Development of a computational framework for Neural Network traning and analysis in the context of the CMS experiment

Reunión Anual de la División de Partículas y Campos (RADPyC2026)
17 de Junio del 2026

Alfredo Castaneda

Universidad de Sonora

"El saber de mis hijos hará mi grandeza"

Motivation

- Provide an analysis framework for newcomers to CMS and people interested in particle physics
- Reduce complexity associated with CMS environment, cluster configuration and software dependencies associated
- Lower the learning curve required to perform a complete CMS analysis workflow
- Enable students and researches to focus on physics rather than software configuration
- Provide tools for people to analyze CERN open data (~5 PB) (in progress)

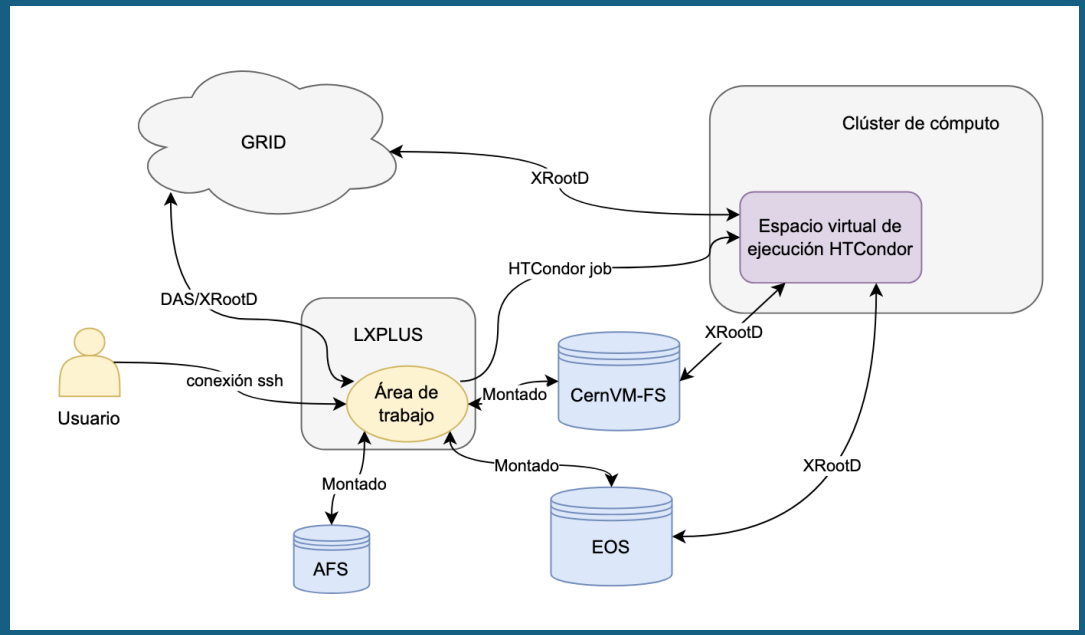
Computational requirements

CMS User?

Yes

No

- Access to LXPLUS CERN computing cluster
  - Official datasets stored and accessible through the GRID
  - **Parallelization** of processes through HTCondor system
  - **Storage** in CERN EOS system

