Eur. Phys. J. C (2024) 84:639
https://doi.org/10.1140/epjc/s10052-024-12978-1

THE EUROPEAN PHYSICAL JOURNAL C

Regular Article - Experimental Physics

Reconstruction of electromagnetic showers in calorimeters using Deep Learning

Polina Simkina[a], **Fabrice Couderc**[b], **Julie Malclès**[c], **Mehmet Özgür Sahin**[d]

IRFU, CEA, Université Paris-Saclay, 91191 Gif-sur-Yvette, France

Received: 24 November 2023 / Accepted: 2 June 2024

Abstract The precise reconstruction of properties of photons and electrons in modern high energy physics detectors, such as the CMS or ATLAS experiments, plays a crucial role in numerous physics results. Conventional geometrical algorithms are used to reconstruct the energy and position of these particles from the showers they induce in the electromagnetic calorimeter. Despite their accuracy and efficiency, these methods still suffer from several limitations, such as low-energy background and limited capacity to reconstruct close-by particles. This paper introduces an innovative machine-learning technique to measure the energy and position of photons and electrons based on convolutional and graph neural networks, taking the geometry of the CMS electromagnetic calorimeter as an example. The developed network demonstrates a significant improvement in resolution both for photon energy and position predictions compared to the algorithm used in CMS. Notably, one of the main advantages of this new approach is its ability to better distinguish between multiple close-by electromagnetic showers.

1 Introduction

In modern hadron collider experiments with a large number of simultaneous collisions and a large number of detector cells, reconstructing the properties of individual particles from the information collected by the detectors poses a significant computational challenge. These experiments use, among other sub-detectors, electromagnetic calorimeters that record the energy deposits left by particles and measure in particular the energy of electrons and photons. Generally, energy deposits in different cells of these detectors must be clustered together to reconstruct the energy and position of the initial particle. Such reconstruction is a complex problem due to the high multiplicity of particles and the overlap of showers, which can be further investigated by taking the Electromagnetic CALorimeter (ECAL) of the Compact Muon Solenoid (CMS) detector as an example.

The CMS experiment is a general-purpose detector situated at the Large Hadron Collider (LHC) [1] at CERN. Its objectives are to probe the main theoretical framework used in particle physics (the Standard Model) and search for physics beyond it. To do so, it is necessary to detect, identify, and determine the kinematic properties of particles produced by proton-proton collisions at center-of-mass energies reaching up to 13.6 TeV.

Numerous physics analyses performed with the data collected by the CMS detector (e.g. relating to the study of the properties of the Higgs boson [2,3]) require precise reconstruction of photon and electron properties. It is done in a multi-step process starting from the reconstruction of single energy deposits in the detector and ending with the identification of the particle type and its origin [4].

The kinematic properties of individual photons are evaluated using a traditional geometrical algorithm that clusters the energies they deposit in different cells of the detector. The algorithm used by CMS is described in Sect. 2 and is called *PFClustering*, *PF* standing for *Particle-Flow* [5]. While this method is accurate and efficient, it has limitations:

1. It has limited ability to accurately distinguish two close-by photons, such as photons produced in neutral pion (π_0) decays, which can mimic the energy pattern of an isolated γ, or decays of potential new particles, for instance, the exotic decay of the Higgs boson $H \to AA \to 4\gamma$ [6], where A is a light scalar or pseudoscalar particle.

[a] e-mail: simkina.polina@gmail.com (corresponding author)
[b] e-mail: fabrice.couderc@cea.fr
[c] e-mail: julie.malcles@cea.fr
[d] e-mail: ozgur.sahin@cea.fr

Published online: 25 June 2024

2. The algorithm has a high background rate at very low energies (< 1 GeV) related to experimental noise. Such backgrounds are expected to increase with the aging of the detector and the performance is expected to further deteriorate.

In order to overcome these limitations and improve the reconstruction of particle properties, machine learning (ML) algorithms can be considered. They have been already widely applied in physics analyses based on photons and electrons in CMS (e.g. [7,8]). However, ML techniques have not been used in CMS for energy clustering, nor optimized for multiple overlapping energy deposits originating from several particles, where new challenges for the reconstruction performance emerge.

In this paper, we present a novel ML algorithm, named DeepCluster, to reconstruct energy deposits in electromagnetic calorimeters. This algorithm is based on convolutional and graph neural networks. The developed model significantly outperforms the PFClustering algorithm in terms of energy and position resolution, while also mitigating the limitations of the traditional approach regarding close-by particle identification. We show the different steps required to develop the final algorithm, explaining the reasoning behind the choice of specific methods. Finally, the performance is shown for various particle types (photons, electrons, and neutral pions), and compared to the performance of the PFClustering algorithm.

2 Particle reconstruction in the CMS electromagnetic calorimeter

The method described in this paper has broad applicability across a variety of clustering tasks. The CMS ECAL serves as a critical instrument for the precise detection and identification of electrons and photons, thus acting as the reference case for the proposed methodology. We first provide an overview of this detector's operational principles and geometric configuration. Then, we delve into the PFClustering algorithm, which is currently employed by the CMS collaboration to reconstruct the properties of electromagnetic deposits. Finally, we discuss the utilized energy correction techniques. This comprehensive overview serves as a foundation for comparison with the novel methodology proposed in this study.

2.1 CMS ECAL

The CMS ECAL is a homogeneous calorimeter consisting of about 75,000 lead tungstate ($PbWO_4$) scintillating crystals. It is divided into two main parts: the barrel (crystal size: $2.2 \times 2.2 \times 23$ cm^3), covering the pseudorapidity region $|\eta| < 1.479$, and the endcaps (crystal size: $2.9 \times 2.9 \times 23$ cm^3), covering the pseudorapidity regions $1.479 < |\eta| < 3.0$. In the barrel, the crystal length corresponds to 25.8 radiation lengths (X_0) and the crystal transverse size matches the Molière radius of PbWO4. A complete description of the CMS experiment and the ECAL is given in Ref. [1]. Crystals are arranged in a quasi-projective geometry with regard to the center of the interaction region: crystals are tilted by 3 degrees with respect to this point to avoid leakage in mechanical gaps.

The main purpose of the ECAL is the characterization of electromagnetic energy deposits. When a photon or an electron enters the calorimeter, it starts an electromagnetic shower [9], ultimately generating scintillation photons that are detected by photodetectors. This energy shower can be spread among multiple crystals around the entry point. A combination of adjacent-crystal energy deposits is called a cluster. The goal of reconstruction algorithms is to estimate as precisely as possible the energy and position of the primary particle entering the calorimeter from the corresponding cluster properties.

2.2 The PFClustering algorithm

The PFClustering algorithm [5] is designed to ensure high reconstruction efficiency even for low-energy particles. The energy clusters reconstructed with the PFClustering algorithm are called PFClusters and they are formed from the following steps:

1. The energy deposit in each crystal is reconstructed as a *hit*, if it is above a threshold $E_{\text{thr}}^{\text{hit}}$.
2. Hits with energies exceeding a certain threshold $E_{\text{thr}}^{\text{seed}}$ and larger than the energy of adjacent hits (either sharing a side or a corner) are selected as *seeds*.
3. Each seed is combined with the hits in the eight neighbouring crystals to create a *topological cluster*.
4. Topological clusters are grown by aggregating all hits with at least a corner or a side in common with a crystal that is already in the cluster.
5. In general, the energy deposited in each crystal of the PFCluster can be caused by more than one particle. If a topological cluster contains multiple seeds (each seed potentially corresponding to a single particle), a PFCluster is attributed to each seed. The fraction f_{ji} of energy of each hit j attributed to PFCluster i in the topological cluster is determined from an expectation-minimization iterative algorithm rooted in the Gaussian-mixture model. Its detailed description can be found in Ref. [5].

The position ($\vec{\mu}^i_{\text{reco}}$) and predicted energy ($E^i_{\text{reco}}$) of the i^{th} cluster are evaluated following the formulas:

Table 1 Input variables to the PFClustering energy regression BDT used in this study

Parameter	Description
E_{reco}	Energy predicted by the PFClustering algorithm
x_{reco}, y_{reco}	Position predicted by the PFClustering algorithm
E_{max}, E_{2nd}	Largest and second largest energy deposits in the crystals within the processed PFCluster
E_{LR}	The energy deposit difference between the left and right crystals in relation to the seed crystal
E_{TB}	The energy deposit difference between the top and bottom crystals in relation to the seed crystal
$cov(X, X)$, $cov(X, Y)$, $cov(Y, Y)$	Covariance values between the PFCluster spread in different directions
$E_{3\times3}$	The sum of energies in a 3×3 matrix around the seed crystal

$$E^i_{\text{reco}} = \sum_{j=1}^{M} f_{ji} A_j \tag{1}$$

$$\vec{\mu}^i_{\text{reco}} = \frac{1}{E^i_{\text{reco}}} \sum_{j=1}^{M} f_{ji} A_j \vec{c}_j, \tag{2}$$

where M is the total number of hits in a topological cluster, A_j is the deposited energy in j_{th} hit, $\vec{c}_j$ is the position of the j_{th} hit.

The parameters of the PFClustering algorithm used in this paper correspond to the tuning performed for the Run 3 (2022–2025 operation) of the LHC [10]: $E^{\text{hit}}_{\text{thr}} = E^{\text{seed}}_{\text{thr}} = 3\,\sigma_n$, where σ_n is the average noise level in a crystal as discussed in Sect. 3.

2.3 Energy correction

Due to the possibility of energy leakage from the shower and the per-crystal energy thresholds implemented in the reconstruction step, the particle energy evaluated by the PFClustering algorithm tends to be underestimated. To compensate for this, an energy correction is applied to the PFClusters obtained in the preceding section. In the CMS experiment, this correction is carried out using a multivariate technique known as *boosted decision trees* (BDT) [11] and trained on simulation. A comprehensive list of all inputs to the BDT used in this study with their brief descriptions is presented in Table 1.

While the reconstruction algorithm implemented in the CMS experiment utilizes a limited subset of variables for energy correction, we have chosen to incorporate additional variables to enhance its performance. Detailed information on the training and optimization of the BDT is available in Appendix A. The correction is applied when comparing the energy resolution of the PFClustering algorithm to the one of the DeepCluster algorithm in Sect. 6.

3 Simulation and dataset

To train the DeepCluster model and compare its performance to the one of the PFClustering algorithm, a calorimeter simulation, referred to as a *toy calorimeter*, is created using the Geant4 software [12].

This simulation uses a simple geometry made of a rectangular shape consisting of 51 × 51 crystals, preserving the physical characteristics of the barrel part of ECAL: $PbWO_4$ crystals with dimensions of $2.2 \times 2.2 \times 23$ cm^3. In each crystal, the measured energy is derived by randomly smearing the deposited energy as obtained from Geant4 (see Sect. 3.2). In the simulation, we neglect the ECAL-crystal tilt. Furthermore, additional interactions (pile-up) are not considered.

