Machine learning techniques for calorimetry

Polina Simkina

On behalf of the CMS collaboration

Université Paris-Saclay (FR)
CEA-IRFU (FR)

CALOR 2022
19/05/2022

Outline

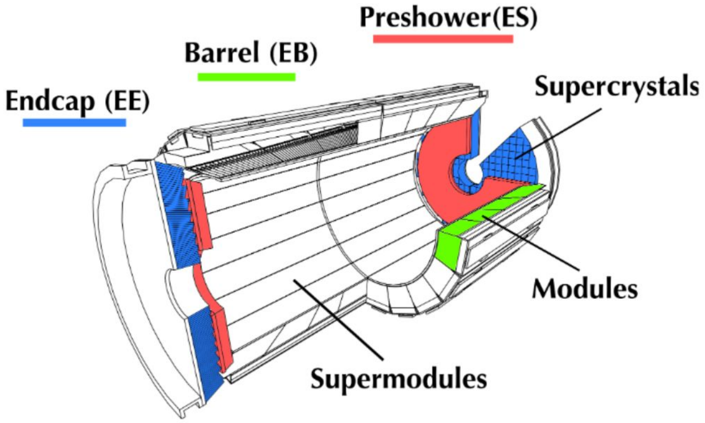
- Introduction to **Graph Neural Networks** (GNN)
- GNNs for **ECAL** Reconstruction

  ***could be used in Run3** for barrel and endcaps*

  - **SuperClustering** Reconstruction
    - SuperClustering in ECAL
    - GNN for SuperClustering
    - GNN Performance
  - **Energy regression** using GNN
    - Energy corrections
    - Dynamic Reduction Network (DRN)
    - DRN Performance

ECAL detector:

- Hermetic homogeneous calorimeter.
- ~ 76 000 lead tungstate (PbWO4) crystals.
- Crystal size:
  Barrel: 2.2 x 2.2 x 23 cm
  Endcaps: 3 x 3 x 22 cm



See the talk by F. Ferri:
Ten years of operations of the CMS ECAL

Neural Networks

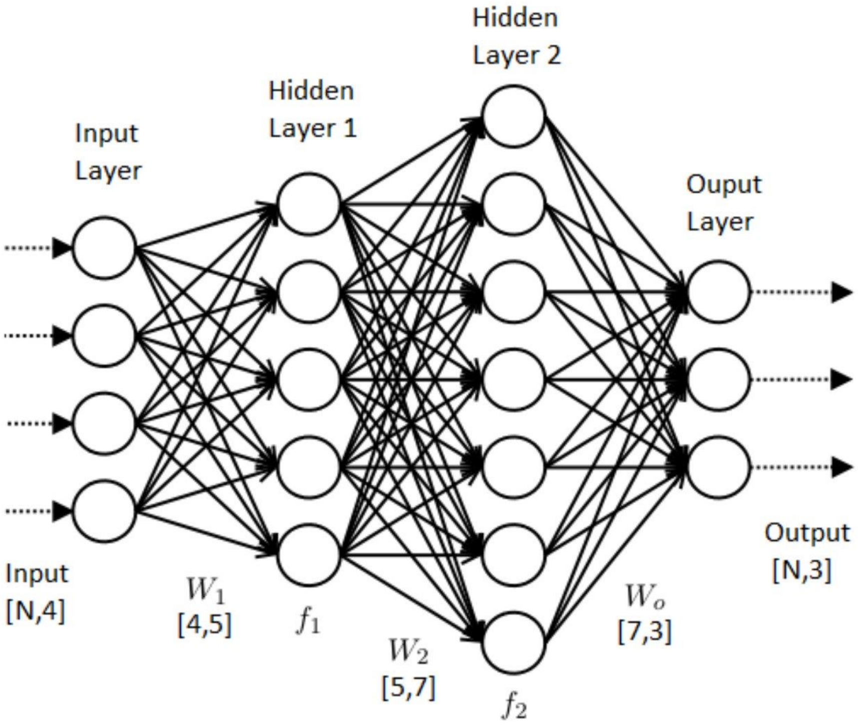
Neural Networks are one of the widely used **Machine Learning** algorithms.

The simplest neural network consists of **an input, an output and one hidden layer.**

If the network has more than one hidden layer, it is called **Deep Neural Network**.

Network training:

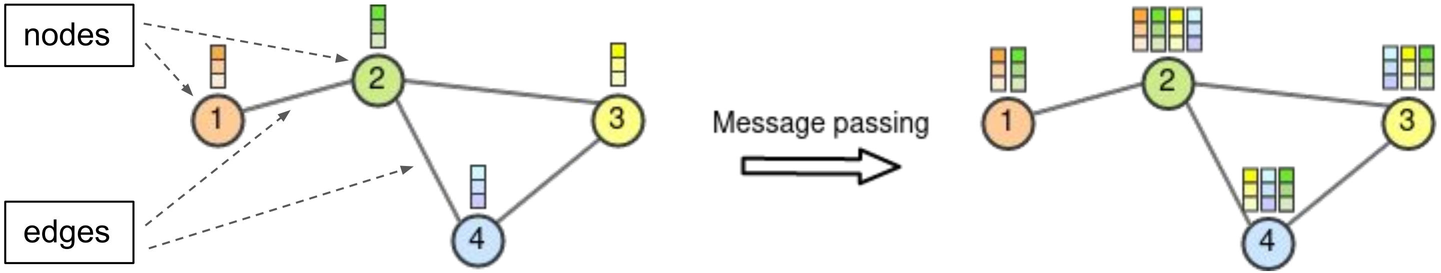
- The input vector is multiplied by a **weight matrix** resulting in an input to a new **(hidden) layer**. This process then can be successively repeated with new layers (each time with different weight matrix).
- The result can be extracted from the output of the last layer. It is compared with the "right" answer and based on the loss function (e.g. Mean Squared Error) the weights are adjusted using method called **backpropagation**.



