Anomaly Detection with AutoEncoders in CMS Data Quality Monitoring

Fourth ML-INFN Hackathon: Advanced Level

13-16 November 2022
Pisa (Italy)

Alkis Papanastassiou - UniFi & INFN Firenze
Benedetta Camaiani - UniFi & INFN Firenze
Piergiulio Lenzi - UniFi & INFN Firenze

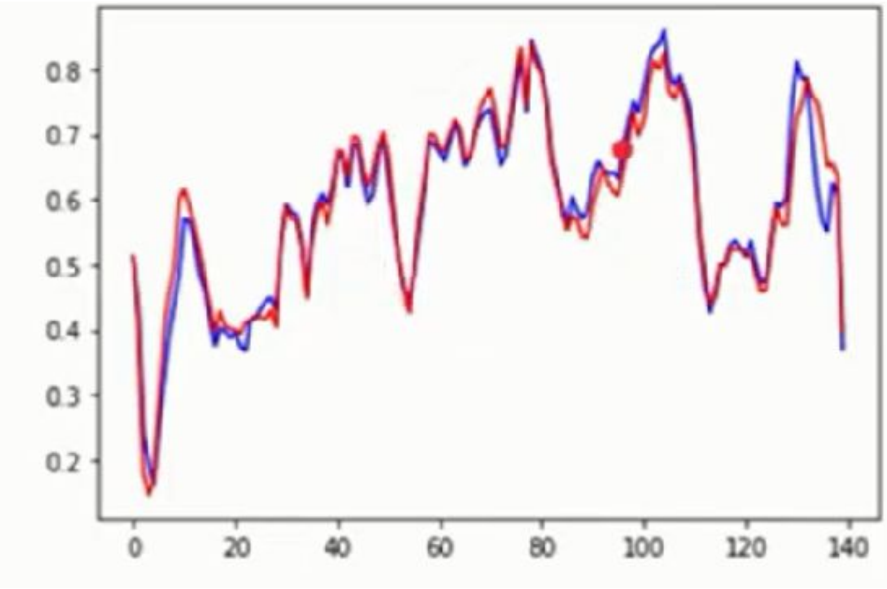
INFN

Introduction

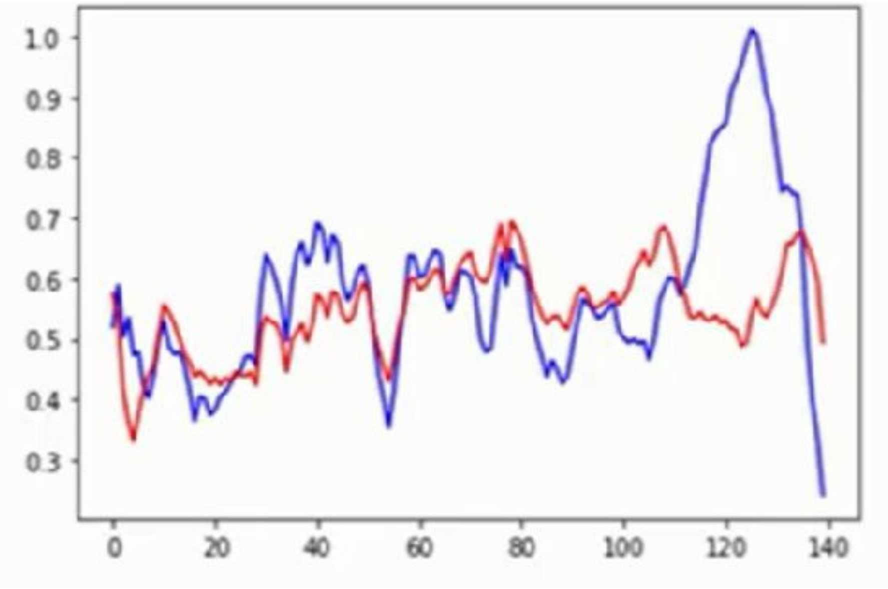
- **Anomaly Detection** (AD) consists of identifying unusual patterns or deviations from the expected behavior within data.
- It is a **crucial task** in various domains such as finance, cybersecurity, and industrial maintenance: it is critical for early detection of faults, fraud, and identifying irregular trends.
- **Time series data** is common and challenging for anomaly detection, as anomalies may not be apparent in individual data points.
- Traditional Approaches vs. Machine Learning
  - Traditional methods like Threshold-based or rule-based detection often **struggle** with complex time series data.
  - Machine Learning, particularly **AutoEncoders** (AEs), provides a powerful tool for time series anomaly detection.

