PUB-2019-033. ATL-COM-PHYS-2019-773, Aug. 30, 2019 (cit. on p. 61).
- [178]Georges Aad, Anne-Sophie Berthold, Thomas Calvet, et al. “Artificial Neural Networks on FPGAs for Real-Time Energy Reconstruction of the ATLAS LAr Calorimeters”. In: *Computing and Software for Big Science* 5.1 (Oct. 2, 2021), p. 19 (cit. on p. 61).
- [179]Johannes Albrecht, Dylan Bourgeois, Victor Coco, et al. “New Approaches for Track Reconstruction in LHCb’s Vertex Locator”. In: *EPJ Web of Conferences* 214 (2019). Ed. by A. Forti, L. Betev, M. Litmaath, O. Smirnova, and P. Hristov, p. 01042 (cit. on p. 61).
- [180]ATLAS Collaboration. *Transforming Jet Flavour Tagging at ATLAS*. May 26, 2025. arXiv: [2505.19689](https://arxiv.org/abs/2505.19689) [hep-ex]. URL: <http://arxiv.org/abs/2505.19689> (visited on June 6, 2025). Pre-published (cit. on p. 62).

- [181]ATLAS Collaboration. *Transformer Neural Networks for Identifying Boosted Higgs Bosons Decaying into $B\bar{b}$ and $C\bar{b}$ in ATLAS*. Geneva: CERN, 2023 (cit. on p. 62).
- [182]Ryan Roberts. *Transformer for Energy Calibration in the ATLAS Electromagnetic Calorimeter*. 2024 (cit. on p. 62).
- [183]Joschka Birk, Frank Gaede, Anna Hallin, et al. *OmniJet- α_C : Learning Point Cloud Calorimeter Simulations Using Generative Transformers*. Jan. 9, 2025. arXiv: [2501.05534](https://arxiv.org/abs/2501.05534) [hep-ph]. URL: <http://arxiv.org/abs/2501.05534> (visited on June 6, 2025). Pre-published (cit. on p. 62).
- [184]Guillaume Quétant, John Andrew Raine, Matthew Leigh, Debajyoti Sengupta, and Tobias Golling. “PIPPIN: Generating Variable Length Full Events from Partons”. In: *Physical Review D* 110.7 (Oct. 21, 2024), p. 076023. arXiv: [2406.13074](https://arxiv.org/abs/2406.13074) [hep-ph] (cit. on p. 62).
- [185]Anja Butter, Nathan Huetsch, Sofia Palacios Schweitzer, et al. “Jet Diffusion versus JetGPT – Modern Networks for the LHC”. In: *SciPost Physics Core* 8.1 (Mar. 3, 2025), p. 026. arXiv: [2305.10475](https://arxiv.org/abs/2305.10475) [hep-ph] (cit. on p. 62).
- [186]Thorben Finke, Michael Krämer, Alexander Mück, and Jan Tönshoff. “Learning the Language of QCD Jets with Transformers”. In: *Journal of High Energy Physics* 2023.6 (June 27, 2023), p. 184. arXiv: [2303.07364](https://arxiv.org/abs/2303.07364) [hep-ph] (cit. on p. 62).
- [187]Charu C. Aggarwal. *Outlier Analysis*. 2nd ed. Cham: Springer International Publishing, 2017 (cit. on p. 63).
- [188]Songqiao Han, Xiyang Hu, Hailiang Huang, Minqi Jiang, and Yue Zhao. “AD-Bench: Anomaly Detection Benchmark”. In: *Neural Information Processing Systems (NeurIPS)*. NeurIPS 2022. 2022 (cit. on pp. 63, 112).
- [189]Gregor Kasieczka, Benjamin Nachman, David Shih, et al. “The LHC Olympics 2020: A Community Challenge for Anomaly Detection in High Energy Physics”. In: *Reports on Progress in Physics* 84.12 (Dec. 1, 2021), p. 124201. arXiv: [2101.08320](https://arxiv.org/abs/2101.08320) [hep-ex, physics:hep-ph, physics:physics] (cit. on p. 63).