Example of Deep Neural Network

Graph Neural Networks

- Type of neural network that can operate on and analyze **graph structures**.
- Unlike other types of networks GNN can be easily applied on sparse data, doesn’t require padding.
- A graph consists of **nodes** (contain features of the object) and **edges** (reflect the relationship between the nodes).
- In GNNs the information can be shared between the neighbors:
  - The vector features of each node are transformed into “messages” (e.g. using dense layers) that are sent to the neighbors (message-passing).
  - In this way, **each node learns information about its neighbors and itself**. The process is carried out in parallel and repeated several times.



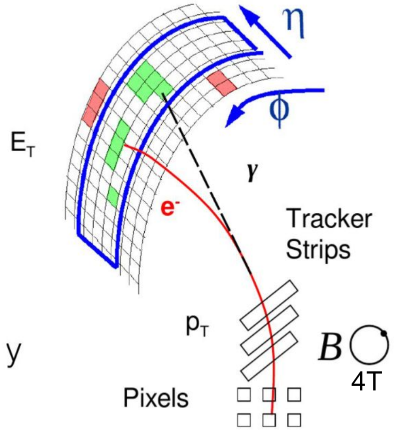
https://uvadlc-notebooks.readthedocs.io/en/latest/tutorial_notebooks/tutorial7/GNN_overview.html

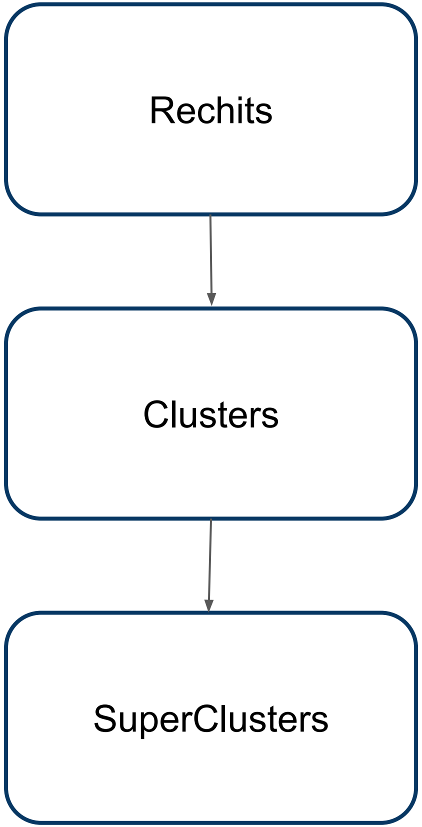
Outline

- ➢ Introduction to Graph Neural Networks (GNN)
- ➢ GNNs for ECAL Reconstruction
  - ○ SuperClustering Reconstruction
    - ■ SuperClustering in ECAL
    - ■ GNN for SuperClustering
    - ■ GNN Performance
  - ○ Energy regression using GNN
    - ■ Energy corrections
    - ■ Dynamic Reduction Network (DRN)
    - ■ DRN Performance

SuperClustering in ECAL

A major step in reconstruction of **electron/gamma**.





Rechits

Clusters

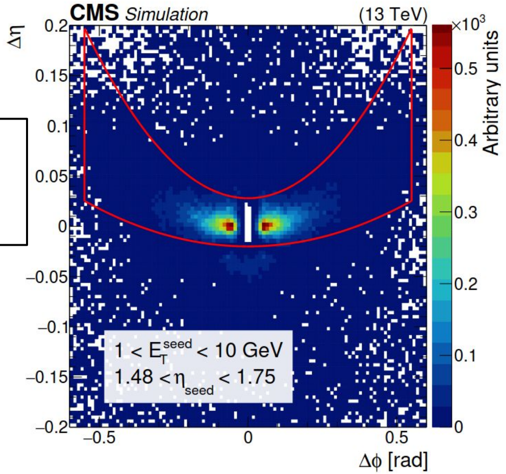
SuperClusters

- ➢ **Reconstructed energy deposits** left by traversing particle in the PbWO4 crystals of the calorimeter (rechits).
- ➢ Rechits are gathered together around the crystal with highest deposited energy to form **clusters**.
- ➢ Each cluster represents a **single particle**.
  - ○ Or several overlapping particles.
- ➢ Due to bremsstrahlung and photon conversion before the ECAL, the individual clusters have to be combined together to form a **SuperCluster**.
- ➢ **The energy of the initial particle can be reconstructed from the SuperCluster.**

Mustache SuperClustering

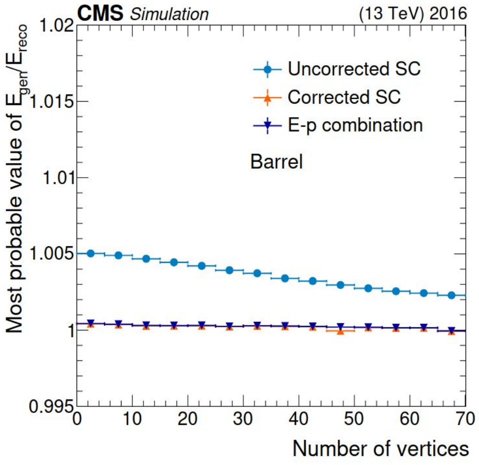
- ➢ The algorithm currently used in CMS for reconstruction of SuperClusters.
- ➢ Purely **geometrical** approach:
  - All the clusters falling into the specified “mustache” shape would be considered as part of the SuperCluster. The size of the area depends on energy and position of the seed.
  - “Mustache” shape due to the CMS magnetic field (spread along φ).

An example of the mustache SC distribution for electron



- ➢ **High efficiency**: the algorithm is able to gather even low-energy clusters.
- ➢ Downside: **suffers from pileup (PU) and noise contamination.**
- ➢ Energy regression is further applied that can **correct PU and noise on average**.

