New Techniques for Reconstruction in the CMS High Granularity Calorimeter

By

Abhirikshma Nandi

Master's Thesis in Physics

Submitted to the

Faculty of Mathematics, Computer Science and Natural Sciences
of the RWTH Aachen University
in December 2022

Written under the supervision of

Prof. Dr. rer. nat. Alexander Schmidt
Physics Institute III A

Dr. rer. nat. Felice Pantaleo
Experimental Physics Department, CERN

Second Examiner
Prof. Dr. rer. nat. Johannes Erdmann
Physics Institute III A

New Techniques for Reconstruction in the CMS High Granularity Calorimeter

Abhirikshma Nandi
Aachen, Decemeber 2022

Writing period
Jun. 2022 – Dec. 2022

First Examiner
Prof. Dr. rer. nat. Alexander Schmidt

Second Examiner
Prof. Dr. rer. nat. Johannes Erdmann

Abstract

The calorimeter endcaps of the CMS detector are to be completely replaced by a High Granularity Calorimeter (HGCAL), as part of the Phase-2 upgrades. A novel reconstruction framework, called The Iterative Clustering (TICL), is under development in CMS Software (CMSSW) and was recently updated to its fourth version (v4) – centered around a new, density-based, parallelizable, algorithm for pattern recognition, CLUE3D. CLUE3D is a single blob shower finder, tuned for high pile-up rejection, meaning showers are often split into several reconstructed 3D objects, called 'tracksters', especially when there are secondary components like the ones initiated by Bremsstrahlung for electromagnetic objects or in hadronic showers. A new linking algorithm was developed and integrated as part of TICL v4 with the goal of accumulating together these tracksters coming from the same shower. It uses the propagation of tracks and tracksters to the same surface to find links geometrically, and then build objects intelligently with those links found. Excellent reconstruction efficiencies were obtained for electromagnetic objects using TICL v4. Separate components of hadronic showers were also observed to be successfully merged using the linking algorithm. Moreover, improved jet energy response and resolution were observed with the new version of TICL. A more advanced approach based on learning structures of showers on a sparse graph of nearby tracksters, using a Graph Neural Network, was also explored. The problem was framed as an edge classification task. Communities were identified in the similarity graph obtained as the model output using spectral clustering. Clustering results better than those obtained with the geometric algorithm have been observed for both events containing two close-by pions and ten randomly chosen particles shot in front of the HGCAL. This approach is complementary to the recent efforts using much lower-level information like hits, and framing the problem as a node classification task.

Contents

List of Figures

1 Introduction

After the end of Long Shutdown 2, the Large Hadron Collider [1] started its Run 3 in early July 2022 and has produced proton-proton collisions at an unprecedented energy of 13.6 TeV. At the end of this run, currently scheduled for the end of 2025, the accelerator and the experiments on it will undergo a major set of upgrades to prepare for the High Luminosity LHC (HL-LHC). The HL-LHC [2] is planned to produce collisions at five times the design luminosity of LHC, and record a dataset ten times as large as the initial LHC goal – greatly increasing its discovery potential. This however introduces huge challenges for the general-purpose experiments CMS [3] and ATLAS [4], in terms of having to deal with an extreme radiation dosage for the detectors and making sense out of a very high number of simultaneous collisions that are expected.

The upgrades planned for CMS [5] are motivated by the requirement to sustain or improve performance even in these very challenging situations. The upgrades encompass the inclusion of timing information, and improved granularity to cope with the high pile-up; usage of radiation hard electronics, and detectors built to survive the radiation dose; improved coverage in the forward region, and the inclusion of tracks in the Level-1 trigger for maintaining sensitivity to physics processes.

As part of the comprehensive set of upgrades, the calorimeter endcaps will be completely replaced with the High Granularity Calorimeter (HGCAL). HGCAL [6] is a novel

detector designed to be radiation hard and capable of handling the very busy environment expected with pile-up in the forward region. The reconstruction for HCGAL is being developed to be as resistant as possible to the challenges that the HL-LHC brings forth, by exploiting the rich information that the detector is able to provide. In this thesis, a new algorithm is developed for linking clustered energy deposits within the HGCAL and those with tracks – improving the global event description. A novel approach based on learning functions on graphs of energy clusters is also studied for the same task.

The thesis is structured as follows: Chapter 2 briefly introduces the CMS experiment and its various sub-detectors. It also contains a description of the upgrades planned for Phase-2, including a separate section on the CMS HGCAL. Chapter 3 contains a description of reconstruction in HGCAL, including the TICL framework [7]. The geometric linking algorithm developed is described in Chapter 4, along with its performance as part of the upgraded version of TICL. Chapter 5 discusses the novel Graph Neutral Network based approach explored for the linking problem. The final Chapter 6 contains the conclusions and outlook for future work.

2

CMS Experiment

The Compact Muon Solenoid (CMS) experiment [3] is one of the two general-purpose experiments on the Large Hadron Collider (LHC) [1] that co-discovered [8] the Higgs Boson in 2012. The LHC being the highest energy particle collider in the world makes it very important as well as challenging that the experiments are able to fully exploit its potential. The CMS experiment was designed with the primary objectives of finding the Standard Model (SM) Higgs Boson, searching for Physics beyond the SM, and precision studies of SM physics processes. The broad detector requirements [9] to meet these physics goals can be summarized as good muon identification performance and momentum resolution, good reconstruction of charged particles close to the interaction point – providing information on the substructure of jets (a spray of particles produced in a narrow cone from the hadronization of quarks or gluons), very good resolution for the measurement of electromagnetic energy deposits and good resolution for the measurement of missing energies from a collision.

The CMS detector is located in a cavern around a hundred meters underground, in the French village of Cessy, at "Point-5" on the LHC (Fig. 2.1). The solenoid in its name refers to the fact that at the heart of the detector lies a superconducting solenoidal magnet, capable of generating a field intensity of 4 T. This drives many of the other design and layout choices for the detector. For example, the tracker, electromagnetic, and most of

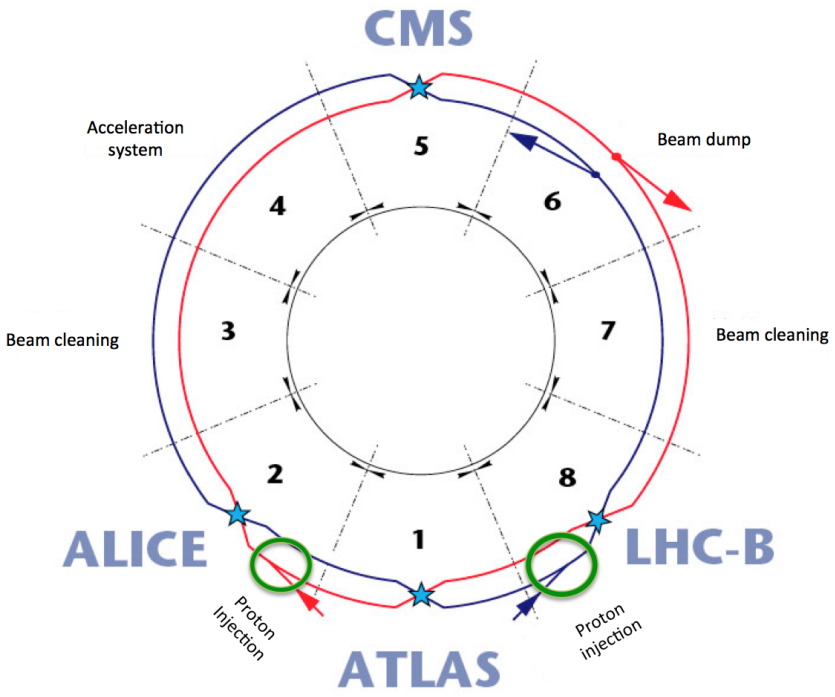


Figure 2.1: A schematic diagram of the LHC with the sectors numbered. The four interaction points around the ring, where the four main experiments are located, are marked with blue stars.

the hadron calorimeter are located inside the magnet. A strong magnetic field is required to ensure good resolution for the momentum measurement of charged particles. Fig. 2.2 shows a cutaway diagram of the CMS detector with its various sub-detectors exposed. It is useful to refer to it as the sub-detectors are briefly described in the next paragraphs.

Nearest to the interaction point lies the silicon tracking system. It consists of pixel detectors (four layers in the barrel and three disks at each end) in the innermost region, followed by ten layers of silicon microstrip detectors in the outer parts. The individual cells get progressively larger as one goes farther away from the beam pipe. This reflects the fact that particle flux decreases with distance and larger sensors can be used to still keep the occupancy under control. Here occupancy refers to the fraction of channels in a sub-detector that have a signal above the threshold. The long bending path through the microstrip detectors combined with the strong magnetic field allows for a measurement of the charged particle momenta with an excellent resolution, and the pixel detectors provide measurement points close to the beam pipe crucial for track seeding and reconstructing secondary vertices in decays of hadrons containing b and c quarks.

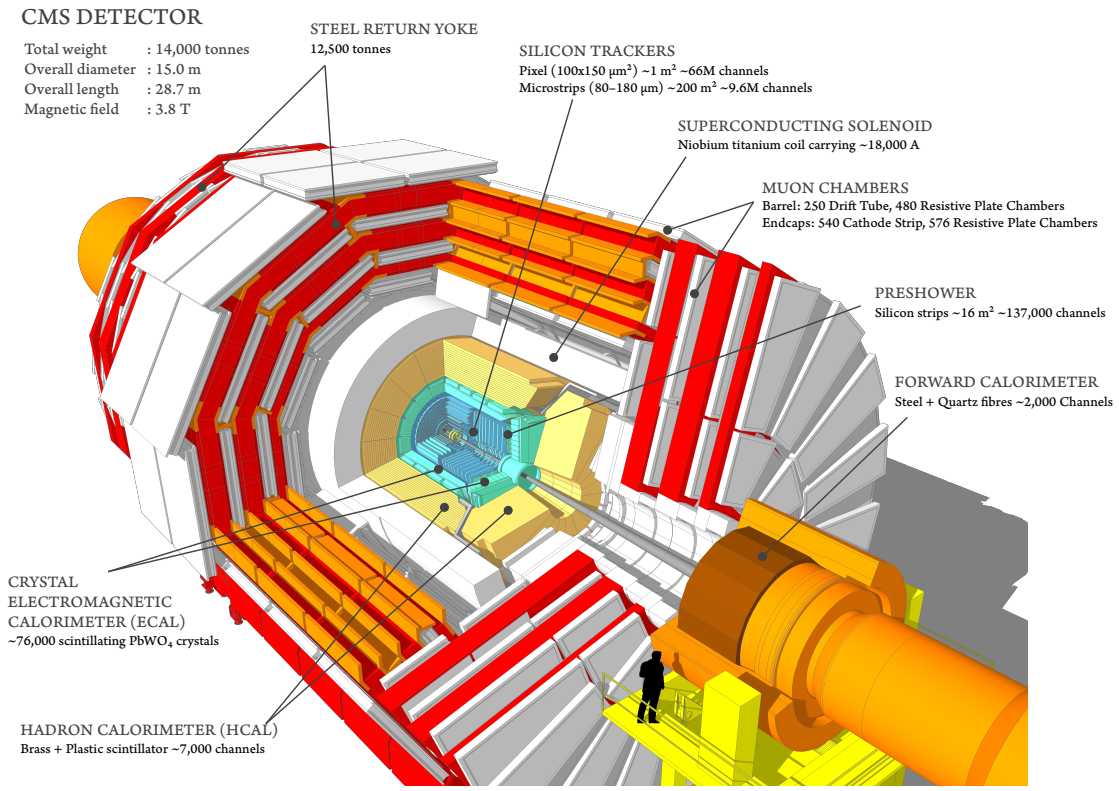


FIGURE 2.2: Cutaway diagram of the CMS detector, with the sub-detectors labeled, as was used in the LHC Run-2 [10].

The Electromagnetic Calorimeter (ECAL) lies just outside the tracker and is made entirely of $\approx$ 76,000 lead tungstate ($PbWO_4$) scintillating crystals. This type of calorimeter in which the whole volume is made of an active material serving both to degrade energies and generate signals is called a homogeneous calorimeter. $PbWO_4$ has a short radiation length (X_0 = 0.89 cm) and Moliere radius (2.2 cm), meaning electromagnetic showers are compact and are easier to contain. The crystals are also fast, allowing the ECAL to be included in the Level-1 (L1) trigger decision. The scintillation light is read out by silicon avalanche photodiodes (APD) in the barrel and vacuum phototriodes (VPT) in the end-caps, both being capable of operating in a magnetic field.

The Hadron Calorimeter (HCAL) surrounds the ECAL system and is located partly inside (Hadron Barrel and Hadron Endcap) and partly outside (Hadron Outer and Hadron Forward) the superconducting solenoid. Like most other hadron calorimeters, the CMS HCAL is also a sampling calorimeter, with active regions, generating signals proportional to the energy deposited by the particles, interleaved by passive absorbers, where most

of the energy is actually deposited. In the Hadron Barrel and Endcaps, scintillator tiles are used as the active material with brass as the absorber. The scintillators are read out by wavelength-shifting fibers, eventually carrying the light out to hybrid-photodiodes (HPD). As mentioned earlier, one of the goals for CMS is to provide good resolution for measurements of missing energy $E_{\mathrm{T}}^{\mathrm{miss}}$; this translates to good containment of showers and hermeticity and is majorly taken care of by the HCAL. The fairly short interaction length of brass and the maximization of the absorbing material inside the solenoid helps with the containment. Since the Hadron Barrel is physically constrained between the ECAL and the inner bore of the solenoid, an additional Hadron Outer, placed just outside the coil of the solenoid, is useful for minimizing leakage from late starting showers. The solenoid coil conveniently acts as an absorber for this, further increasing the material encountered by a particle before escaping the calorimeters. To be sensitive also in the very forward region, $3.0 < \eta < 5.0$, the Hadron Forward calorimeter is used. It uses quartz fibers embedded in steel absorbers, where the signal is the Cherenkov light generated upon the passage of a relativistic charged particle. The material choices are motivated by the requirement to have narrow and shallow shower profiles – necessitated by the high occupancy, very high radiation resistance, and fast signal generation.

As reflected in the name, the muon systems play a crucial part in the experiment. They are the outermost sub-detectors and are present completely outside the solenoid, in the return yoke of the superconducting magnet. The primary purpose of the 10000-ton steel return yoke is to safely confine most of the strong magnetic field so that the field outside in the experimental cavern stays at a manageable level. It also provides a region of uniform magnetic field outside the solenoid which can be exploited to measure the momenta of muons. Since muons, unlike most other particles, can pass through large amounts of matter almost unharmed, the huge mass of steel also acts as a filter for other particles produced directly or indirectly in a collision. Muon stations are placed in the gaps between consecutive layers of the return yoke, both in the barrel and endcap regions. A few different technologies of gaseous detectors are used, depending on the magnetic field, radiation levels, and muon incidence rates: Drift Tubes (DT) in the barrel, Cathode Strip Chambers (CSC), Resistive Plate Chambers (RPC), and a recent addition, Gas Electron Multipliers (GEM) in the endcaps. Gaseous detectors work on the general principle of capturing and amplifying charges produced by ionization when a charged particle passes through a gaseous volume placed in a high-voltage region. The momenta of muons can be measured both in the inner tracker as well as using the muon systems. At low momenta, the measurement from the tracker outperforms that from the muon chambers by an order of magnitude in terms of resolution. However, the muon systems become important at

high momenta, when its combination with the tracker improves the resolution. The fast signals from the muon systems are also used to determine the bunch crossing as well as in the L1 trigger decision. A dedicated muon trigger allows CMS to trigger on and record final states with high momenta muons, a very important feature towards fulfilling the physics objectives of the experiment.