- [190]Benjamin Nachman. *Anomaly Detection for Physics Analysis and Less than Supervised Learning*. Oct. 27, 2020. arXiv: [2010.14554](https://arxiv.org/abs/2010.14554) [hep-ex, physics:hep-ph, physics:physics]. URL: <http://arxiv.org/abs/2010.14554> (visited on Apr. 25, 2023). Pre-published (cit. on p. 63).
- [191]Vichayanun Wachirapusitan. “Machine Learning Applications for Data Quality Monitoring and Data Certification within CMS”. In: *Journal of Physics: Conference Series* 2438.1 (Feb. 1, 2023), p. 012098 (cit. on pp. 63, 64, 114).
- [192]Anna Hallin, Joshua Isaacson, Gregor Kasieczka, et al. “Classifying Anomalies through Outer Density Estimation”. In: *Physical Review D* 106.5 (Sept. 6, 2022), p. 055006 (cit. on p. 64).
- [193]O. Knapp, O. Cerri, G. Dissertori, et al. “Adversarially Learned Anomaly Detection on CMS Open Data: Re-Discovering the Top Quark”. In: *The European Physical Journal Plus* 2021 136:2 136.2 (Feb. 2021), pp. 1–18 (cit. on p. 64).
- [194]FastML Team. *Fastmachinelearning/Hls4ml*. Version v1.0.0. Zenodo, 2024 (cit. on p. 64).

- [195]Farah Fahim, Benjamin Hawks, Christian Herwig, et al. “HLS4ML: An Open-Source Codesign Workflow to Empower Scientific Low-Power Machine Learning Devices”. 2021. arXiv: [2103.05579](#) (cit. on p. 64).
- [196]Kamil Deja. “Using Machine Learning Techniques for Data Quality Monitoring in CMS and ALICE”. In: *Proceedings of Science*. 7th Annual Conference on Large Hadron Collider Physics - LHCP2019. Puebla, Mexico, May 20, 2019 (cit. on p. 64).
- [197]David M. W. Powers. *Evaluation: From Precision, Recall and F-measure to ROC, Informedness, Markedness and Correlation*. Oct. 11, 2020. arXiv: [2010.16061](#) [cs]. URL: <http://arxiv.org/abs/2010.16061> (visited on Feb. 17, 2025). Pre-published (cit. on p. 65).
- [198]Siwon Kim, Kukjin Choi, Hyun-Soo Choi, Byunghan Lee, and Sungroh Yoon. *Towards a Rigorous Evaluation of Time-series Anomaly Detection*. Jan. 4, 2022. arXiv: [2109.05257](#) [cs, stat]. URL: <http://arxiv.org/abs/2109.05257> (visited on May 7, 2024). Pre-published (cit. on pp. 65, 113, 118, 124).
- [199]Davide Chicco and Giuseppe Jurman. “The Advantages of the Matthews Correlation Coefficient (MCC) over F1 Score and Accuracy in Binary Classification Evaluation”. In: *BMC Genomics* 21.1 (Jan. 2020), pp. 1–13 (cit. on p. 65).
- [200]Noman Bashir, Priya Donti, James Cuff, et al. “The Climate and Sustainability Implications of Generative AI”. In: *An MIT Exploration of Generative AI* (Mar. 27, 2024) (cit. on p. 66).
- [201]Christoph Molnar. *Interpretable Machine Learning*. 3rd. Leanpub, Feb. 26, 2018 (cit. on p. 67).
- [202]W. James Murdoch, Chandan Singh, Karl Kumbier, Reza Abbasi-Asl, and Bin Yu. “Definitions, Methods, and Applications in Interpretable Machine Learning”. In: *Proceedings of the National Academy of Sciences* 116.44 (Oct. 29, 2019), pp. 22071–22080 (cit. on p. 67).
- [203]Ribana Roscher, Bastian Bohn, Marco F. Duarte, and Jochen Garcke. “Explainable Machine Learning for Scientific Insights and Discoveries”. In: *IEEE Access* 8 (2020), pp. 42200–42216 (cit. on p. 67).