Examples of AD: fraud detection

Genuine transactions

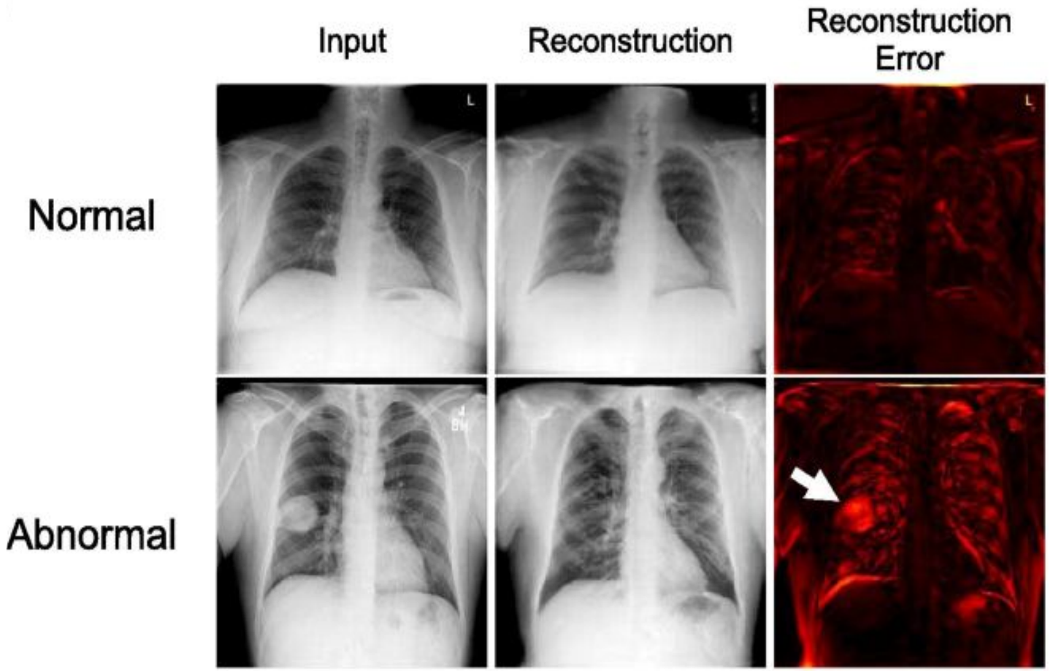


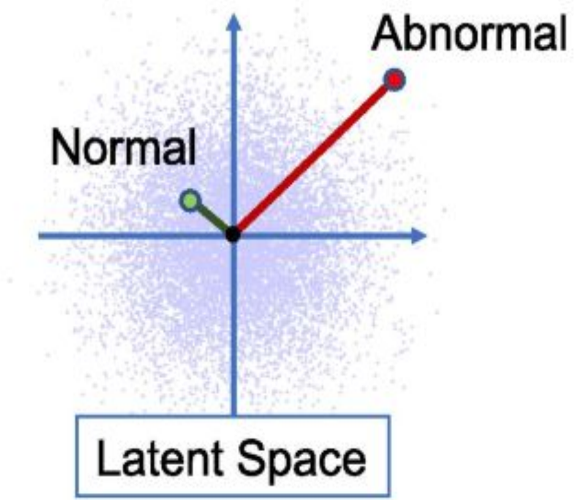
Fraudulent transaction



— input
— prediction

Examples of AD: medical anomalies





https://doi.org/10.1007/s10278-020-00413-2

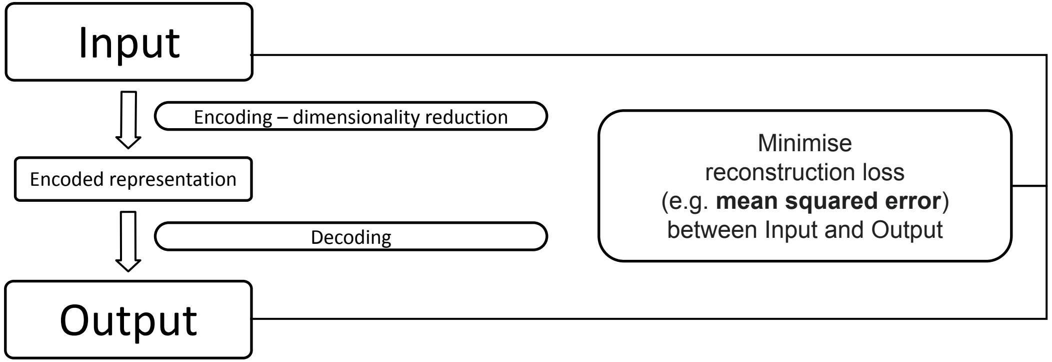
Anomaly detection with AutoEncoders

The idea:

Implementation of AEs to detect deviations from the “normal” behavior of data.

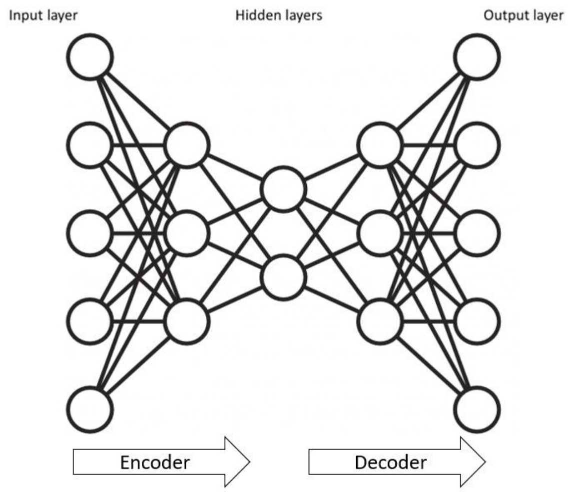
AE: a particular kind of **unsupervised** neural network capable of learning efficient codings of **unlabelled** data.

→ During **training**, the task is to reconstruct the input approximately, preserving only the most relevant aspects of it.



→ During **testing**, the AE is presented with possibly anomalous data: peaks in the reconstruction loss are probably related to anomalies.