Performance of SuperClustering and energy correction for 2016 MC samples



https://iopscience.iop.org/article/10.1088/1748-0221/16/05/P05014

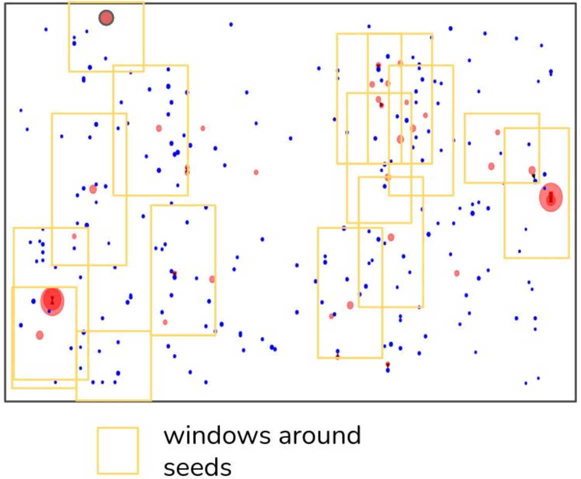
GNN for ECAL SuperClustering

New algorithm for SuperClustering: **DeepSuperCluster ML model**

- Based on Graph Neural Network. It can receive and combine the information from all the clusters in the window.
- **Maintains the efficiency while improving PU and noise rejection.**

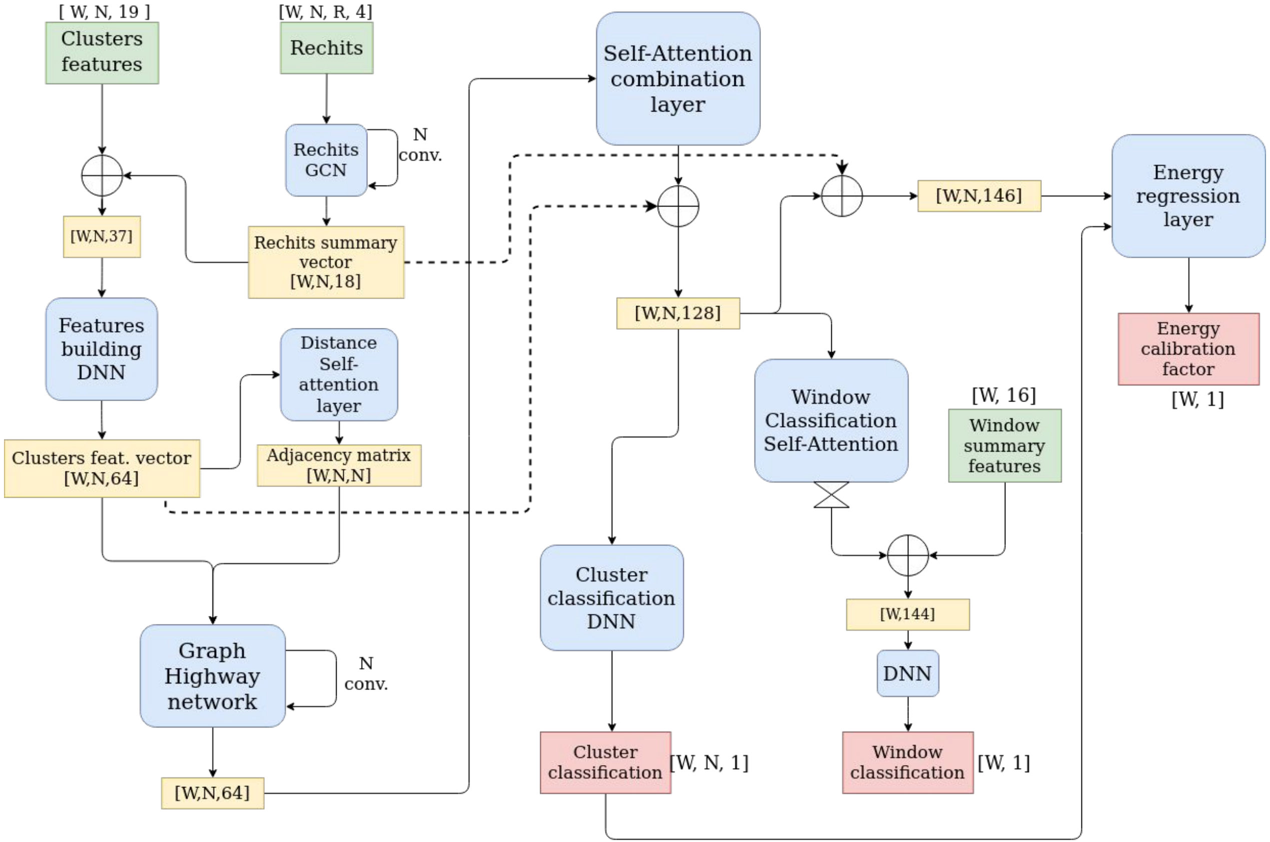
For the training and testing the **dataset** was created:

- Electrons and photons are generated uniformly in p_T = [1,100] GeV.
- PU uniformly distributed between [55,75] interactions is used.



- Windows are opened around all the clusters with E_T > 1 GeV (**seeds**).
  - Window dimensions are η-dependent.
  - The model has to process each window and give a prediction for it.
- The **inputs** for the model are:
  - Cluster information (E, E_T, η, φ, z, number of crystals, relative to seed: is_seed_flag, Δη, Δφ, ΔE, ΔE_T).
  - List of Rechits for each cluster.
  - Summary window features (max, min, mean of the crystal variables: E_T, E, Δη, Δφ, ΔE, ΔE_T)
- The **outputs**: cluster classification (in/out of SC), window classification (electron/photon/jet), energy regression.