The protons in the LHC are accelerated in bunches. Pairs of oppositely circulating bunches are then made to interact at the four interaction points (IPs) along the ring (Fig. 2.1), where the four major experiments reside. The design bunch crossing rate for the LHC is 40 MHz, which corresponds to a bunch separation of 25 ns in time. It is technically unfeasible to store the data generated from all the bunch crossings, because of the huge bandwidth and enormous storage required to do so. Moreover, only a tiny fraction of the bunch crossings actually give rise to hard collisions (with sufficiently high momentum transfer) where interesting physics processes could take place.

The CMS trigger and data acquisition system has thus been designed to discard uninteresting events, thereby greatly reducing the data rate, before they are written out onto permanent storage. The trigger system is designed in two levels, the Level-1 (L1) trigger and the High-Level Trigger (HLT). The L1 trigger provides the first level of filtering, reducing the 40 MHz bunch crossing rate to 100 kHz, and is implemented on fast custom hardware processors like Application Specific Integrated Circuits (ASICs) and Field Programmable Gate Arrays (FPGAs). The L1 trigger decision is taken based on the presence of certain "trigger primitive" objects like electrons, photons, muons, and jets above p_{T} thresholds. These trigger primitive objects are constructed using simplified information from the calorimeter and muon systems. The total time allocated for the L1 trigger signals from the detectors to reach the processors, a decision to be taken, and communicated back to the detector is 3.2 μs, called its latency. Accounting for the delay in signal propagation, less than 1 μs is available for the actual L1 trigger calculation. During this latency time, all the high-resolution data from the detectors are held in buffers, waiting to be either passed on to the HLT upon L1 Accept or rejected. The HLT runs on a computing farm consisting of commercial computing hardware and reduces the L1 output rate of 100 kHz to a few kHz for final storage. Software for the HLT is developed akin to the full offline reconstruction to gain the most from the sophisticated algorithms but streamlined with time-saving strategies in mind. The HLT algorithms follow a sequence of increasingly complex reconstruction and filtering, called a "trigger path". Events are discarded and the rest of the trigger path is skipped as soon as a filter fails. This helps in reducing the average event processing time by running the more time-consuming parts of the reconstruction for fewer events. Moreover, the reconstruction is restricted to narrow regions

around L1 or higher-level objects saving more CPU time.

Over the past decade or so the CMS experiment has collected around 200 fb^{-1} of proton-proton (pp) collision data and contributed to some very important scientific advances. The detector has also undergone several upgrades to maintain as well as improve its performance. Following the discovery of the Higgs boson, the scientific effort has shifted more in the direction of studying the properties of the Higgs boson as well as searching for new physics in the energy frontier. The LHC provides an unparalleled opportunity to explore these goals and this was strongly reaffirmed in the European Strategy for Particle Physics Update (ESPPU) of 2013 [11] and 2020 [12] as well as in the American Snowmass process of 2013-2015 [13]. To fully exploit the capabilities of the machine, it was decided that the LHC will be upgraded to run at five times its original design luminosity, as what is called the High Luminosity LHC (HL-LHC) [2] or "Phase-2" of its running. This dramatically increases the physics reach of the experiments by integrating an additional 3000 fb^{-1} by the end of the LHC program. A brief overview of the high-luminosity program (Fig. 2.3) and the upgrades planned for the CMS experiment in preparation for Phase-2 operations are given in the next section.

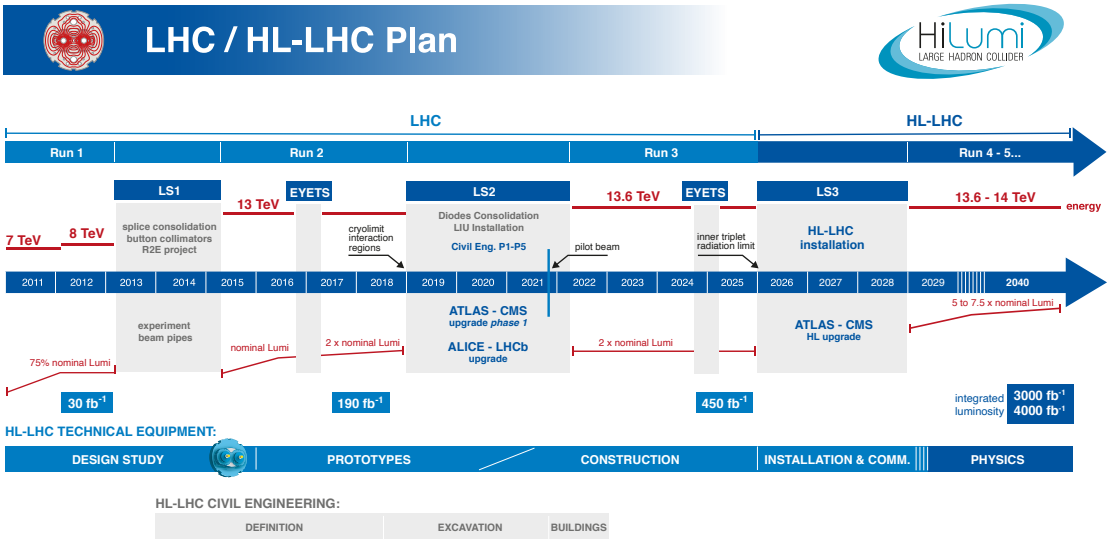


Figure 2.3: LHC/HL-LHC plan, as of February 2022, showing the collision energy (upper line) and instantaneous luminosity (lower line) through the periods of running and shutdowns for the next decade and beyond.

2.1 Phase-2 Upgrades

Instantaneous luminosity $\mathcal{L}$ is defined as the number of potential collisions per unit area per unit time and gives the event rate dR/dt when multiplied by the cross-section of the process σ_p, $\frac{dR}{dt} = \mathcal{L}\,\sigma_p$. The LHC was originally designed with an instantaneous luminosity goal of 1×10^{34} cm^{-2}s^{-1} and has reached a peak of twice that during operations in 2018. The HL-LHC is planned to run at a nominal value of 5×10^{34} cm^{-2}s^{-1}, with all pieces of equipment being designed with a margin that can allow pushing the machine luminosity to 7.5×10^{34} cm^{-2}s^{-1} – a so-called "ultimate" performance scenario. This introduces immense challenges in terms of very high radiation damage to the detectors and a major increase in pile-up, the number of simultaneous collisions during a bunch crossing, for the experiments. At the nominal HL-LHC luminosity, the average pile-up (PU) expected is 140 which increases to 200 for the ultimate scenario. Additionally, a substantial increase in computing requirements, both at the online and offline levels and for data transfer capabilities is expected. Several parts of the current CMS detector will reach the end of their lifetimes by the end of Run 3 and several other parts and subsystems are simply not capable to handle the conditions expected during Phase-2. To sustain or improve its performance, the CMS experiment is planned to go through an extensive set of upgrades [5] during the three years of Long Shutdown 3, after the end of Run 3 (Fig. 2.3).

The tracking detectors, which are placed closest to the interaction point, receive the highest dose of radiation. In silicon sensors radiation damage introduces defects in the crystal lattice, changing its bulk electrical properties. These changes manifest as increased leakage currents, and reduced charge collection efficiency – leading to reduced signals and an unsustainable increase in the full depletion voltage (required to make the full thickness of the sensor depleted, making it fully sensitive to charged particles). This forces operation with partial depletion, further reducing signals. The current tracker will receive enough dose by the end of Run 3 that it will need to be completely replaced for Phase-2. The tracker for Phase-2 [14] is designed to be more granular to cope with the higher number of tracks from the high pile-up. This is achieved by using strip detectors that are shorter in length and with smaller pixels. The Outer Tracker is also made significantly lighter, in terms of the material seen by a particle traversing through it, decreasing the rate of photon conversions as well as improving the p_{T} resolution. A significant improvement is the inclusion of tracking in the L1 trigger, where the Outer Tracker provides information at 40 MHz to L1. Additionally, the coverage of the tracker is extended to $|\eta| = 4$ to better match that of the calorimeter.

The current DT, CSC, and RPC detectors in the muon system can work throughout

Phase-2 and do not need to be replaced. The upgrades instead focus on improving performance in the higher rate environments expected in Phase-2 [15]. The measurement in the forward region is augmented by adding new detectors: GEMs for the first two and improved RPC (iRPC) detectors for the next two muon stations. These would provide high-resolution measurement points improving the standalone muon track reconstruction, and also provide an opportunity to trigger on muons produced outside the tracker from decays of long-lived particles. An additional GEM detector, placed immediately outside the new endcap calorimeter, is to be used to extend the coverage of the muon system to $|\eta| = 2.8$.

A new Minimum Ionizing Particle (MIP) Timing Detector (MTD) [16] will be placed in front of the barrel and endcap calorimeters. The MTD will be used to provide precise timing information for tracks. With a resolution of 30-40 ps, at the beginning of the HL-LHC, this helps in simplifying the problem of assigning tracks to primary vertices. The individual interactions in a bunch crossing are spread over a time of 180-200 ps due to the longitudinal extent of the beams. The time information of a track from the MTD narrows the time window over which the algorithms have to look for a compatible vertex, thereby reducing the combinatorics and confusion introduced by high pile-up.

Most of the current barrel ECAL and HCAL are to be retained for Phase-2, with upgrades planned only for the detector electronics [17]. However, the endcaps, where the radiation damage is much higher, are not suitable for use in their current condition during Phase-2. The calorimeter endcaps are thus to be completely replaced with a novel, radiation hard, High Granularity Calorimeter (HGCAL) [6]. Since the reconstruction for the HGCAL is the main subject of this thesis, its design is discussed in more detail in the next section.

The latency of the L1 trigger [18] is to be increased from 3.2 μs to 12.5 μs, allowing more time for tracks from the Outer Tracker to be reconstructed and matched to calorimeter deposits or muon tracks. Machine learning-based trigger algorithms, exploiting the additional available information, are also under development for event selection and anomaly detection already at the hardware level. High-speed optical links will be used to retrieve the detector information. The front-end electronics of all the existing subdetectors, that are to be retained, have to be upgraded to meet the new requirements of the L1 trigger. To maintain thresholds comparable to what is currently in place, the maximum L1 trigger rate will increase from 100 kHz to 500 kHz at PU 140 and 750 kHz at PU 200. All the detector components are being designed with read-out capabilities to handle the conditions at PU 200. The HLT [19] has to deal with the challenges of increased event rate and complexity and efficiently perform a reduction by a factor of

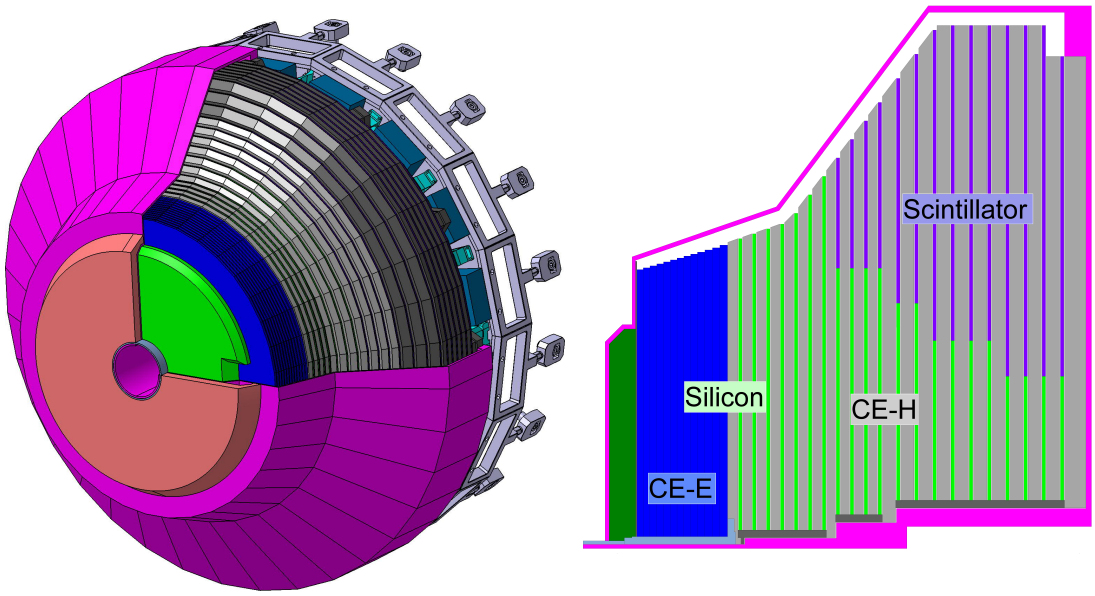
100, the same as in Phase-I. Algorithms are being developed or overhauled in view of the new constraints. Additionally, heterogeneous architectures with Graphics Processing Units (GPUs) accelerating code wherever possible, are being exploited to adhere to the strict time constraints. Evidently, the data acquisition system has to be upgraded as well to be able to handle the increased data rate.

2.2 High Granularity Calorimeter (HGCAL)

Calorimeter performance in the endcaps is important for achieving the physics goals of CMS during Phase-2. For example, the reactions initiated by Vector Boson Fusion (VBF) give rise to narrow jets, with large angular separation, usually ending up in the detector endcaps. Despite the busy environment caused by the high pile-up, the endcap calorimeters should be able to trigger on and reconstruct VBF jets, or merged jets like those from the hadronic decays of the W and Z bosons. It is also necessary to maintain a good electromagnetic energy resolution, a feature of the current ECAL.

The High Granularity Calorimeter is being developed as a replacement for the calorimeter endcaps which can maintain its performance even after an integrated luminosity of 3000 fb^{-1}. This requires a good inter-cell calibration with MIPs, necessitating the use of small sensors where the electronics noise after irradiation is still low. Additionally, the entire calorimeter will be operated at $-30\,^{\circ}\text{C}$ to keep the energy equivalent of the electronic noise low. The high transverse granularity caused by the small sensors is useful also in rejecting energy from pile-up and, with the high density of the calorimeter, in resolving close-by showers. The HGCAL is a sampling calorimeter and the high longitudinal granularity is required to maintain the electromagnetic energy resolution and aid in software compensation for hadronic showers. The high granularity opens possibilities for using pattern recognition and machine learning algorithms that benefit from the high data dimensionality, as well as from using the calorimeter to calculate the directional information of showers.

