



Finanziato  
dall'Unione europea  
NextGenerationEU



Ministero  
dell'Università  
e della Ricerca



Italiadomani

PIANO NAZIONALE  
DI RIPRESA E RESILIENZA



Centro Nazionale di Ricerca in HPC,  
Big Data and Quantum Computing



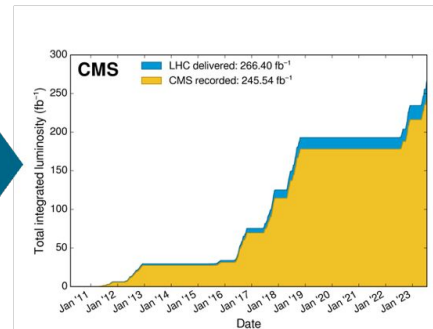
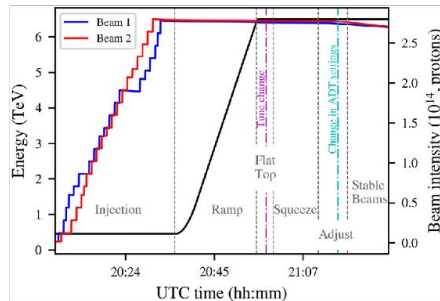
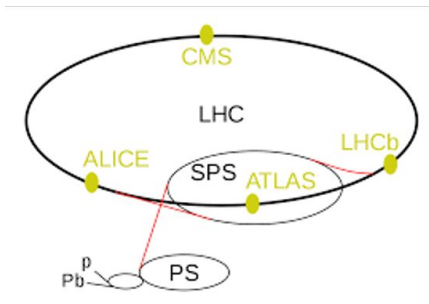
Centro Nazionale di Ricerca in HPC,  
Big Data and Quantum Computing

## Anomaly detection for the online monitoring of the CMS Muon System (UC 2.2.1)

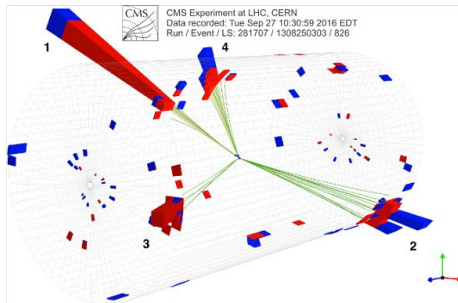
**Marco Buonsante, Federica Simone, Rosamaria Venditti** (INFN e Università/Politecnico di Bari)

Spoke2 WP2 Meeting, 21 Maggio 2024

# Data taking at CMS



Recorded event



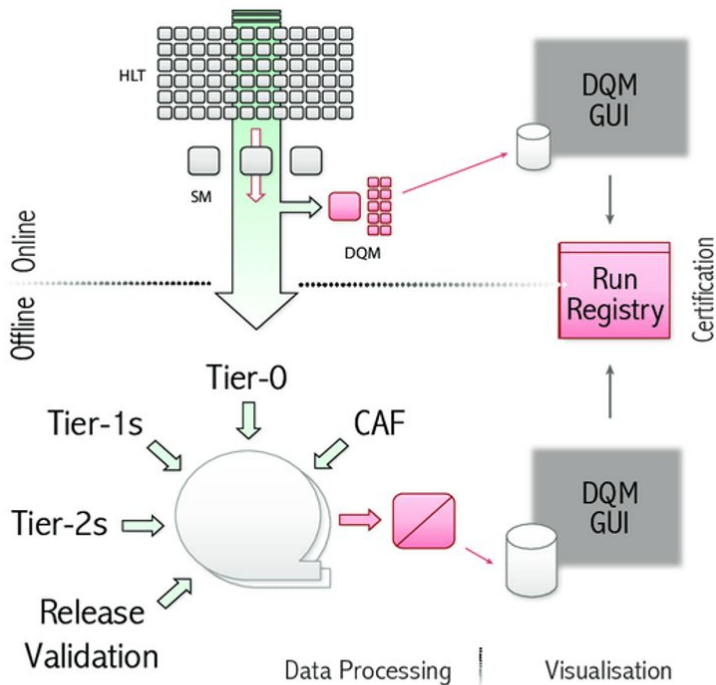
Lumisection (LS)