GNN for ECAL SuperClustering architecture



- W : number of windows in the batch
- N: number of clusters
- R: number of rechits
- [X,Y,Z] tensor dimension

🟩 Inputs

🟦 Trainable layers

🟨 Tensors with dimensions

🟥 Outputs

→ Tensor *flow*

⇢ Skipped connection

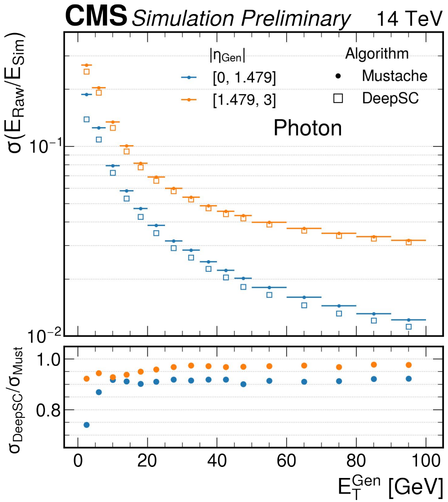
⊕ Concatenation

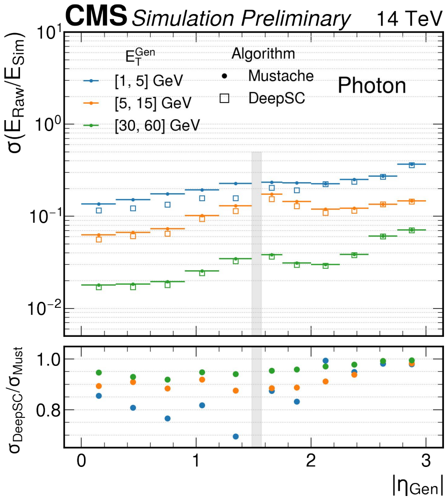
⧗ Aggregation (sum over clusters dimension)

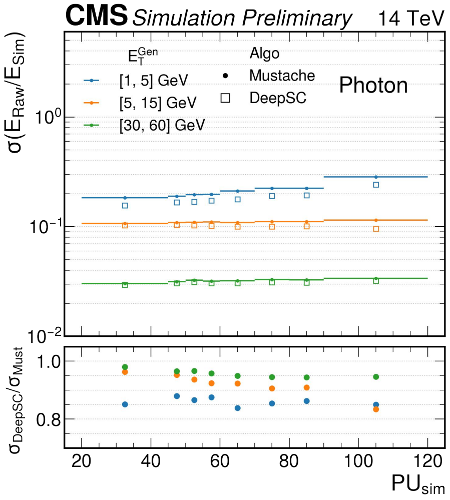
Performance: energy resolution

Resolution of the reconstructed uncorrected **SuperCluster energy** (E_{Raw}) divided by the **true energy deposits** in ECAL (E_{Sim}) versus:

- the transverse energy of the gen-level particle E_T^{Gen} (left)
- the gen-level particle position $|\eta_{Gen}|$ (center)
- the number of simulated PU interactions (right)







The resolution is computed as half of the difference between the 84% quantile and the 16% quantile (one σ) of the E_{Raw} /E_{Sim} distribution in each bin.

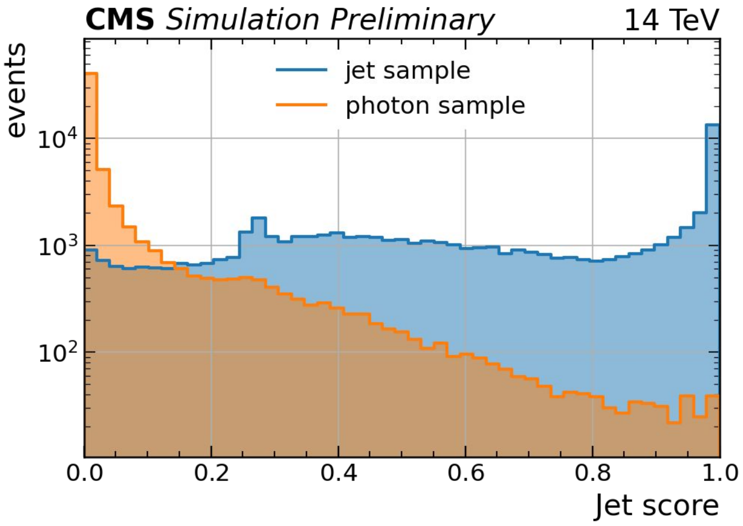
The lower panel shows the ratio of the resolution of the two algorithms:

$$\sigma_{DeepSC} / \sigma_{Mustache}$$

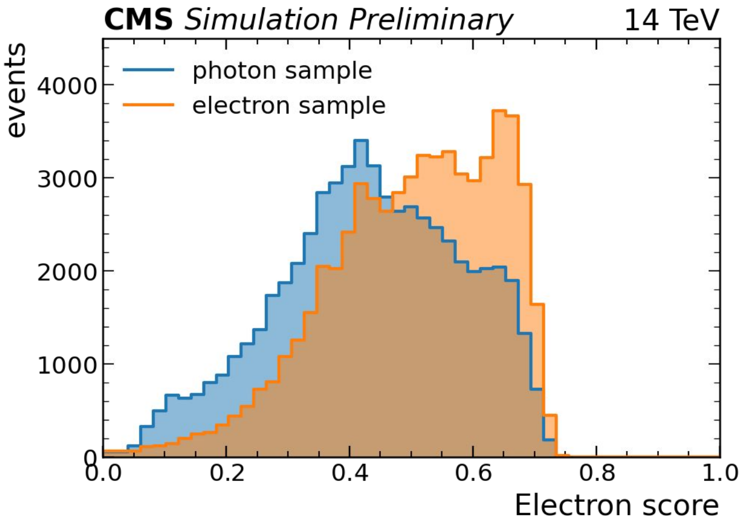
The DeepSC algorithm shows **significantly improved resolution**, particularly for low E_T signals and at high PU.

Particle identification

- ➢ Same network can be used to identify the **flavor of the particle**.
- ➢ An extra sample containing **jets** was generated (same energy/PU as for electron/photon sample).
- ➢ The goal is to identify the clusters belonging to jets.
- ➢ In order to avoid the performance degradation for electrons/photons in terms of cluster selection, **Transfer Learning** was used to re-train only the ID part of the network.