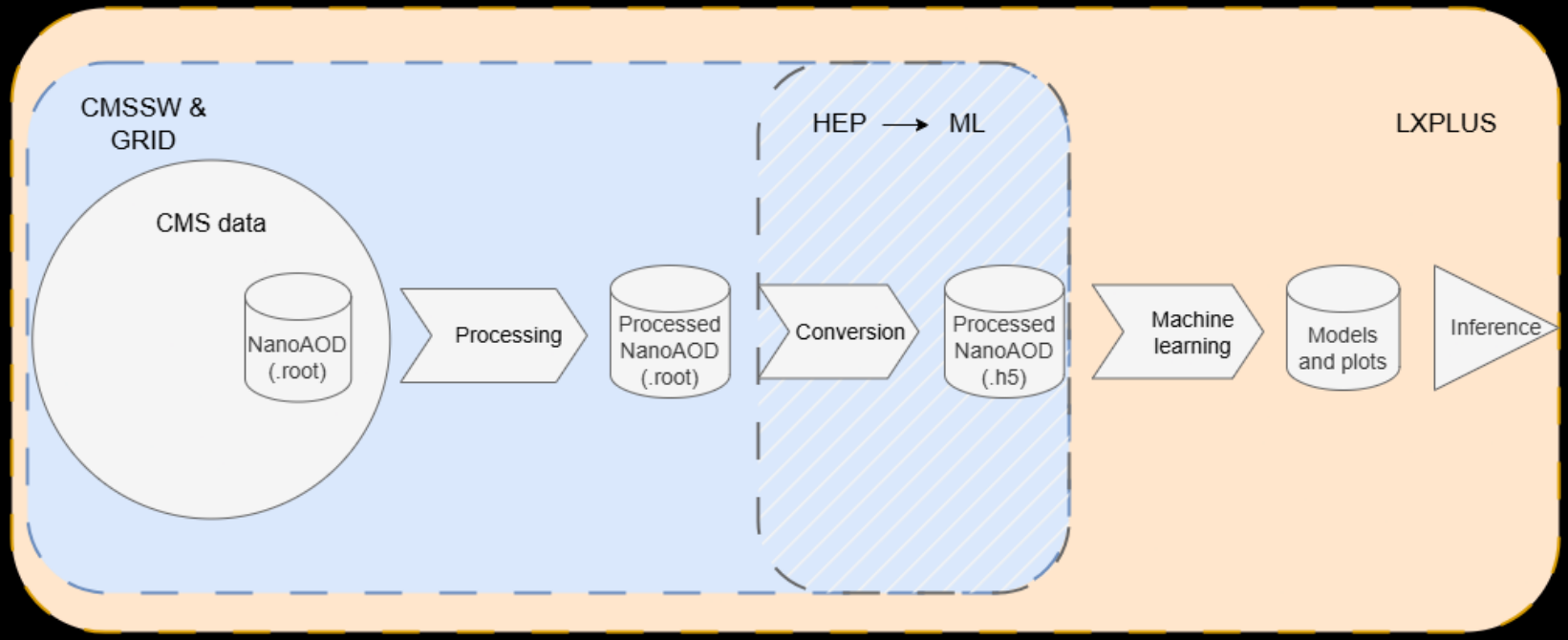


- Access to common university HCP cluster
  - Datasets from CERN Open data stored locally or accessible to server connection
  - **Parallelization** or process (SLURM, HTCondor, other)
  - **Storage** system (Lustre, other)

Yuca: Clúster de cómputo de alto rendimiento y virtualización

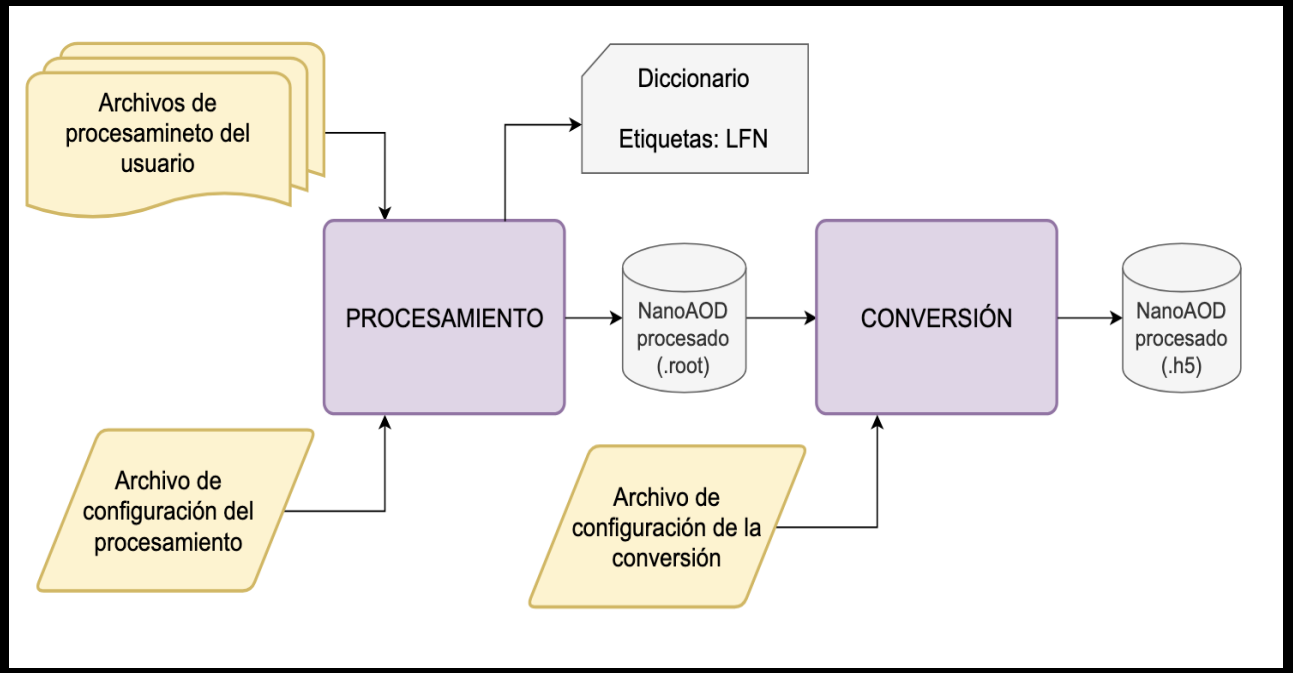
Host Name: yuca.unison.mx
Open OnDemand: https://yucaood.acarus.unison.mx
IP: 148.225.111.153
Sistema Operativo: Alma Linux 9.4
Número de Nodos: 24 (CPU) + 6 (GPU) + 6 (Virtualización) + 2 (Maestro) + 2 (Login) = 40 nodos
Número de Cores: 1536 (CPU) + 384 (Virtualización) + 288 (GPU) CPU = 2208 CPU cores
+ 6656 GPU cores
Número de GPGPU: 6 nc
Memoria Total Ram: 24 +
Almacenamiento: Almace
Almacenamiento SSD: 450