Different types of AutoEncoders: Undercomplete



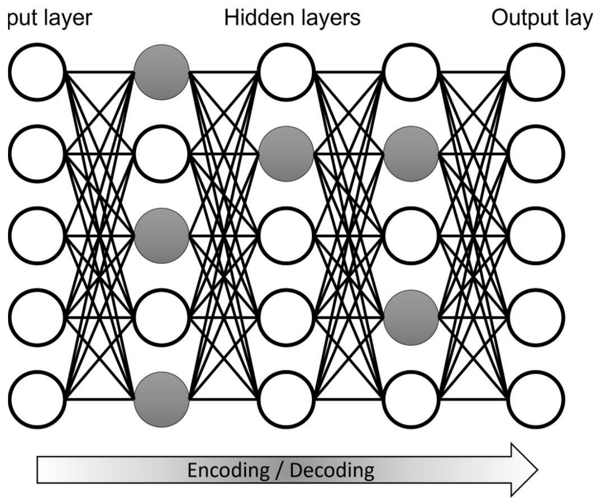
Undercomplete AutoEncoder

Encoding via dimensionality reduction

The simplest architecture for constructing an AE:

- **Constrain** the number of nodes present in the hidden layer(s) of the network, **limiting** the amount of information that can flow through it.
- Provides a more powerful (nonlinear) generalization of **PCA**.
- Has no explicit regularization term - we simply train our model according to the reconstruction loss.
- The only way to ensure that the model isn't memorizing the input data is to sufficiently restrict the number of nodes in the hidden layer(s).

Different types of AutoEncoders: Sparse



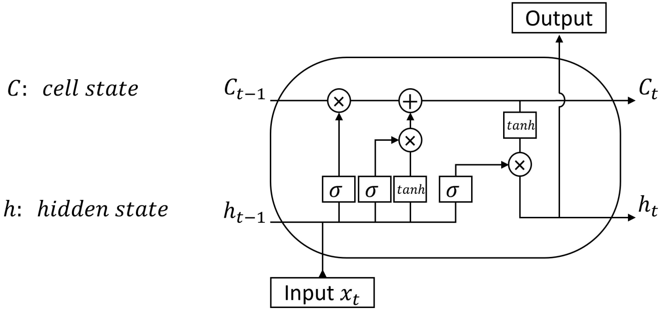
Sparse AutoEncoder

Encoding via
sparsity constraints

An alternative method for introducing an information bottleneck:

- Build the loss function such that activations within a layer are penalized.
- For any given observation, the network have to learn an encoding and decoding relying only on the activation of a small number of neurons.
- Allows for a separation between latent state **representation** and **regularization** of the network: one can choose a latent state representation (ie. encoding dimensionality) in accordance with what makes sense given the context of the data while imposing regularization by the sparsity constraint.

Different types of AutoEncoders: LSTM

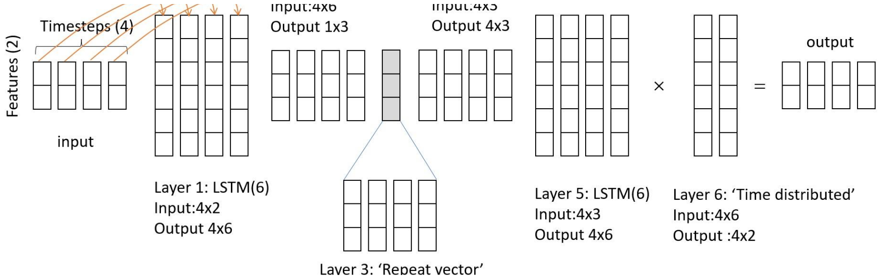


A model capable of learning the complex dynamics within the temporal ordering of input sequences.

- LSTM is a type of Recurrent Neural Network (RNN) in which each neuron is built as multiple copies of the same unit.
- Each unit uses an internal memory to remember or use information across long input sequences.
- Each layer see not one sample at a time but a certain window of them

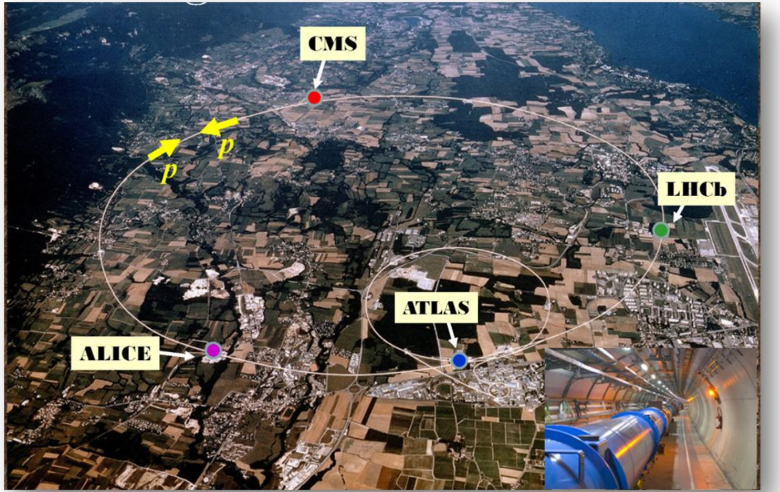
LSTM AutoEncoder

More suitable for time-series

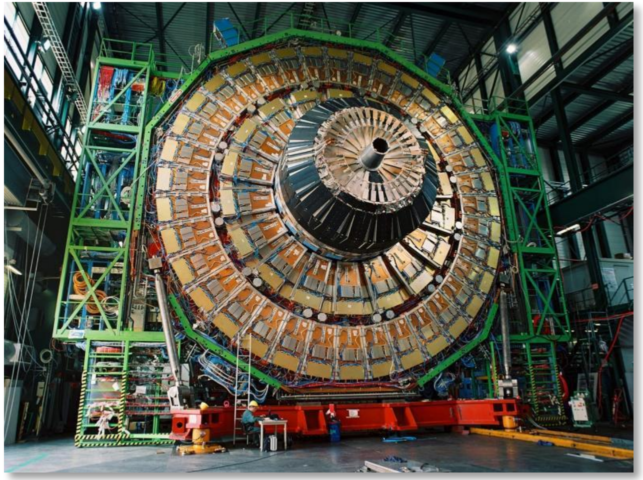


Application of AD to HEP: Data Quality Monitoring @ CMS

LHC and the CMS experiment



- The Large Hadron Collider (LHC) is the largest and most powerful particle accelerator in the world, situated at the CERN near Geneva
- The LHC accelerates two beams of protons that are made to collide at four points, around which the main experiments are located