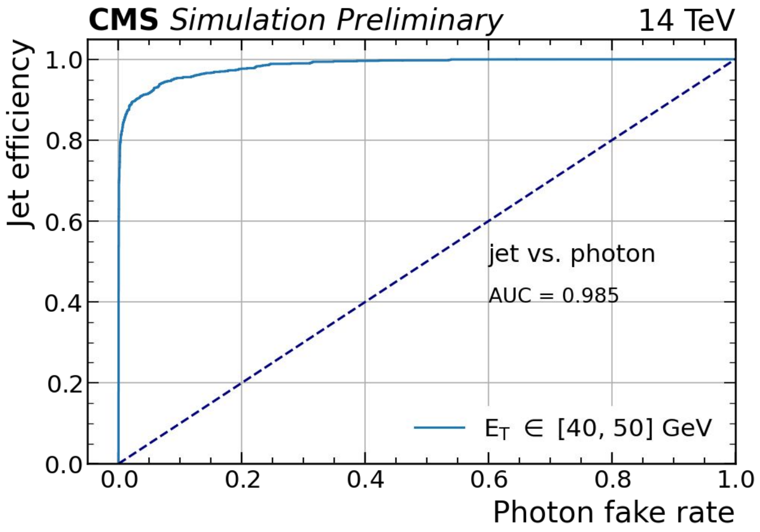
Likelihood to be predicted as jet (score) for the jet and photon sample.

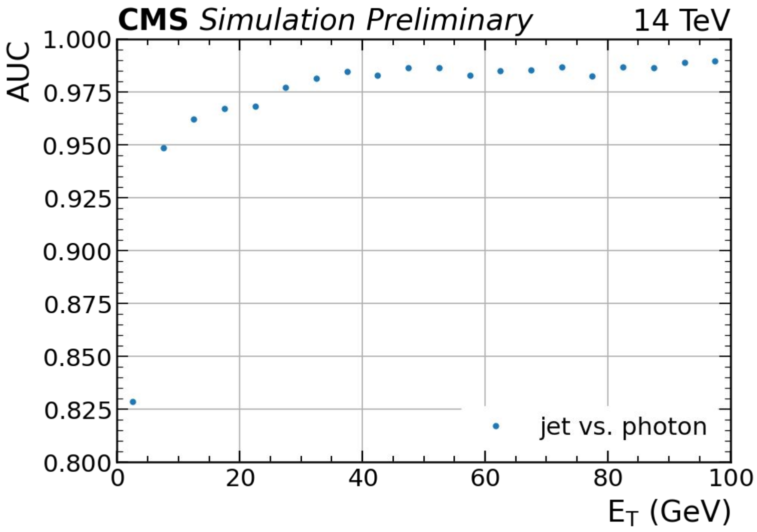


Likelihood to be predicted as electron (score) for the photon and electron sample.

Performance: particle identification

- ROC curve obtained from the discriminator for jet vs. photon for E_T = [40, 50] GeV (left).
- Summary performance obtained by calculating **Area Under the ROC curve** (AUC) for different energy ranges (right).





High performance for jet vs. photon discrimination.

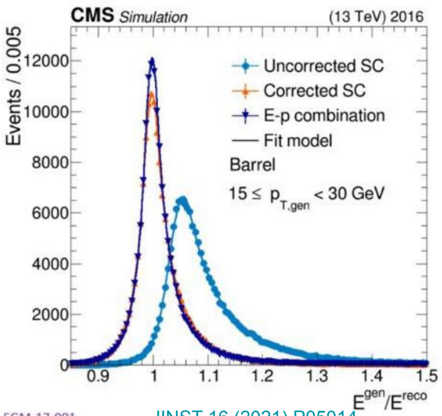
- AUC levels for photon vs. electron discriminator are ~63%.
- Only ECAL variables are used.
- The output of the model can be used in the global event reconstruction of CMS.

Outline

- Introduction to Graph Neural Networks (GNN)
- GNNs for ECAL Reconstruction
  - SuperClustering Reconstruction
    - SuperClustering in ECAL
    - GNN for SuperClustering
    - GNN Performance
  - Energy regression using GNN
    - Energy corrections
    - Dynamic Reduction Network (DRN)
    - DRN Performance

Energy corrections using Dynamic Reduction Network

- After SuperClusters are formed, **energy correction** has to be applied
  - Account for energy lost in gaps and upstream material, longitudinal energy leakage, finite thresholds to suppress noise and unclustered energy.
- Currently done per particle using **Boosted Decision Trees** with a semiparametric regression.
  - Uses ~30 high-level input features to describe electromagnetic shower.



EGM-17-001 JINST 16 (2021) P05014

New Machine-Learning algorithm developed: **graph-based Dynamic Reduction Network** (DRN).

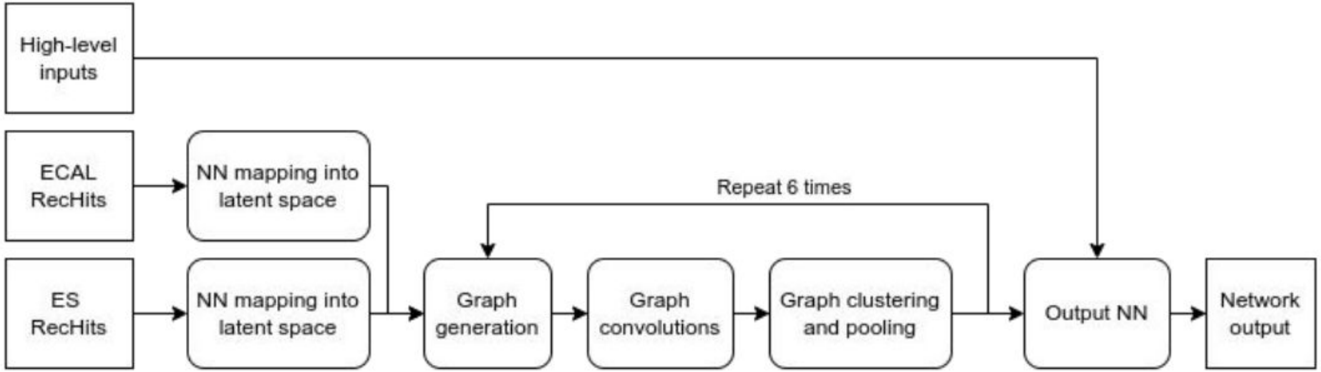
- Allows learning on **low-level detector features** (rechits) associated with a given particle.
- Input includes RecHits from both ECAL and ECAL preshower (ES), as well as additional features to describe information not encoded in the hit collection (pileup, leakage into HCAL).