Overview of the framework



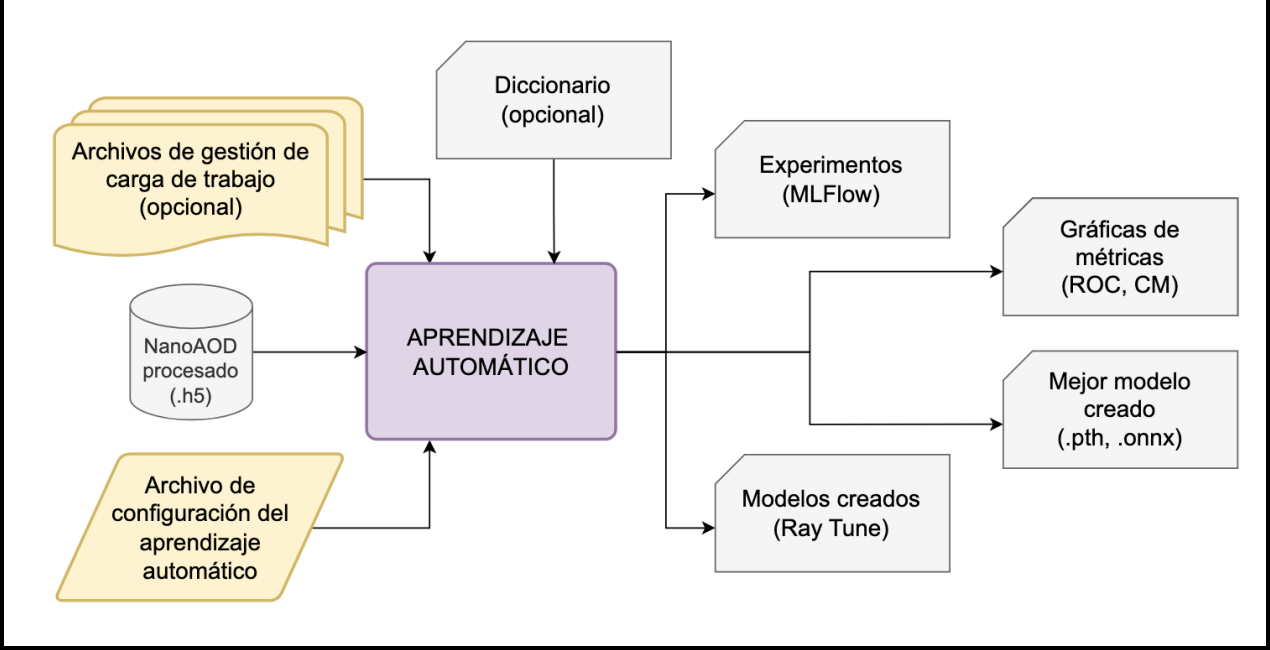
- Modular framework
  - Processing of datasets (nanoAOD)
  - Conversion to HDF5 datasets (if needed)
  - ML training