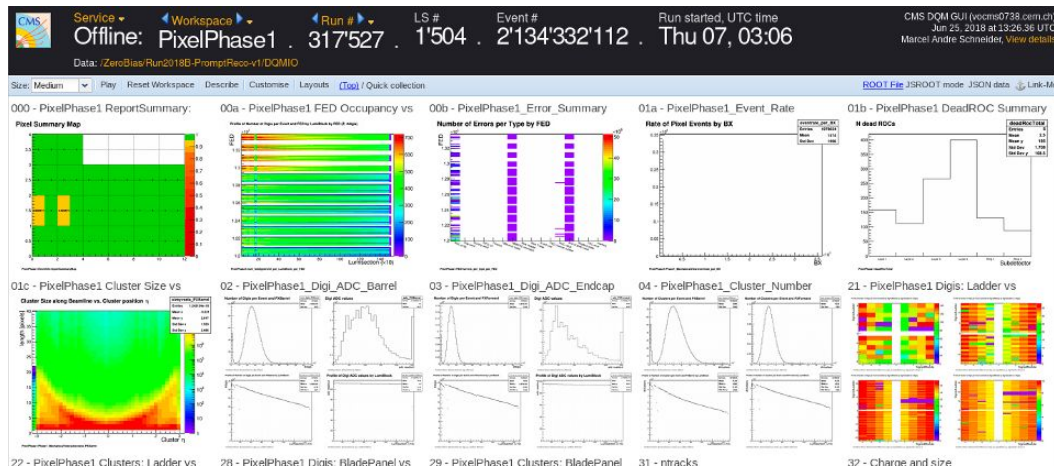
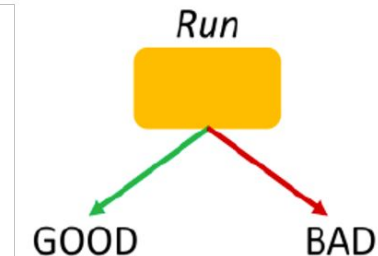
"Run" → thousands of LS