The HGCAL consists of an electromagnetic compartment (CE-E), followed by a hadronic compartment (CE-H) behind it. It was found that silicon as a sensitive material can work in the radiation levels expected at the endcap, so most of the HGCAL is instrumented with silicon sensors of size $\approx$0.5-1 cm^2. Three different silicon thicknesses 120, 200, and 300 μm are used respectively in the areas of decreasing fluence. Where the radiation levels are expected to be low, at large radii at the back of CE-H, several layers are instrumented using plastic scintillators ($\approx$4-30 cm^2 in size) with individual Silicon Photomultiplier (SiPM) readouts. The silicon sensors have another advantage of having fast response times and



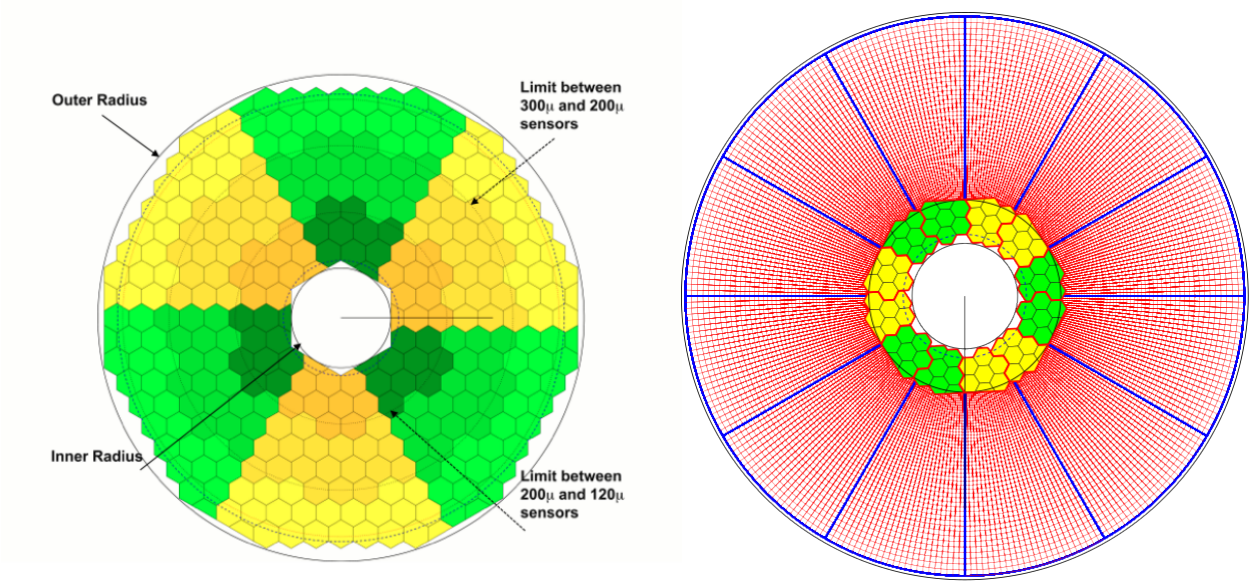
(a) (b)

Figure 2.4: (a) An isometric view of one endcap of the HGCAL. (b) The longitudinal structure of the upper half of an endcap shows the electromagnetic (CE-E) and hadronic (CE-H) compartments as well as the different sensor technologies used.

with the front-end electronics can give precise time stamps (with 25 ps resolution) of energy deposits greater than a certain threshold. As has been discussed before, this helps in pile-up rejection and in assigning energy deposits to the correct vertex.

In its present design, there are 26 layers in CE-E and 21 in CE-H. In the CE-E the hexagonal silicon sensors are sandwiched between a WCu baseplate and a printed circuit board carrying the front-end electronics, forming a silicon module. The silicon modules are tiled on either side of a copper cooling plate, and lead absorbers clad in stainless steel are placed on both sides of this module-cooling plate-module sandwich. This makes the total electromagnetic compartment around 27.7 X_0 or $\sim 1.5\lambda$ deep. In CE-H, the silicon sensors and (in the later layers) scintillators are mounted only on one side of a copper cooling plate. Stainless steel absorber plates enclose the sensor-cooling plate structure on both sides, making the total HGCAL around 10λ thick.

The hexagonal silicon sensors will be fabricated on 8-inch silicon wafers. The layout of the wafers in a layer of CE-E, with only silicon sensors, is shown in Fig. 2.5a. The scintillators, formed as small tiles, are arranged in a r-ϕ grid, with the tiles getting progressively larger in size with increasing radii. A layer of CE-H, with both silicon and scintillator sensors, is shown in Fig. 2.5b. For ease of engineering and assembly, the layers



(a) (b)

Figure 2.5: (a) Layout of the hexagonal silicon wafers in a layer with only silicon sensors. The 60° arcs with different colors represent the cassettes. The three thicknesses of silicon sensors used are also shown as different shades. (b) The layout of silicon wafers and scintillator tiles in a layer where both are present. The tiles are arranged in an r,ϕ grid, and get progressively larger with increasing r. The 30° arcs show the cassettes in CE-H [6].

in CE-E are subdivided into 60° wide wedges and those in CE-H into 30° wide wedges called "cassettes". The cassettes in CE-E additionally include the absorbers while those in CE-H do not.

Information from the HGCAL will be used as part of the L1 Trigger. Energy sums by combining groups of either 4 (2×2) or 9 (3×3) adjacent silicon cells and 2 scintillator tiles are calculated by the on-detector front-end electronic systems. These sums in the so-called "trigger cells" are transmitted to and given as inputs to the off-detector Trigger Primitive Generator (TPG) present in the service cavern. Energy deposits in all layers of CE-H but only the alternate layers of CE-E are used for this. The TPG outputs a list of 3D clusters, reconstructed from the trigger cells, and an energy map of the summed energy in η, ϕ bins which are then used further in the central L1 Trigger system to form the trigger decision.

3

Reconstruction in HGCAL

The high pile-up expected during Phase-2 makes reconstruction a very challenging task. The unprecedented granularity of the HGCAL helps reduce occupancy, helping resolve energy deposits in a busy environment, but introduces new challenges as well. Naive reconstruction algorithms exploring every possible combination and path are expected to fail due to timing or memory explosion. Highly granular calorimeters are ideal for Particle Flow-like reconstruction [20, 21], in which information from different parts of the detector is matched via sophisticated pattern recognition algorithms, akin to tracing the path of a particle through the detector. Moreover, the 5-Dimensional measurements (E, x, y, z, t) of energy deposits, that the HGCAL is capable to offer, make it a prime candidate for doing so.

The signals from the silicon cells or scintillator tiles are digitized and calibrated to produce the Reconstructed Hits (`HGCRecHit`). For data in real life, this would be done by the front-end ASIC called HGCROC, which reads out and digitizes the sensor charge at every bunch crossing. The time of arrival would also be measured and digitized. The triggered data is then sent to another digital ASIC, ECON-D, for zero suppression, before being transmitted off the detector over optical links. In the simulation, a similar treatment is done by electronic emulations and algorithms. Reconstructed hits form the input to the reconstruction. Those are first clustered in their respective layers, using a

density-based algorithm called CLUE, to form layer-clusters (`LayerCluster`). The layer-clusters from different layers can be connected in several ways, inside what is called The Iterative Clustering (TICL) framework, to finally output a list of particles along with their identification probabilities and kinematic properties. TICL also takes into account non-calorimeter objects like tracks and muons to perform a Particle Flow like treatment of calorimeter data.

The CLUE algorithm is described in Section 3.1 followed by a brief discussion on the modular TICL framework in Section 3.2. CLUE3D, the density-based pattern recognition algorithm made default in the latest update of the TICL framework, is described in Section 3.2.1.

3.1 CLUE

CLUstering of **E**nergy or CLUE [22] is used as the layer-clustering algorithm in HGCAL reconstruction. It is inspired by the Imaging Algorithm [23] and is designed to be completely parallelizable and optimized using data structures supporting fast queries. As a density-based algorithm, it is based on the idea that cluster centers are areas of high density and are relatively distant from other high-density areas.

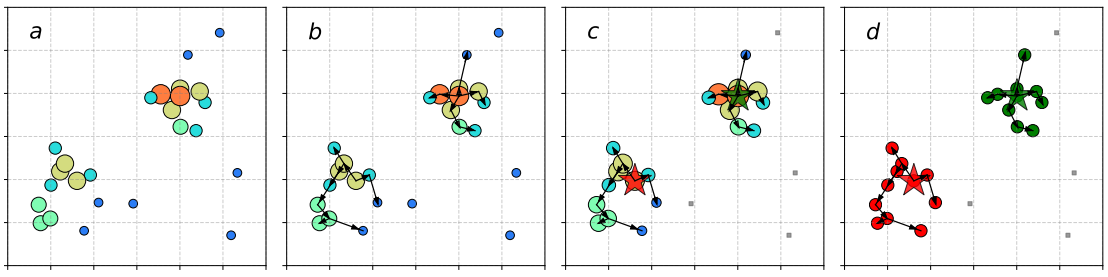


FIGURE 3.1: The steps of the CLUE algorithm: (a) calculation of local densities, the color, and size of points represent their local densities; (b) calculation of nearest highers, the arrows point from the nearest higher to the point in question; (c) labeling of seeds and outliers, marked with stars and grey squares respectively; (d) assignment of clusters, colors representing cluster indices [22].

The first step is the calculation of local densities for every detector cell (Fig. 3.1a). The local density ρ_i of a cell i is the weighted sum of energy in its neighboring cells N_{d_c}

(including itself) which are closer than a distance cut-off d_c.

$$\rho_i = \sum_{j:j \in N_{d_c}(i)} \chi(d_{ij}) w_j,$$

where w_j is the weight (energy) of cell j, d_{ij} is the distance between cells i and j, and $\chi(d_{ij})$ is a convolution kernel, which can be chosen according to the lateral size of showers and the specific application.

Next, for every cell, the distance δ to its nearest cell with a higher density is calculated (Fig. 3.1b). If a cell has local density greater than a cut-off ρ_c and δ higher than δ_c, it is flagged as a *seed*, forming the cores of clusters. The ones with local density less than ρ_c and δ greater than δ_o are marked as *outliers* (Fig. 3.1c). These are cells that are neither close to other clusters nor have enough density to form clusters on their own. Each outlier and all its descendant followers are disallowed to be included in any cluster, providing noise-rejection from low-density deposits. Every other cell becomes a *follower* of its nearest cell with higher density.

Each seed forms a separate cluster with all of its followers (Fig. 3.1d). Cluster indices are then iteratively passed down from the seeds through its chain of followers. For instance, if cell `X` is a seed and `a` is a follower of `X` and `b` is a follower of `a`, `X` forms one cluster with both `a` and `b`.

CLUE uses a fixed grid spatial index for neighborhood queries. The spatial index, much like a two-dimensional histogram, is constructed layer-wise by collecting the indices of the reconstructed hits into square bins according to their 2D coordinates. The neighborhood search around a point (x_i, y_i), within a distance threshold d_c, is then restricted within the bins touched by the window $[x_i \pm d_c, y_i \pm d_c]$. This prevents a sequential scan over all the points, making the process significantly more efficient.

If k clusters are found from n reconstructed hits, the calculation of local densities and distances to nearest cells with higher densities can be done with n-way parallelization, independently for every hit, while the expansion of clusters from seeds can be done with k-way parallelization, independently for every seed found.

The four parameters of the algorithm: the distance cut-off for local density calculation d_c, one density cut-off ρ_c, and two distance cut-offs δ_c and δ_o, respectively, for seed and outlier labeling, are chosen based on the physics requirements (shower sizes, separations) and the level of noise rejection desired.

Two layer-clusters built by CLUE for a 400 GeV photon shower are shown in Fig. 3.3a. CLUE has been extensively tested, and has been found to work very well in terms of clustering performance for the HGCAL [24], even in the presence of a high number of

simultaneous collisions. Its performance scales linearly, with respect to the number of hits to be clustered, preventing an explosion in timing or memory in a high occupancy environment. It has also been shown to effectively reject hits from electronic noise. The number of layer-clusters produced is typically an order of magnitude less than the number of hits, greatly reducing the problem size for the rest of the reconstruction.

3.2 The Iterative CLustering (TICL)

The **I**terative **CL**ustering [7] is a modular framework for reconstruction in HGCAL, being developed within CMS Software (CMSSW). Its modularity plays several important functions. By separating functionalities into modules, as shown in Fig. 3.2, troubleshooting and debugging becomes easier and the core infrastructure of the CMSSW framework is abstracted away from the developer. It also allows for quick switching of algorithms, making the testing and development cycle fast. Moreover, TICL is being developed with data structures and algorithms keeping parallelism in mind, and porting of code to heterogeneous architectures is simultaneously being done at a healthy pace.

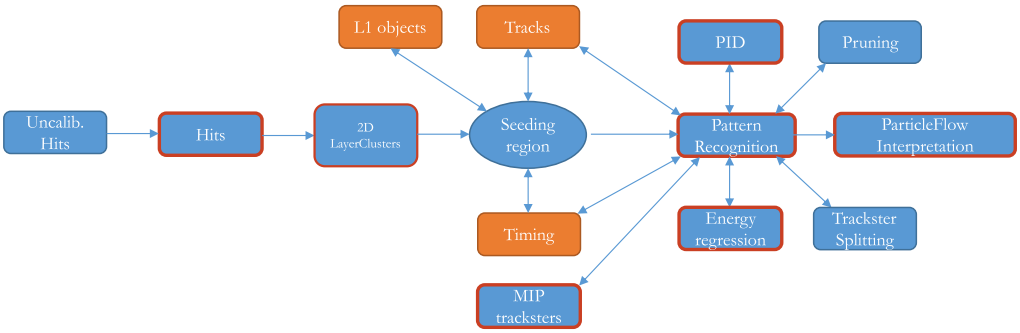


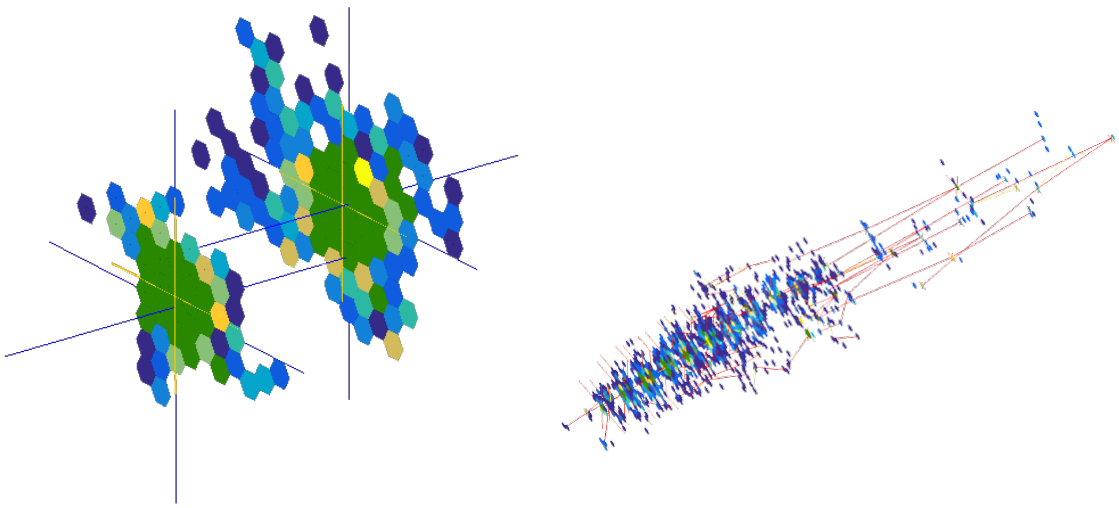
FIGURE 3.2: Flow diagram of the TICL framework. Modules in orange refer to information obtained from outside the calorimeter. Modules with a red border have dedicated validation and visualization functionalities in CMSSW [7].

TICL takes the 2D layer-clusters found by CLUE as input. The next step is the so-called "Pattern Recognition", which connects the layer-clusters across HGCAL layers to form 3D particle shower-like objects called "tracksters" (`Trackster`). There is a possibility to filter the layer-clusters going into the pattern recognition step to reconstruct special objects like secondary tracks (using layer-clusters with less than 3 hits) or restrict it around L1 objects or tracks (using seeding regions). The latter is useful for fast, regional reconstruction in the HLT. Currently, there are several choices available for the algorithm to be used for pattern recognition: CLUE3D, Cellular Automaton, and FastJet [25]. CLUE3D

is discussed in more detail in the following subsection. Cellular Automaton uses layer-clusters from neighboring layers and forms *doublets*. If these doublets are aligned and point back to the seeding region, they become neighbors of each other. Neighbors of doublets form a graph that is then visited by a depth-first search to form the tracksters. FastJet [25] features a number of cone-based jet clustering algorithms and can be used to both make both jets and tracksters.

Algorithms can be switched and configured using the *plug-in* system of CMSSW. Pattern recognition can also be done in iterations; a floating point mask for layer-clusters keeps track of their usage and allows an iterative workflow to be built. Additionally, the floating point mask lets layer-clusters be shared between more than one tracksters.