Dataset processing module



- **User provides the dataset name (LFN)** corresponding to a CMS NanoAOD sample
- A **logic file (Python script)** defines the event selection, filtering criteria, and variable extraction according to the analysis goals.
- The framework processes the original **NanoAOD datasets** and applies the user-defined logic.
- A new **skimmed dataset** is produced, containing only the relevant events and variables.
- The previous is done using HTCondor for parallelization
- The skimmed ROOT files are converted to **HDF5 format** to facilitate efficient access from Python-based machine learning tools.
- Simplified interaction via YAML configuration files

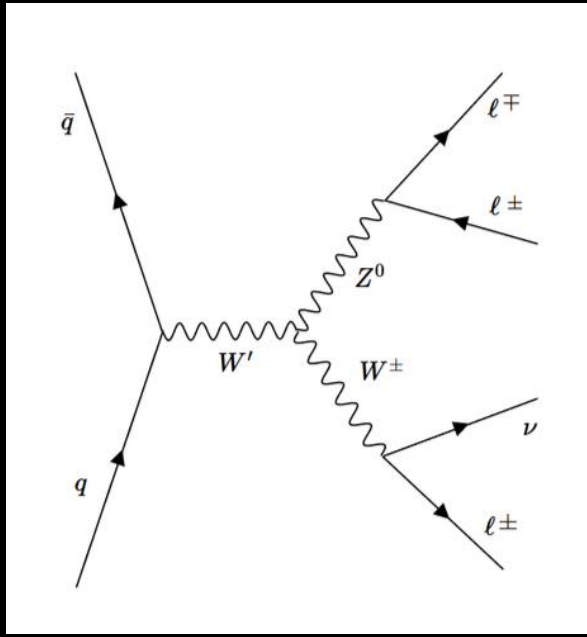
ML training module



- Generate an **Apptainer container** that encapsulates the complete execution environment.
- Execute the ML training pipeline using the container and a user-defined YAML configuration file.
- Load and preprocess the training and validation datasets.
- Configure **Ray Tune** for distributed hyperparameter optimization.
- Train multiple models using different hyperparameter combinations.
- Track experiments and model performance using **MLflow**.
- Save the best-performing model in **PyTorch (.pth)** and **ONNX** formats.
- Evaluate the trained model using an independent test dataset.
- Generate performance metrics

Physics user case

- Search for a **heavy charged resonance (W')** decaying into a WZ boson pair.
- Study the fully leptonic final state **W' → WZ → ℓνℓℓ**.
- Probe possible deviations from the **Standard Model** in the low-to-intermediate mass region (**0.4–4 TeV**).
- Benefit from a **clean experimental signature** with low background contamination.
- Achieve better signal discrimination compared to fully hadronic final states.
- Demonstrate the application of machine learning techniques in searches for physics beyond the Standard Model.



Background	Sample	XSec [pb]
$WZ \to l\nu ll$	WZTo3LNu_mllmin4p0_TuneCP5_13TeV-powheg-pythia8	4.664e+00
$ZZ \to 4l$	ZZTo4L_M-1toInf_TuneCP5_13TeV_powheg_pythia8	1.381e+01
$t\bar{t}$	TTTo2L2Nu_TuneCP5_13TeV-powheg-pythia8	8.829e+01
	TTToSemiLeptonic_TuneCP5_13TeV-powheg-pythia8	365.974e+00
$Z\gamma \to ll\gamma$	ZGToLLG_01J_5f_TuneCP5_13TeV-amcatnloFXFX-pythia8	5.557e+01
$Z + Jets$	DYJetsToLL_M-50_HT-70to100_TuneCP5_PSweights_13TeV-madgraphMLM-pythia8	1.398e+02
	DYJetsToLL_M-50_HT-100to200_TuneCP5_PSweights_13TeV-madgraphMLM-pythia8	1.405e+02
	DYJetsToLL_M-50_HT-200to400_TuneCP_PSweights_13TeV-madgraphMLM-pythia8	3.839e+01
	DYJetsToLL_M-50_HT-400to600_TuneCP5_PSweights_13TeV-madgraphMLM-pythia8	5.213e+00
	DYJetsToLL_M-50_HT-600to800_TuneCP5_PSweights_13TeV-madgraphMLM-pythia8	1.265e+00
	DYJetsToLL_M-50_HT-800to1200_TuneCP5_PSweights_13TeV-madgraphMLM-pythia8	5.683e-01
	DYJetsToLL_M-50_HT-1200to2500_TuneCP5_PSweights_13TeV-madgraphMLM-pythia8	1.328e-01
	DYJetsToLL_M-50_HT-2500toInf_TuneCP5_PSweights_13TeV-madgraphMLM-pythia8	2.987e-03
VVV	WWW_4F_TuneCP5_13TeV-amcatnlo-pythia8	2.158e-01
	WWZ_4F_TuneCP5_13TeV-amcatnlo-pythia8	1.707e-01
	WZZ_TuneCP5_13TeV-amcatnlo-pythia8	5.709e-02
	ZZZ_TuneCP5_13TeV-amcatnlo-pythia8	1.746e-02
$gg \to 4ll$	GluGluToContinToZZTo2e2mu_TuneCP5_13TeV-mcfm701-pythia8	5.4e-03
	GluGluToContinToZZTo4e_TuneCP5_13TeV-mcfm701-pythia8	2.7e-03
	GluGluToContinToZZTo4mu_TuneCP5_13TeV-mcfm701-pythia8	2.7e-03
	GluGluToContinToZZTo4tau_TuneCP5_13TeV-mcfm701-pythia8	2.7e-03
$Single Top$	ST_s-channel_4f_leptonDecays_TuneCP5_13TeV-amcatnlo-pythia8	3.549e+00
	ST_tW_antitop_5f_inclusiveDecays_TuneCP5_13TeV-powheg-pythia8	3.251e+01
	ST_tW_top_5f_inclusiveDecays_TuneCP5_13TeV-powheg-pythia8	3.245e+01