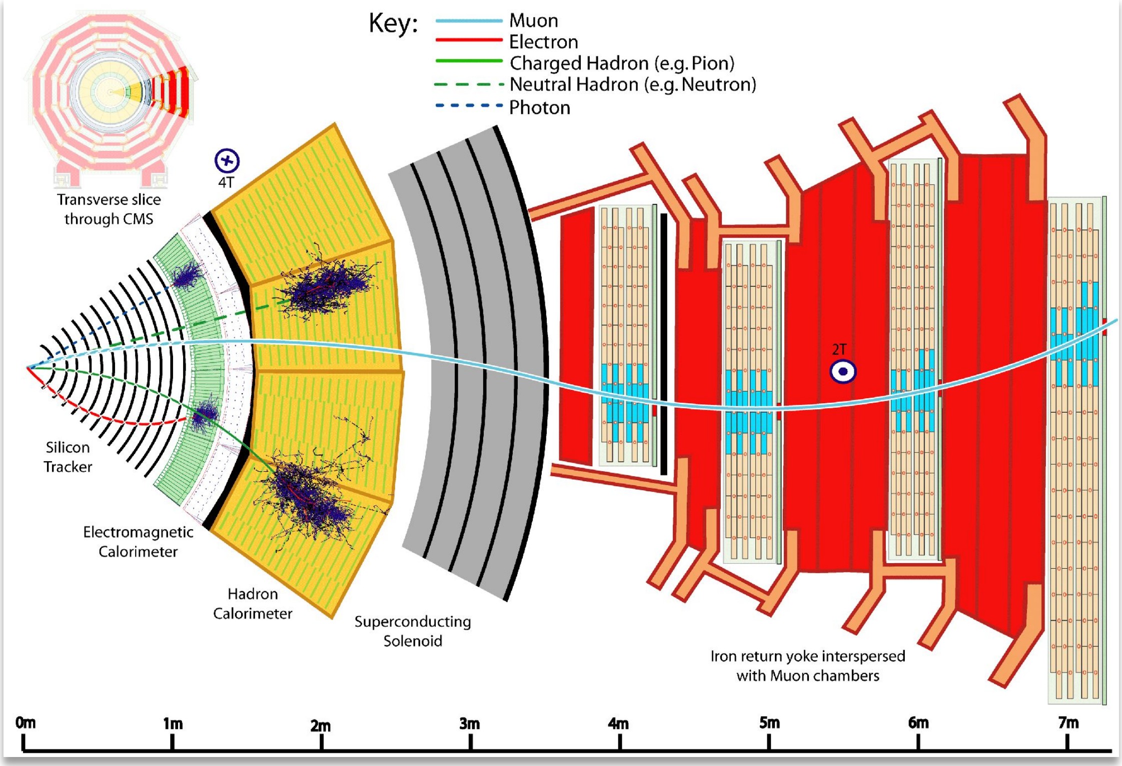


- The CMS experiment is one of the main detectors installed at the LHC and it is a multi-purpose apparatus for high energy physics

CMS sub-detectors

- CMS is composed of a complex system of sub-detectors to detect electrons, photons, muons and hadronic jets

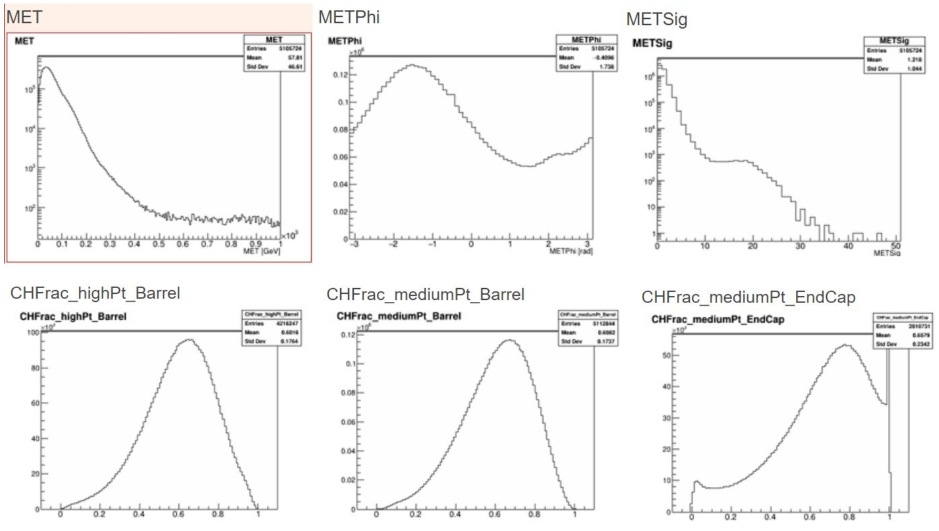
- The only particles that escape from detection are neutrinos but their presence can be deduced by the imbalance of the total transverse momentum (MET)



LHC & CMS operations

- ✔ LHC operates in «fills»: each fill is a proton injection inside the accelerator. CMS runs during LHC fills.
- ✔ Data gathered in luminosity sections, **lumisections** in short (LSs), corresponding to **~23 s of data taking**.
- ✔ LSs are grouped in **runs**.
- ✔ Issues in the different detectors can arise due to various factors, such as radiation damage, electronic noise, and aging of components.
- ✔ **Data Quality Monitoring (DQM)** essential for data quality assessment.