Formally, tracksters are directed acyclic graphs with layer-clusters or hits as nodes, whose edges depend on the pattern recognition algorithm used. Once tracksters are built using the algorithm of choice, their aggregated properties are computed. These properties include the barycenter, direction, and shape using energy weighted Principal Component Analysis (PCA) of the layer-cluster positions, transverse momentum using the PCA direction, time, particle identification (PID) probabilities, and regressed energies. A 400 GeV trackster reconstructed by CLUE3D, corresponding to a photon, is shown in Fig. 3.3b.



(a) (b)

FIGURE 3.3: (a) Two layers-clusters built by CLUE belonging to a 400 GeV photon shower. The hexagonal silicon sensors can be recognized. (b) A 400 GeV trackster produced by CLUE3D for a photon. The red lines denote edges formed by CLUE3D between the layer-clusters. Higher energies in both images are shown with warmer colors.

3.2.1 CLUE3D

CLUE3D [24] is a density-based pattern recognition algorithm developed with a logic very similar to CLUE. Contrary to how its name might be understood, it does not cluster reconstructed hits directly in three dimensions. Rather, it takes the layer-clusters as input and clusters them together across layers. It starts with first calculating the local density of a layer-cluster, by searching for other layer-clusters in its adjacent layers lying within a distance(projected) window of the original layer-cluster. The next step is the search for the nearest layer-cluster with higher density. The nearest higher is found by exploring a grid in binned η-ϕ space on the same set of adjacent layers. Seeds, outliers, and followers are identified in the same way as in CLUE. Finally, tracksters are created from every seed and all its followers.

Several performance improvements [26] were observed by using CLUE3D as the pattern recognition algorithm with respect to the Cellular Automaton algorithm, which was the default in the previous version of TICL. These include improvements in the separation of tracksters, the efficiency of energy reconstruction, and scaling of timing with pile-up. Moreover, the graphs of layer-clusters in CLUE3D tracksters depict the energy flow better (with directed edges connecting layer-clusters to their nearest higher, which can be followed back to their respective seeds) and can potentially be used in the future for trackster characterization and to subsequently trigger splitting, cleaning or linking. Owing to the improvements in performance, CLUE3D was made the default pattern recognition algorithm in TICL version 4 (v4). In the rest of this thesis, it should be assumed that tracksters are obtained using CLUE3D if not stated otherwise.

4

Linking in TICL

Particle showers often have secondary components, like the ones initiated by Bremsstrahlung for electrons, the electromagnetic and hadronic components of a hadronic shower, or from particles that start showering before entering the calorimeter. For example in HGCAL, the electromagnetic (CE-E) and hadronic (CE-H) compartments have different material compositions for the absorbers as well as different sensors. This results in different energy densities and the tracksters obtained from CLUE3D are often split at the boundary. Moreover, CLUE3D has been tuned for high pile-up rejection, producing pure but split tracksters.

This calls for a mechanism that can accumulate spatially separate clusters of energy deposits coming from the same shower or particle. There is also the requirement that energy deposits reconstructed in the calorimeter need to be matched to objects reconstructed elsewhere in the detector, e.g. tracks. As has been alluded to before, HGCAL is well-equipped for performing this kind of Particle Flow-like reconstruction [6, 20]. This approach is especially necessary to maintain/improve jet energy resolutions in the busy environment expected during Phase-2. Once objects reconstructed in the sub-detectors are matched, this works by taking the most precise measurement of a particle in the detector, for example, from the tracker for charged particles and from the (electromagnetic) calorimeter for (photons) neutral hadrons.

Due to the high occupancy, this linking step in TICL is not a trivial task. There are various ways for an algorithm to get confused or explode in its time or memory consumption. Confusion can be introduced in two ways. Firstly, if deposits from a neutral particle close to those from a charged particle are not resolved and are merged with the latter. The energy measurement will be from the momentum of the track, thus losing the energy of the neutral. Secondly, if energy deposits from a single charged particle are not matched to the track. These then form fake neutrals and contribute to double counting of the energy, which was already once accounted for from the track.

As part of the TICL v4 upgrade, a new geometric iterative linking algorithm was developed and deployed in CMSSW. The algorithm is discussed in Section 4.2, the Particle Flow interpretation of the objects built from the linking is described in Section 4.3, and the performance of the linking algorithm is reported in Section 4.4. In the next, Section 4.1, a brief overview is given of the *plug-in* system introduced for the deployment of multiple linking algorithms in TICL.

4.1 Plug-in System

The plug-in system allows several different algorithms to exist simultaneously and be conveniently *plugged-in* when required by the user or developer. The same structure is used for the several algorithms that exist for pattern recognition in TICL. `LinkingAlgoFactory.cc` is the so-called `PluginFactory` where different linking algorithms can be loaded as plug-ins. The one plug-in, corresponding to the only linking algorithm present now, is registered as a `ValidatedPlugin` which uses the `ParameterSet` validation, an extra layer of checks when passing parameters to an algorithm. This also makes the parameters easily modifiable from the Python configuration files. Once multiple algorithms are present as separate plug-ins, the one to be used can also be easily changed by specifying in the Python configuration.

4.2 Geometric Iterative Linking Algorithm

The problem at hand is to link tracks to tracksters or tracksters to other tracksters and exists in three dimensions. The general idea of the algorithm is to reduce the problem into two dimensions, by propagating the tracks and tracksters onto a common surface, and try to find neighbors geometrically. This is the *link-finding* step. Once the entire graph of *links* is known, it is then iteratively explored, aided by considerations of energy and time compatibility, to create the final objects.

The algorithm is motivated by the assumption that tracks and tracksters belonging to the same particle would also be geometrically close once projected onto the same surface. Although not perfect, this is a good enough place to start a general approach. Other ideas for the future might include 2D clustering of the projected objects or using secondary tracks reconstructed in the calorimeter to connect separate blobs of energy. The graph structure of layer-clusters in a trackster, corresponding to the energy flow in the shower, might also be exploited to aid linking – by inferring whether a trackster is incomplete, and needs to be linked to other neighboring tracksters, or not.

4.2.1 Propagation of objects

For propagation of tracks, the `PropagatorWithMaterial` is used. It takes into account the material effects,[1] like multiple scattering and energy loss, as well as the bending due to the magnetic field. However, for tracksters, a simple linear extrapolation of its barycenter (energy-weighted average of positions of all its layer-clusters) back to the origin $(0, 0, 0)$ is used instead. The first principal component obtained from the PCA of the layer-clusters in the trackster was also used as the trackster direction but produced significantly worse results. This might be due to the fact that the directions obtained from the PCA for small tracksters are not reliable, and that the rest have to undergo some sort of cleaning (of the layer-clusters considered for its calculation) to obtain a good enough estimate of its direction [27]. The two surfaces chosen for propagation are the layer 1 of HGCAL and the interface between CE-E and CE-H (`lastLayerEE`), a natural choice given the geometry of the problem. The tracksters propagated to a surface are stored in a fixed grid data-structure TICLLayerTile (`TICLLayerTile`), binned in η-ϕ space, similar to that used in CLUE and CLUE3D, for faster querying during link-finding.

4.2.2 Link-finding

For the link-finding step at a given surface, a *seeding* collection of either tracks or tracksters, propagated to that surface, is used to look for propagated tracksters in a configurable η-ϕ window around each seed. The TICLLayerTile, corresponding to the correct side of the calorimeter, can be conveniently queried for its contents in a window around a certain η,ϕ. The contents returned are further checked for their distance from the seed, for a more accurate selection according to the size of the window specified. Every object found inside the window is called a *link*. The indices of all those found are stored in a collection of

[1] At the time of writing the effects arising from the material inside the HGCAL are not accounted for in the propagator.

C++ type `std::vector<std::vector<unsigned>>`, where every element, corresponding to a seed, is a vector of the indices of objects found as *links* to that seed, if any.

It is useful to think of a *link* here as a possible connection between two objects purely based on their geometrical distance in η-ϕ space. The links found to a seed are sorted in ascending order of their distance from that seed, so that the ones closer get used up earlier in the link exploration step.

Four separate such collections are generated for the track-to-trackster or trackster-to-trackster links found at the two surfaces:

- Tracks to all tracksters, propagated at layer 1
- Tracks to all tracksters, propagated at `lastLayerEE`
- Electromagnetic tracksters to Hadronic tracksters, propagated at `lastLayerEE`
- Hadronic tracksters amongst themselves, propagated at `lastLayerEE`

Here Electromagnetic (EM) or Hadronic (HAD) simply refer to tracksters that have their barycenters in CE-E or CE-H respectively. For example, to build the collection second in the list, the collection of tracks propagated to the `lastLayerEE` is used as the seeding collection, and for each of those, the links found to all tracksters (propagated to the same surface) are stored.

Tracks are only considered in the linking algorithm if they are in the HGCAL acceptance, have transverse momenta p_{T} above 1 GeV, are `highPurity`, have less than five missing outer hits[2] and are not associated to muons. Additionally, only those with energy (using the pion mass hypothesis) above 2 GeV are propagated and allowed to be linked with tracksters. The ones that are below this threshold are made charged hadrons by default.

4.2.3 Exploration of links

The links found before are explored in two iterations to build the final TICLCandidates (`TICLCandidate`). TICLCandidates are lightweight physics objects that are one of the outputs of TICL and are used downstream in the reconstruction chain. A TICLCandidate has zero (created with just a track) or more included tracksters and can possibly include a track. In addition to kinematic information, it contains the charge, particle identification probabilities, and time of the candidate. The first iteration of link exploration is for candidates that are charged, i.e. have a track associated and the next is for neutrals.

[2] hits expected but not recorded in the tracker after the last (farthest from the interaction point) hit of a track

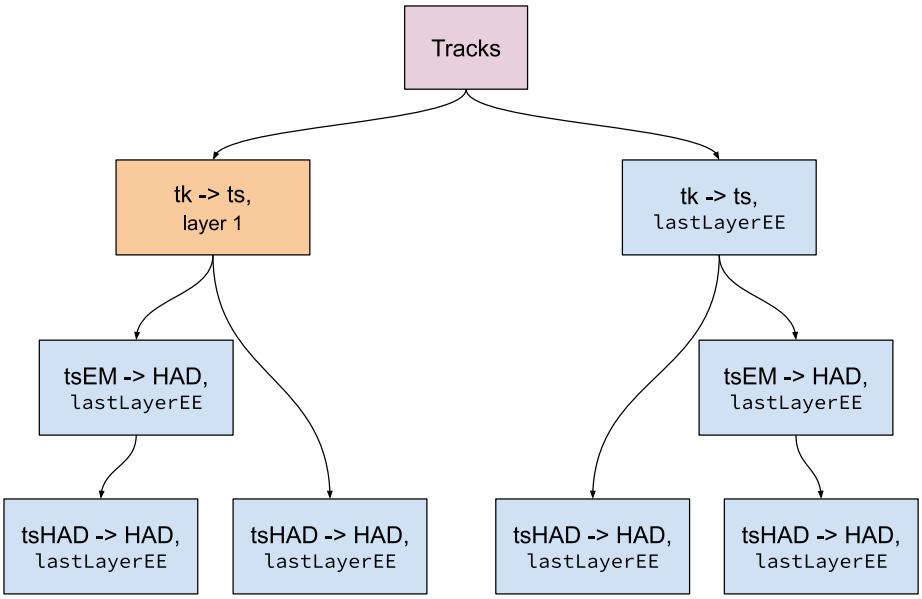


Figure 4.1: The exploration of *links* found to build charged candidates, beginning from the collection of tracks and exploring sequentially the deeper collections. Arrows correspond to queries in the collection pointed to by it, using an element in the collection from which it originates. The exploration is depth-first. For a given track, the branch on the left-hand side is executed before the one on the right. Tracksters found are added to a candidate if they are energy and time compatible (with the track) and have not already been used before.

Given a track/trackster, one can quickly find the list of objects (tracksters) found as links to it, at a surface, by asking in the appropriate collection.

This forms the basic operation for exploring the graph of links. For making the charged candidates, this exploration starts with the tracks. For every track, the track to trackster collection at layer 1 is asked for any links found to it. Every trackster found as a link here is used to query the EM to HAD trackster collection at `lastLayerEE`, and any trackster found here is used to ask in the final HAD to HAD trackster collection at `lastLayerEE`. It is a depth-first search, so any link found is immediately used to query further in the deeper collection before proceeding with other links at the current level. This forms the branch on the extreme left in Fig. 4.1. The tracksters found as links to the track at layer 1 could also be HAD (i.e. have barycenters in CE-H), and thus are used directly to check in the trackster HAD to HAD collection at `lastLayerEE` – forming the second branch from left in Fig. 4.1. This branch is executed after visiting all the deeper links of the trackster found as a link to the track, in the track to trackster collection at layer 1, in the previous branch.

This procedure is repeated, with the same track, starting the exploration at the track

to trackster collection at `lastLayerEE` and exploring further links to those in a similar fashion – shown by the two branches, third and fourth from the left in Fig. 4.1. The order of exploration is the same as before, i.e., in Fig. 4.1, the fourth branch from left is executed before the third. The rationale behind the repetition is the observation that this makes the track-to-trackster linking more robust, recovering some of those links that were missed in the steps before. If nothing is found linked to a track or all its links were used up in previous candidates, a charged candidate is made from just the track.

A trackster can be found as a link to more than one track/trackster if it is close to both. The same link can also be explored more than once due to the iterative logic. To prevent double counting, a hitherto unused trackster when found as a link is masked, preventing its reuse.

The collection of tracks in the event is sorted in the decreasing order of their p_{T}, and this order is maintained while making the charged candidates. Meaning, if a trackster is found with links to more than one track it is preferably associated with the track with a higher p_{T} (given that it passes the other compatibility checks, described below).

A trackster is only added to a charged candidate if it is compatible in energy and time with the track. If a trackster found as a link is incompatible, it is not added to the TICLCandidate, but the exploration continues with any further links to it in deeper collections. The energy compatibility checks if the incoming trackster makes the total energy of the candidate greater than the track momentum by a threshold; if so, it is flagged incompatible. The threshold currently in use is 10% of the energy of the incoming trackster or 10 GeV, whichever is smaller. Ideally, this threshold should be calculated by looking at the energy resolution of the pattern recognition algorithm. For time compatibility, the trackster and track times, if available for both, are required to be within 3σ of each other. This can help in rejecting spurious links to out-of-time pile-up. Out-of-time pile-up refers to the pile-up contributions from temporally adjacent bunch crossings.

The neutral candidate creation proceeds in exactly the same way except that the exploration starts with using the tracksters to query in the trackster EM to HAD collection at `lastLayerEE`. Only tracksters that were not used up in a charged candidate are allowed to be in a neutral. Tracksters once included in a neutral candidate also become unavailable for later use. The logic for exploring the rest of the links is exactly the same as was used in the charged candidate creation and is shown diagrammatically in Fig. 4.2.

The geometric linking algorithm returns the collection of TICLCandidates made to the `TrackstersMergeProducer`, from where it was called, and the candidates are finalized there. This process is discussed in Section 4.3. A separate collection of tracksters, labeled `ticlTrackstersMerged`, is produced by merging the tracksters included in the

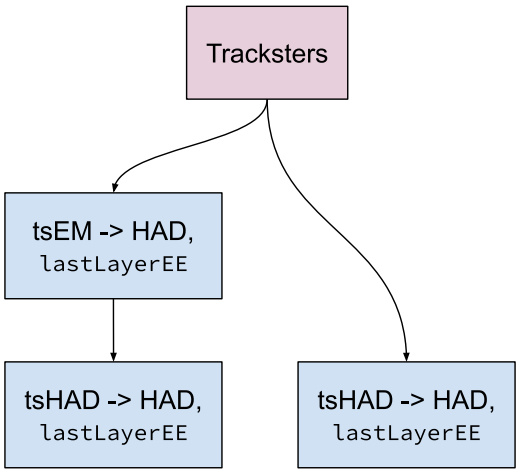


Figure 4.2: The exploration of links for making a neutral candidate starts from the collection of tracksters and sequentially visits the collections of tracksters linked to it. For a given trackster, the branch on the left-hand side is executed before the one on the right. The exploration is depth-first. Tracksters found are added to a candidate if they were not already used up in a charged or another neutral candidate.

same candidate. If a candidate contains only one trackster it is passed through to the merged collection as it is.