Dynamic Reduction Network

- Input features (rechits) are transformed into **high dimensional latent space**.
- Graphs are **dynamically generated** in the latent space (recomputed at each iteration).
- The graph convolutions are performed (includes message-passing).
- **The information is aggregated** over the graph using clustering and pooling.

Input: Rechit features (energy, x, y, z).

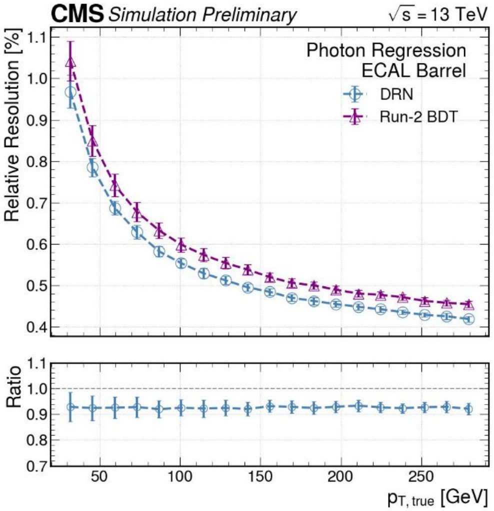
Output: Predict probability density of energy correction value.

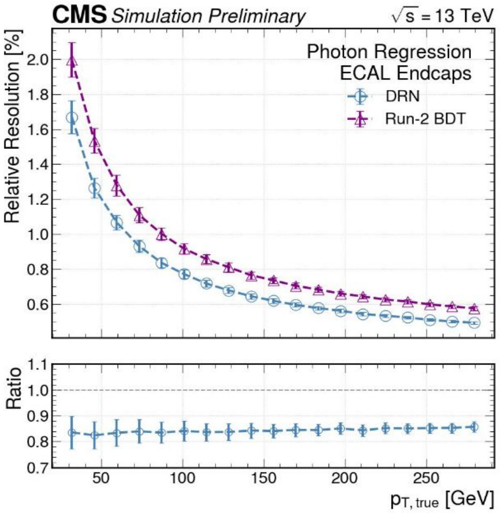


Energy correction: performance

Mean response E_{Pred} /E_{True} estimated with Dynamic Reduction Network (DRN) and Boosted Decision Tree (BDT) performance in the ECAL barrel (left) and endcaps (right) as a function of transverse momentum.

Performance evaluated on photon gun simulation with ideal detector calibration.
Error bars represent fitting uncertainties.



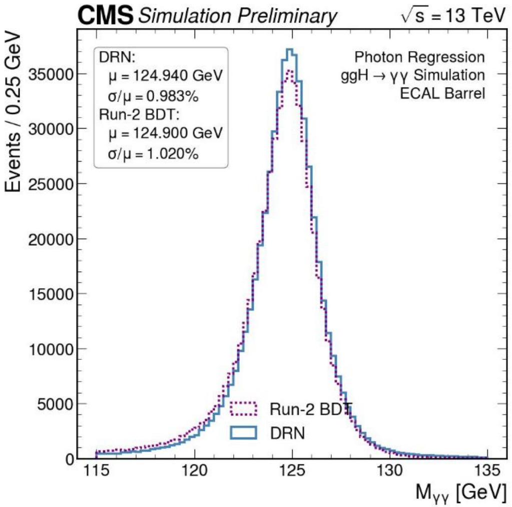


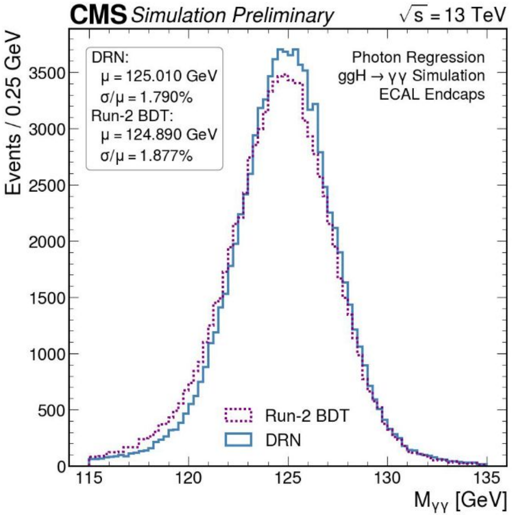
The DRN obtains a better resolution than the BDT by **a factor of ≈ 10%** at all values of p_T.

Energy correction: performance

Di-photon invariant mass distributions of H→γγ events in 2018 Legacy simulation for both the Dynamic Reduction Network (DRN) and Boosted Decision Tree (BDT) architectures.

The Higgs peak is fit with a Cruijff function to parameterize the detector response and resolution.





The DRN obtains an **improved resolution with respect to the BDT by a factor of about 5%** in both detector regions.