- [204]Jeff Danoy. *Isolated Jet Tree*. May 2018 (cit. on p. 70).
- [205]Kevin Greif. *Root2HDF5*. Mar. 2023 (cit. on p. 71).
- [206]Daniel Holmberg, Dejan Golubovic, and Henning Kirschenmann. “Jet Energy Calibration with Deep Learning as a Kubeflow Pipeline”. In: *Computing and Software for Big Science* 7.1 (Dec. 2023), p. 9. arXiv: [2308.12724](#) [hep-ex, physics:hep-ph, physics:physics] (cit. on p. 74).
- [207]Laura Boggia and Bogdan Malaescu. *Simultaneous Jet Calibration with Machine Learning Including in Situ Jet Energy Resolution Measurement*. Geneva: ATLAS Collaboration, 2024 (cit. on pp. 107, 164, 168).
- [208]Can Akkaya and Dionysis Goularas. “Evaluation of Machine and Deep Learning Methods for Fraud Detection”. In: *2024 8th International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*. 2024 8th International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT). Nov. 2024, pp. 1–6 (cit. on p. 109).

- [209]Kuangyi Gu. “Deep Learning Techniques in Financial Fraud Detection”. In: *Proceedings of the 7th International Conference on Cyber Security and Information Engineering*. ICCSIE2022: 7th International Conference on Cyber Security and Information Engineering. Brisbane QLD Australia: ACM, Sept. 23, 2022, pp. 282–286 (cit. on p. 109).
- [210]Yann-Aël Le Borgne, Wissam Siblini, Bertrand Lebichot, and Gianluca Bontempi. *Reproducible Machine Learning for Credit Card Fraud Detection - Practical Handbook*. Université Libre de Bruxelles, 2022 (cit. on p. 109).
- [211]European Central Bank. “Sixth Report on Card Fraud”. In: (Aug. 14, 2020) (cit. on p. 109).
- [212]M.V. Araújo, M. Begalli, W.S. Freund, et al. “Lorenzetti Showers - A General-Purpose Framework for Supporting Signal Reconstruction and Triggering with Calorimeters”. In: *Computer Physics Communications* 286 (May 2023), p. 108671 (cit. on pp. 110, 138–140, 143).
- [213]Manish Gupta, Jing Gao, Charu C. Aggarwal, and Jiawei Han. “Outlier Detection for Temporal Data: A Survey”. In: *IEEE Transactions on Knowledge and Data Engineering* 26.9 (Sept. 2014), pp. 2250–2267 (cit. on pp. 111, 117).
- [214]Raghavendra Chalapathy and Sanjay Chawla. *Deep Learning for Anomaly Detection: A Survey*. Jan. 23, 2019. arXiv: 1901.03407 [cs]. URL: <http://arxiv.org/abs/1901.03407> (visited on Apr. 30, 2025). Pre-published (cit. on p. 111).
- [215]Ane Blázquez-García, Angel Conde, Usue Mori, and Jose A. Lozano. “A Review on Outlier/Anomaly Detection in Time Series Data”. In: *ACM Comput. Surv.* 54.3 (Apr. 2021) (cit. on pp. 111, 117).
- [216]Sebastian Schmidl, Phillip Wenig, and Thorsten Papenbrock. “Anomaly Detection in Time Series: A Comprehensive Evaluation”. In: *Proceedings of the VLDB Endowment* 15.9 (May 2022), pp. 1779–1797 (cit. on p. 111).
- [217]Mohammad Braei and Sebastian Wagner. *Anomaly Detection in Univariate Time-series: A Survey on the State-of-the-Art*. Apr. 1, 2020. arXiv: 2004.00433 [cs]. URL: <http://arxiv.org/abs/2004.00433> (visited on Apr. 30, 2025). Pre-published (cit. on p. 111).