We first create an initial dataset composed of single high-energy γ directed perpendicularly towards the detector surface. Their energies are uniformly distributed within the range of 1–100 GeV. The position at which the particles enter the toy calorimeter is randomly chosen, avoiding the edges of the detector to ensure that the particle deposits most of its energy in the active material of the calorimeter.

We demonstrate the validity of our toy detector by comparing the energy deposit profiles obtained with our simulation to the ones obtained from the nominal CMS simulation, see Appendix B.

3.1 Datasets

In order to train and test the DeepCluster model, we create two separate datasets (using pandas [13] and numpy [14] libraries from Python programming language) from the aforementioned initial dataset:

- Single-photon dataset.
  Each entry of the dataset consists of one photon. It is used both for training and as a primary check of DeepCluster performance regarding coordinate and energy resolution. It corresponds to the case of isolated particles in the calorimeter.
- Two-photon dataset.
  Each entry is created by superimposing two different samples from the initial dataset. In this dataset, only particles with positions located within a maximum distance

($\Delta R = \sqrt{\Delta x^2 + \Delta y^2}$) of 3 crystals from each other are selected, excluding cases where the two particles enter the calorimeter in the same crystal. This dataset represents two close-by photons in the calorimeter, mimicking the signature of a π_0 decay, for instance.

In this study, the training (resp. validation) dataset consists of a random mixture of 600 k (resp. 200 k) samples from the single-photon dataset and 300 k (resp. 100 k) samples from the two-photon dataset. The training dataset is used to train the DeepCluster model while the validation dataset is used to select the best model ensuring no overtraining. In addition, we estimate the performance with a test dataset composed of 100 k single photons and 50 k two-photon samples.

Moreover, in order to gain a comprehensive understanding of the model performance across various potential use cases, extra datasets used only for evaluation are created:

- Electron dataset.
  This dataset is created in a similar way as the photon one. Each electron has an energy randomly chosen from a uniform distribution in the range [1, 100] GeV. Since electrons are primarily reconstructed within the ECAL, it is important to check the algorithm's performance for these particles.
- Neutral pion dataset.
  To create this dataset, we simulate π^0s emitted at a distance of 130 cm from the toy calorimeter front face and with energies uniformly distributed in the range [1, 100] GeV. It consists of 180k samples.

3.2 Per-crystal energy

Finally, as the simulation does not include all the steps of the ECAL readout chain, the per-crystal energy is obtained by smearing the true deposited energy. In case of two overlapping samples (e.g. two-photon dataset), the true deposited energies are first summed and then smeared. This provides a realistic simulation of the calorimeter response and energy resolution, including the simulation of the readout-electronic noise.

The ECAL energy resolution is parametrized as follows [15]:

$$\left(\frac{\sigma_E}{E}\right)^2 = \left(\frac{a}{\sqrt{E}}\right)^2 + \left(\frac{\sigma_n}{E}\right)^2 + c^2, \tag{3}$$

where a, σ_n, and c are called respectively stochastic, noise, and constant terms. The energy E_{xtal} measured in a crystal $xtal$ is given by:

$$E_{\text{xtal}} = E_{\text{xtal}}^{\text{true}} \times \mathcal{N}\left(\mu = 1; \sigma = \frac{\sigma_{E_{\text{xtal}}^{\text{true}}}}{E_{\text{xtal}}^{\text{true}}}\right), \tag{4}$$

where $\mathcal{N}(\mu;\sigma)$ is a random number following a Gaussian distribution with mean μ and standard deviation σ, and $E_{\text{xtal}}^{\text{true}}$ is the true energy deposited in the crystal $xtal$ obtained from the Geant4 program. The chosen parameters correspond to the ECAL conditions for Run 3 [10,16]: $a = 0.03\ \text{GeV}^{\frac{1}{2}}$, $\sigma_n = 0.167$ GeV, $c = 0.0035$. A lower cut at 50 MeV is applied on the smeared energy to mitigate the noise.

4 DeepCluster model

There has been a surge in the application of machine learning techniques in the field of particle physics over recent years due to the capacity of these algorithms to extract complex patterns. In particular, techniques such as deep neural networks (DNN) and convolutional neural networks (CNN) have enabled unprecedented levels of accuracy and efficiency in tasks such as particle tracking and calorimetry clustering. Graph neural networks (GNN) have also recently gained significant popularity in high-energy physics applications (e.g. [17,18]) due to the following factors:

- they can handle sparse data coming from complex detector geometries,
- they are applicable to non-Euclidean data structures with variable input sizes.

To achieve optimal performance, we combine multiple state-of-the-art machine learning methods in the DeepCluster architecture. First, we use the excellent pattern recognition abilities of CNNs. As energy deposited in the crystals of the calorimeter can be represented as pixel intensities of an image, CNN can be easily and naturally applied to treat calorimeter data. Secondly, we use a GNN to allow information transfer between neighbouring particles.

In the DeepCluster model, the particle-reconstruction task is divided into two consecutive steps:

1. Extract small windows (7×7 crystals), named *seed windows*, whose energy deposits potentially originate from a real particle and not from noise. This is performed by a first NN called the *seed-finder NN* described in Sect. 4.2.
2. For each seed window predict the kinematic properties of the corresponding generated particle. This is done with a second NN called *center-finder NN*. In this work, we develop two different approaches for the center-finder NN:
   - The first one, based on a CNN, is described in Sect. 4.3.
   - To circumvent the limitations of this CNN-based center-finder, a second center-finder, using a GNN, is introduced in Sect. 4.4.

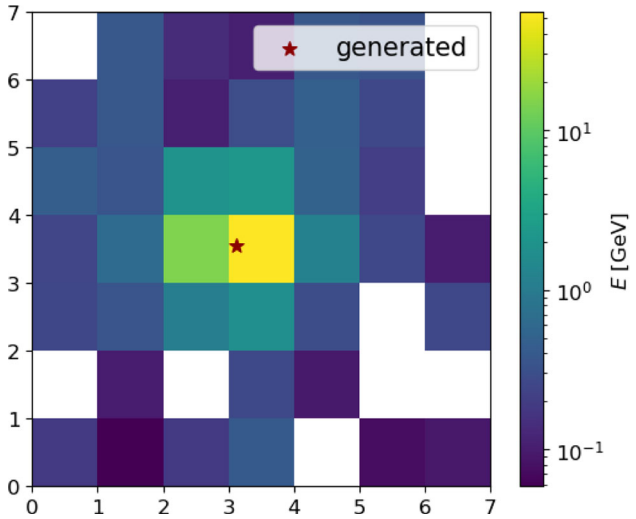

Fig. 1 Example of a seed window

With this approach, the networks process only small crystal matrices as opposed to the full calorimeter matrix (51 × 51 crystals). This significantly reduces the need for computational power and allows to easily scale this method to a real detector (the ECAL barrel contains 170 × 360 crystals).

4.1 *Seed windows* definition

The inputs to the DeepCluster model are called *seed windows* and are obtained for each sample as follows:

1. All the crystals from the toy calorimeter with energy deposits $E_{\text{xtal}} > 0.5$ GeV are selected and defined as *seed crystals*. The selected 0.5 GeV threshold value corresponds to $E_{\text{thr}}^{\text{seed}}$ used in the PFClustering algorithm as tuned for Run 3 operations.
2. For each seed crystal, a seed window is created. This is a matrix of size 7 × 7 crystals centered on the seed crystal. An example of a seed window is shown in Fig. 1.

From one simulated sample corresponding to the full toy calorimeter, several seed windows can be created. They originate from a real particle or from noise.

In order to train the different networks, we need to associate the seed windows to their corresponding generated particles and the subsequent truth labeling has to be defined. We first check if the impact position of any generated particle in the considered sample lies within the boundaries of the seed crystal:

- In such a case, the corresponding particle (at most one by construction) is associated to the seed window. The window is labeled *true seed window* and assigned three kinematic variables corresponding to the true particle: generated position $(x_{\text{gen}}, y_{\text{gen}})$ and energy E_{gen}.
- Otherwise, it is labeled as background.

For true seed windows, the local position of the generated particle inside the seed window $(x_{\text{loc}}, y_{\text{loc}})$ is defined as:

$$\begin{aligned} x_{\text{loc}} &= x_{\text{gen}} - x_{\text{win}} \\ y_{\text{loc}} &= y_{\text{gen}} - y_{\text{win}} \end{aligned} \tag{5}$$

where $(x_{\text{win}}, y_{\text{win}})$ corresponds to the position of the seed window center.

4.2 Seed-finder NN

The seed-finder NN is the first network in the DeepCluster model. This is a CNN, whose goal is to select the true seed windows and discard background ones.

The network takes a seed window as input and assigns to it a seed-finder score (P_{seedSF}), indicating the likelihood to be a true seed window. The seed windows with $P_{\text{seedSF}} < P_{\text{seedSF}}^{\text{thr}}$, where $P_{\text{seedSF}}^{\text{thr}}$ is a tunable threshold, are discarded, the other ones are passed to the center-finder NN.

The seed-finder-NN architecture consists of two convolutional and two dense layers. LeakyReLU is chosen as the activation function [19] and a dropout of 10% is applied after the first convolutions. For the output of the seed-finder NN, a sigmoid activation function is used. A detailed view of the model architecture is presented in Fig. 2.

The model is trained using the Adam optimizer [20] with a learning rate of 0.0001 and a batch size of 64. We use the binary cross entropy as the loss function. The network is trained for ≈100 epochs and the epoch yielding the best result on the validation dataset is chosen.

Compared to the seeding step used in the PFClustering algorithm, the seed-finder NN provides several advantages:

- As the condition for the seed to be a local maximum is removed, the seed-finder NN provides a better possibility to reconstruct close-by photons.
- The seed-finder NN performs a refined seed window selection that helps to significantly eliminate the low-energy background coming from electronic noise.

4.3 Center-finder NN: convolutional neural network

The center-finder NN is the second step of the DeepCluster model. It predicts the position $(x_{\text{reco}}^i, y_{\text{reco}}^i)$ and energy E_{reco}^i of the generated particle associated to the seed window i; the corresponding generated quantities are named $(x_{\text{loc}}^i, y_{\text{loc}}^i)$, E_{gen}^i. The global coordinates can be further inferred by inverting Eq. 5.