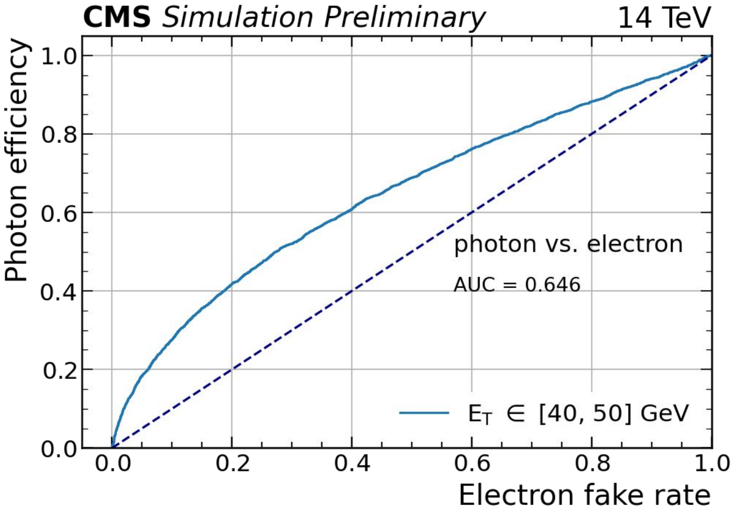
Conclusion

- Graph Neural Networks are **new advanced algorithms** for the calorimeter reconstruction.
- The implementations and the performances are shown with application to various tasks:
  - Particle reconstruction and identification in ECAL.
  - Energy correction of the electrons and photons in ECAL.
- All of them show **significant improvement** w.r.t. Run-2 algorithms.
- The goal is to use the presented algorithms during the Run-3 and beyond.

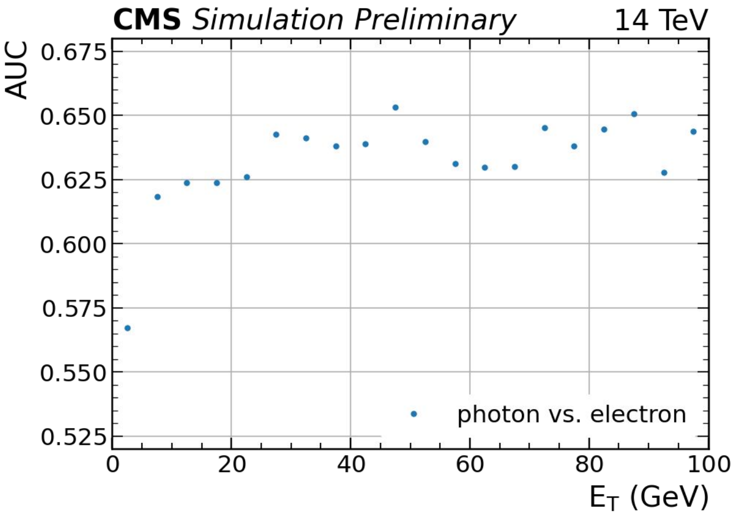
Backup

Discrimination photon vs. electron

ROC curve and AUC summary obtained using the ID output of DeepSC model.



ROC curve for photon vs. electron sample in the energy range [40, 50] GeV



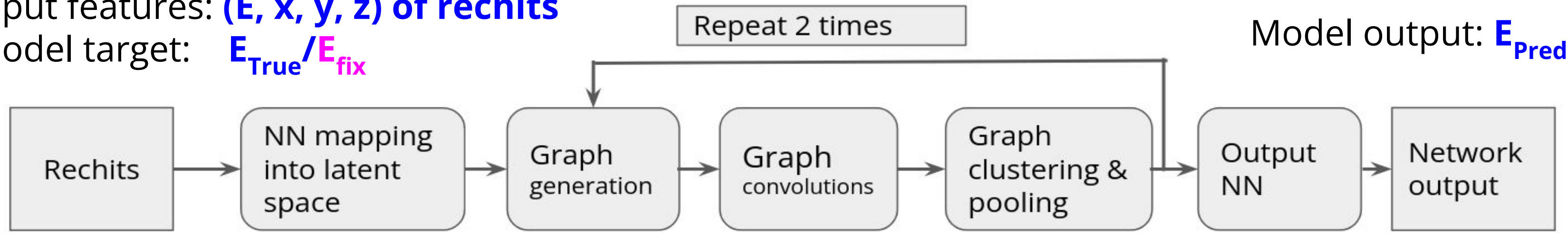
Summary AUC plot for photon vs. electron

Dynamic Reduction Neural (DRN) network

Exploiting the imaging power of the detector by using measured energy and position of each readout cell

Input features: **(E, x, y, z) of rechits**
Model target: E_{True}/E_{fix}

Model output: E_{Pred}/E_{fix}



Repeat 2 times

Rechits → NN mapping into latent space → Graph generation → Graph convolutions → Graph clustering & pooling → Output NN → Network output

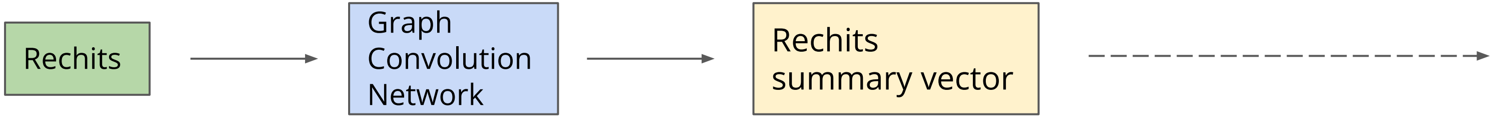
where E_{True} is true energy of particle, E_{fix} is reconstructed energy using detector level calibration and E_{Pred} is the energy reconstruction using DRN weights.

- ➢ This DRN model is trained on a flat energy sample of 10-350 GeV with a total of 4.1M events simulated using GEANT4 v10.4.3 and FTFP BERT EMN hadronic physics list.
  - ○ Out of 39 sampling layers of AHCAL, only 10 layers are sampled (consistent with the final HGCAL geometry).
- ➢ The loss functions is defined as $$\frac{(target - predic)^2}{target}$$
- ➢ A constant learning rate of 10^{-4} is used & time taken to train the model per epoch for 4M events is ~30 mins using 4 V100 GPU cards with 16 GB of memory each with data parallelism of pytorch.