- [218]John Paparrizos, Yuhao Kang, Paul Boniol, et al. “TSB-UAD: An End-to-End Benchmark Suite for Univariate Time-Series Anomaly Detection”. In: *Proceedings of the VLDB Endowment* 15.8 (Apr. 2022), pp. 1697–1711 (cit. on pp. 111, 117).
- [219]Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. “Isolation Forest”. In: *2008 Eighth IEEE International Conference on Data Mining*. 2008 Eighth IEEE International Conference on Data Mining (ICDM). Pisa, Italy: IEEE, Dec. 2008, pp. 413–422 (cit. on p. 111).
- [220]Bernhard Schölkopf, Robert C Williamson, Alex Smola, John Shawe-Taylor, and John Platt. “Support Vector Method for Novelty Detection”. In: *Advances in Neural Information Processing Systems*. Vol. 12. MIT Press, 1999 (cit. on p. 111).

- [221]Takaaki Nakamura, Makoto Imamura, Ryan Mercer, and Eamonn Keogh. “MER-LIN: Parameter-Free Discovery of Arbitrary Length Anomalies in Massive Time Series Archives”. In: *2020 IEEE International Conference on Data Mining (ICDM)*. 2020 IEEE International Conference on Data Mining (ICDM). Sorrento, Italy: IEEE, Nov. 2020, pp. 1190–1195 (cit. on pp. 111, 112, 183).
- [222]Kwei-Herng Lai, Daochen Zha, Junjie Xu, et al. “Revisiting Time Series Outlier Detection: Definitions and Benchmarks”. In: *Thirty-Fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*. June 8, 2021 (cit. on pp. 111, 112, 116, 117).
- [223]Varun Chandola, Arindam Banerjee, and Vipin Kumar. “Anomaly Detection: A Survey”. In: *ACM Computing Surveys* 41.3 (July 2009), pp. 1–58 (cit. on p. 111).
- [224]Julien Audibert, Pietro Michiardi, Frédéric Guyard, Sébastien Marti, and Maria A. Zuluaga. “USAD: UnSupervised Anomaly Detection on Multivariate Time Series”. In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. KDD ’20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. Virtual Event CA USA: ACM, Aug. 23, 2020, pp. 3395–3404 (cit. on pp. 111, 119, 189).
- [225]Shreshth Tuli, Giuliano Casale, and Nicholas R. Jennings. “TranAD: Deep Transformer Networks for Anomaly Detection in Multivariate Time Series Data”. In: *Proc. VLDB Endow.* 15.6 (Feb. 1, 2022), pp. 1201–1214 (cit. on pp. 111, 112, 120, 125, 183).
- [226]Ailin Deng and Bryan Hooi. “Graph Neural Network-Based Anomaly Detection in Multivariate Time Series”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 35.5 (May 18, 2021), pp. 4027–4035 (cit. on p. 111).
- [227]Hang Zhao, Yujing Wang, Juanyong Duan, et al. “Multivariate Time-Series Anomaly Detection via Graph Attention Network”. In: *2020 IEEE International Conference on Data Mining (ICDM)*. 2020, pp. 841–850 (cit. on p. 111).
- [228]Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, et al. *Generative Adversarial Networks*. June 10, 2014. arXiv: 1406.2661 [cs, stat]. URL: <http://arxiv.org/abs/1406.2661> (visited on Apr. 26, 2023). Pre-published (cit. on p. 111).
- [229]Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. “The Graph Neural Network Model”. In: *IEEE Transactions on Neural Networks* 20.1 (2009), pp. 61–80 (cit. on p. 111).
- [230]Bo Zong, Qi Song, Martin Renqiang Min, et al. “Deep Autoencoding Gaussian Mixture Model for Unsupervised Anomaly Detection”. In: *International Conference on Learning Representations*. 2018 (cit. on p. 111).
- [231]Ya Su, Youjian Zhao, Chenhao Niu, et al. “Robust Anomaly Detection for Multivariate Time Series through Stochastic Recurrent Neural Network”. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. KDD ’19: The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. Anchorage AK USA: ACM, July 25, 2019, pp. 2828–2837 (cit. on pp. 111, 112, 116, 183).