Data Quality Monitoring (DQM)

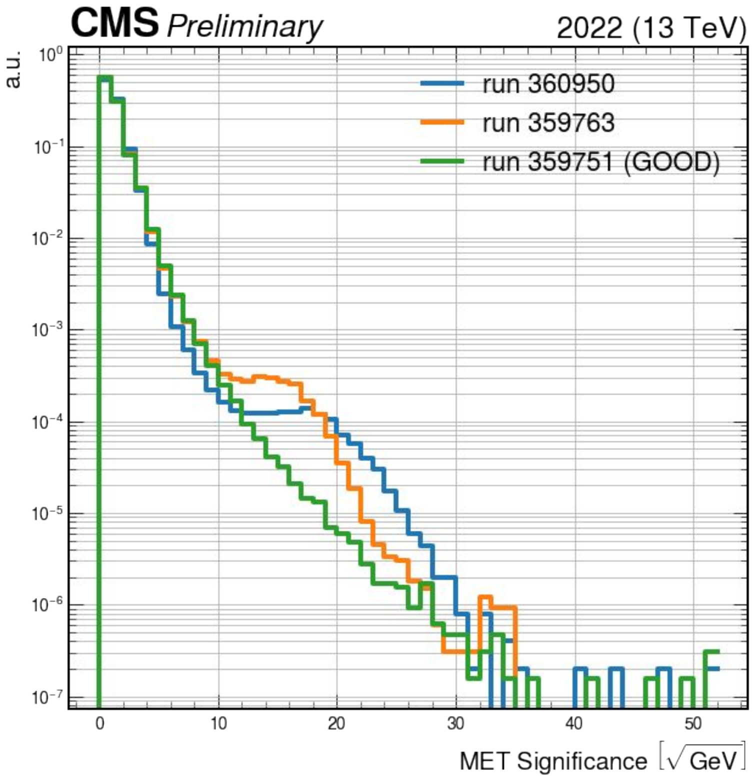


The monitoring of data quality is crucial both **online**, during the data taking, to promptly spot issues and act on them, and **offline**, to provide analysts with datasets that are cleaned against the occasional failures that may have crept in.

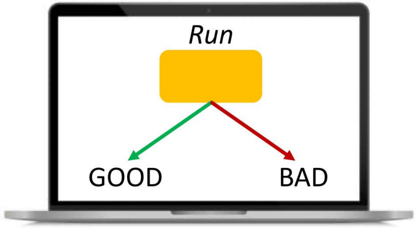
The quantities that are checked are usually referred to as **Monitor Elements** (MEs)

MEs frequently used for DQM purposes are for example quantities pertaining to **hadronic jets** and **missing transverse momentum** (**MET**): *Jet e missing energy* (JME) ME

GOOD and *BAD* runs



For the specific case of quantities pertaining to hadronic jets and MET, an issue in a few LSs would cause the entire run to be flagged as problematic (***BAD***), and thus removed from the pool of "good-for-analysis" data (***GOOD***).

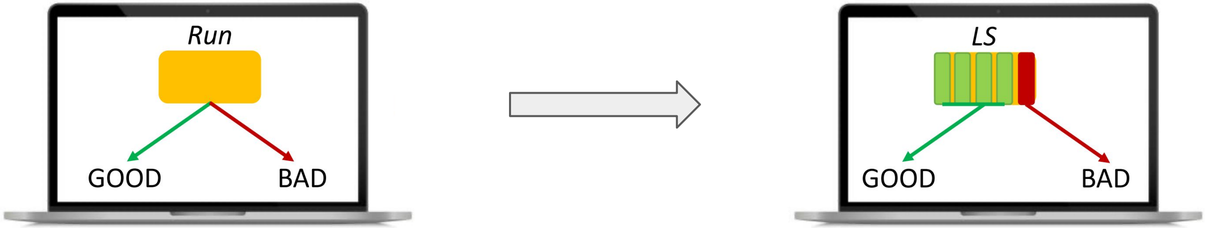


In the plot: histograms of a ME (MET Significance) **for three different runs** chosen as example, one flagged ***GOOD*** (**green**) and two presenting an anomaly, therefore flagged ***BAD*** (**blue** and **orange**).

Per-Lumisection data

The possibility of accumulating MEs per-LS enables the saving of higher amounts of data from runs presenting only a limited set of anomalous LSs.