# Data Quality Monitoring (DQM)

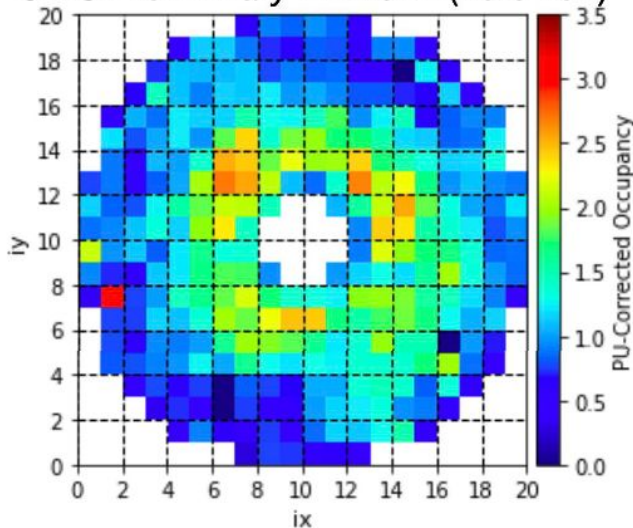


**Online DQM:** promptly raise alarms during data taking



# Monitoring Elements (ME)

**CMS Preliminary EE 2022 (13.6 TeV)**



**ME:** quantities (i.e. histograms) which are inspected either automatically or by experts

**ME in the online DQM:** histograms are filled in real time during the run.

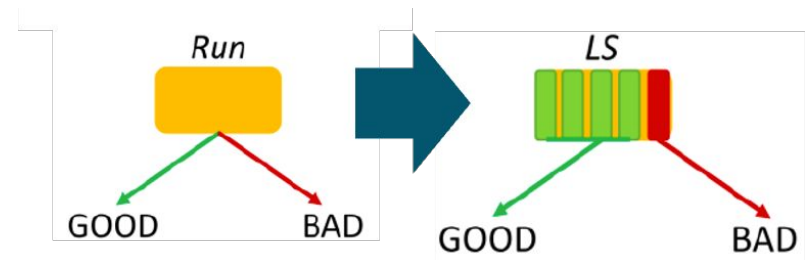
**ME in the offline DQM:** histograms are available for inspection with per-run granularity

ECAL occupancy images

# DQM challenges and limitations



- Online monitoring is a highly time-sensitive operational task
- Impossible to foresee or simulate all potential failure scenarios
- Limited time granularity (run) can potentially hide transient issues only affecting few lumisections
  - Drawback: per-LS approach increases the number of MEs by a factor  $\mathcal{O}(10^3)$

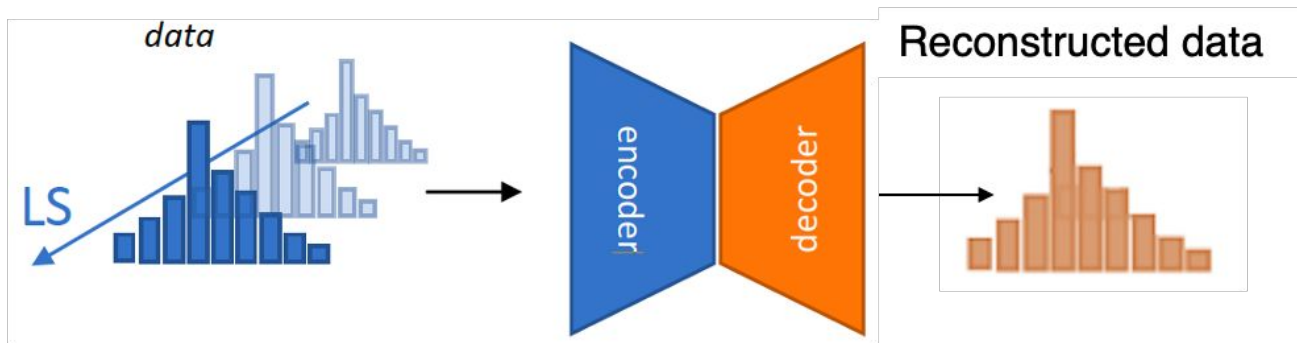


# Machine Learning for the Online DQM

6

- High number of features
- Large class imbalance (most data is good)
- Non-exhaustive definition of failures

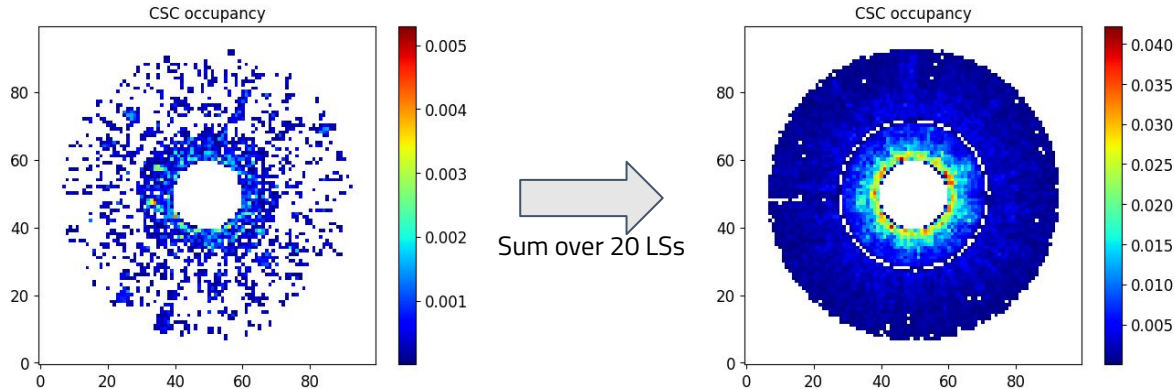
Unsupervised learning for anomaly detection