- [232]Ruei-Jie Hsieh, Jerry Chou, and Chih-Hsiang Ho. “Unsupervised Online Anomaly Detection on Multivariate Sensing Time Series Data for Smart Manufacturing”. In: *2019 IEEE 12th Conference on Service-Oriented Computing and Applications (SOCA)*. 2019 IEEE 12th Conference on Service-Oriented Computing and Applications (SOCA). Kaohsiung, Taiwan: IEEE, Nov. 2019, pp. 90–97 (cit. on p. 111).
- [233]Qingsong Wen, Tian Zhou, Chaoli Zhang, et al. *Transformers in Time Series: A Survey*. May 11, 2023. arXiv: 2202.07125 [cs, eess, stat]. URL: <http://arxiv.org/abs/2202.07125> (visited on Apr. 26, 2024). Pre-published (cit. on pp. 112, 120).
- [234]Sabeen Ahmed, Ian E. Nielsen, Aakash Tripathi, et al. “Transformers in Time-Series Analysis: A Tutorial”. In: *Circuits, Systems, and Signal Processing* 42.12 (Dec. 2023), pp. 7433–7466 (cit. on pp. 112, 120).
- [235]Xixuan Wang, Dechang Pi, Xiangyan Zhang, Hao Liu, and Chang Guo. “Variational Transformer-Based Anomaly Detection Approach for Multivariate Time Series”. In: *Measurement* 191 (Mar. 15, 2022), p. 110791 (cit. on p. 112).
- [236]Hongwei Zhang, Yuanqing Xia, Tijin Yan, and Guiyang Liu. “Unsupervised Anomaly Detection in Multivariate Time Series through Transformer-based Variational Autoencoder”. In: *2021 33rd Chinese Control and Decision Conference (CCDC)*. 2021, pp. 281–286 (cit. on p. 112).
- [237]Zekai Chen, Dingshuo Chen, Xiao Zhang, Zixuan Yuan, and Xiuzhen Cheng. “Learning Graph Structures with Transformer for Multivariate Time Series Anomaly Detection in IoT”. In: *IEEE Internet of Things Journal* 9.12 (June 15, 2022), pp. 9179–9189. arXiv: 2104.03466 [cs] (cit. on p. 112).
- [238]Jiehui Xu, Haixu Wu, Jianmin Wang, and Mingsheng Long. *Anomaly Transformer: Time Series Anomaly Detection with Association Discrepancy*. June 29, 2022. arXiv: 2110.02642 [cs]. URL: <http://arxiv.org/abs/2110.02642> (visited on Mar. 22, 2024). Pre-published (cit. on pp. 112, 120).
- [239]Hyeongwon Kang and Pilsung Kang. “Transformer-Based Multivariate Time Series Anomaly Detection Using Inter-Variable Attention Mechanism”. In: *Knowledge-Based Systems* 290 (Apr. 2024), p. 111507 (cit. on pp. 112, 125).
- [240]N. Laptev, S. Amizadeh and Y. Billawala. *S5 - A Labeled Anomaly Detection Dataset, Version 1.0 (16M)*. Y Webscope. Mar. 2015. URL: <https://webscope.sandbox.yahoo.com/catalog.php?datatype=s&did=70&guccounter=1> (visited on Feb. 10, 2025) (cit. on p. 112).
- [241]Subutai Ahmad, Alexander Lavin, Scott Purdy, and Zuha Agha. “Unsupervised Real-Time Anomaly Detection for Streaming Data”. In: *Neurocomputing* 262 (June 1, 2017) (cit. on p. 112).
- [242]Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. “Detecting Spacecraft Anomalies Using LSTMs and Nonparametric Dynamic Thresholding”. In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. July 19, 2018, pp. 387–395. arXiv: 1802.04431 [cs, stat] (cit. on pp. 112, 116, 117, 125, 183).