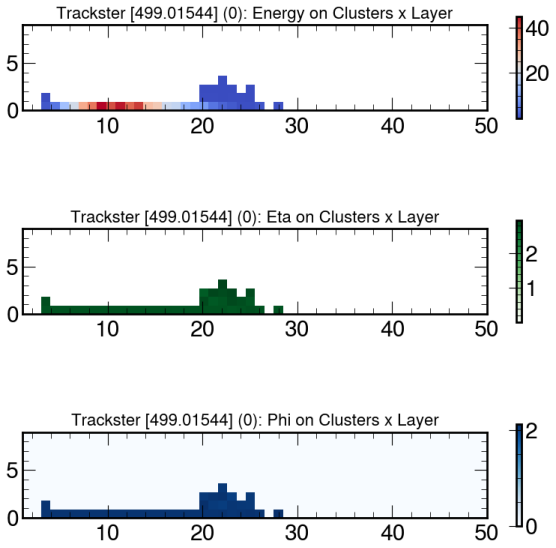


Figure 4.3: The trackster "image" for a 500 GeV photon, input to the CNN for particle identification and energy regression. The three "colors" energy, η and ϕ are shown separately. The horizontal axes represent the HGCAL layers and the vertical axes correspond to the layer-clusters in the layer [28].

The energy regression and particle identification are run on the merged tracksters using a Convolutional Neural Network (CNN) model [29], and the result is also used to set the energy and PDG id of the candidates. The CNN model takes an "image" (Fig. 4.3) of the tracksters as an input, where the two dimensions correspond to the HGCAL layers and up to ten layer-clusters per layer. The images have three "colors" corresponding to the energy, η and ϕ of the layer-clusters. The model (Fig. 4.4) simultaneously predicts both the particle type and regressed energy using two separate branches, each with two fully connected layers, on top of a shared stack of three convolution and two fully connected layers.

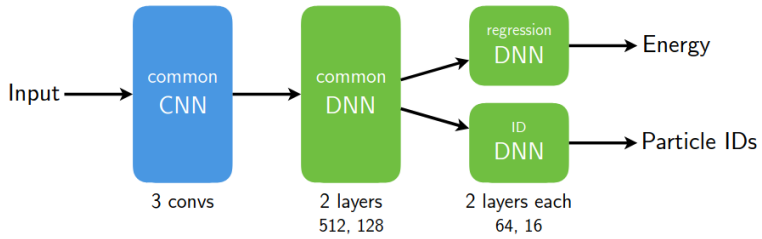


FIGURE 4.4: The architecture of the CNN model used for particle identification and energy regression [29].

4.2.4 Parameters

The algorithm takes a few parameters, the most important of which are the window sizes for the link-finding step. The four windows, corresponding to the four collections generated, are separately configurable. The window sizes used for the version of the linking algorithm integrated into CMSSW, as part of the TICL v4 upgrade, are given in the table below.

Link-Finding Window	Half-width
Track - trackster at layer 1	0.02
Track - trackster at `lastLayerEE`	0.03
Trackster EM - HAD at `lastLayerEE`	0.03
Trackster HAD - HAD at `lastLayerEE`	0.03

The window sizes were largely determined by choosing the minimum size that works satisfactorily for single particle samples. As a validation measure, it was checked that the sizes are not too big to unnecessarily merge separate showers in more complicated event topologies. A larger window is used at `lastLayerEE` due to the decreased occupancy there.

The other parameters of the algorithm are the thresholds for energy and time compatibility checks for charged candidates and the minimum quality for the track time, obtained from the MTD, to be considered legitimate. A more rigorous tuning of all the parameters has not been done yet and one might expect some improvements in performance with it.

4.3 Post-processing Candidates

After the TICLCandidates are built by the linking algorithm, their kinematic and other aggregated attributes can be calculated. This is done after the particle identification probabilities, regressed energies, and times for the merged tracksters are known. For a charged candidate, its 4-momentum is set using the track momentum and the regressed energy of the corresponding merged trackster. For neutrals, the combined barycenter of the accumulated tracksters is used as the direction, along with the regressed energy of the merged tracksters. A candidate is marked hadronic if the sum of the probabilities of it being an electron or a photon is smaller than 0.5. The probabilities are the ones obtained by running the particle identification on the corresponding merged trackster. If hadronic, charged candidates are labeled as pions ($\pi^{\pm}$, PDG id ± 211) and neutrals as kaons (K^0, 130). If not, candidates become electrons ($e^{\pm}$, ± 11) or photons (γ, 22) depending on if they are charged or not. The charge ($\pm$) is set from the charge of the track. As mentioned before, tracks without any linked trackster are promoted as pions.

4.4 Performance

The performance validation of the linking algorithm ideally needs to be done at two levels, how well the tracksters reconstructed from the same particle are linked together and how well the reconstructed track of a charged particle is linked to its energy deposits reconstructed in the calorimeter. The complete machinery for the latter task is not currently in place, so that could not yet be quantitatively verified. However, the HGCAL validation modules are well suited for the former task, and the performance plots obtained from those are discussed later in this section.

Fig. 4.5 shows the tracksters reconstructed in HGCAL for a single charged pion produced at the vertex, before and after the linking. The tracksters are visualized at the detector hit level, to get a better idea of the extent of the shower. As can be clearly seen, a majority of the tracksters, and most importantly the two high energy components, labeled Tracksters 0 and 1 in the image on the left, have been merged together after the linking. Some small tracksters have not been merged since they are not aligned with the

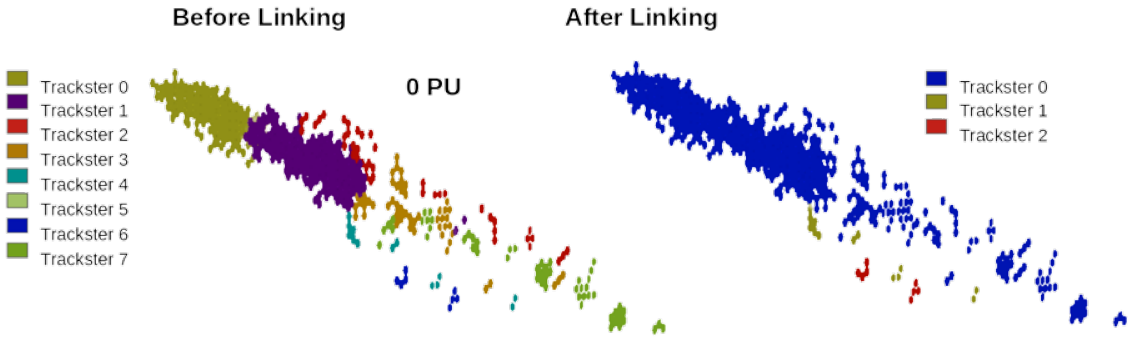


FIGURE 4.5: Tracksters reconstructed from a single charged pion produced at the vertex and visualized using the Fireworks utility for CMS event display. On the left, the reconstructed tracksters (in different colors) are shown. The merged tracksters obtained after the linking procedure are shown on the right.

other ones.

For validation in HGCAL, the association score between a CaloParticle and a reconstructed trackster is relevant. CaloParticle refers to the parent of a particle in the simulated decay graph nearest to the vertex. The association score between a CaloParticle i and a reconstructed trackster j is defined as

$$\text{score}_{i,j} = \frac{\sum_{\text{DetId}\, k \in \text{CP}_i} \min\left(\left(fr_k^{reco} - fr_k^{MC}\right)^2, \left(fr_k^{MC}\right)^2\right) \times \epsilon_k^2}{\sum_{\text{DetId}\, k \in \text{CP}_i} \left(fr_k^{MC}\right)^2 \times \epsilon_k^2},$$

where DetId refers to a detector cell and fr_k^{reco} and fr_k^{MC} are the fractions of the cell k that has been assigned to the reconstructed trackster j and to the CaloParticle i respectively. ϵ_k is the energy deposited in cell k. A score equal to 0 thus means that the fractions belonging to the reconstructed and simulated objects match exactly, implying that the two objects are completely matched as well. On the contrary, a score equal to 1 indicates a complete mismatch. There is also a similarly defined reco-to-sim score, where the summation (and the normalization in the denominator) is over the reconstructed object. Pairs of objects with a score smaller than 0.2 are considered *associated*.

The *efficiency* metric is defined as the number of CaloParticles associated with reconstructed tracksters divided by the total number of generated CaloParticles. Fig. 4.6 shows the efficiency of merged tracksters, produced from the linking, for single photons of energy between 10 GeV and 600 GeV. The photons are produced at the vertex, and efficiencies are calculated for both PU 0 and PU 200 scenarios. The four distributions are with respect to the energy, p_T, ϕ, and η of the generated particle. The first three distributions are plotted

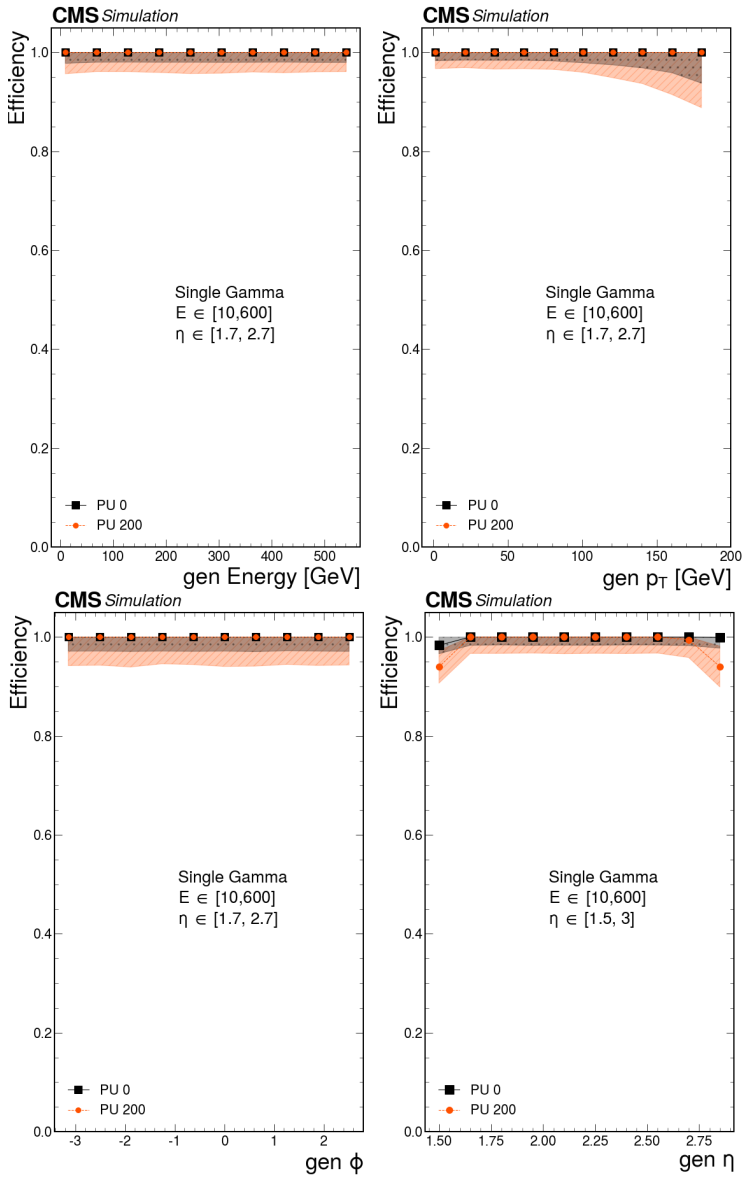


Figure 4.6: Efficiencies of merged tracksters obtained from the linking procedure for single photons generated from the vertex, in PU 0 and PU 200, with energies between 10 and 600 GeV, and distributed uniformly in the HGCAL acceptance. Efficiency distributions are shown with respect to the energy, p_T, ϕ, and η of the generated particle.

for the η range between 1.7 and 2.7 while the last one, with respect to η, is between 1.5 and 3.0. This is to disentangle the effects at the HGCAL boundary. The efficiency stays at 1 for the entire range of energy, p_T and ϕ, at both 0 and 200 PU, indicating excellent reconstruction performance for electromagnetic objects. The efficiency falls off at low η,

presumably due to shower leakage, and at high η, for PU 200, due to the high occupancy there.

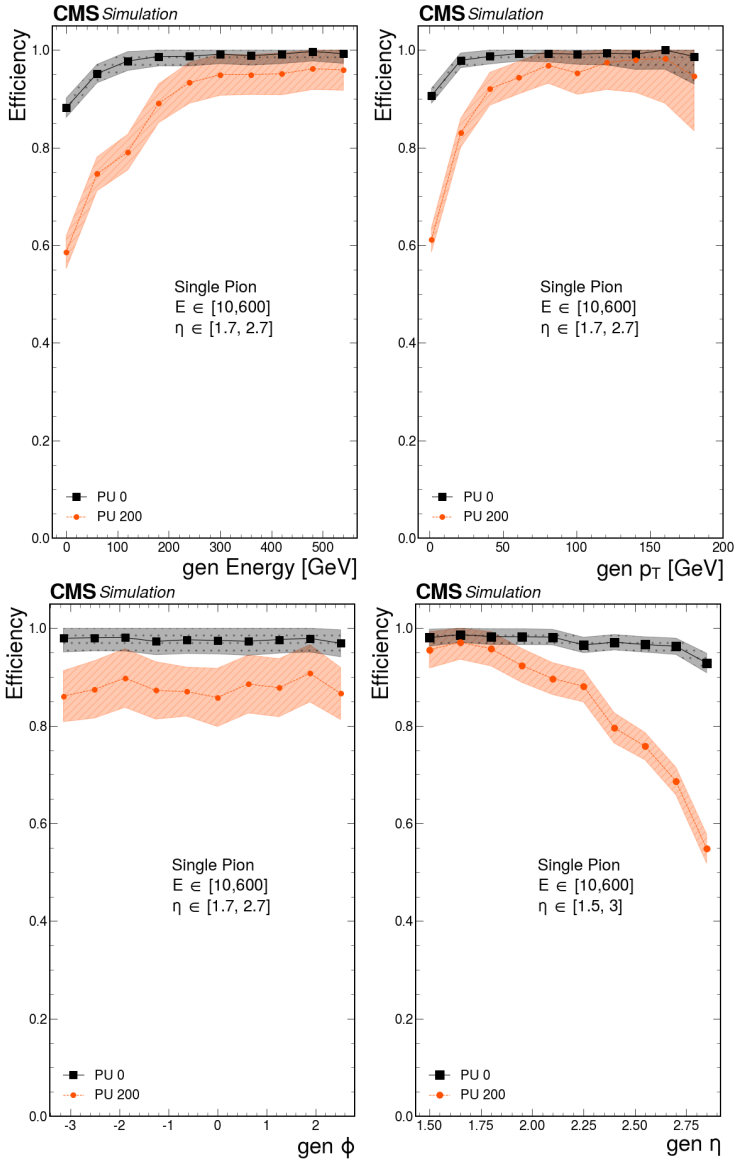


FIGURE 4.7: Efficiencies of merged tracksters obtained from the linking procedure for single charged pions generated from the vertex, in PU 0 and PU 200, with energies between 10 and 600 GeV, and distributed uniformly in the HGCAL acceptance. Efficiency distributions are shown with respect to the energy, p_T, ϕ, and η of the generated particle.