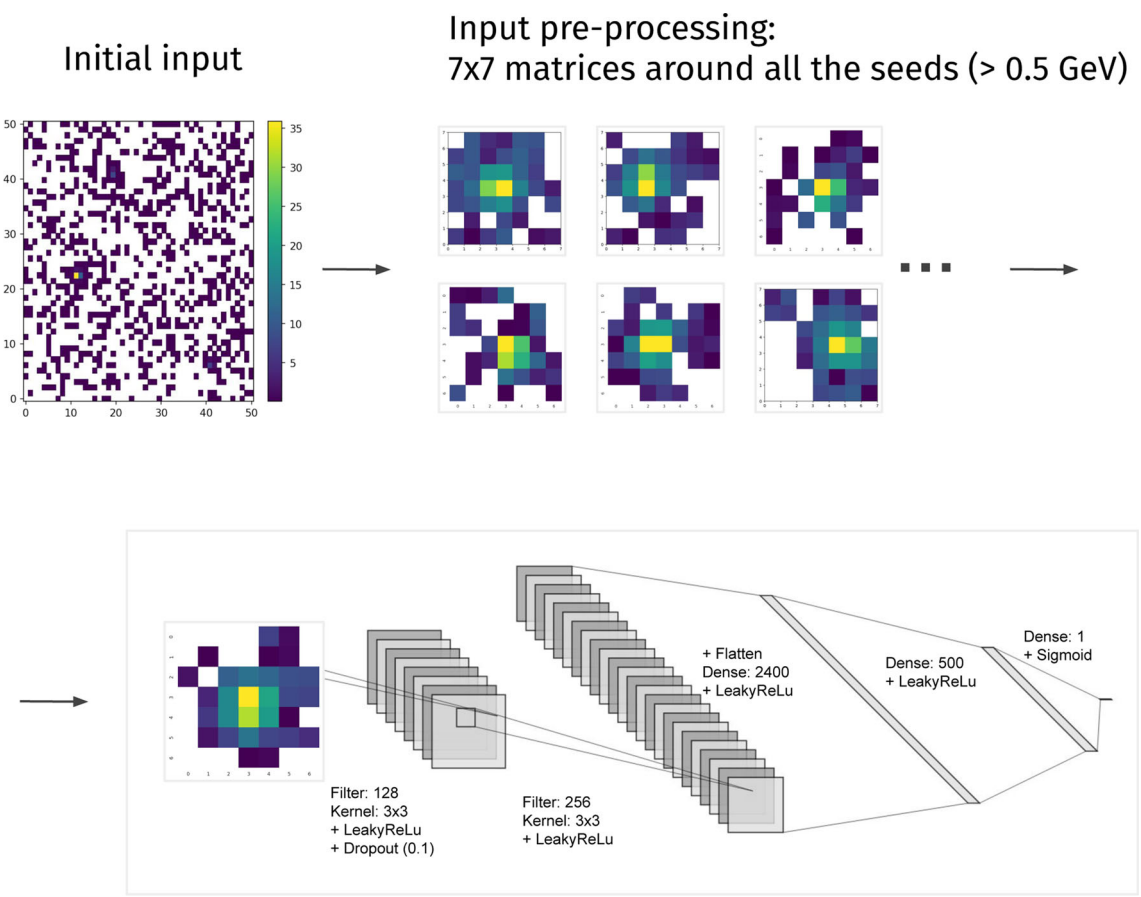

Fig. 2 Seed-finder NN architecture. 7×7 seed windows are first selected around all possible seeds ($E_{\text{xtal}} > 0.5$ GeV). They are separately passed as input to the seed-finder NN. The input is processed by two convolutional layers until the vector of summary features is extracted. This vector is further passed to two dense layers, resulting in the network output: the seed-finder score P_{seedSF}. It represents the likelihood of the input to originate from a generated particle. Detailed information on the number of nodes at each layer is presented in the figure

Similarly to the seed-finder NN, the CNN-based center-finder takes seed windows as input. All the inputs are processed independently from each other. In the training phase, the inputs are limited to the true seed windows. However, for the evaluation, the center-finder NN processes all the seed windows with the seed-finder scores passing $P^{\text{thr}}_{\text{seedSF}}$.

The architecture of the center-finder NN is close to the one of the seed-finder NN: it consists of multiple convolutional layers, followed by dense layers that are divided into two parts: one resulting in coordinate prediction and another in energy prediction. LeakyReLU activation function is applied everywhere except for the output layer, where tanh and sigmoid functions are used respectively for position and energy predictions [19]. The dropout level is set to 10% everywhere except for the last energy prediction layer, where it is set to 30%. The full network architecture with precise details on the number of nodes is presented in Fig. 3.

The network is trained for 1000 epochs with a batch size N_b of 64 seed windows. We use the mean absolute error loss defined by:

$$\mathcal{L}_{\text{kin}} = \frac{1}{N_b}\sum_{i=1}^{N_b}\frac{1}{2}\left(|x^i_{\text{reco}} - x^i_{\text{loc}}| + |y^i_{\text{reco}} - y^i_{\text{loc}}|\right) + |E^i_{\text{reco}} - E^i_{\text{gen}}|. \quad (6)$$

The training is performed using the Adam optimizer with a learning rate of 0.0001. The chosen epoch is the one providing the best performance according to the validation dataset.

4.3.1 Results

The results of the DeepCluster model and the PFClustering algorithm are compared with the single- and two-photon test datasets. For the seed-finder NN, we set $P^{\text{thr}}_{\text{seedSF}} = 0.3$. This latter threshold is tuned for the final DeepCluster model as discussed in Sect. 5.

The performance for the position prediction is presented in Fig. 4 for the single-photon dataset on the left and for the two-

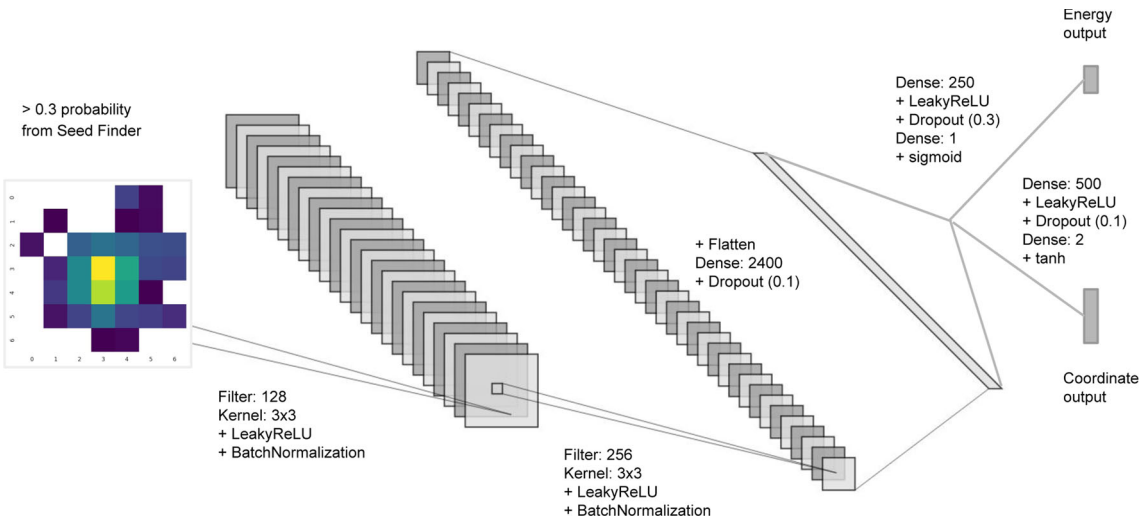

Fig. 3 Center-finder NN architecture. The seed windows are passed separately as input to the network. Each input is processed by two convolutional layers until the vector of summary features is extracted. This vector is passed through one dense layer and further sent separately to two different branches (coordinate and energy predictions). In each branch, it passes through two additional dense layers. Detailed information on the number of nodes at each layer is presented in the figure

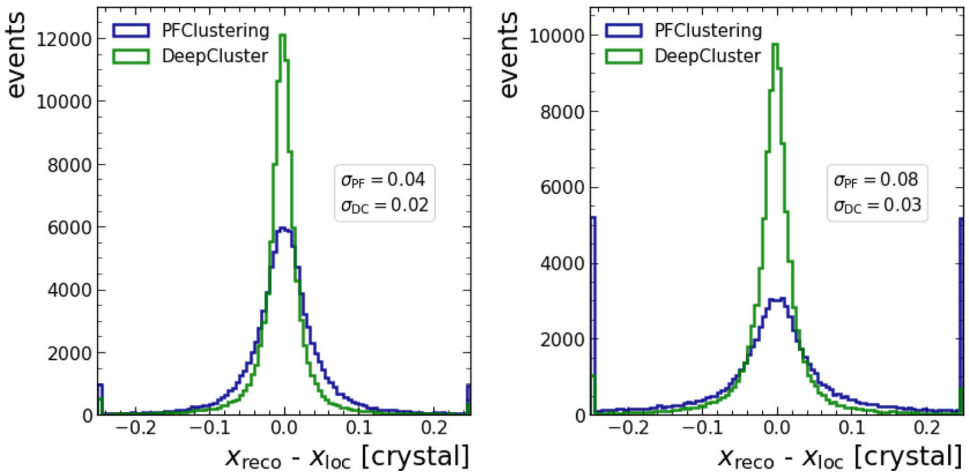

Fig. 4 Distribution of the variable $x_{reco} - x_{loc}$. The results are obtained by applying the PFClustering algorithm and DeepCluster network on the single-photon (left) and two-photon (right) test datasets. The resolutions (see text) are reported in the figures

photon dataset on the right. Each plot shows the distribution of the difference $x_{reco} - x_{loc}$. Similar results are obtained for the y-coordinate. We associate reconstructed objects to generated objects using a matching procedure described in Appendix C.

The DeepCluster network significantly outperforms the PFClustering algorithm. The coordinate resolution (evaluated as half the interval containing 68% of the distribution and centered on the median) for the DeepCluster is 0.02 crystal compared to 0.04 crystal for the PFClustering algorithm for the single-photon dataset and 0.03 crystal versus 0.08 crystal for the two-photon dataset.

The position reconstruction for the two-photon dataset is a more difficult task than for the single-photon one because of overlapping energy clusters for close-by generated particles. This explains the performance degradation in the two-photon dataset.

Concerning the energy prediction, the performance of the PFClustering and DeepCluster algorithms is closer. They are presented for the optimized DeepCluster algorithm in Sect. 6.

4.3.2 Per-event energy overestimation

Because the network processes only seed windows, this approach can be easily extended to a real calorimeter. How-

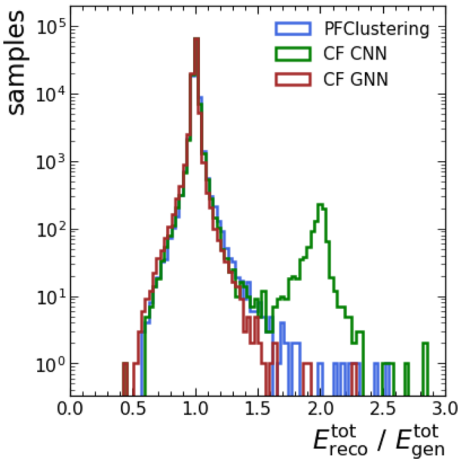

Fig. 5 Distributions of the ratio R_{en} between the total reconstructed energy E^{tot}_{reco} and the total generated energy E^{tot}_{gen}. The results are obtained with the PFClustering algorithm, the DeepCluster model with a CNN-based center-finder CNN (CF CNN), and with a GNN-based center-finder (CF GNN). The distributions are obtained from the single-photon test dataset. A second peak $R_{en} \approx 2$ arises for the CF CNN, while it is eliminated with the CF GNN

ever, it also raises a major issue described in this section. Contrary to the PFClustering, the local maximum condition is omitted for the seed crystal in the DeepCluster. As a consequence, two (or more) neighbouring crystals can be selected as seeds. While this allows for efficient reconstruction of close-by particles, this can also create two or more separate seed windows corresponding to a single generated particle, which would share energy among several crystals. For the majority of these latter cases, the seed-finder NN is able to identify the local maximum itself and predict a high seed-finder score P_{seedSF} for the corresponding seed window (in which the local maximum is the central crystal) and a low score for its neighbour window.