- [243]Renjie Wu and Eamonn Keogh. “Current Time Series Anomaly Detection Benchmarks Are Flawed and Are Creating the Illusion of Progress”. In: *IEEE Transactions on Knowledge and Data Engineering* (2021), pp. 1–1 (cit. on pp. 112, 116, 183, 184).
- [244]Mohamed El Amine Sehili and Zonghua Zhang. *Multivariate Time Series Anomaly Detection: Fancy Algorithms and Flawed Evaluation Methodology*. Nov. 1, 2023. arXiv: 2308.13068 [cs]. URL: <http://arxiv.org/abs/2308.13068> (visited on June 28, 2025). Pre-published (cit. on p. 113).
- [245]B. P. Abbott, R. Abbott, T. D. Abbott, et al. “Observation of Gravitational Waves from a Binary Black Hole Merger”. In: *Physical Review Letters* 116.6 (Feb. 2016) (cit. on p. 113).
- [246]Elena Cuoco, Marco Cavaglià, Ik Siong Heng, David Keitel, and Christopher Messenger. “Applications of Machine Learning in Gravitational-Wave Research with Current Interferometric Detectors”. In: *Living Reviews in Relativity* 28.1 (Feb. 27, 2025), p. 2 (cit. on p. 113).
- [247]Filip Morawski, Michał Bejger, Elena Cuoco, and Luigia Petre. “Anomaly Detection in Gravitational Waves Data Using Convolutional AutoEncoders”. In: *Machine Learning: Science and Technology* 2.4 (Dec. 1, 2021), p. 045014. arXiv: 2103.07688 [astro-ph, physics:gr-qc] (cit. on p. 113).
- [248]Xiang-Ru Li, Wo-Liang Yu, Xi-Long Fan, and G. Jogesh Babu. “Some Optimizations on Detecting Gravitational Wave Using Convolutional Neural Network”. In: *Frontiers of Physics* 15.5 (June 10, 2020), p. 54501 (cit. on p. 113).
- [249]Eric A Moreno, Bartłomiej Borzyszkowski, Maurizio Pierini, Jean-Roch Vlimant, and Maria Spiropulu. “Source-Agnostic Gravitational-Wave Detection with Recurrent Autoencoders”. In: *Machine Learning: Science and Technology* 3.2 (Apr. 2022), p. 025001 (cit. on p. 114).
- [250]Ryan Raikman, Eric A Moreno, Ekaterina Govorkova, et al. “GWAK: Gravitational-Wave Anomalous Knowledge with Recurrent Autoencoders”. In: *Machine Learning: Science and Technology* 5.2 (Apr. 2024), p. 025020 (cit. on p. 114).
- [251]Vasileios Skliris, Michael R. K. Norman, and Patrick J. Sutton. *Real-Time Detection of Unmodelled Gravitational-Wave Transients Using Convolutional Neural Networks*. June 18, 2024. arXiv: 2009.14611 [astro-ph]. URL: <http://arxiv.org/abs/2009.14611> (visited on June 30, 2025). Pre-published (cit. on p. 114).
- [252]Adrian Alan Pol, Gianluca Germinara, Cecile Germain, Maurizio Pierini, and Agrima Seth. *Detector Monitoring with Artificial Neural Networks at the CMS Experiment at the CERN Large Hadron Collider*. July 27, 2018. arXiv: 1808.00911 [physics, stat]. URL: <http://arxiv.org/abs/1808.00911> (visited on Jan. 16, 2024). Pre-published (cit. on p. 114).
- [253]Adrian Alan Pol. “Machine Learning Anomaly Detection Applications to Compact Muon Solenoid Data Quality Monitoring”. U. Paris-Saclay, 2020 (cit. on p. 114).
- [254]Mulugeta Weldezigina Asres. “Anomaly Detection, Prognostics, and Diagnostics: Machine Learning for the Hadron Calorimeter at the CMS Experiment”. University of Agder, May 2024 (cit. on p. 114).