Per-LS data is analysed offline during **Data Certification** (DC), i.e. the final step of quality checks performed by DQM on recorded collision events



If we look at a given run with a per-LS granularity, we can identify the LSs showing the anomaly and flag them as *BAD*

Given the **high number** of LSs to be analyzed for each run, an **automated approach** for DC is required

AEs can represent the perfect tool for this purpose

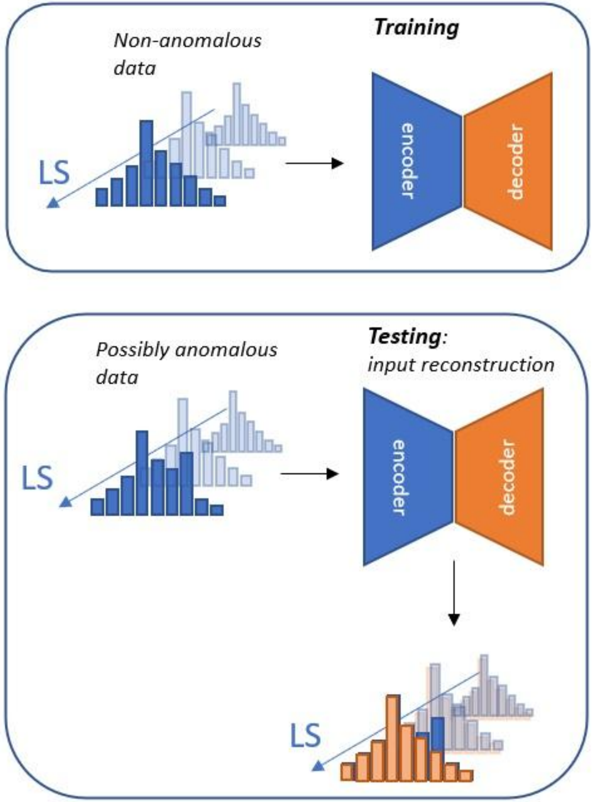
Implementing AEs for per-Lumisection AD

Main objective

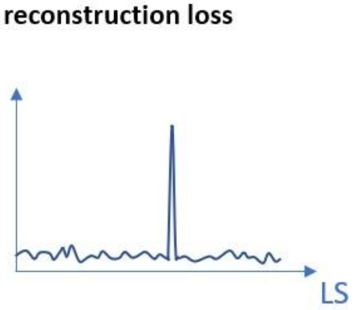
Develop a **model** capable of analyzing problematic runs with a per-LS granularity, filtering out anomalous LS data (flagging it as *BAD*) and retaining what's *GOOD*.

- Based on AEs.
- Trained on *GOOD* data.
- Tested on anomalous or possibly anomalous runs.
- Optimized using known anomalies.
- Able to perform well independently of the drop in luminosity during the run.

How does it work?



- **Training** on non-anomalous data from *GOOD* runs: histograms of specific MEs are fed to the model with a LS granularity to allow the AE to learn a **«normal» non-anomalous behavior** of that specific ME.
- Minimization of the **reconstruction loss**, a measure of the distance between the input and output of the AE: $MSE = \sum_{i=1}(y_i - \hat{y}_i)^2/n$, where **y** and **ŷ** are respectively the input and the output of the AE and *n* is the #bins.



Possibly anomalous runs under investigation are tested by looking again at the reconstruction loss: **peaks** in this function **indicate LSs containing histograms that deviate from the learned behavior**.

→ **Deviations** from the learned «normal behaviour» could in principle be **detected**.

Inputs & rescaling

- We look at per-LS data for a specific run (train or test).
- Some of the MEs used are:

  'CHFrac_highPt_Barrel', 'CHFrac_mediumPt_Barrel', 'JetMass_highPt_Barrel', 'MET', 'METSig', 'METPhi'.

- **Get rid of luminosity dependence**

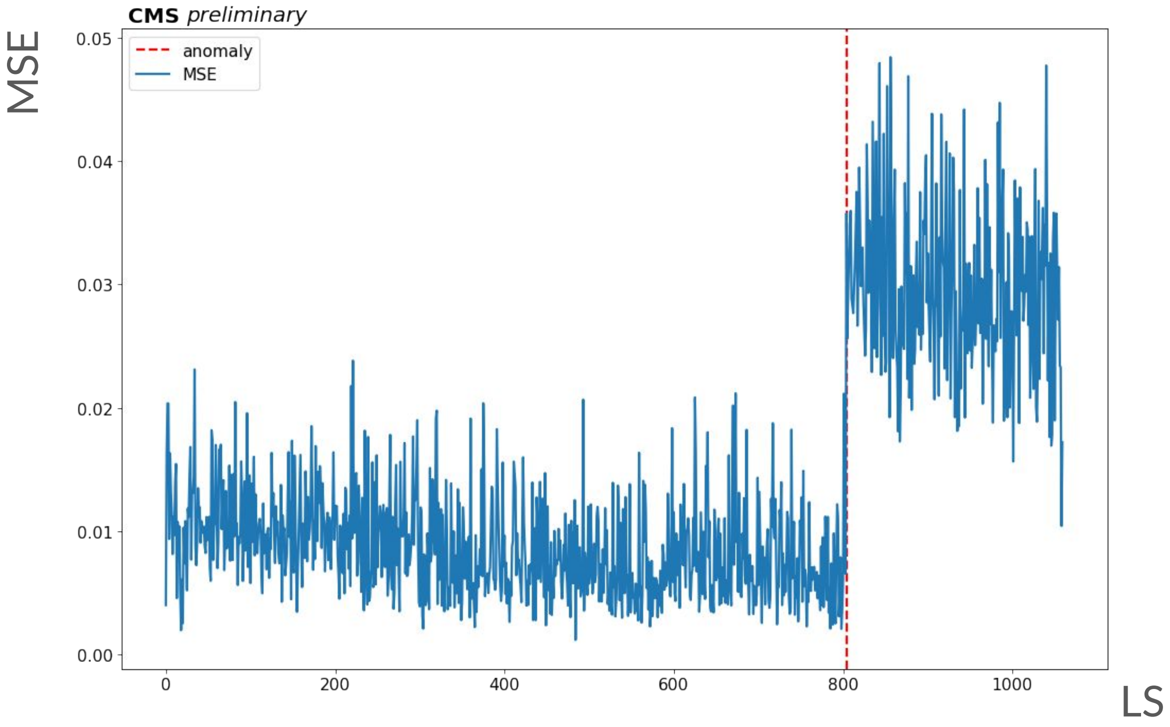
  For each ME the input is structured by adding the luminosity of the given lumisection as a **further bin**, resulting in an input shape:

  (#LSs, #bins+1)

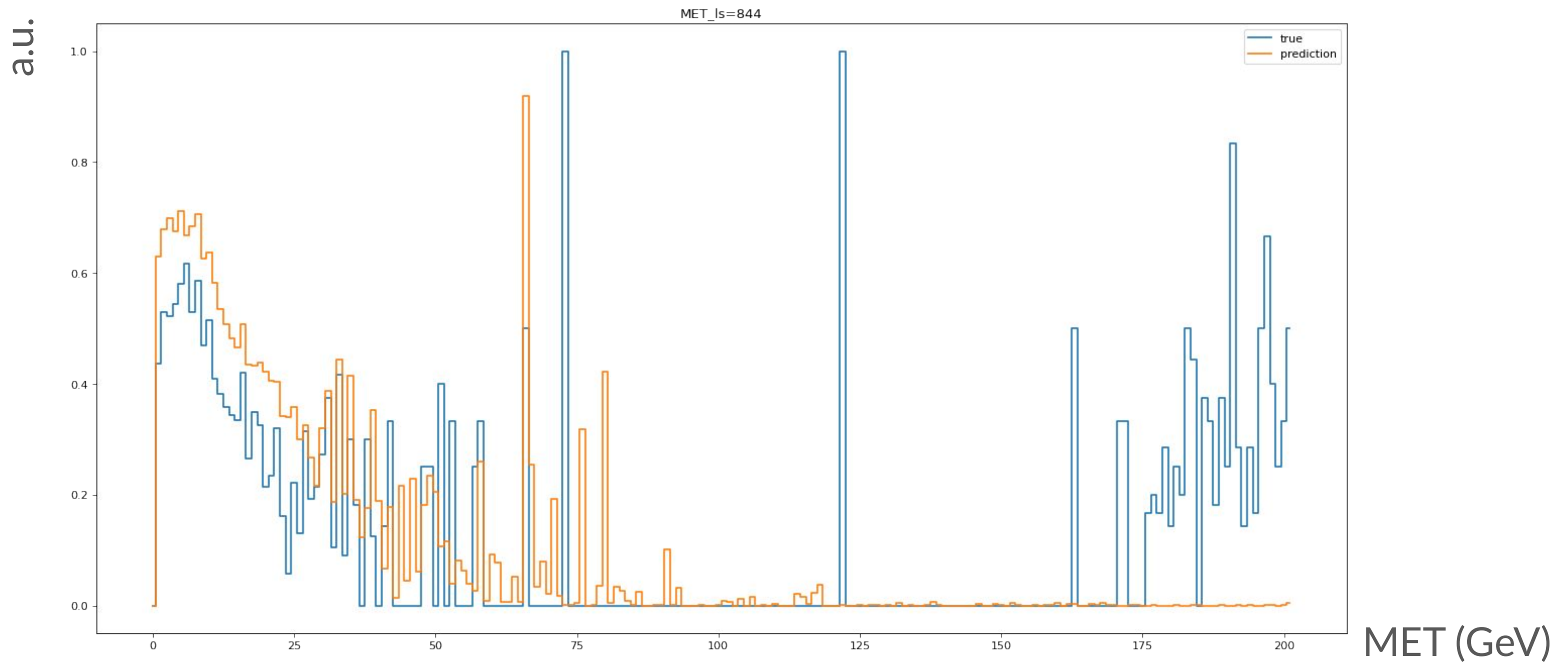
- The **input is rescaled** to the [0,1] interval bin by bin:

$$\boxed{input = \frac{input - \min_{run} input}{\max_{run} input - \min_{run} input}}$$