Processing step

- Minimal 2 datasets (signal and background)
- The script that generate the event selection and new variables in skimmed ntuples is provided by the user
- In this case a selection is done using the leptonic final state search for W’

```yaml
---
proxy:
  generate: 1 # 1 yes, 0 no
  voms: "cms"
  proxy_time: "192:00"
  proxy_path: "$HOME/.globus/x509up_$(id -u)"

data_processing:
  condor_params:
    executable_file: "run_filter.sh"
    cpus: 1
    gpus: 0
    mem: 1.5GB
    disk: 2GB
    job_flavour: "espresso" # 20 minutes
  processing_script: "src/ml_framework/example_files/main_process/filterNanoAOD.py"
  eos_output_dir: "/eos/user/v/vminjare/test_dataprocessing"
  afs_cms_base: "/afs/cern.ch/user/v/vminjare/CMSSW_13_3_0"
  redirector: "cms-xrd-global.cern.ch"
  datasets:
    - FLN: "/ZGToLLG_01J_5f_TuneCP5_13TeV-amcatnloFXFX-pythia8/RunIISummer20UL18Na
      ID: 0
      amount: 3 # -1 is all ROOT files
    - FLN: "/WprimeToWZToWlepZlep_narrow_M1000_TuneCP5_13TeV-madgraph-pythia8/RunI
      ID: 1
      amount: -1
```

Convertion step

- A one to one conversion between .root format to HDF5
- Is optional but recommended before performing ML training
- The user can define which branches to use for the ML training (saving storage space)

```yaml
---
convertion:
  input_dirs:
    - "/eos/user/v/vminjare/test_dataprocessing/WprimeToWZToWlepZlep_narrow_M1000_
    - "/eos/user/v/vminjare/test_dataprocessing/ZGToLLG_01J_5f_TuneCP5_13TeV-amcat

  tree_name: "Events"

  branches: "all"

# If you do not have jagged arrays leave it blank. If you use, the default value i
  max_jagged_len:

  eos_output_dir: "/eos/user/v/vminjare/test_conversionh5"

condor_params:
  executable_file: "run_conversion.sh"
  cpus: 1
  gpus: 0
  mem: 1.5GB
  disk: 2GB
  job_flavour: "espresso" # 20 minutes
```