- [255] Franz Kevin Stehle, Wainer Vandelli, Felix Zahn, Giuseppe Avolio, and Holger Fröning. “DeepHYDRA: A Hybrid Deep Learning and DBSCAN-Based Approach to Time-Series Anomaly Detection in Dynamically-Configured Systems”. In: *Proceedings of the 38th ACM International Conference on Supercomputing*. ICS ’24: 2024 International Conference on Supercomputing. Kyoto Japan: ACM, May 30, 2024, pp. 272–285 (cit. on p. 114).
- [256] Yong Liu, Tengge Hu, Haoran Zhang, et al. “iTransformer: Inverted Transformers Are Effective for Time Series Forecasting”. In: *The Twelfth International Conference on Learning Representations*. 2024 (cit. on pp. 115, 120, 121, 136, 155, 185).
- [257] Aditya P. Mathur and Nils Ole Tippenhauer. “SWaT: A Water Treatment Testbed for Research and Training on ICS Security”. In: *2016 International Workshop on Cyber-physical Systems for Smart Water Networks (CySWater)*. 2016 International Workshop on Cyber-physical Systems for Smart Water Networks (CySWater). Vienna, Austria: IEEE, Apr. 11, 2016, pp. 31–36 (cit. on pp. 116, 184).
- [258] Chuadhry Mujeeb Ahmed, Venkata Reddy Palleti, and Aditya P. Mathur. “WADI: A Water Distribution Testbed for Research in the Design of Secure Cyber Physical Systems”. In: *Proceedings of the 3rd International Workshop on Cyber-Physical Systems for Smart Water Networks*. CySWATER ’17. New York, NY, USA: Association for Computing Machinery, Apr. 21, 2017, pp. 25–28 (cit. on pp. 116, 184).
- [259] JulienAu. *Anomaly Detection Tutorial*. 2020. URL: https://github.com/JulienAu/Anomaly_Detection_Tuto/blob/master/Data/serie2.json (cit. on pp. 116, 184).
- [260] Andrea Dal Pozzolo, Olivier Caelen, Reid A. Johnson, and Gianluca Bontempi. “Calibrating Probability with Undersampling for Unbalanced Classification”. In: *2015 IEEE Symposium Series on Computational Intelligence*. 2015 IEEE Symposium Series on Computational Intelligence. Dec. 2015, pp. 159–166 (cit. on pp. 116, 183).
- [261] Thomas Bartz-Beielstein. *Internet of Things: Online Anomaly Detection for Drinking Water Quality*. GECCO challenge. 2018 (cit. on pp. 116, 183).
- [262] Addison Howard, Bernadette Bouchon-Meunier, IEEE CIS, et al. *IEEE-CIS Fraud Detection*. 2019. URL: <https://kaggle.com/competitions/ieee-fraud-detection> (cit. on pp. 116, 183).
- [263] Prince Grover, Julia Xu, Justin Tittelfitz, et al. *Fraud Dataset Benchmark and Applications*. Sept. 22, 2023. arXiv: 2208.14417. URL: <http://arxiv.org/abs/2208.14417> (visited on Oct. 25, 2024). Pre-published (cit. on pp. 116, 183).
- [264] Amazon-Science/Fraud-Dataset-Benchmark: Repository for Fraud Dataset Benchmark (cit. on p. 116).
- [265] Guansong Pang, Chunhua Shen, Longbing Cao, and Anton van den Hengel. “Deep Learning for Anomaly Detection: A Review”. In: *ACM Computing Surveys* 54.2 (Mar. 31, 2022), pp. 1–38. arXiv: 2007.02500 [cs, stat] (cit. on p. 117).
- [266] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. “Are We Really Making Much Progress? A Worrying Analysis of Recent Neural Recommendation Approaches”. In: *Proceedings of the 13th ACM Conference on Recommender Systems*. RecSys ’19: Thirteenth ACM Conference on Recommender Systems. Copenhagen Denmark: ACM, Sept. 10, 2019, pp. 101–109 (cit. on p. 119).