However, when the position of the generated particle is close to the edge of a crystal, its energy deposit can be very similar in the two neighbouring crystals. Both of them get a high P_{seedSF}, giving rise to two distinct seed windows. These windows are separately passed to the center-finder NN that predicts similar coordinates and energies for both of them. The same generated particle can therefore be reconstructed twice by the DeepCluster algorithm, thus overestimating the total energy reconstructed in the event. In the following, this is referred to as energy *double counting*.

This effect is illustrated in Fig. 5, which presents the distribution of the ratio $R_{en} = E^{tot}_{reco}/E^{tot}_{gen}$ where E^{tot}_{reco} and E^{tot}_{gen} are respectively the total reconstructed energy E^{tot}_{reco} (sum of the energies of all the reconstructed particles in the event) and the total generated energy E^{tot}_{gen} (sum of the energies of all the generated particles in the event). The distribution of R_{en} is obtained from the single-photon dataset. The additional peak around two observed for the CNN-based center-finder (CF CNN on the figure) originates from the energy double counting.

The double counting issue is cured by changing the center-finder NN architecture as presented in the next section.

4.4 Center-finder NN: graph neural network

The energy double counting is due to the fact that the CNN-based DeepCluster model does not receive information from the rest of the event, as a consequence, it is not aware of the existence of several seed windows corresponding to the same generated particle.

To solve the issue, we pass several neighbouring seed windows as input to the network. This is achieved using a GNN architecture for the center-finder NN while the seed-finder NN remains unchanged. The GNN implementation provides in addition message-passing capabilities [21] enabling information sharing between the different seed windows.

The inputs to the GNN-based center-finder are constructed as follows. First, the list of seed windows in the event is ordered based on the energy of its center crystal E_{seed}. This list is processed as follows:

1. A center-finder input is initialized from the window w_{ref} with the highest energy E_{seed} in the list. At this step, the input shape is $1 \times 7 \times 7$.
2. For each remaining seed window w_{alt} in the list, the ΔR distance between w_{ref} and w_{alt} is computed. If $\Delta R < 3$, w_{alt} is added to the input. After this step, the input shape is $N_w \times 7 \times 7$ where N_w is the total number of selected seed windows (including w_{ref}).
3. All of the seed windows included in the input are removed from the list of seed windows. The process is iterated until this list is empty.

In this work, N_w is chosen to be at most 4. This maximum can be easily adjusted to higher values as well. For each input, if $N_w < 4$, it is completed with 7×7 empty windows, i.e. with null crystal energies. In such a way, the input shape is fixed to $4 \times 7 \times 7$.

The 4 seed windows of the input represent the nodes of the graph. In the center-finder NN, each seed window j is first separately processed by a chain of convolutional layers (identical to the CNN center-finder implementation) in order to extract the vector of summary features v_j. The message-passing is implemented as the concatenation of these vectors. It results in 4 updated vectors $\bar{v}_j$, each of them containing

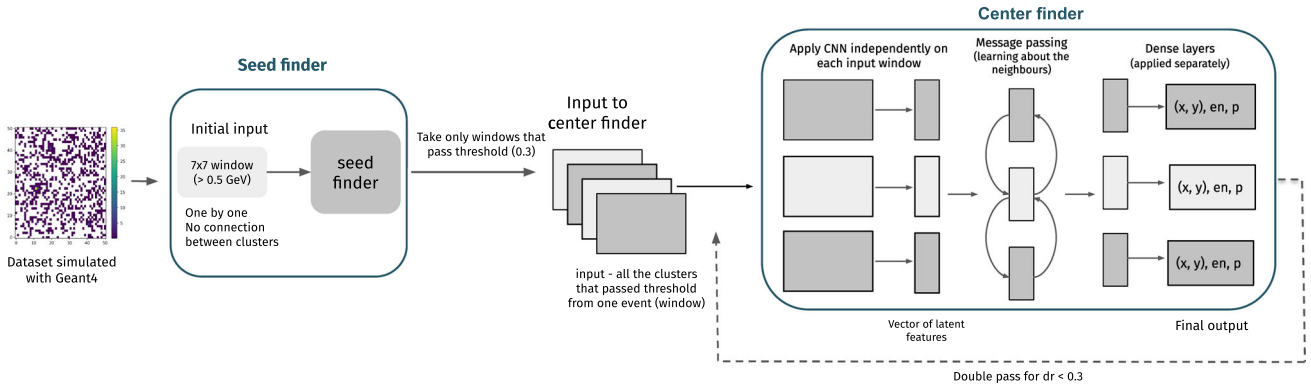

Fig. 6 Flow chart of the DeepCluster model. 7×7 seed windows are first selected around all possible seeds (>0.5 GeV) in the event. They are separately passed as input to the seed-finder NN predicting P_{seedSF} for each seed window. Selected seed windows with $P_{\text{seedSF}} > P^{\text{thr}}_{\text{seedSF}}$ are combined into groups of 4 with their neighbours and passed to center-finder NN which predicts the coordinates x_{reco}, y_{reco}, the energy E_{reco} and a new seed score P_{seedCF} for each seed window

information about their neighbours:

$$\bar{v}_1 = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{pmatrix}, \quad \bar{v}_2 = \begin{pmatrix} v_2 \\ v_1 \\ v_3 \\ v_4 \end{pmatrix}, \quad \bar{v}_3 = \begin{pmatrix} v_3 \\ v_1 \\ v_2 \\ v_4 \end{pmatrix}, \quad \bar{v}_4 = \begin{pmatrix} v_4 \\ v_1 \\ v_2 \\ v_3 \end{pmatrix}, \tag{7}$$

The combined vectors $\bar{v}_j$ are then passed independently to a set of dense layers until the final output is extracted. The GNN-based center-finder NN predicts 4 values for each seed window j in the input i: the kinematic variables (x^{ij}_{reco}, y^{ij}_{reco}), E^{ij}_{reco} and a new seed score P^{ij}_{seedCF}, indicating the likelihood to be associated to a true generated particle. In this version of the DeepCluster model, the seed-finder NN serves as an initial filter, separating signal from background, while the GNN-based center-finder corrects for the wrong predictions related to double counting: eventually only objects with $P_{\text{seedCF}} > P^{\text{thr}}_{\text{seedCF}}$ are selected. The optimization of $P^{\text{thr}}_{\text{seedCF}}$ is presented in Sect. 5. The GNN-based center-finder architecture is presented in Fig. 6. The loss used in the training is the sum of three terms related to the coordinates, the energy, and the seed probability predictions. The two first terms are based on a mean absolute error loss while the last one is based on a focal cross-entropy loss [22]. The combined loss is computed as:

$$\mathcal{L}_{\text{pos}} = \frac{1}{4 \cdot N_b} \sum_{i=1}^{N_b} \sum_{j=1}^{4} \frac{1}{2} \times \left(|x^{ij}_{\text{reco}} - x^{ij}_{\text{loc}}| + |y^{ij}_{\text{reco}} - y^{ij}_{\text{loc}}| \right)$$

$$\mathcal{L}_{\text{en}} = \frac{1}{4 \cdot N_b} \sum_{i=1}^{N_b} \sum_{j=1}^{4} |E^{ij}_{\text{reco}} - E^{ij}_{\text{gen}}|$$

$$\mathcal{L}_{\text{seed}} = -\frac{1}{4 \cdot N_b} \sum_{i=1}^{N_b} \sum_{j=1}^{4} \alpha (1 - P^{ij}_{\text{seedCF}})^{\gamma} \log(P^{ij}_{\text{seedCF}}),$$

$$\mathcal{L}_{\text{tot}} = \mathcal{L}_{\text{pos}} + \mathcal{L}_{\text{en}} + k_{\text{s}} \cdot \mathcal{L}_{\text{seed}}, \tag{8}$$

where N_b is the number of batches, the focal loss parameters γ and α are chosen to be $\gamma = 2$, $\alpha = 0.25$, and k_{s} is an adjustable parameter associated with the seed loss, it is referred to as *seed-loss weight*.

4.4.1 Results

In the GNN implementation of the center-finder NN, the model receives information about all the neighbouring seed windows simultaneously. With this adjustment, the network is able to make a more informed decision for each seed window and additionally better attribute energy fractions for different windows.

The related improvement is shown in Fig. 5 where the distribution of R_{en} obtained with the single-photon dataset is presented. For the center-finder GNN (CF GNN) only the seed windows with $P_{\text{seedCF}} > 0.4$ are selected. One can notice the disappearance of the peak at 2 signalling the resolution of the double-counting issue.

5 Network optimization

The DeepCluster model has a number of adjustable parameters. This section describes the optimization of these parameters to achieve the best possible performance. This starts with the optimization of the seed loss weight, followed by

the optimization of the different seed-probability thresholds: $P^{\text{thr}}_{\text{seedSF}}$ and $P^{\text{thr}}_{\text{seedCF}}$. Then another step is added to the algorithm in order to suppress multiple predicted particles arising from the same generated one. Eventually, the adjustment of the hyperparameters which underlie the different NNs is presented.

5.1 Seed-loss weight

The GNN-based center finder contains in its loss a term related to the predicted seed probability which is weighted by the parameter k_s. The definition of the total loss is given by Eq. 8, where the parameter k_s is introduced. The optimal value of k_s is obtained by comparing the performance achieved with different seed-loss weights k_s. The evolution of the losses for the training and validation datasets are shown in Appendix D for different values of k_s.

5.2 Seed-probability thresholds

In the final DeepCluster model implementation, the seed-finder NN and the center-finder NN predict two different seed scores: P_{seedSF}, P_{seedCF}. Solely the seed windows fulfilling the criteria $P_{\text{seedSF}} > P^{\text{thr}}_{\text{seedSF}}$ and $P_{\text{seedCF}} > P^{\text{thr}}_{\text{seedCF}}$ are kept for further analysis, the other ones are discarded as background windows. The energy and position resolutions as well as the signal efficiencies and background rates are studied on the single-photon test sample for different sets of thresholds. We retain the values $P^{\text{thr}}_{\text{seedSF}} = 0.3$ and $P^{\text{thr}}_{\text{seedCF}} = 0.4$ as they ensure excellent performance in terms of position and energy resolutions ($\sigma_x \sim 0.02$ crystal and $\sigma_E \sim 0.56$ GeV) while maintaining a high signal efficiency ($\epsilon \sim 99.5\ \%$).

5.3 Particle splitting

One of the advantages of the DeepCluster model resides in its ability to reconstruct close-by particles. Still, for both the DeepCluster and the PFClustering algorithms, the closer the particles are, the harder it is to disentangle them in the reconstruction process. There is a limit to the distance between two particles below which the reconstruction is not reliable.