# The Muon System use-case: data pre-processing

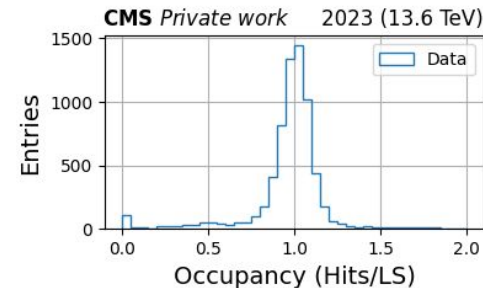
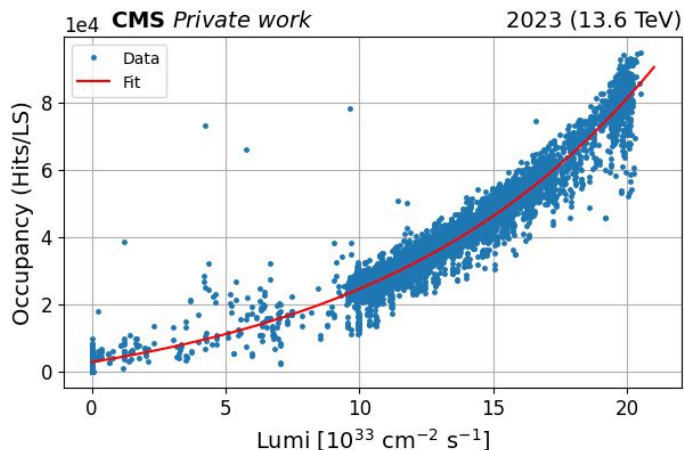
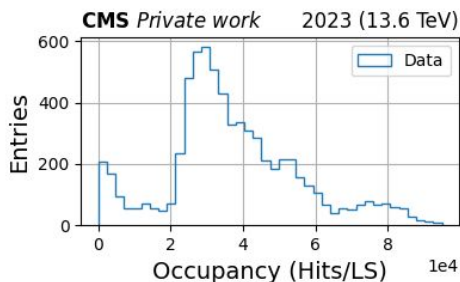
7

- Basic idea: train a model on occupancy plots from the different detector regions
- Need for preprocessing:
  - Muon System characterised by low occupancies (can't use a single lumisection for training)
  - Occupancy vs lumisection not uniform (depends on instantaneous luminosity)
  - Need to correct for the expected geometrical distribution of incoming particles (strong eta dependency)