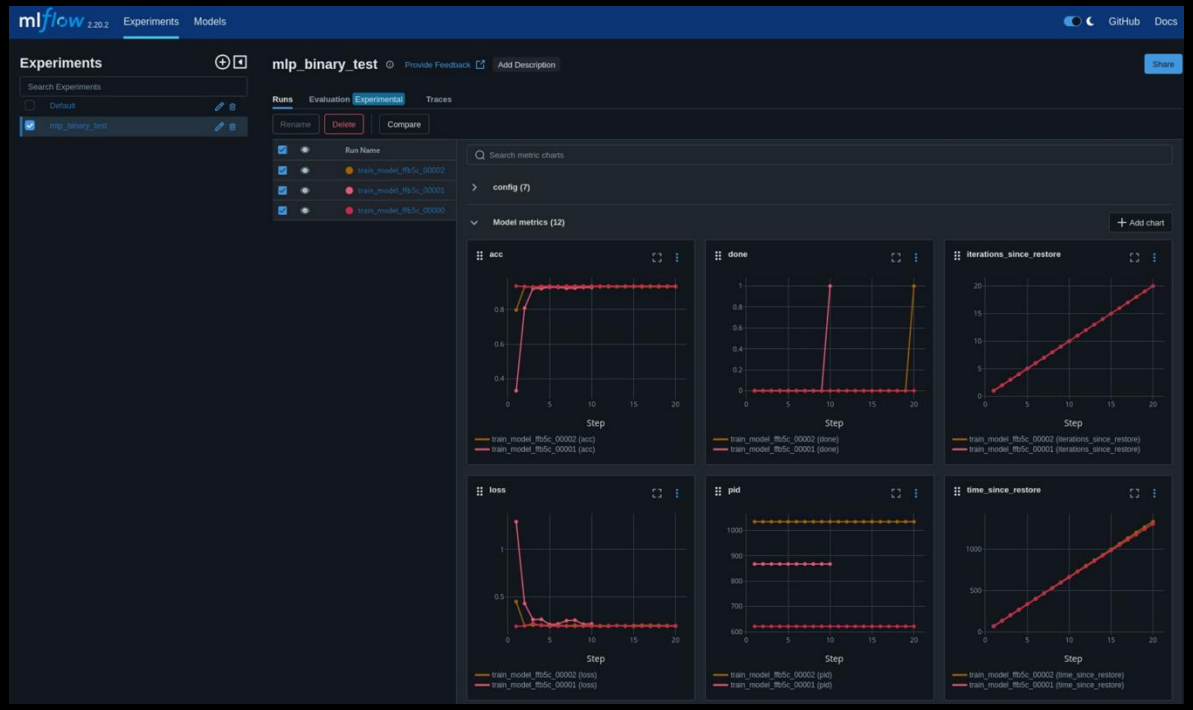
ML training step (classification)

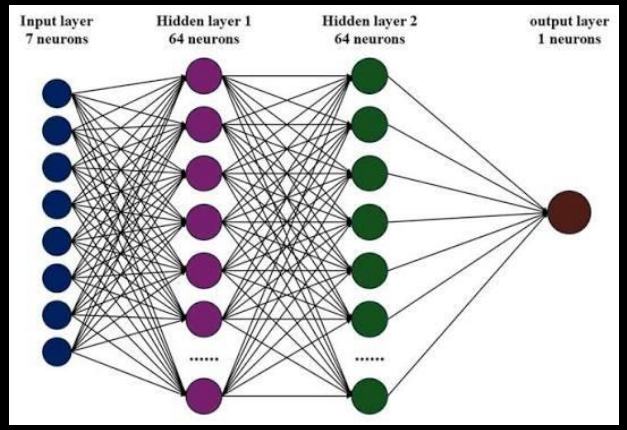
- Configurable depending on the neural network arquitecture
- A configuration YAML provided for each scenario
- So far MLP and autoencoder are available

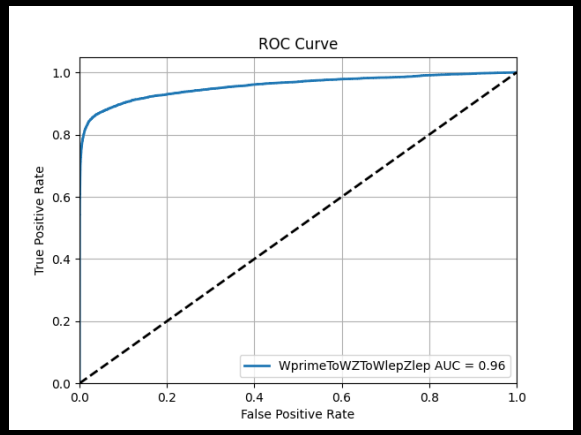
```yaml
---
data:
  input_paths:
    - "test_conversionh5/WprimeToWZToWlepZlep_narrow_M1000_TuneCP5_13TeV-madgraph-p
    -  "test_conversionh5/ZGToLLG_01J_5f_TuneCP5_13TeV-amcatnloFXFX-pythia8_RunIISu


  output_path: "results"

  features:
    - "A_Dr_Z"
    - "A_Zmass"
    - "MET_pt"
    - "B_Zmass"
    - "B_Dr_Z"


  label: "Dataset_ID"

  num_classes: 2

  # Leave this blank if you do not want to apply a mapping
  label_mapping: "short_name_mapping.json"

model:
  name: "mlp_binary_test"

  type: "mlp"

  ideal_accuracy: 95

  num_models: 3
```