This limit is explored in Fig. 7 by comparing the distance between two properly reconstructed objects (i.e. associated to two generated particles) to the distance between two reconstructed objects associated to the same particle. The latter case corresponds to a generated particle giving rise to two very close-by clusters, thus splitting in parts the energy of the original particle, this is referred to as particle splitting. From this figure, one can see that for $\Delta R < 0.3$, the reconstruction of two close-by clusters mostly corresponds to the splitting of a single particle (given that the number of photons in single- and two-particle datasets is the same). On the opposite, for $\Delta R > 0.3$, close-by particles are properly reconstructed and single-particle splitting is negligible.

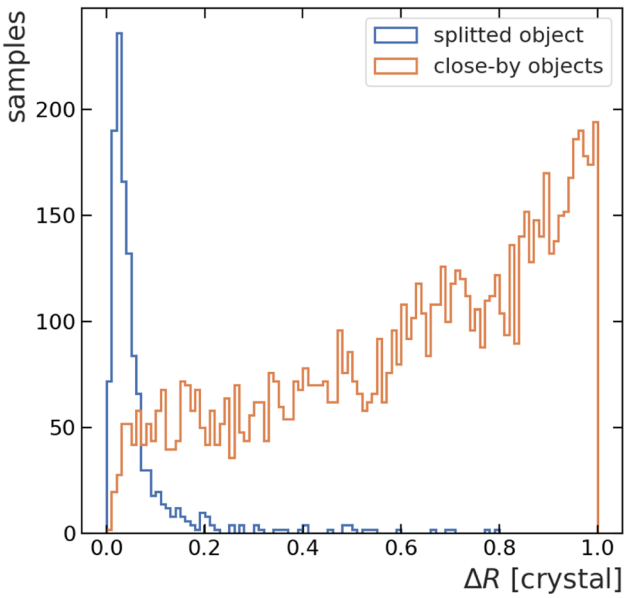

Fig. 7 Distributions of ΔR between two properly reconstructed objects associated to two generated particles (labeled close-by objects), obtained from the two-photon dataset, and ΔR between two close-by reconstructed objects associated to the same particle in the single-photon dataset (labeled splitted object)

In order to mitigate single-particle splitting while maintaining high signal efficiency for two close-by photons, a dedicated procedure is implemented. We first group reconstructed objects with $\Delta R < 0.3$. As aforementioned, these group contain essentially particle-splitting clusters. In these groups, the highest energy object is kept while the others are discarded. The seed window associated to the corresponding selected object is passed a second time in the center-finder NN. This second pass allows to keep a single reconstructed object for a single particle while properly predicting its energy thus avoiding the particle-splitting phenomena.

5.4 Hyperparameters

The tuning of the hyperparameters of the center-finder network is performed using a Bayesian optimization [23]. The optimal parameters are presented in Table 2. The final DeepCluster model is trained using the LAMB optimizer [24]. The epoch achieving the lowest value in the validation dataset is selected.

6 DeepCluster performance

This section presents the performance of the optimized DeepCluster model and compares it to the one obtained with

Table 2 Final values for the DeepCluster model after the Bayesian hyperparameter optimization was performed for center-finder NN

Parameter	Value
1st convolutional layer	number of filters: 128, kernel size: 3 + batch normalization
2nd convolutional layer	number of filters: 112, kernel size: 1 + batch normalization
Common dense layers	nodes: 1100, dropout: 0.1
Center dense layer	nodes: 500, dropout: 0.3
Energy dense layer	nodes: 100, dropout: 0.1
Seed dense layer	nodes: 250, dropout: 0.3
Batch size	512
Learning rate	0.0001

Table 3 Description of the variables used for evaluating the performance of the algorithms

Name	Description
Signal efficiency	Number of correctly reconstructed objects divided by the number of generated particles, calculated for 100 k photons
Particle-splitting yield	Number of events where one particle is reconstructed as two different objects, reported for 100 k photons
Background yield	Number of reconstructed objects that do not correspond to any generated particle, reported for 100 k toy simulations (51 crystals $\times$ 51 crystals)

the PFClustering algorithm. The energy and position resolutions are evaluated as half the interval containing 68% of the distribution and centered on the median. In addition to the position and energy resolutions, other important metrics are investigated: signal efficiency, background yield, and particle-splitting yield, as defined in Table 3. The matching procedure linking the reconstructed objects with the true generated particles to determine if reconstructed objects are to be considered as signal or background is described in Appendix C. Each of the reconstructed objects is linked to at most one generated particle, while the generated particle can be linked to multiple reconstructed objects. In the latter case, the objects are tagged as particle-splitting. The same matching procedure is applied for the objects reconstructed by the DeepCluster model and PFClustering algorithm.

First, the DeepCluster model is tested with photon datasets that are comparable to the ones used for training. In a second step, the DeepCluster reconstruction is tested with electrons and finally, the algorithm is applied to the reconstruction of π^0-meson decays producing collimated photons in the detector.

6.1 Performance for photons

The performance for single-photon and two-photon test datasets is presented in this section.

Figure 8 shows the distributions of the difference between the reconstructed and generated values for position (left) and energy (right) in the single-photon dataset. The position resolution is improved by 50% with regard to the PFClustering algorithm while the energy resolution is improved by about 10%. The improvement is more drastic for the two-photon dataset, where both the position and energy resolutions are improved by about 60%.

This is illustrated in Figs. 9 and 10 where the standard deviation (left) and median (right) of the distributions of the difference between the reconstructed and generated values for energy and position are presented as a function of the generated energy of the particle E_{gen}, for the single- and two-photon datasets. For the energy, the relative resolution is shown. In the two-photon case, the resolutions, for energies above 20 GeV, are improved by more than 70 % for the energy and 60% for the position. The bias observed for the median of the distributions for the DeepCluster model is small compared to the corresponding resolution. It can be either fixed by further optimizing the parameters of the network or corrected for.

The reason for the poor performance of the PFClustering algorithm for the two-photon dataset is two-fold. Firstly, the BDT used for energy correction is trained only on a single-photon dataset (as it is done in the current CMS implementation). Secondly, with higher energy, there is a bigger chance for two separate clusters to be misreconstructed as a single particle by the PFClustering algorithm, thus, largely overestimating energy.

Figure 11 presents the signal efficiency, particle-splitting yield, and background yield for the DeepCluster model and the PFClustering algorithm for the single-photon dataset on the left and for the two-photon dataset on the right. The results are presented as a function of the energy of the generated particle E_{gen} for the signal efficiency and splitting yield and as a function of the energy of the seed crystal E_{seed} for the background yield. In the single-photon case, the signal efficiency is the same as for PFClustering starting from about 5 GeV, while the background rate, coming from noise reconstructed as low energy clusters, is improved by a factor of about 2000. In the two-photon case, the signal efficiency is largely improved, up to a factor of two at low energies. The splitting yield is slightly increased at high energies but the rate stays rather low (in total $\approx 0.3\%$). The splitting yield is a relevant metric that is important to consider, for example, in the context of di-photon resonance searches.

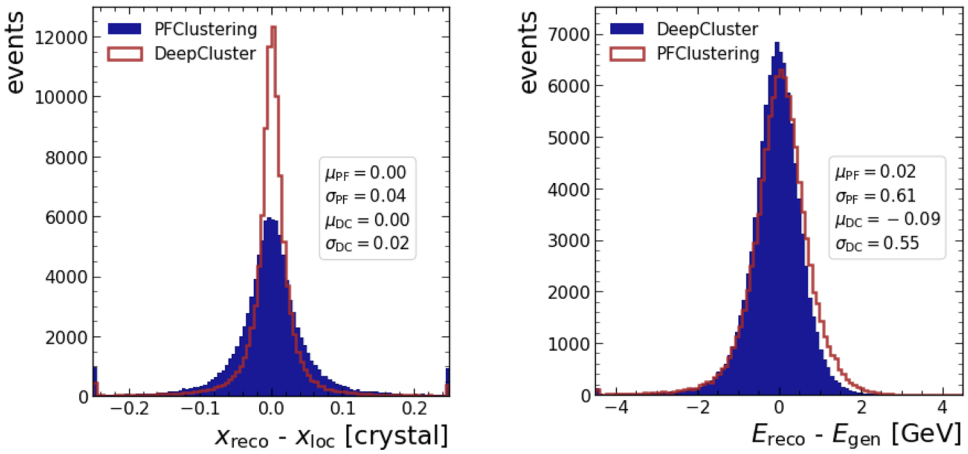

Fig. 8 Left: distribution of the difference between the reconstructed position x_{reco} and the generated position of the particle x_{loc}. Right: distribution of the difference between the reconstructed energy E_{reco} and the generated energy of the particle E_{gen}. The results are obtained by applying the PFClustering algorithm and DeepCluster model on the single-photon test dataset

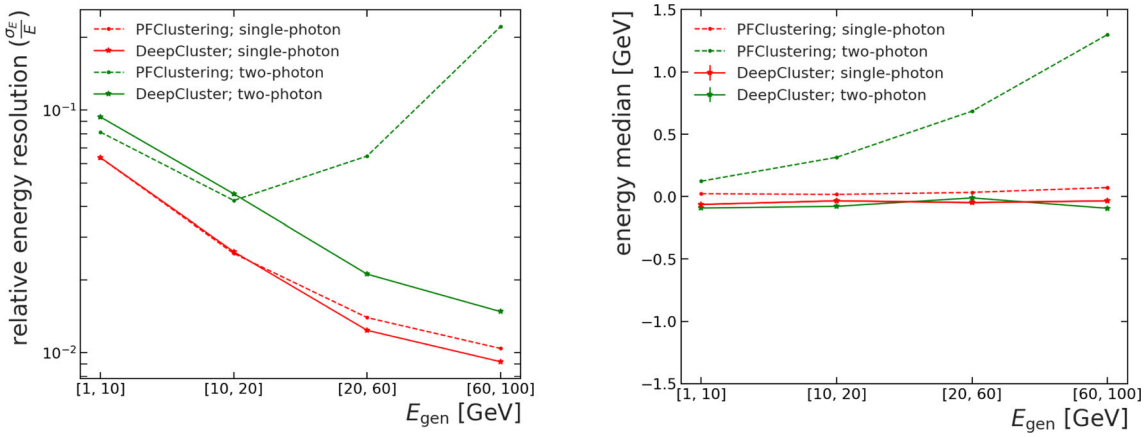

Fig. 9 Relative energy resolution (left) and energy median (right) obtained with the DeepCluster model and PFClustering algorithm applied on the single- and two-photon test datasets. The results are shown in the bins of generated energy E_{gen}