# The Muon System use-case: data pre-processing

Entries normalised to instantaneous luminosity

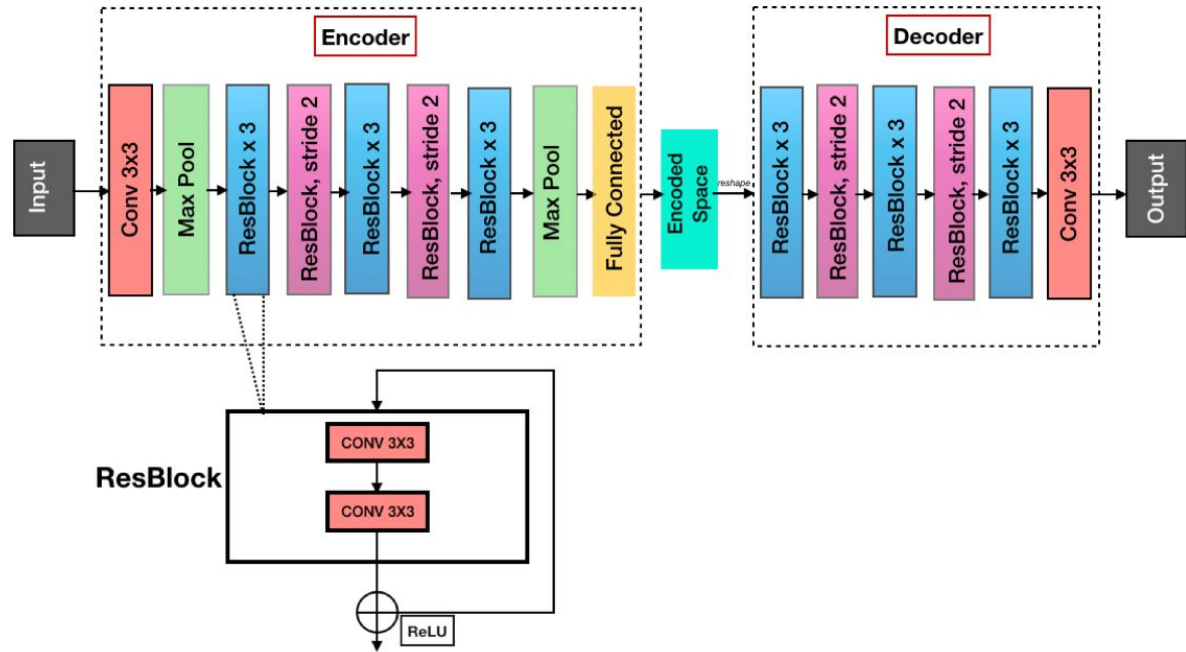




# Model architecture

ResNet for unsupervised reconstruction of images

**Input:** 100x100 pre-processed images (small dataset, 300 examples)

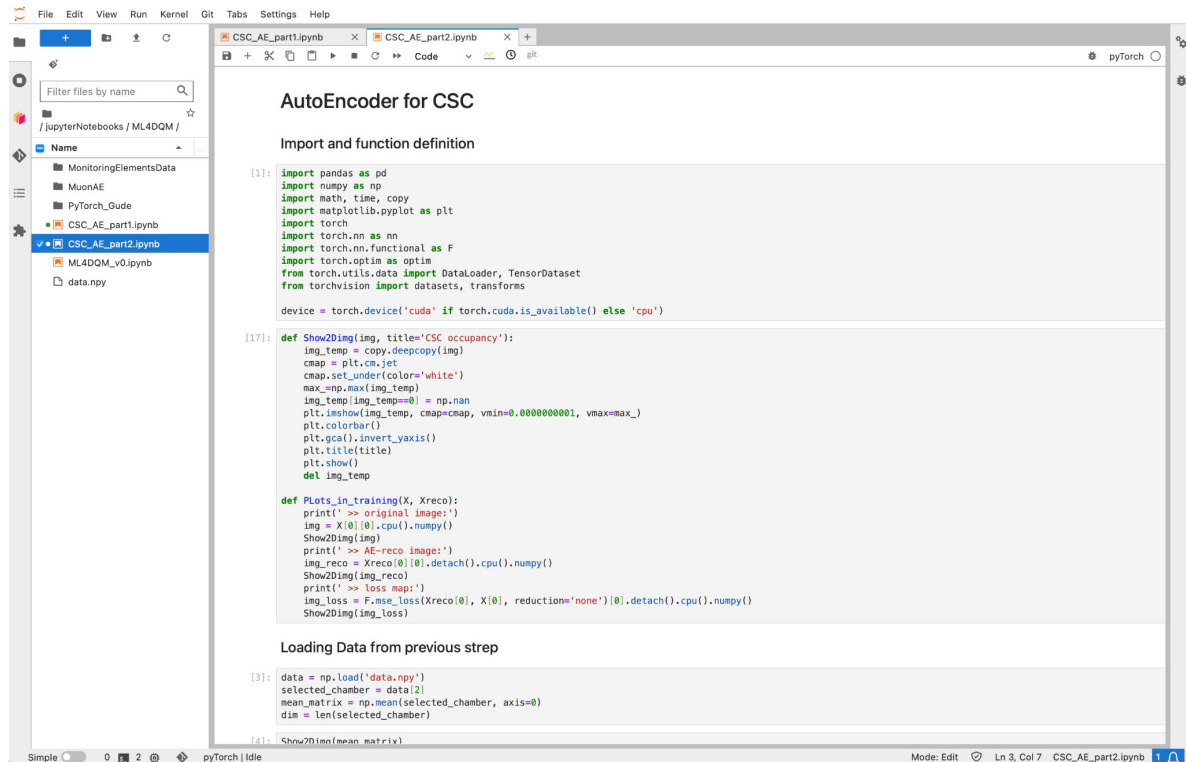


<https://doi.org/10.48550/arXiv.2309.10157>

# Computing resources for training

Leveraging the ReCas JupyterHub service with GPU

[https://jvino.github.io/cluster-hpc-gpu-guide/interactive\\_services/jupyter-lab-for-recas-users/](https://jvino.github.io/cluster-hpc-gpu-guide/interactive_services/jupyter-lab-for-recas-users/)



The screenshot displays a JupyterLab environment with a file browser on the left and a code editor on the right. The code editor is titled 'AutoEncoder for CSC' and contains Python code for training an autoencoder. The code includes imports for pandas, numpy, math, time, copy, matplotlib, and torch. It defines a device variable, a Show2Ding function for visualization, and a Plots\_in\_training function for monitoring training progress. The code also shows data loading from a previous step.

```
File Edit View Run Kernel Git Tabs Settings Help
CSC_AE_part1.ipynb CSC_AE_part2.ipynb
Filter files by name
/jupyterNotebooks / ML4DQM
MonitoringElementsData
MuonAE
PyTorch_Guide
CSC_AE_part1.ipynb
CSC_AE_part2.ipynb
ML4DQM_v0.ipynb
data.npy