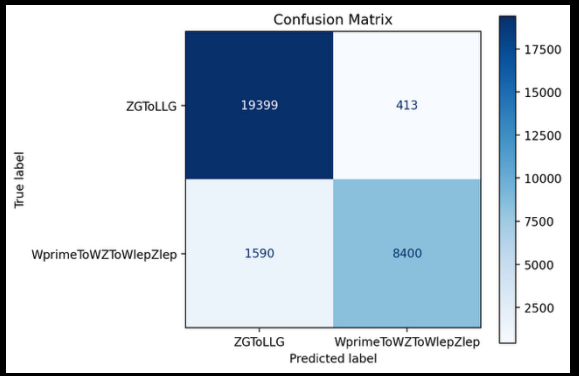
Results (classification)

- **Common statistical metrics for evaluating NN performance**
- **Users can additionally monitor and explore the optimization process through MLflow.**







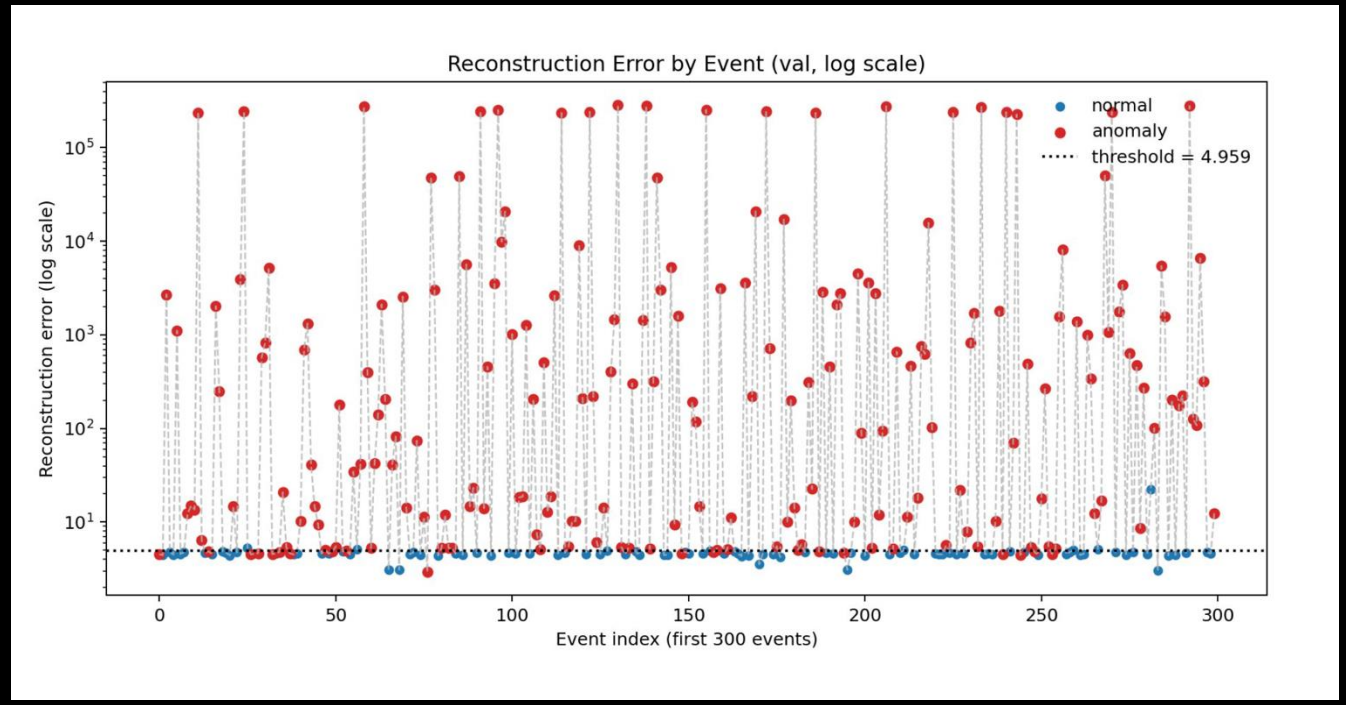


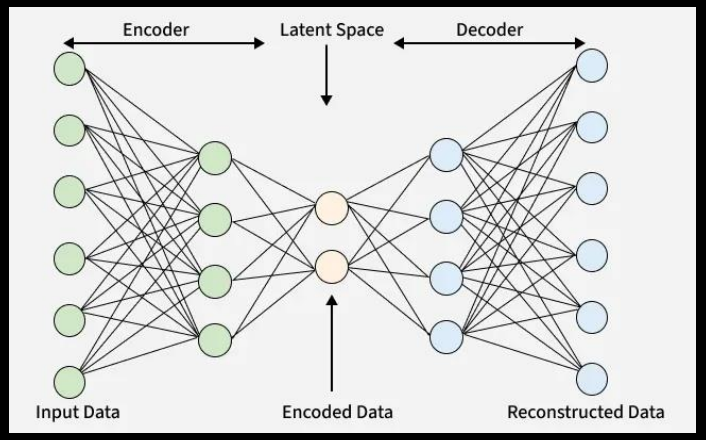
ML training step (anomaly detection)

- Configurable depending on the neural network arquitecture
- A configuration YAML provided for each scenario
- So far MLP and autoencoder are available

```
---
data:
  input_paths: []

  input_paths_normal:
    - "test_conversionh5/ZGToLLG_01J_5f_TuneCP5_13TeV-amcatnloFXFX
1_NANOAODSIM"

  input_paths_anomaly:
    - "test_conversionh5/WprimeToWZToWlepZlep_narrow_M1000_TuneCP5
stic_v16_L1v1-v1_NANOAODSIM"

  output_path: "results_autoencoder"

  features:
    - "A_Dr_Z"
    - "A_Zmass"
    - "MET_pt"
    - "B_Zmass"
    - "B_Dr_Z"

  label: "Dataset_ID"
  num_classes: 2
  label_mapping: "short_name_mapping.json"

model:
  name: "ae_zgamma_vs_wprime"
  type: "autoencoder"
  ideal_loss: 0.01
  num_models: 1

anomaly_detection:
  normal_labels: [0]
  anomaly_labels: [1]
  val_size: 0.2
  test_size: 0.2
  seed: 16
[castaned@lxplus945 ml_training]$
```

Results (Anomaly detection)





- Autoencoder trained on background-like events to learn the nominal event topology.
- Reconstruction error used as an anomaly score for event classification.
- A threshold was defined from the validation sample to identify anomalous events.
- Most normal events remain below the threshold, while anomalies exhibit significantly larger reconstruction errors.
- Clear separation between normal and anomalous events demonstrates the effectiveness of the approach.
- The method provides a model-independent strategy for identifying rare or unexpected signatures in high-energy physics data.

Conclusions

- **End-to-end framework for ML-based physics analyses.**
- **Portable, reproducible, and easy to deploy through containers.**