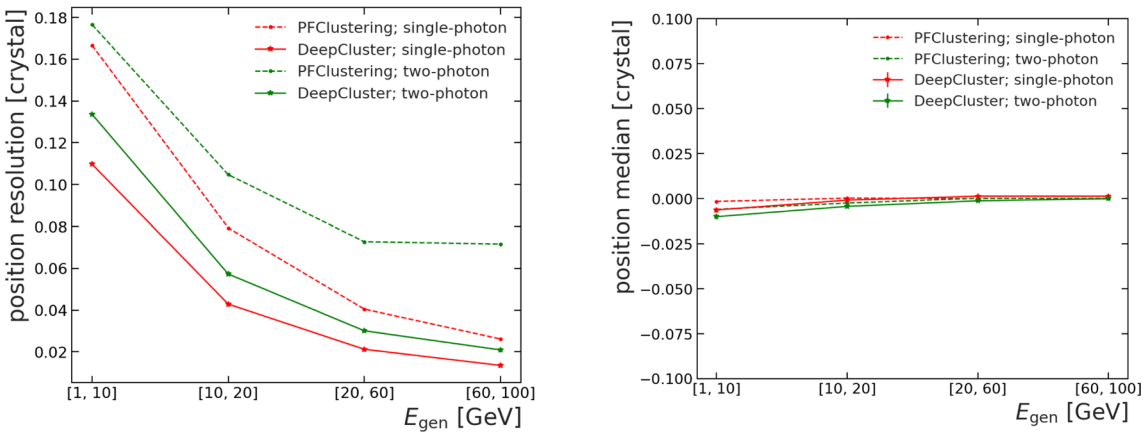

Fig. 10 Position resolution (left) and position median (right) obtained with the DeepCluster model and PFClustering algorithm applied on the single- and two-photon test datasets. The results are shown in the bins of generated energy E_{gen}

A summary of the performance is presented in Table 4. The DeepCluster model outperforms the PFClustering algorithm in terms of position and energy resolution both for the single- and two-photon cases. Most notably, the signal efficiency for the two-photon dataset obtained with the DeepCluster model is 97.0%, while with PFClustering, it is only 82.0%.

All presented results evaluate the performance of each photon within the two-particle dataset without requiring both photons to be properly reconstructed.

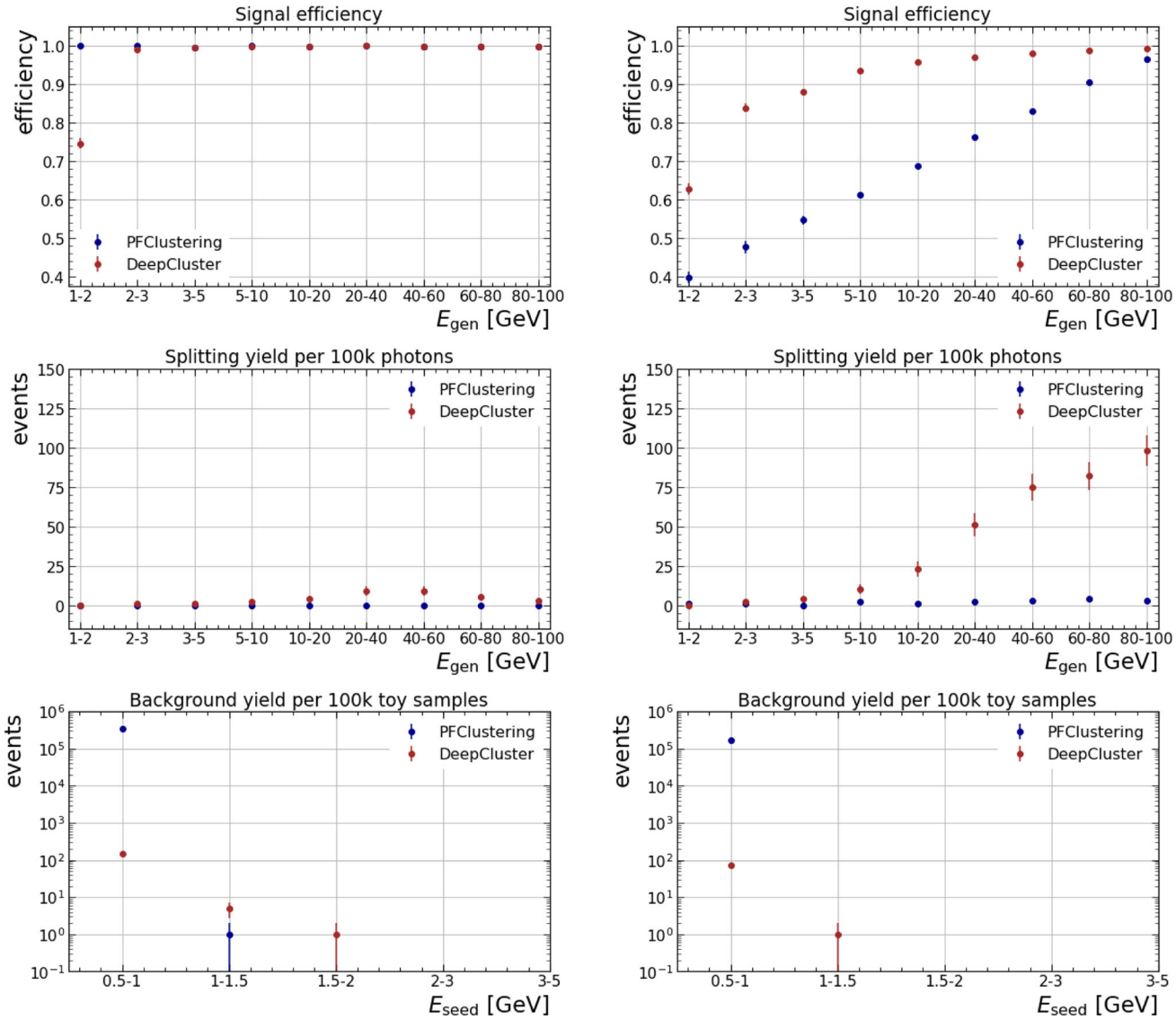

Fig. 11 Signal efficiency (top), splitting yield for 100 k photons (middle), and background yield for 100 k toy simulations (bottom) values. The results are obtained by applying the PFClustering and DeepCluster model on the single-photon (left) and two-photon (right) test datasets

Table 4 Performance comparison for position and energy resolutions, signal efficiency, splitting yield for 100k photons, and background yield for 100k toy simulation between PFClustering and DeepCluster algorithms for single- and two-photon datasets

	Single-photon		Two-photon	
	PFClustering	DeepCluster	PFClustering	DeepCluster
σ_x [crystal]	0.04 ± 0.00	0.02 ± 0.00	0.08 ± 0.00	0.03 ± 0.00
σ_E [GeV]	0.61 ± 0.00	0.55 ± 0.00	6.24 ± 0.01	0.92 ± 0.00
ϵ	99.8 ± 0.3 %	99.5 ± 0.3%	82.0 ± 0.3%	97.0 ± 0.3%
N_{split}/100k photons	0	34 ± 6	17 ± 4	345 ± 19
N_{bkg}/100k toy samples	350k $\pm$ 19k	153 ± 12	320k $\pm$ 18k	146 ± 12

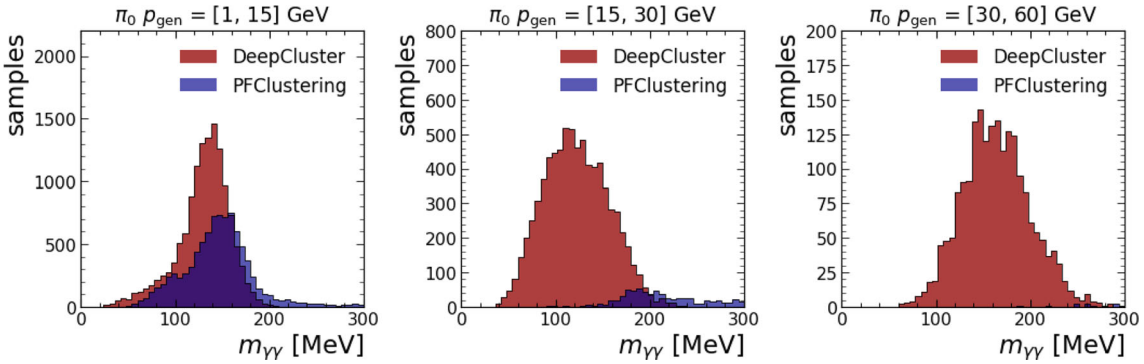

Fig. 12 $m_{\gamma\gamma}$ mass distributions reconstructed with DeepCluster model and PFClustering algorithm on the π_0 dataset. The results are shown in bins of the generated momentum p_{gen} of the π_0

6.2 Electrons

The DeepCluster algorithm is tested on the electron dataset. Although the network is not specifically trained on it, it still shows excellent performance. The resolutions in energy and position, the signal efficiency, and the splitting and background yields are extremely similar to the ones obtained for photons, as expected in absence of magnetic field and material in front of the calorimeter.

6.3 Neutral pions

Finally, the results achieved on the π_0 sample are presented. In this case, the reconstruction algorithms have to detect both of the photons originating from the π_0 decay and correctly estimate their energy. With this information, the mass of the π_0 can be reconstructed as:

$$m_{\gamma\gamma} = \sqrt{2E^1_{\text{reco}}E^2_{\text{reco}}\,(1-\cos\theta_{\text{reco}})}, \tag{9}$$

where E^1_{reco}, E^2_{reco} are the reconstructed energies of two photons and θ_{reco} is the reconstructed angle between them.

The diphoton mass distributions reconstructed with the DeepCluster model and PFClustering are presented in Fig. 12. The results are shown in bins of the generated momentum p_{gen} of the π_0. When the π_0 momentum increases, the two photons get closer and are harder to reconstruct separately. The DeepCluster model achieves excellent results, outperforming the PFClustering π_0 detection efficiency by a factor of more than two. Moreover, the diphoton mass resolution is significantly better with the model.

As in the case of electrons, the DeepCluster model is not specifically trained on the π_0 dataset. Moreover, the photons enter the toy calorimeter under different angles and not perpendicularly as in the training sample. This further underscores the robustness of the network.

7 Conclusion

This paper introduces an innovative machine-learning algorithm, called the DeepCluster model, to measure the energy and position of photons and electrons based on convolutional and graph neural networks, taking the geometry of the CMS electromagnetic calorimeter as an example.

To develop the DeepCluster model and evaluate its performance, a dedicated simplified simulation of the ECAL is created, and different implementations of the model are tested. A two-step network strategy, incorporating both CNN and GNN architectures, delivers the best performance and effectively addresses all identified issues. The final model is tested on datasets with single photons and two overlapping photons, as well as on datasets with electrons or neutral pions. In all cases, the DeepCluster shows superior performance compared to the method currently in use in CMS in terms of coordinate and energy resolutions, as well as background rejection and signal efficiency. In particular, the DeepCluster model demonstrates excellent results in distinguishing between closely spaced particles, reconstructing approximately twice as many π^0 as the traditional approach.

These results demonstrate that this approach is very promising to enhance the performance of calorimeters in high-energy physics experiments. Moreover, as the network processes only small 7×7 windows, the scalability for the full ECAL will not pose a problem. The particles being either far apart in the calorimeter (single-photon dataset) or in close proximity (two-photon dataset), all physics cases are covered. Finally, the performance gain of this approach could be even larger in presence of pile-up interactions, as the local information could allow the network to better mitigate their impact.