AutoEncoder for CSC
Import and function definition
[11]: import pandas as pd
import numpy as np
import math, time, copy
import matplotlib.pyplot as plt
import torch
import torch.nn as nn
import torch.nn.functional as F
import torch.optim as optim
from torch.utils.data import DataLoader, TensorDataset
from torchvision import datasets, transforms

device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')

[17]: def Show2Ding(img, title='CSC occupancy'):
    cmag = copy.deepcopy(img)
    cmap = plt.cm.jet
    cmap.set_under(color='white')
    max_val = np.max(img)
    img_val[img_val==0] = np.nan
    plt.imshow(img_val, cmap=cmap, vmin=0.000000001, vmax=max_val)
    plt.colorbar()
    plt.gca().invert_yaxis()
    plt.title(title)
    plt.show()
    del img_val

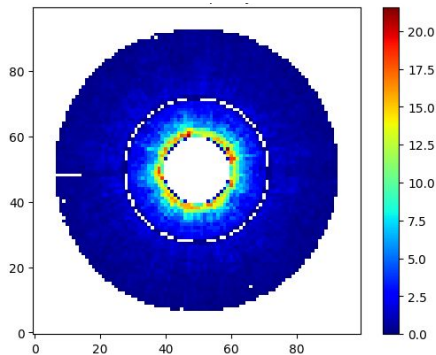
def Plots_in_training(X, Xreco):
    print(' >> original image:')
    img = X[0][0].cpu().numpy()
    Show2Ding(img)
    print(' >> AE-reco image:')
    img_reco = Xreco[0][0].detach().cpu().numpy()
    Show2Ding(img_reco)
    print(' >> loss map:')
    img_loss = F.mse_loss(Xreco[0], X[0], reduction='none')[0].detach().cpu().numpy()
    Show2Ding(img_loss)

Loading Data from previous step
[31]: data = np.load('data.npy')
selected_chamber = data[2]
mean_matrix = np.mean(selected_chamber, axis=0)
dim = len(selected_chamber)

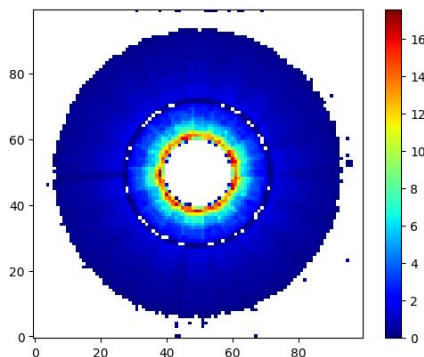
[41]: Show2Ding(mean_matrix)
```