Fig. 4.7, on the other hand, shows the efficiency of merged tracksters for single charged pions produced from the vertex and in the same energy range as the photons. Similar η

ranges as in Fig. 4.6 are used to plot the different distributions. The efficiencies at 0 PU stay very close to 1 except at low energies or p_{T}-s. At low energies, the tracksters are less aligned and therefore the angular windows of the linking algorithm fail to accumulate those, resulting in degraded performance. The efficiencies drop off at higher η. The efficiencies are lesser in case of the 200 PU scenario, and the drops at low energy and high η are more pronounced, due to the more challenging conditions. At high η there is presumably again the effects arising from the detector boundary and the high occupancy contributing to the loss in efficiencies. To summarize, the algorithm works quite well in linking the separate tracksters reconstructed for charged pions without pile-up, but high pile-up negatively affects its performance, especially in the high η region.

There is possibly another effect contributing to the dropping efficiencies at higher η-s. The physical size of the η-ϕ windows does not remain the same everywhere in the detector and indeed drops off with increasing η. This might be constraining severely the tracksters that can be found as geometrical *links* at higher η, eventually affecting the efficiencies. A potential workaround to this could be using the projective coordinates that CLUE3D uses – $(\frac{r}{z}z_r, \frac{r}{z}z_r\phi)$, where the r/z from any layer can be projected onto a specific layer z_r allowing the use of absolute values on the projected layer for deltas.

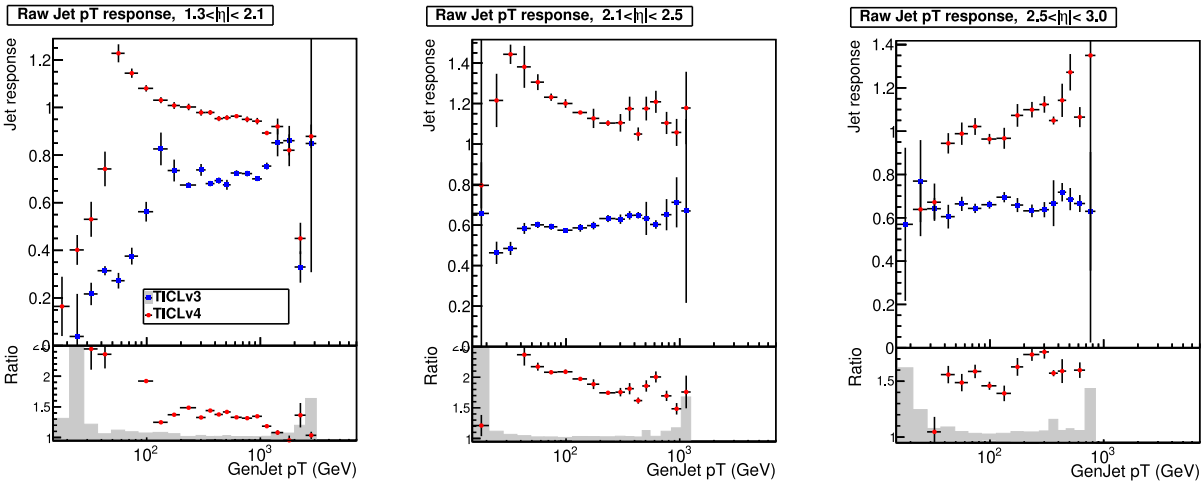


Figure 4.8: Jet p_{T} response with PUPPI enabled for TICL v4 (red circles) and v3 (blue squares) using a flat QCD sample with 200 pile-up.

To understand how the changes introduced in TICL translate to performance at a higher level, Particle Flow validation is used. Among other things, it produces the plots of jet response and jet p_{T} resolution with respect to the p_{T} of particle-level jets obtained from simulation. Jets in this context refer to Particle Flow jets obtained by clustering

the reconstructed Particle Flow candidates, which in the case of HGCAL are very similar to the TICLCandidates. The jet response is the ratio of the reconstructed jet p_T to that of the particle-level jet. Fig. 4.8 compares the jet response obtained from the Particle Flow Validation, using the latest version of TICL (v4, with a new linking algorithm and CLUE3D as the pattern recognition algorithm) and the older v3. The plots are obtained with pile-up per particle identification[3] (PUPPI) enabled and without any jet energy corrections[4] (JEC) applied. A flat QCD sample with 200 pile-up is used. The η ranges cover the region relevant for HGCAL. It can be observed that the jet responses from TICL v4 are better (higher) than those from v3.

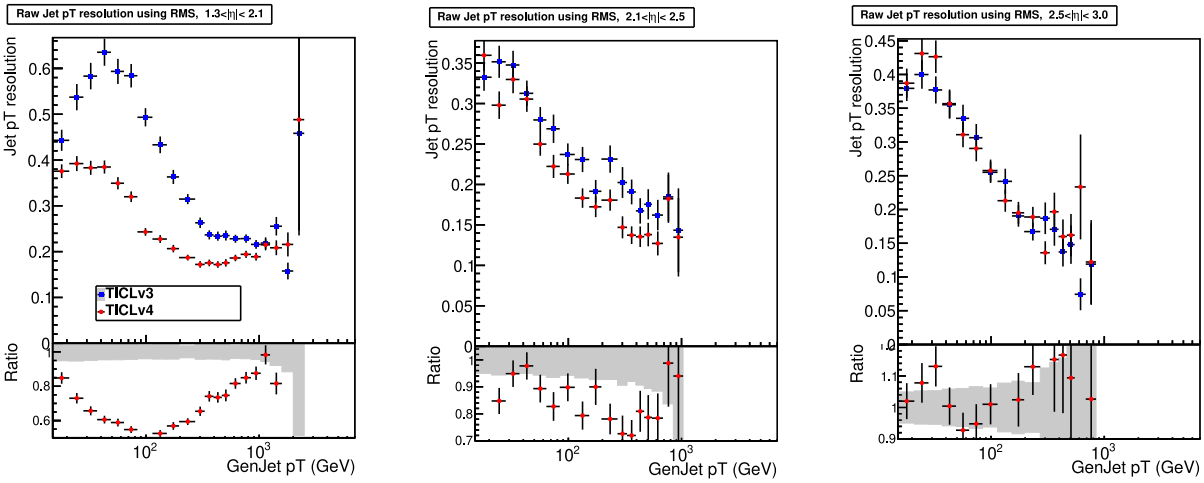


FIGURE 4.9: Jet p_T resolution from RMS with PUPPI enabled for TICL v4 (red circles) and v3 (blue squares) using a flat QCD sample with 200 pile-up.

Similarly in Fig. 4.9, the jet p_T resolutions – obtained from the root mean square (RMS) of deviations of the jet responses from the mean – using TICL v3 and v4, are compared using the same sample. Once again, the plots are obtained with PUPPI and without any JEC. It can be observed that the resolutions with TICL v4 are better (lower) than with v3. The difference is more pronounced in the lower η range. At higher η, there are presumably the same effects of high occupancy, smaller windows, and the effects arising from the HGCAL boundary influencing the performance. But it is clear that with respect to the last version of TICL, the introduction of CLUE3D as the pattern recognition algorithm

[3]PUPPI [30] is a pile-up mitigation technique at the per particle level. In addition to excluding charged particles associated with vertices from PU, it calculates the probability for each neutral particle to have originated from PU (based on the information of its surrounding particles) and scales their energy based on that.

[4]JEC is derived from simulations to bring the measured response of jets to that of generated particle-level jets on average.

along with the new linking algorithm leads to gains in reconstruction performance also at the level of higher-level objects.

The computational performance of the linking algorithm was also investigated. The time measured is the one required to process one event when running the reconstruction on a single core of a CPU – in this case, an AMD EPYC 75F3 32-Core processor. For a TTbar ($t\bar{t}$) sample, with 200 pile-up, the `TrackstersMergeProducer` takes a little less than 400 ms, out of a total of 1120 ms for the entire HGCAL reconstruction. This amounts to around 35% of the HGCAL reconstruction and 1% of the entire CMS offline reconstruction. The `TrackstersMergeProducer` includes the linking algorithm along with the subsequent merging of tracksters included in the same TICLCandidate, calculation of the aggregated properties of the merged tracksters – like PCA and regressed energy, and the post-processing step described in Section 4.3. The time required for the linking algorithm is thus well under control even in the scenario with 200 simultaneous collisions.

5

Graph Neural Network for Trackster Linking

Data from particle detectors are usually sparse, meaning detector channels are predominantly zero, and can be naturally represented as a set – an unordered collection of elements. For example, hits from the tracking detector or energy deposits in the calorimeter. This avoids having to represent the data in a grid-like data structure or imposing an order onto it, both of which might require some loss in the information or simply not possible due to the arbitrary geometry of the detectors. A graph, broadly speaking, consists of nodes and edges encoding relationships between the nodes. A set can be easily transformed into a graph using some relations between the elements, usually from the knowledge of the detector. Graphs are thus an attractive way of representing data from particle detectors.

Graph Neural Networks (GNNs) [31–33] are a class of deep learning architectures that are specialized for learning functions on graphs. The functions can be of different types predicting node-wise, edge-wise, or global targets. This is in contrast to the other popular deep learning architectures like the fully connected network (FC) [34], convolutional neural network (CNN) [35, 36], or recurrent neural network (RNN) [37] that work on data represented as vectors, grids, and sequences respectively. Due to their representation power and the ability to learn on graphs, GNNs are starting to be explored as the deep learning architecture of choice for various problems in particle physics, including event reconstruction.

In this chapter, such an attempt to solve the trackster linking problem by learning structures on a graph of tracksters is described. The linking task is cast as an edge classification problem, predicting edges connecting tracksters to be linked, and graph clustering to subsequently find partitions of the graph. The chapter is structured as follows: Section 5.1 contains a short overview of the general GNN architecture and some of its applications in particle physics reconstruction, the construction of the input graph for the GNN is described in Section 5.2, the GNN model and training in Section 5.3, followed by the graph clustering applied on the output of the GNN in Section 5.4, and finally the results obtained from this approach in Section 5.5.

5.1 Overview of Graph Networks

The graph network (GN) formalism from Ref. [32] is used here to describe the fundamental operations of a GNN. GNs are functions that operate on graphs and preserve the node and edge structure of the graph in its output. They are therefore graph-to-graph functions. A graph can be represented by $G = (\mathbf{u}, V, E)$, where $\mathbf{u}$ represents the global features, V the set of N_v vertices/nodes and E the set of N_e edges. Every node in V is characterized by its node attributes and every edge in E by its edge attributes and the indices of the nodes that the edge is incident on.

The GN works by message passing over the graph. The edge connectivity of the graph dictates how the messages are passed between the nodes. Formally, the GN consists of the *update* and *aggregation* functions. The update functions take a fixed-size input to generate a fixed-size output. The aggregation functions take a variable size input to produce an output of a fixed size. The general function of a GN block can be understood with the steps listed below and shown in Fig. 5.1:

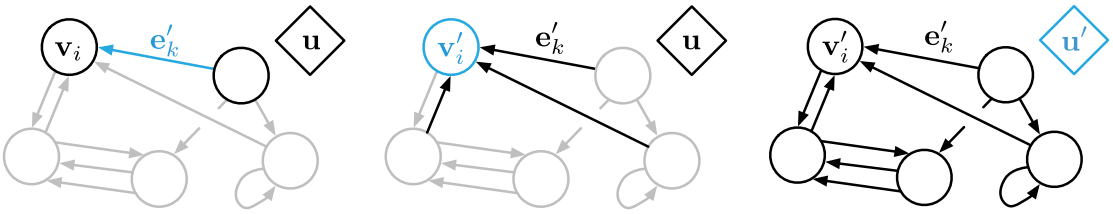
1. The computation starts with the edge update function calculating the updated edge attributes for every edge. For each edge, the update function can take the edge attribute, the attributes of the connected nodes, and the global feature as inputs. This is represented in Fig. 5.1a.

2. The next step is the aggregation of the calculated edge updates per node. For every node, the aggregation function considers the edges that are incident on that node.

3. For every node, the aggregated edge attributes from the last step and the current node attribute are used by the node update function to compute updates. This and the previous step are combinedly shown in Fig. 5.1b.

4. The set of all updated edge attributes and the set of all updated node attributes are aggregated separately and, along with the current global feature, are used to compute the next global update. This updates the global feature $\mathbf{u}$ of the graph. This is shown in Fig. 5.1c.

The update functions are usually implemented as trainable neural networks, hence the word "neural" in GNNs. The aggregation functions are permutation invariant, such as an element-wise sum, mean, or maximum, and ensure that the GN block itself remains invariant to the permutation of the nodes. Note that there are six functions in a GN block, one each for the update and aggregation of nodes, edges, and global features. The exact kind of function depends on the particular GNN architecture used. Problems can be cast as classification or regression using node, edge, or global features.



(a) Edge update (b) Node update (c) Global update

Figure 5.1: Updates in a GN block. The element being updated is represented in blue and the elements that contribute to that update are shown in black. The previous state of the element being updated is also used in its update [32].

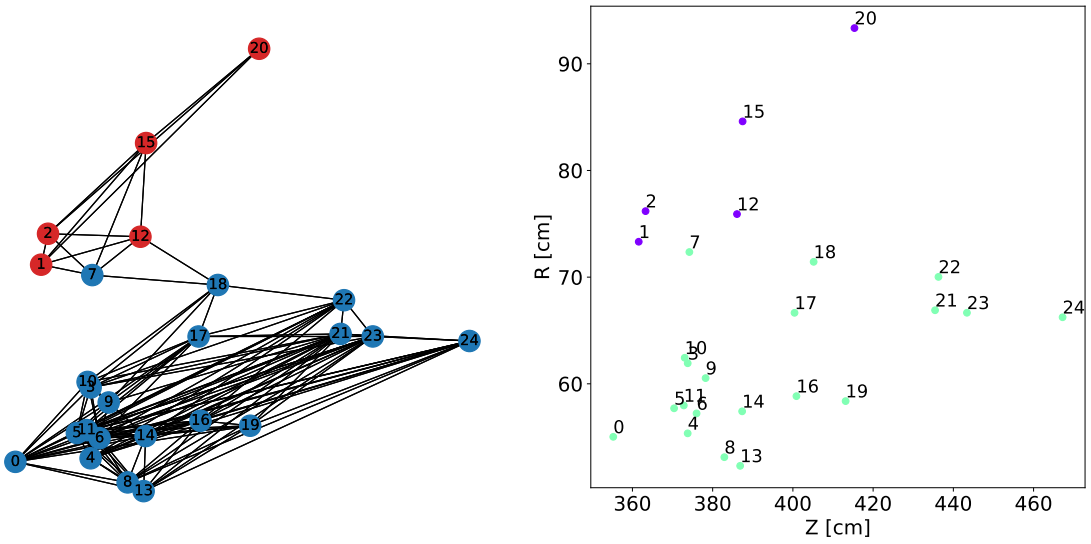
A specific GNN architecture worth discussing separately is the Edge Convolution or *EdgeConv* model [38]; this is also the architecture used here for the trackster linking problem. The specialty of this model is that the *edge features* are calculated using a learnable function operating on the node features of its endpoints. If the n nodes are represented by the set $X = \{\mathbf{x_1}, ..., \mathbf{x_n}\}$, where $\mathbf{x_i}$ is the feature vector of node i, and an edge e_{ij} connects nodes i and j, the edge feature $\mathbf{e_{ij}}$ is calculated as $\mathbf{e_{ij}} = h_\Theta(\mathbf{x_i}, \mathbf{x_j})$. Here h_Θ is a nonlinear function with a set of learnable parameters Θ. The updated node features $\mathbf{x'_i}$ are then calculated by aggregating the edge features $\mathbf{e_{ij}}$ for every node j that has an edge with node i. The symmetric aggregation operation is usually a sum or max. There can be a few different choices of h, and those are described as follows. $h_\Theta(\mathbf{x_i}, \mathbf{x_j}) = h_\Theta(\mathbf{x_i})$, here only the global information of the graph is encoded and the network is completely oblivious to the local connectivity of the graph; on the other hand a choice like $h_\Theta(\mathbf{x_i}, \mathbf{x_j}) = h_\Theta(\mathbf{x_j} - \mathbf{x_i})$ only encodes the local information and loses global structure; whereas $h_\Theta(\mathbf{x_i}, \mathbf{x_j}) = h_\Theta(\mathbf{x_i}, \mathbf{x_j} - \mathbf{x_i})$ captures both the global (in $\mathbf{x_i}$) and the

local structure (in $\mathbf{x_j} - \mathbf{x_i}$). The last choice of h is the one that is used in *EdgeConv* model in Ref. [38] as well as in the trackster linking studies described here.