Acknowledgements This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 800945 - NUMERICS - H2020-MSCA-COFUND-2017. We gratefully thank the CMS experiment from which this work was inspired. We also acknowledge the CEA (France) for financial and computing support and Dr. James Rich and Dr. Nathalie Besson for their careful reading of this manuscript.

Data Availability Statement Data will be made available on reasonable request. [Authors' comment: The datasets generated during the current study are available from the corresponding author on reasonable request.]

Code Availability Statement This manuscript has associated code/software in a data repository. [Authors' comment: The code generated during the current study is available in the DeepCluster repository, https://github.com/polinasimkina/DeepCluster.git.]

Funded by SCOAP[3].

Appendix A

The BDT is trained on a single-photon dataset with a flat energy distribution between 1 and 250 GeV. The hyperparameters of the model, such as *learning rate*, *number of estimators*, *minimum split*, *minimum leaf*, and *maximum depth*, are optimized to achieve the best performance using a random grid search.

The resulting regression scores for 20 different sets of parameters are calculated. The full summary of optimization is presented in Table 5 and parameters corresponding to the highest score (trial 10) are selected based on these results.

The performance in terms of energy for the final optimized model is shown in Fig. 13 along with the comparison to the raw PFClustering prediction. With the BDT correction, the energy resolution is drastically improved compared to the raw results: by 35%, 36%, 43%, and 26% for energies of generated particles $E_{\text{gen}} = [1, 10]$ GeV, $E_{\text{gen}} = [10, 20]$ GeV, $E_{\text{gen}} = [20, 60]$ GeV, and $E_{\text{gen}} = [60, 100]$ GeV,

Table 5 Results for the BDT hyperparameter optimization. The algorithm is applied on the single-photon validation dataset, and the regression score measures its performance. The parameters of the 10th trial are chosen as optimal as they correspond to the highest value of the regression score

Trial	Learning rate	Number of estimators	Minimum split	Minimum leaf	Maximum depth	Score
1	0.20	100	20	5	20	0.7799
2	0.05	300	10	10	2	0.7676
3	0.20	300	10	5	10	0.7815
4	0.05	300	20	2	2	0.7756
5	0.20	300	2	10	5	0.7886
6	0.05	500	4	5	20	0.7797
7	0.20	100	10	2	10	0.7952
8	0.05	100	20	2	5	0.7874
9	0.20	500	20	2	5	0.7947
10	0.10	100	2	2	10	0.7953
11	0.20	300	10	2	10	0.7887
12	0.10	300	2	10	2	0.7754
13	0.10	300	4	10	10	0.7850
14	0.20	300	2	10	10	0.7782
15	0.20	500	20	2	2	0.7913
16	0.05	100	2	5	10	0.7913
17	0.10	300	4	5	5	0.7951
18	0.10	500	20	10	10	0.7828
19	0.20	300	4	10	10	0.7782
20	0.05	300	2	2	5	0.7952

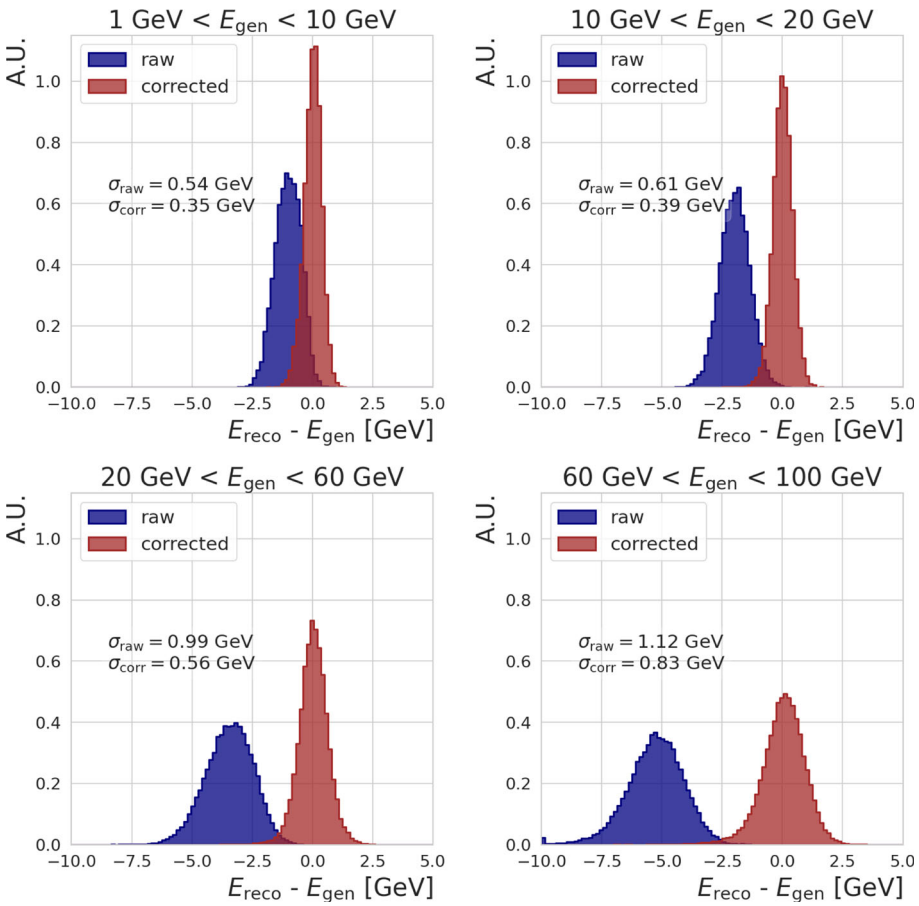

Fig. 13 Distribution of the difference between the reconstructed energy E_{reco} and the generated energy of the particle E_{gen}. The results are obtained by applying the PFClustering algorithm (raw) and the PFClustering algorithm with additional energy regression (corrected) on the single-photon test dataset. The distributions are presented in four bins of the generated photon energy E_{gen}: [1, 10] GeV, [10, 20] GeV, [20, 60] GeV, and [60, 100] GeV. The resolutions evaluated from these distributions are reported on the plots

respectively. Moreover, the corrected energy distributions are centered around zero, unlike the ones obtained only from the raw PFClustering algorithm.

Appendix B

To validate the simulation of the toy calorimeter, the energy deposit profile is compared to the simulation of the actual ECAL. To do so, 1,000 electrons at 100 GeV are shot at the center of the detector, and the average deposited energy in each cell of the 5 x 5 crystal matrix around the central crystal is calculated. The results are shown in Fig. 14 (left) and they are compared to the ones achieved with the real ECAL simulation presented in Fig. 14 (right).

The obtained results are comparable: the average ratio of energy deposits of initial electrons in the central crystal (E1), 3 × 3 matrix (E3), and 5 × 5 (E5) matrix around it are approximately 78%, 94%, 97% for toy calorimeter and 79%, 95%, 98% for ECAL simulation [25], respectively. This demonstrates that the toy calorimeter can be used as a proxy for the real ECAL.

The asymmetry in the energy deposition around the central crystal is not present for the toy calorimeter (unlike in the ECAL simulation) as it does not include crystal tilt.

Appendix C

A matching procedure that links the reconstructed objects with the true generated particles is applied to determine if reconstructed objects are to be considered as signal or background. A "matching" variable r_{match} is defined:

$$r_{match} = \sqrt{\left(\frac{\Delta R}{\langle \Delta R \rangle}\right)^2 + \left(\frac{E_{reco} - E_{gen}}{\langle E \rangle}\right)^2}, \quad (10)$$

where $\Delta R = \sqrt{(x_{reco} - x_{loc})^2 + (y_{reco} - y_{loc})^2}$, $\langle \Delta R \rangle$ is the mean value of the ΔR distribution and $\langle E \rangle$ is the mean value of the (E_{reco} - E_{gen}) distribution. These values are obtained from a preliminary truth association based only on the position. For each reconstructed object in the sample:

- r_{match} is computed for the considered reconstructed object and all the generated particles;

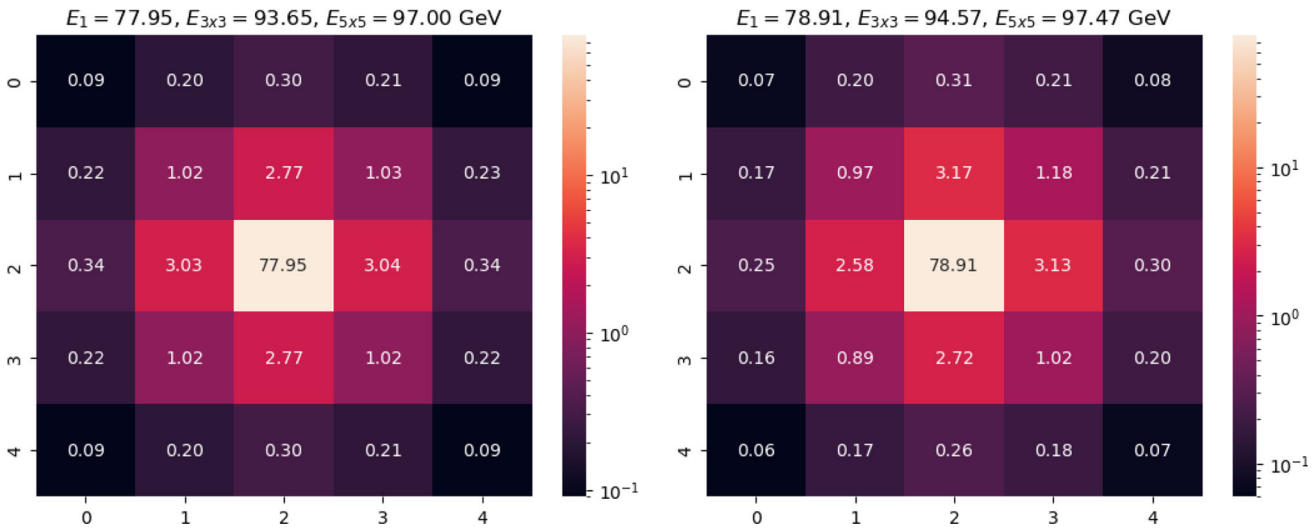

Fig. 14 Energy deposit profiles from the toy calorimeter (left) and from Geant4 simulation of ECAL (right) [25]. The profiles are obtained with an electron beam

- A link is created between the reconstructed object and the particle with the smallest r_{match};
- If $\Delta R > 1.5$ crystal, the link is removed and the reconstructed object is considered as background.