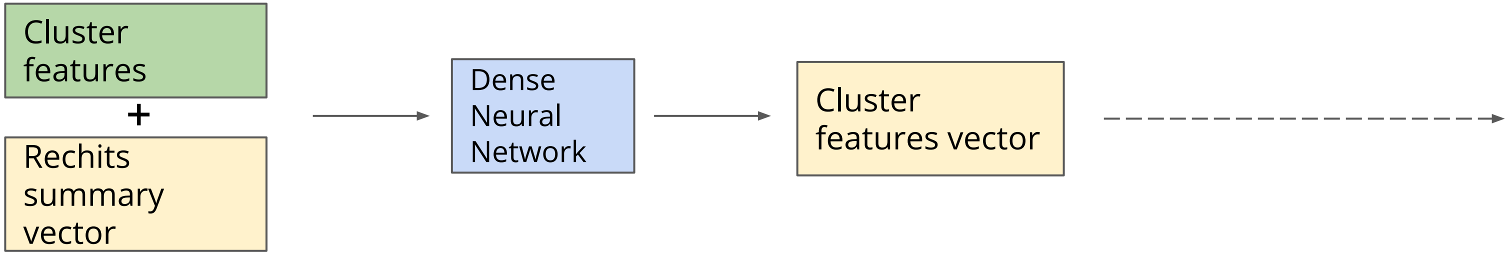
Graph Neural Network architecture - I

- 🟩 Inputs
- 🟦 Trainable layers
- 🟨 Tensors with dimensions
- 🟥 Outputs

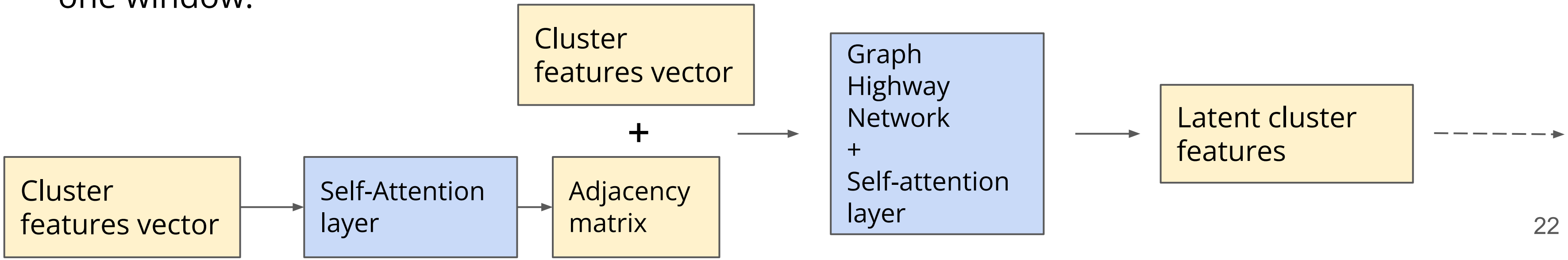
1. Get the summary vector features for the rechits of each cluster in the window.



2. Extract the latent features for every cluster.



3. Get the adjacency matrix for clusters in the window and share the features between clusters in one window.



Graph Neural Network architecture - II

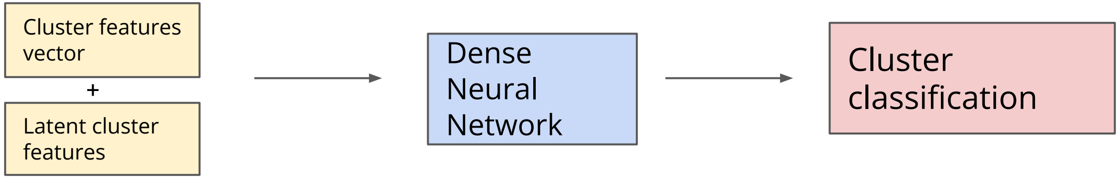
🟩 Inputs

🟦 Trainable layers

🟨 Tensors with dimensions

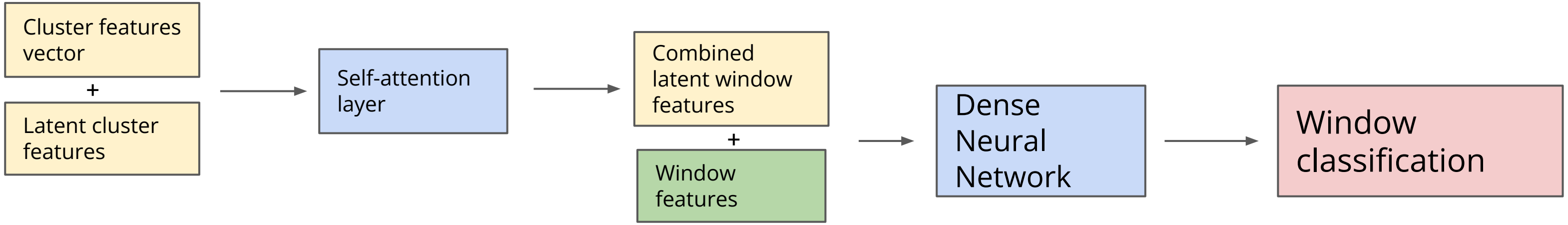
🟥 Outputs

4. Get the classification output (in/out of SC) for each cluster in the window.



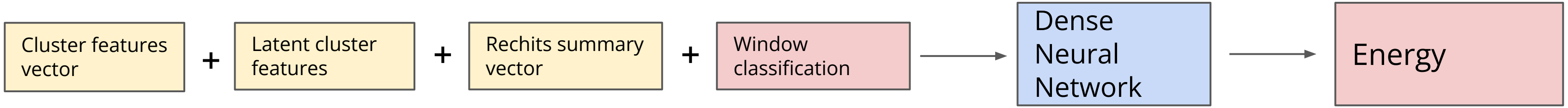
Cluster features vector + Latent cluster features → Dense Neural Network → Cluster classification

5. Get the window classification (photon/electron) based on combined window features.



Cluster features vector + Latent cluster features → Self-attention layer → Combined latent window features + Window features → Dense Neural Network → Window classification

6. Get the energy regression factor.



Cluster features vector + Latent cluster features + Rechits summary vector + Window classification → Dense Neural Network → Energy

Cruijff function

Centered Gaussian with different left-right resolutions and non-Gaussian tails

$f(x) = \exp((x\text{-}m)^2/(2\sigma^2_{L,R} + \alpha_{L,R}(x\text{-}m)^2))$