It is also imperative to consider the literature already existing on the use of GNNs for particle physics reconstruction and contrast it with the approach described in here. Ref. [39] uses a distance-weighted dynamic GNN to reconstruct particle showers in a highly granular calorimeter model, "loosely inspired" by the HGCAL. A dynamic GNN means that the graph connectivity is recomputed after every update of the node features, using the neighbors in the feature space of the nodes. Additionally, the model takes as input directly the hits in the calorimeter cells. In the approach described here, the graph of tracksters, instead of hits, is static and does not change after the updates of node features. The number of tracksters is $\mathcal{O}(10^2)$ less than the number of hits, reducing significantly the problem size. The approach described here is quite similar to that used in Ref. [40] in that it also uses a static graph and casts the task of calorimeter clustering into an edge classification problem. But Ref. [40] differs in their use of hits as the input (instead of tracksters) and by considering only single particle scenarios. In general, the approach described in this thesis is also the first to use graph clustering on the output graph of the GNN to build the final clusters.

5.2 Graph Building

The input graph for the GNN model is built by using loose angular compatibility to connect nodes with an edge. Fig. 5.2 shows the input graph for an event with two nearby pions shot in front of the HGCAL. Every trackster reconstructed by CLUE3D becomes a node in the graph. Centered on every trackster, every other trackster lying inside a 0.2 η-ϕ window is connected to it with an edge. Therefore, the edges connect a trackster to all its neighbors in η-ϕ space. The angular compatibility is motivated by the fact that tracksters reconstructed from the same particle would mostly lie inside such an angular window and we want to guide the GNN to connect tracksters that are longitudinally aligned. The window is kept sufficiently large so that pairs of tracksters that originate from the same particle have an edge between them or are connected through intermediate nodes in the input. If a trackster is still too far away, it is connected to its nearest neighbor (in three-dimensional physical space) with an edge. This can be a trackster that is complete, and does not need to be linked to any other trackster or a distant part of a shower. Having an edge, along with the input features of the nodes, helps the model to learn either of these scenarios. The input node features are the Cartesian coordinates of the trackster barycenter, first eigenvector, and eigenvalues from PCA (representing the direction and



(a) (b)

Figure 5.2: (a) The input graph structure of an event with two pions shot in front of the HGCAL. (b) Tracksters in the same event are visualized in R-z coordinates. The numbers and colors of the nodes in both represent respectively the index of that trackster in the event collection and the CaloParticle that the trackster is associated with.

shape respectively), size in the number of included layer-clusters, raw energy and raw electromagnetic energy (energy deposited in the CE-E). The inclusion of two additional features, shower maximum – the HGCAL layer with maximum energy deposition for a trackster, and shower length – the extent of a trackster in terms of HGCAL layers, did not improve the model performance. This is probably because shower maximum is strongly correlated to the z coordinate of the barycenter and shower length to the size, as shown for double pion events in Fig. 5.3.

It is also interesting to note the positive correlation between the x (y) coordinate of the trackster barycenter and the x (y) component of the first eigenvector – indicating trackster directions pointing back to the z-axis. Eigenvalues 1 and 2, which measure the extent of the trackster in the plane perpendicular to its principal axis, are strongly correlated – indicating the symmetry of tracksters in the transverse plane. The trackster size is negatively correlated to shower maximum and the z coordinate of barycenter – implying tracksters that are reconstructed later in the calorimeter are usually smaller. There is also an obvious positive correlation between trackster size and energy.

For the truth labeling, edges in the input graph connecting two tracksters associated to the same `SimTrackster` from `CaloParticle` are labeled as true else false. To remind the reader, CaloParticle refers to the parent of a particle in the simulated decay graph nearest

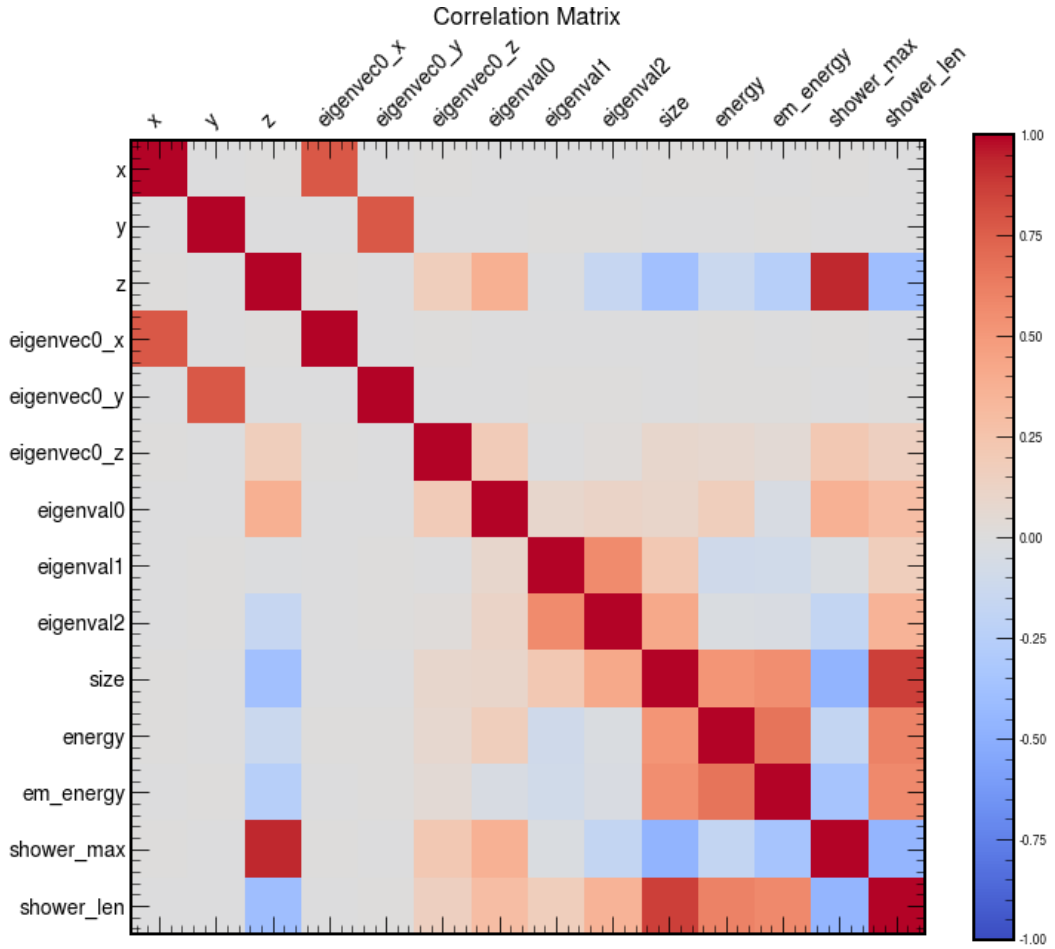


FIGURE 5.3: Correlation matrix of the node features used for the GNN, with two additional variables `shower_max` and `shower_len`, explained in the text. The correlations are calculated for tracksters reconstructed for two close-by pions shot in front of the HGCAL.

to the vertex. A SimTrackster from CaloParticle is a trackster made with all the layer-clusters associated with the same CaloParticle. So if a particle starts showering even before entering the HGCAL, all the reconstructed tracksters from those showers will be associated with the same SimTrackster from CaloParticle. Clearly, the tracksters reconstructed from the secondary components of a shower inside the calorimeter will also be associated with the same SimTrackster from CaloParticle. The associations of a reconstructed object (e.g. layer-cluster, trackster) to a simulated one (e.g. CaloParticle, SimTrackster) are obtained from the validation modules of HGCAL by an energy fraction matching done at the detector hit level.

5.3 Model and Training

The PyTorch Geometric [41] Graph Neural Network library, built upon PyTorch [42], is used for the model and training. The structure of the model used is fairly simple. An initial two-layer fully connected (FC) network is used to obtain the latent representation of the nodes. It takes the node features described before as input and outputs the 64-dimensional latent representation for every node. A static graph *EdgeConv* block is used for message passing in the graph. A two-layer FC network is used to calculate the messages (from the latent representation of the connected nodes) and the aggregation function used is the summation. Since there are no natural directions of the edges in the input graph, they are made undirected before the message passing. Additionally, self-loops are added to allow the information of a node to be included in its update. Four iterations of the EdgeConv block are used. The number of iterations defines the receptive field of a node, and the idea is to allow a trackster to have information about both its immediate surroundings (in the same shower) as well as from showers in its neighborhood. After the message passing, another three-layer FC network is used to get the edge representations from the embeddings of connected nodes and subsequently obtain a classification score per edge, representing its probability of being legitimate. A legitimate edge would be one connecting tracksters coming from the same particle.

Since there is an almost 3:1 imbalance in the true-to-false edges in the training data (cross shower edges are discouraged due to the angular window used to build the edges), a focal loss with $\alpha = 0.25$ and $\gamma = 2$ is used to stress more on the misclassified examples. The model is trained with a batch size of 32 and for 50 epochs separately on two types of simulated events: two close-by pions shot in front of the detector (around 100,000 events) and ten randomly selected particles from a list of electrons, photons, charged and neutral hadrons shot in front of the detector (around 50,000 events). In both cases, generated particles are uniformly distributed in energy between 10 and 600 GeV, in η between 1.8 and 2.5, and between 0 and 2π in ϕ. For double pions the showers could be overlapping and are shot in a 30 cm wide window, opened at an η and ϕ randomly chosen from the intervals mentioned above. The Adam optimizer is used with the initial learning rate of 10^{-3} and weight decay of 5×10^{-4}. A learning rate scheduler is configured to decrease the learning rate by a factor of 0.1 once the validation loss stops decreasing, with a patience of 5 epochs. The choice of hyperparameters was made based on testing a few different combinations and finding the ones with the best performance.

5.4 Building Clusters

The model predicts a score for every edge in the input graph representing the probability of that edge being true. With the edge scores, the graph of tracksters becomes a *similarity graph* – the scores being a measure of how "similar" the connected nodes are – or in this case, the probability of those to belong to the same shower. One naive way to proceed from here would be to find the *connected components* in the output graph, perhaps after dropping the edges with low scores. A connected component of a graph is a set of nodes, in which every pair is either connected with an edge or there is a path between the two via other intermediate vertices. Every connected component would thus form a cluster containing the tracksters that the model predicts should be linked. This was seen to work when showers are well separated and the model can confidently classify any cross-cluster edges as false. But in other scenarios, this can easily go wrong if there is even one misclassified edge between two otherwise well-separated clusters – eventually merging those two showers.

Even though the model predictions are not perfect at the per-edge level, it was observed that the model successfully learns the community structure of showers. The community structure in a graph or network refers to the much stronger connections within a community (shower, in our case) compared to those between two communities. Apart from just true or false edges (after thresholding at a certain score), the continuum of edge scores obtained from the model provide another dimension to this, encoding rich information about the structure. This information can then be used to detect communities in the graph. This is indeed a well-established domain in network science, known by several names such as community detection or graph clustering.

5.4.1 Graph Clustering

To reiterate, the general idea is to find a partitioning of the graph such that the edges between groups have very low weights compared to those within a group. There can be various ways to solve this and one of them is a *Spectral Clustering* of the graph. The word spectral refers to the use of the eigenvectors of the graph connectivity matrix, in this case, the graph Laplacian. The (unnormalized) graph Laplacian L is defined as

$$L = D - W,$$

where $W = (w_{ij})_{i,j=1,\dots,n}$ is the weighted adjacency matrix with w_{ij} being the score of the edge connecting nodes i and j (if any, else is zero) and D is the diagonal degree matrix

with elements

$$d_i = \sum_{j=1}^{n} w_{ij}.$$

The normalized Laplacian is defined as $D^{-1}L$. There can be several different spectral clustering algorithms using different graph Laplacians (unnormalized/normalized) but a general type is described below. For implementation in code, `SpectralClustering` from the scikit-learn toolkit [43] is used for the graph clustering step, which uses a slightly different normalized graph Laplacian. Additionally, the edges with scores below 0.5 are dropped from the graph before finding communities in it.

The general logic of a spectral clustering algorithm is as follows: given the Laplacian (of shape $n \times n$, n being the number of nodes in the graph), its first k eigenvectors $u_1, ..., u_k$ are calculated, where k is the desired number of clusters to construct. A matrix, called U, is constructed containing the k eigenvectors as its columns. Hence U is of shape $n \times k$. If $y_i \in \mathbb{R}^k$ is a vector corresponding to the i-th row of U, the n points $(y_i)_{i=1,...,n}$ are clustered to form k clusters $C_1, ..., C_k$, using any standard algorithm such as k-means. The clusters of the n original nodes can be assigned from their one-to-one correspondence with y_i. Due to the properties of the graph Laplacian, the change of representation to $y_i \in \mathbb{R}^k$ enhances the cluster properties which can then be easily identified using any simple clustering algorithm.

The desired number of clusters to input to the spectral clustering algorithm is still unknown. The *eigengap* heuristic is used to infer this from the connectivity of the graph. This uses the spectrum of eigenvalues of the graph Laplacian to find the number k such that all the eigenvalues $\lambda_1, ..., \lambda_k$ are very small but λ_{k+1} is relatively large. A justification for this can be given based on perturbation theory. For a graph with k completely disconnected clusters (connected components), the eigenvalues from 1 through k are zero and there is a gap to the $(k+1)$-th eigenvalue (top row of Fig. 5.4). The same graph, where the k components are not fully disconnected, but are connected only by a few edges of low weights, can be represented as a perturbed case of the previous Laplacian. Perturbation theorems state that the eigenvalues or eigenvectors of the original and the perturbed matrix are similar and the "distance" between them is bounded by the magnitude of the perturbation times a constant – which usually depends on the eigenvalue under consideration and how far it is from the rest of the spectrum. Thus, for a small enough perturbation and a large enough eigengap $|\lambda_k - \lambda_{k+1}|$ in the original Laplacian, the first k eigenvalues of the perturbed matrix, though not all zero, will still have a gap to the $(k+1)$-th (bottom row of Fig. 5.4).