Appendix D

The three terms in the loss function referenced in Eq. 8 collectively contribute to the minimization of the total loss function. The part of the neural network responsible for predicting the seed probability has shown a tendency for overfitting. This observation is supported by the divergence of the validation sample loss, as depicted in Fig. 15. Predominantly, a higher contribution from the seed probability loss dominates the total validation loss. This behavior results in two undesirable effects. First, the total loss begins to diverge on the validation sample, causing the epoch corresponding to the minimum validation total loss to represent a less-than-ideal configuration for the network's other two outputs. Secondly, the gradients are disproportionately affected by the seed loss, leading to a suboptimal training for the remaining outputs. To mitigate these effects, we introduced a penalty term into the seed probability loss function. Based on the physics performance metrics associated with different penalty terms shown in Fig. 16, we determined that applying a penalty term of $k_s = 0.05$ to the seed probability loss produces a reduced cumulative loss across all three networks. The model configuration at the epoch n $= 321$ is considered as the optimal one since it has the smallest validation loss value and all of the three terms of the loss function are converged.

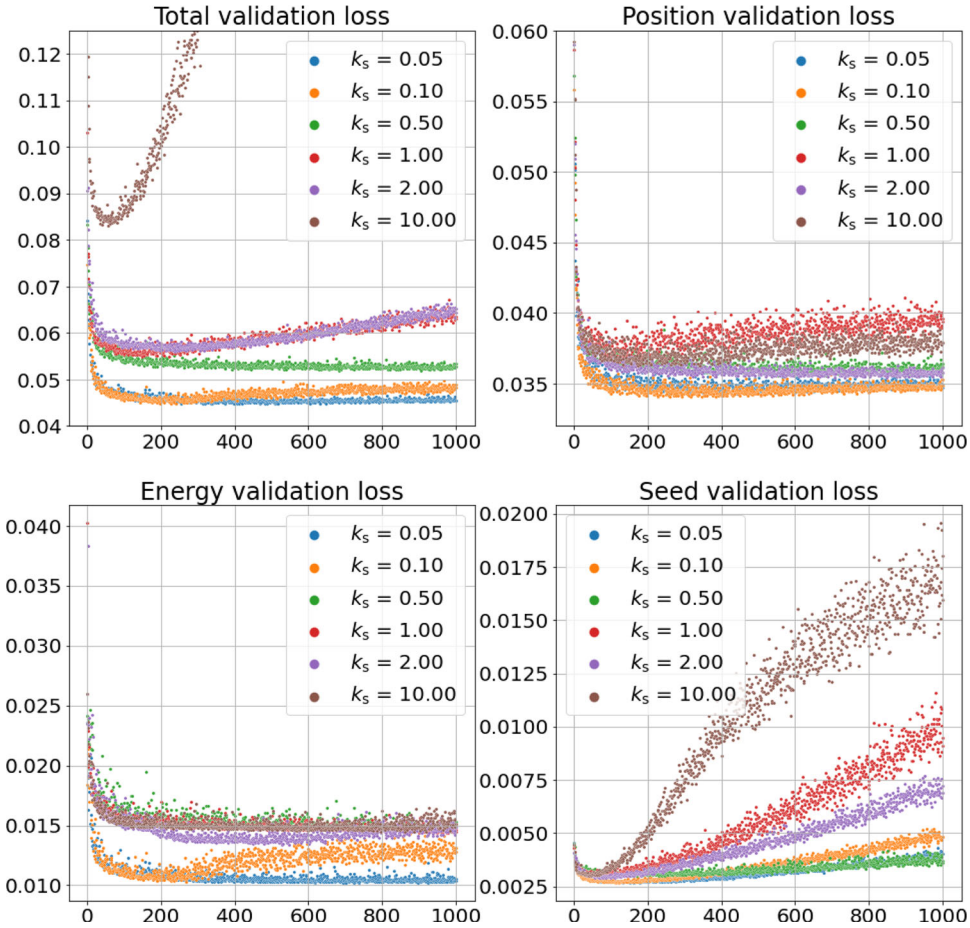

Fig. 15 The validation loss functions evolution versus epoch, presented for different seed weights (k_s). The position, energy, seed, and total validation losses are shown

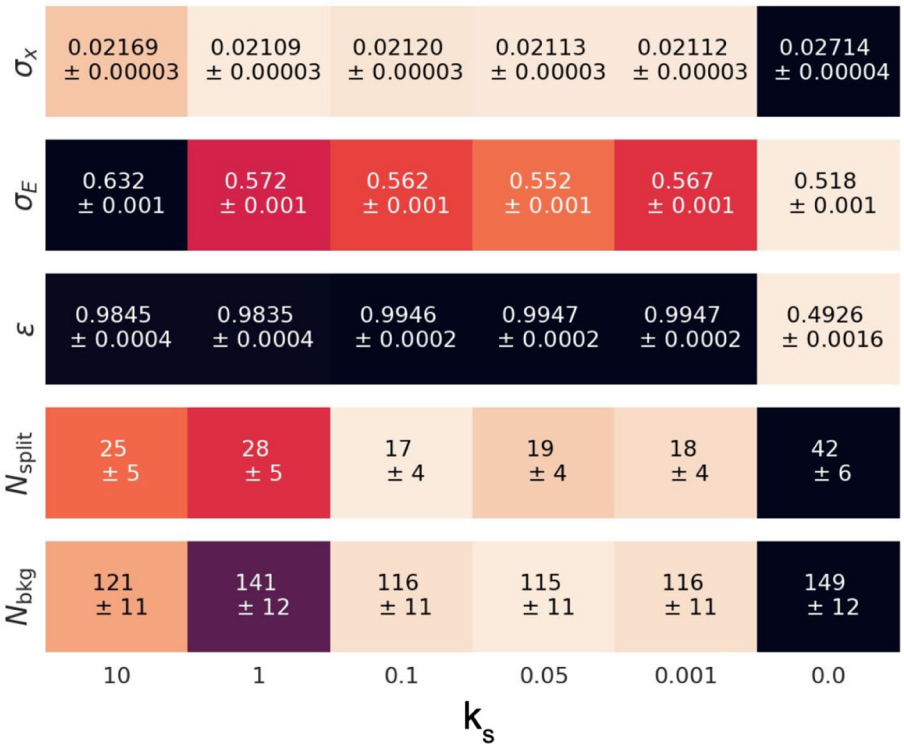

Fig. 16 Values corresponding to different physics performance metrics, as detailed in Sect. 6, are obtained from scanning various k_s penalty terms. $k_s = 0.05$ is found to be the value yielding the best performance

References

1. CMS Collaboration, The CMS experiment at the CERN LHC. J. Instrum. **3**(08), 08004 (2008). https://doi.org/10.1088/1748-0221/3/08/S08004
2. CMS Collaboration, Measurements of the Higgs boson production cross section and couplings in the W boson pair decay channel in proton-proton collisions at $\sqrt{s} = 13$ TeV. Eur. Phys. J. C **83**(7), 667 (2023). https://doi.org/10.1140/epjc/s10052-023-11632-6
3. CMS Collaboration, A measurement of the Higgs boson mass in the diphoton decay channel. Phys. Lett. B **805**, 135425 (2020). https://doi.org/10.1016/j.physletb.2020.135425
4. CMS Collaboration, Electron and photon reconstruction and identification with the CMS experiment at the CERN LHC. JINST **16**(05), 05014 (2021). https://doi.org/10.1088/1748-0221/3/08/S08004
5. CMS Collaboration, Particle-flow reconstruction and global event description with the CMS detector. J. Instrum. **12**(10), 10003 (2017). https://doi.org/10.1088/1748-0221/12/10/P10003
6. CMS Collaboration, Search for exotic Higgs boson decays $H \to \mathcal{A}\mathcal{A} \to 4\gamma$ with event containing two merged diphotons in proton-proton collisions at $\sqrt{s} = 13$ TeV. Phys. Rev. Lett. **131**(10), 101801 (2023). https://doi.org/10.1103/PhysRevLett.131.101801
7. D. Belayneh et al., Calorimetry with deep learning: particle simulation and reconstruction for collider physics. Eur. Phys. J. C **80**(7), 688 (2020). https://doi.org/10.1140/epjc/s10052-020-8251-9
8. N. Akchurin, C. Cowden, J. Damgov, A. Hussain, S. Kunori, On the use of neural networks for energy reconstruction in high-granularity calorimeters. JINST **16**(12), 12036 (2021). https://doi.org/10.1088/1748-0221/16/12/P12036
9. C.W. Fabjan, F. Gianotti, Calorimetry for particle physics. Rev. Mod. Phys. **75**, 1243–1286 (2003). https://doi.org/10.1103/RevModPhys.75.1243
10. CMS Collaboration, ECAL Clustering for run 3 (2022). https://cds.cern.ch/record/2812783
11. Y. Coadou, Boosted decision trees, in *Artificial Intelligence for High Energy Physics* (World Scientific, 2022), pp. 9–58. https://doi.org/10.1142/9789811234033_0002
12. S. Agostinelli et al., Geant4-a simulation toolkit (2003). https://doi.org/10.1016/S0168-9002(03)01368-8
13. The pandas development team: "pandas-dev/pandas: Pandas". https://doi.org/10.5281/zenodo.3509134
14. R.H. Charles et al., Array Programming with NumPy. https://doi.org/10.1038/s41586-020-2649-2
15. CMS Collaboration, The CMS electromagnetic calorimeter project: Technical Design Report. Technical report, CERN (1997). https://cds.cern.ch/record/349375?ln=en
16. CMS Collaboration, ECAL 2016 refined calibration and Run2 summary plots (2020). https://cds.cern.ch/record/2717925
17. J. Shlomi, P. Battaglia, J.-R. Vlimant, Graph neural networks in particle physics. Mach. Learn. Sci. Technol. **2**(2), 021001 (2021). https://doi.org/10.1088/2632-2153/abbf9a
18. S. Qasim, J. Kieseler, Y. Iiyama, M. Pierini, Learning representations of irregular particle-detector geometry with distance-weighted graph networks. Eur. Phys. J. C (2019). https://doi.org/10.1140/epjc/s10052-019-7113-9
19. S.R. Dubey, S.K. Singh, B.B. Chaudhuri, Activation Functions in Deep Learning: A Comprehensive Survey and Benchmark (2022). https://doi.org/10.48550/arXiv.2109.14545
20. D.P. Kingma, J. Ba, Adam: A Method for Stochastic Optimization (2017). https://doi.org/10.48550/arXiv.1412.6980
21. J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, M. Sun, Graph Neural Networks: A Review of Methods and Applications (2021). https://doi.org/10.48550/arXiv.1812.08434
22. L. Tsung-Yi, P. Goyal, R. Girshick et al., Focal Loss for Dense Object Detection (2018). https://doi.org/10.48550/arXiv.1708.02002
23. A. Kapoor, A. Gulli, S. Pal et al., Deep Learning with TensorFlow and Keras: Build and Deploy Supervised, Unsupervised, Deep, and Reinforcement Learning Models. Packt Publishing Ltd (2022)
24. You Y. Yang, J. Li, S. Reddi et al., Large Batch Optimization for Deep Learning: Training BERT in 76 minutes (2020). https://doi.org/10.48550/arXiv.1904.00962
25. T. Frisson, P. Mine, H4SIM, a Geant4 simulation program for the CMS ECAL supermodule (2005). http://geant4.in2p3.fr/2005/Workshop/UserSession/P.Mine.pdf