# Preliminary results

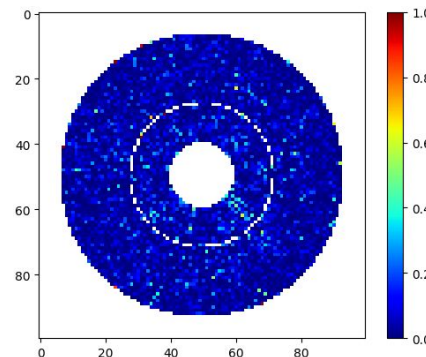
Original image:



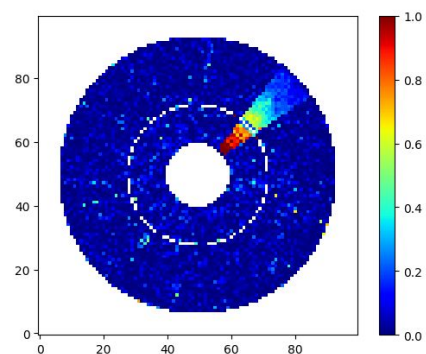
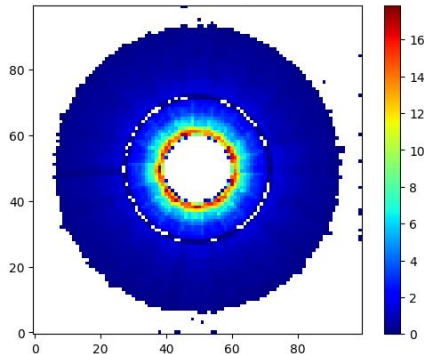
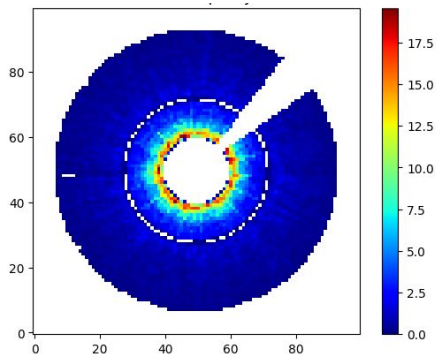
AE Reco-image:



Normalized loss map:



Predictions on a good example:



Predictions on a trivial example with an "artificial hole":

## To-dos and summary

- Ongoing development for automatised anomaly detection for the Muon System <https://github.com/fsimone91/MuonAE>
- Requested some muon-related ME to be included in the DQM file production → larger statistics for training will be available soon <https://github.com/cms-sw/cmssw/pull/44610>
- Preliminary results on a specific detector region (CSC endcap)
  - To-do: extend to other regions/subdetectors
  - To-do: simulate anomalies (e.g. dead channels) to access performance
- This development will be included in the next CMS ML Hackathon dedicated to DQM (July 2024)
- Abstract submitted to CHEP2024 <https://indico.cern.ch/event/1338689/abstracts/175693/>