- **Integrates training, optimization, monitoring, and evaluation tools in a single environment.**
- **Designed to shorten the learning curve for newcomers and accelerate the transition from analysis ideas to scientific results.**
- **Allows students and researchers to focus on physics challenges rather than software and infrastructure complexities.**

Future implementations

- **Extension of the framework to support advanced deep-learning architectures such as CNNs, Graph Neural Networks (GNNs), Transformers, and other state-of-the-art models.**
- **Continuous debugging, validation, and performance optimization through real use cases and community feedback.**
- **Active involvement of students and new users to test the framework, identify bugs, and propose new features.**
- **Development of comprehensive documentation, tutorials, and example workflows to facilitate adoption.**
- **Native compatibility with CERN Open Data formats and analysis workflows.**

Current manpower

Manpower Involved

- **1 master student in Data science (main developer)**

- **3 undergraduate students in Physics**
  - Contribute to the improvement of the framework documentation and user guides.
  - Perform systematic testing and validation of the framework to identify bugs, usability issues, and potential optimizations.
  - Develop research projects and potential bachelor theses by applying the framework to analyses based on CERN Open Data.

- **2 undergraduate students in Computer Science**
  - Participate in software testing, debugging, and code validation.
  - Contribute to the design and implementation of additional machine learning architectures, including more advanced neural network models and supporting software components.

Anyone interested in testing or contributing to the framework, either as a user or developer, is welcome to participate.

Documentation

https://castaned.github.io/ML-integration-CMSSW/

thanks