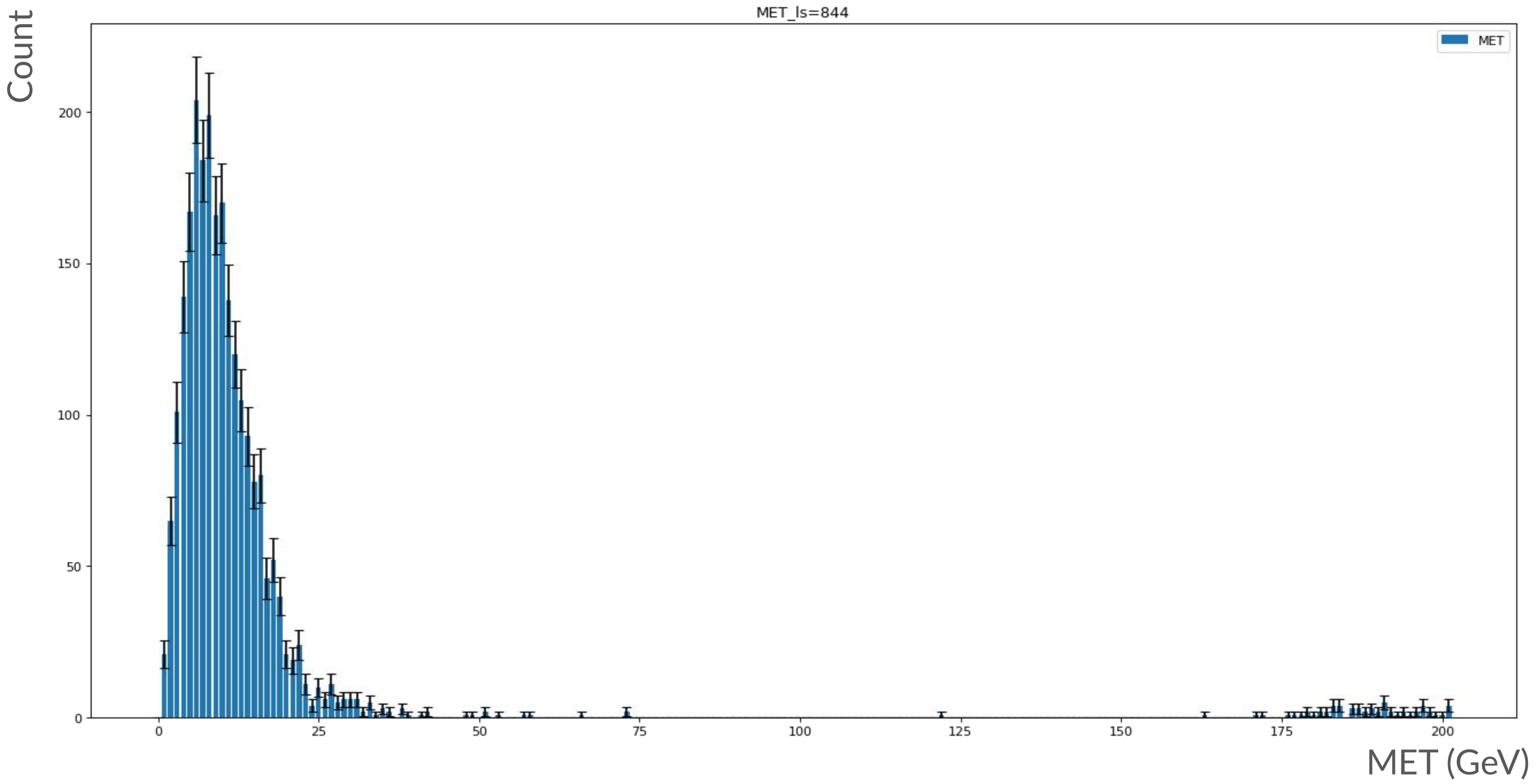
One *BAD* run



One major jump in **MSE** for ‘MET’: the model was unable to reconstruct correctly the input for the following test lumisections.



The corresponding histogram contains a “bump” anomaly that the AE **couldn’t reconstruct**, leading to the MSE jump.
All the following lumisections contain the evolution of this anomaly.


MET_ls=844
MET
Count
0
50
100
150
200
0
25
50
75
100
125
150
175
200
MET (GeV)

Optimization

A ***metric*** must be chosen in order to perform the **optimization** of the different architectures.

→ we quantify how steep is the step in the MSE obtained by *testing* the AEs on runs containing known anomalies.

We want our *metric* such that:

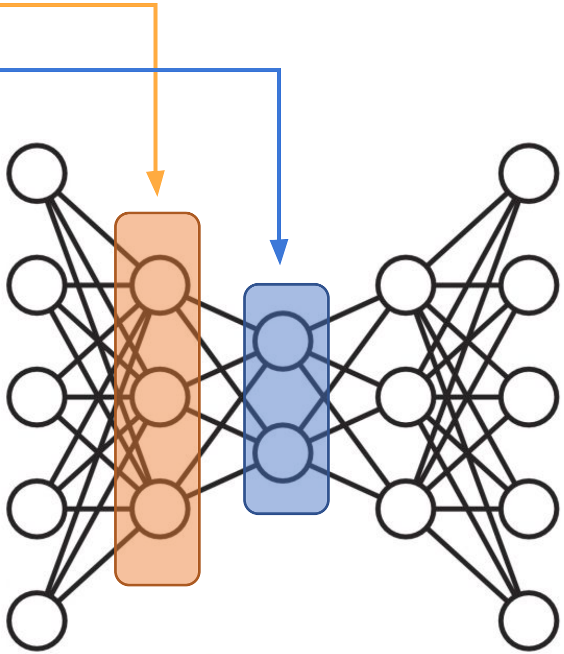
1. It increases with the size of the step.
2. It decreases with standard deviation of the MSE before the step.

$$metric = \frac{|mean(MSE_{after}) - mean(MSE_{before})|}{stddev(MSE_{before})}$$

Optimized Dense Undercomplete Autoencoder

Hyperparameters to be optimized

- encoding_dim
- encoding_dim_2
- batch_size



Optimized:
encoding_dim = 103
encoding_dim_2 = 82
batch_size = 42

learning_rate = 10^{-7}

Summary

Summary

A few relevant points:

- **Anomaly detection** is important both in the discovery of **relevant patterns** and in maintaining **data quality** and reliability across diverse applications, included **HEP**.
- Challenges are posed by anomalies in large-scale apparatuses, especially in **complex systems** like the CMS detector at CERN.
- Detecting anomalies at a finer granularity, specifically at the per-LS level, to enhance precision and recovery of **valuable** data.
- **Automation** for per-LS analysis: necessary due to the high number of LSs.
- The adoption of **machine learning** as a powerful solution for anomaly detection, considering its ability to handle complex patterns and large datasets.
- **AutoEncoders** as a robust unsupervised learning technique for anomaly detection, able to capture complex patterns within data, in particular **temporal sequences of histograms**.