- [267]Shereen Elsayed, Daniela Thyssens, Ahmed Rashed, Hadi Samer Jomaa, and Lars Schmidt-Thieme. *Do We Really Need Deep Learning Models for Time Series Forecasting?* Oct. 20, 2021. arXiv: [2101.02118 \[cs\]](#). URL: <http://arxiv.org/abs/2101.02118> (visited on Jan. 9, 2025). Pre-published (cit. on p. 119).
- [268]Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. *Are Transformers Effective for Time Series Forecasting?* Aug. 17, 2022. arXiv: [2205.13504 \[cs\]](#). URL: <http://arxiv.org/abs/2205.13504> (visited on Apr. 23, 2024). Pre-published (cit. on p. 119).
- [269]Abhimanyu Das, Weihao Kong, Andrew Leach, et al. *Long-Term Forecasting with TiDE: Time-series Dense Encoder*. Apr. 4, 2024. arXiv: [2304.08424 \[stat\]](#). URL: <http://arxiv.org/abs/2304.08424> (visited on Mar. 11, 2025). Pre-published (cit. on p. 119).
- [270]Jan Beirlant, Yuri Goegebeur, Johan Segers, and Jozef L. Teugels. *Statistics of Extremes: Theory and Applications*. John Wiley & Sons, Oct. 15, 2004. 516 pp. Google Books: [GtIYLA1TcKEC](#) (cit. on pp. 121, 124).
- [271]Alban Siffer, Pierre-Alain Fouque, Alexandre Termier, and Christine Largouet. “Anomaly Detection in Streams with Extreme Value Theory”. In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’17: The 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Halifax NS Canada: ACM, Aug. 13, 2017, pp. 1067–1075 (cit. on pp. 121–124).
- [272]James Pickands Iii. “Statistical Inference Using Extreme Order Statistics”. In: *The Annals of Statistics* 3.1 (Jan. 1975), pp. 119–131 (cit. on p. 122).
- [273]A. A. Balkema and L. de Haan. “Residual Life Time at Great Age”. In: *The Annals of Probability* 2.5 (Oct. 1974), pp. 792–804 (cit. on p. 122).
- [274]Zeyan Li, Wenxiao Chen, and Dan Pei. “Robust and Unsupervised KPI Anomaly Detection Based on Conditional Variational Autoencoder”. In: *2018 IEEE 37th International Performance Computing and Communications Conference (IPCCC)*. 2018 IEEE 37th International Performance Computing and Communications Conference (IPCCC). Nov. 2018, pp. 1–9 (cit. on p. 125).
- [275]Marco Cuturi and Mathieu Blondel. *Soft-DTW: A Differentiable Loss Function for Time-Series*. Feb. 20, 2018. arXiv: [1703.01541 \[stat\]](#). URL: <http://arxiv.org/abs/1703.01541> (visited on Apr. 14, 2025). Pre-published (cit. on p. 127).
- [276]Peter J. Huber. “Robust Estimation of a Location Parameter”. In: *The Annals of Mathematical Statistics* 35.1 (Mar. 1964), pp. 73–101 (cit. on p. 127).
- [277]Hiroaki Sakoe and Seibi Chiba. “A Dynamic Programming Approach to Continuous Speech Recognition”. In: 1971 (cit. on p. 127).
- [278]Hiroaki Sakoe. “Dynamic Programming Algorithm Optimization for Spoken Word Recognition”. In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 26 (1978), pp. 159–165 (cit. on p. 127).
- [279]Damien Garreau, Rémi Lajugie, Sylvain Arlot, and Francis Bach. “Metric Learning for Temporal Sequence Alignment”. In: *Advances in Neural Information Processing Systems*. Vol. 27. Curran Associates, Inc., 2014 (cit. on p. 127).

- [280]E Fullana, J Castelo, V Castillo, et al. *Optimal Filtering in the ATLAS Hadronic Tile Calorimeter*. Geneva: CERN, 2005 (cit. on p. [143](#)).
- [281]Matthew Inkawhich. *PyTorch: Saving and Loading Models*. Aug. 2018 (cit. on p. [173](#)).