The eigengap heuristic is implemented to find the first occurrence of a gap between

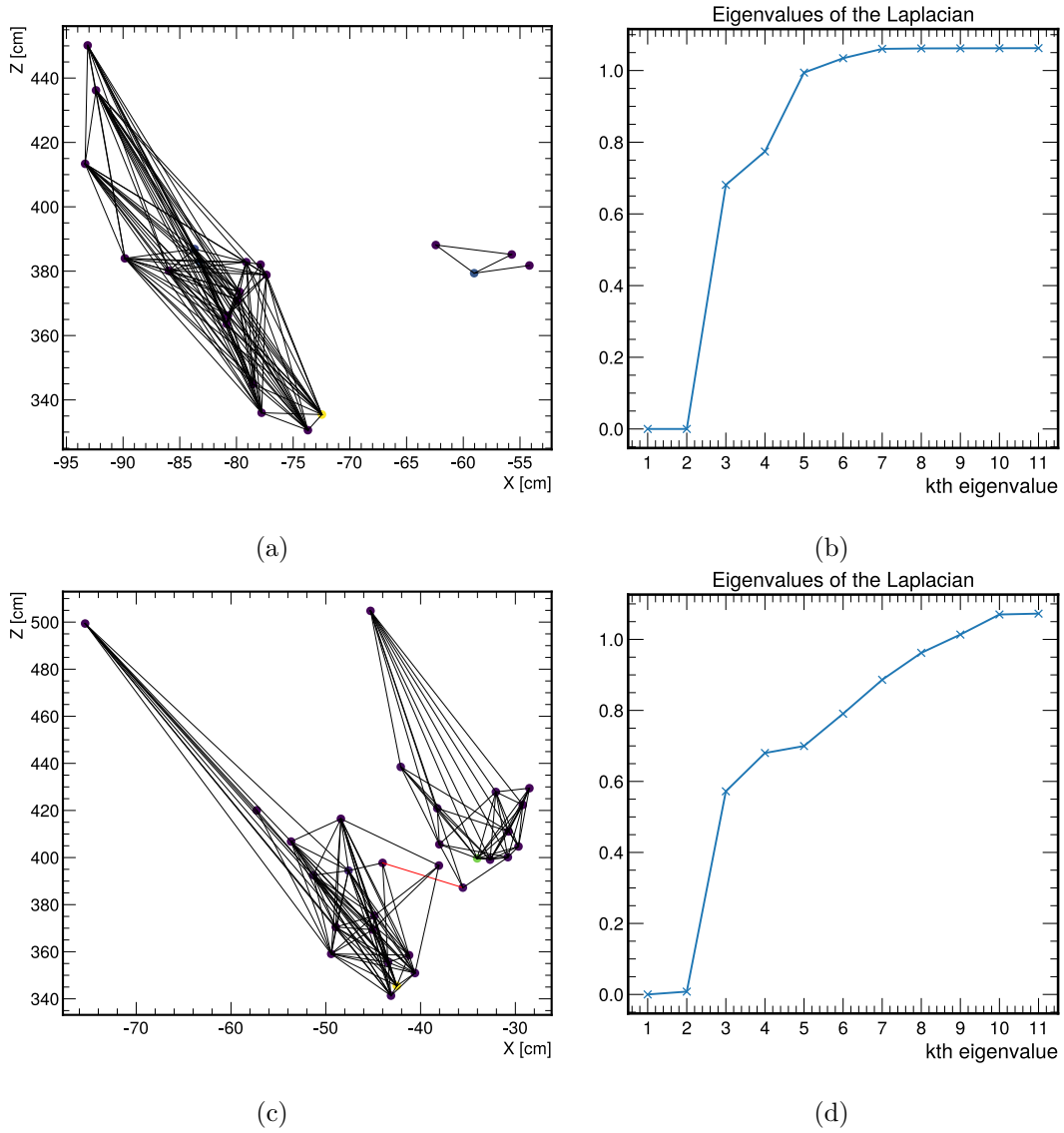


FIGURE 5.4: (a) A graph of tracksters with two completely disconnected components and (b) the eigenvalue spectrum of its normalized Laplacian, showing the first two eigenvalues to be exactly zero and the eigengap between eigenvalues two and three. (c) A similar graph with a single edge (in red) connecting the two communities with (d) the first eigenvalue of its normalized Laplacian still zero (corresponding to the only one connected component), and the second eigenvalue greater than zero but still separated from the third by a large gap.

consecutive eigenvalues to be more than a threshold, set at 0.2. But if the following difference is more than thrice this threshold, the next gap is considered to be relevant –

corresponding to a more accurate description of the community structure of the graph. The thresholds were set empirically from the observed eigenvalue spectra of the graph Laplacians.

The eigengap method works best when communities in the graph are balanced – the number of nodes and their inter-connectivities are similar in clusters. The nodes with no edges left after thresholding the edge scores from the model are removed from the Laplacian before inferring the number of clusters in the rest of the graph. The number of nodes removed is added back when passing the number of clusters to the spectral clustering step.

A more involved discussion of the properties of the graph Laplacian, spectral clustering algorithms, or the eigengap heuristic is beyond the scope of this thesis. The interested reader is warmly encouraged to consult Ref. [44], which gives an excellent overview of the subject, or the numerous references therein.

5.5 Results

The clusters obtained from the GNN with the graph clustering approach, for two pions shot in front of the detector, are shown and compared to the truth in Fig. 5.5. One can

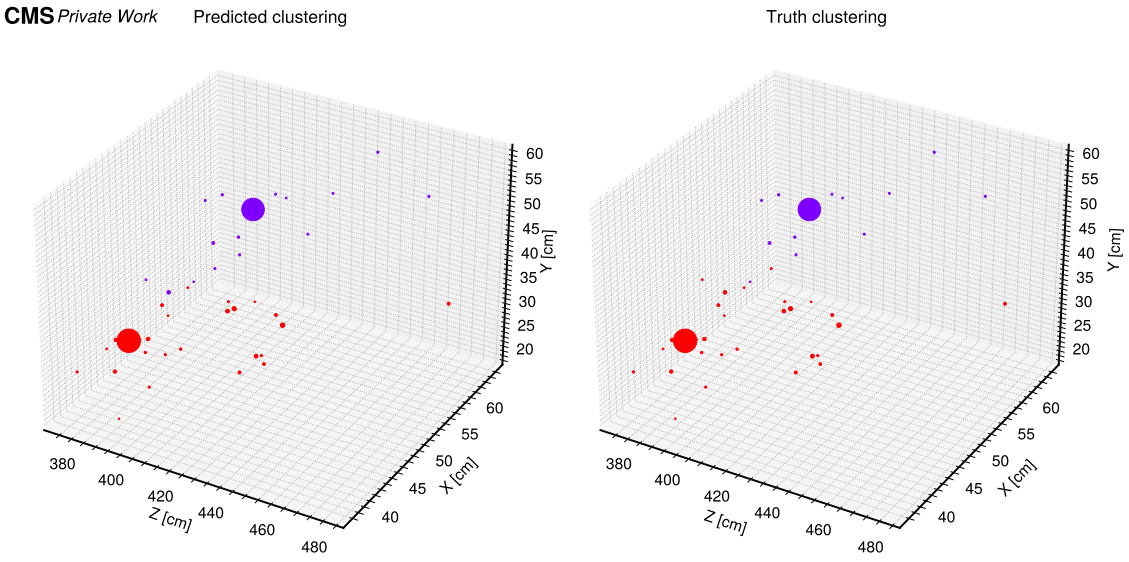


Figure 5.5: Predicted (left) clusters from the GNN with graph clustering approach compared with the truth (right) for two pions shot in front of the HGCAL. Every point is a trackster with sizes proportional to its energy. Different colors represent different clusters.

visually observe that the clusters obtained are quite similar to the truth, and most of the

smaller, distant tracksters are also assigned to the correct cluster.

To quantify the quality of clusters obtained from this approach, an energy intersection over union (EIoU) score is calculated between the clusters of tracksters, *SuperTracksters* for convenience, and the truth – *SimTrackster* from CaloParticle. To remind oneself, the association of a reconstructed trackster to a SimTrackster is obtained from the validation modules of HGCAL. The score between a SuperTrackster i and a SimTrackster j (Eqn. 5.1) is obtained by dividing the sum of the energy of tracksters that are common to both by the sum of energy of tracksters that are contained in their union.

$$score(\text{supts}_i, \text{simts}_j) = \frac{\sum_{t_n \in \text{supts}_i \cap \text{simts}_j} E_{t_n}}{\sum_{t_n \in \text{supts}_i \cup \text{simts}_j} E_{t_n}} \tag{5.1}$$

So a score of 1 would mean a perfect match between the two and 0 indicates no intersection. For a SuperTrackster, this score is initially calculated to all the SimTracksters in the event, and the highest among those is taken as its final score. The scores for the tracksters merged with the geometric linking is calculated in the same way.

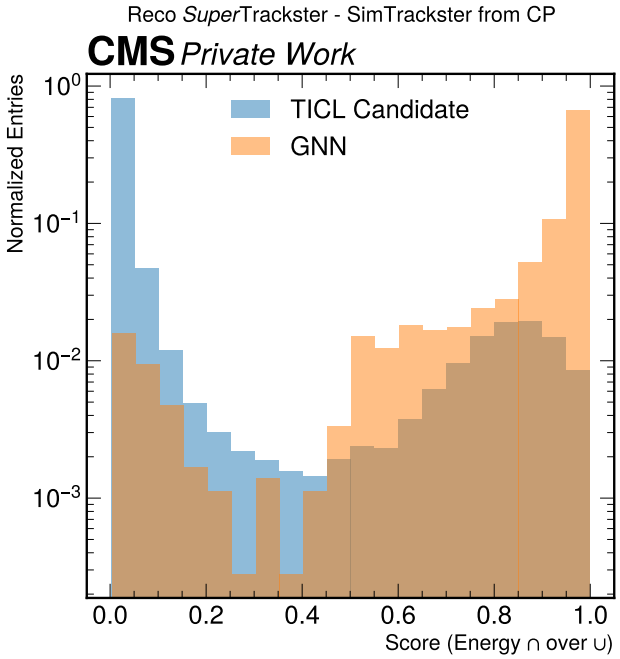
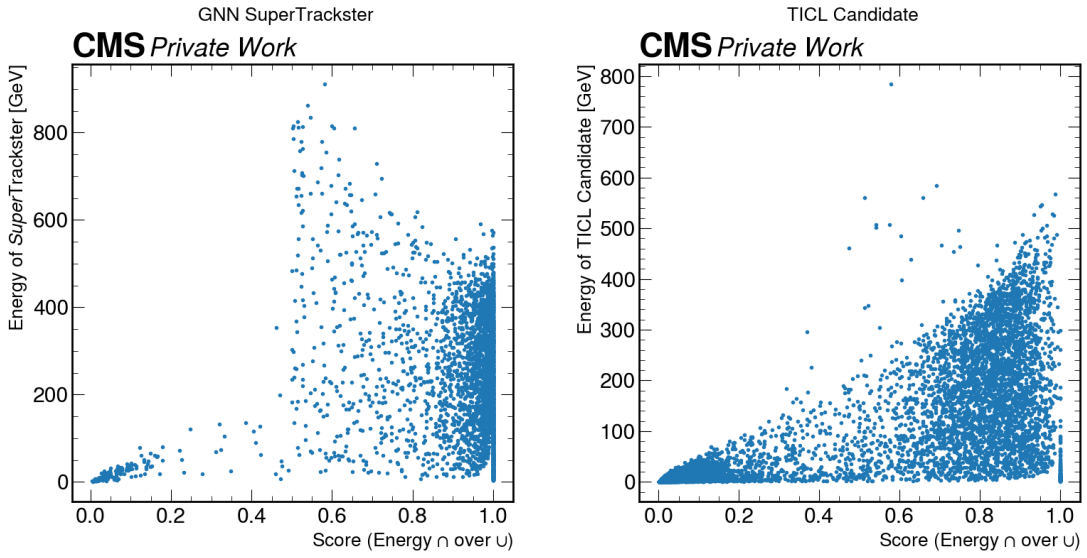


FIGURE 5.6: The histogram of EIoU scores of SuperTracksters obtained from the GNN approach compared with that of the scores of the merged tracksters from TICLCandidates obtained using the geometric linking – with 2000 events of double pions shot in front of the detector. Each histogram is separately normalized by its total number of entries.

The comparison of scores obtained for the GNN SuperTracksters and the merged tracksters from the TICLCandidates (geometric linking), for two pions shot in front of the detector, is shown in Fig. 5.6. The model used was also trained on a (different) sample of double pions. The number of clusters produced from the geometric linking is more than an order of magnitude larger than using the GNN approach and hence the histograms shown are normalized separately by their total number of entries. It can be observed that a much higher fraction of clusters produced using the GNN approach have scores closer to 1 – implying that those are better matched to the truth than the ones obtained from geometric linking.



(a) (b)

Figure 5.7: Energy vs. EIoU score distributions of (a) SuperTracksters and (b) merged tracksters from TICLCandidates, plotted for 2000 events of double pions shot in front of the detector.

To understand how the quality of the produced clusters depend on their energy, the energy versus score distributions for the GNN SuperTracksters and the merged tracksters from TICLCandidates are shown in Fig. 5.7. These are plotted with the same double pion sample and for 2000 events. As expected from Fig. 5.6, a significant fraction of the SuperTracksters have scores near 1 and the fraction is larger compared to the case of the merged tracksters. One can further notice two more things, firstly that the low score clusters from the GNN are of low energy, and secondly that the GNN merges more than the geometric linking, noticeable from the points at the higher energy regions. This also corroborates the observation that on an average the GNN approach produces around 1.8 SuperTracksters per event. In comparison, the geometric linking produces around 18.9

TICLCandidates from around 28.1 tracksters per event.

Another way to evaluate performance over a full event is by looking at the event-wise fraction of energy correctly clustered. A cluster is labelled as correct if its score is greater than 0.5. The fraction of energy correctly clustered per event is obtained by dividing the total energy contained in clusters which are labelled correct, by the total energy of tracksters reconstructed in that event. Fig. 5.8 compares the distribution of the fraction of energy correctly clustered per event, over 2000 events, from the GNN with that in the tracksters merged with the geometric linking. The double pion sample used is the same as before. In almost all of the events, the GNN manages to cluster close to all of the energy correctly. The peak at the highest bin is quite sharp, with the next higher bin containing almost two orders of magnitude less events. The geometric linking, on the other hand, manages to cluster a bit more than 80% of the energy in most of the events. But the peak in this case is much broader. It is thus quite clear that the GNN approach can correctly cluster a higher fraction of energy in a significantly more number of events when compared to the geometric linking.

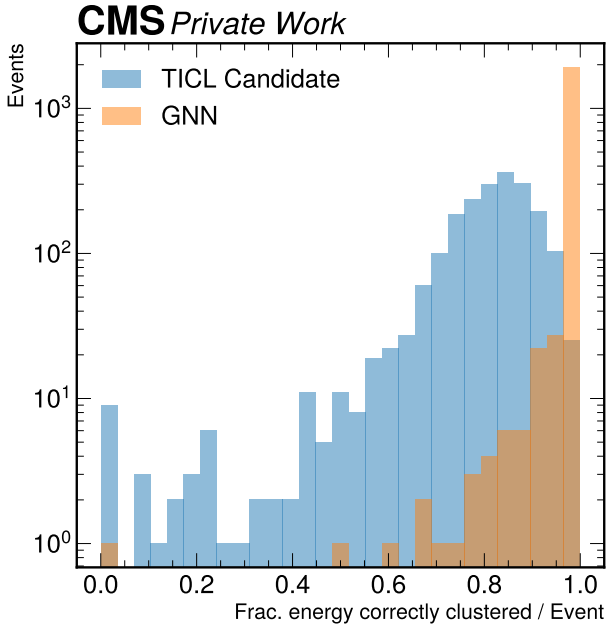


FIGURE 5.8: Histograms of the fraction of energy correctly clustered in an event using the GNN approach and in the merged tracksters from TICLCandidates, for 2000 events of double pions shot in front of the detector.

As mentioned in Section 5.3, the model was also trained on a sample of 10 randomly

selected particles shot in front of the HGCAL. The same metrics as before are calculated to test the performance of the approach on this sample. Fig. 5.9 compares the scores obtained for the SuperTracksters and the merged tracksters from geometric linking. The higher fraction of scores close to 1 for the SuperTracksters indicates that the clusters from this approach are better matched to the truth. However, when compared to the SuperTrackster scores for double pions (Fig. 5.6), we can see a slightly degraded performance – observable as the higher fraction of SuperTracksters with low scores. Also, a linking algorithm with slightly larger windows was used for all the plots with the multiparticle sample, due to the unavailability of these samples with the latest version of the linking. This is presumably[1] why a higher fraction of merged tracksters have scores closer to 1 in the case of the multiparticles when compared to the double pions.

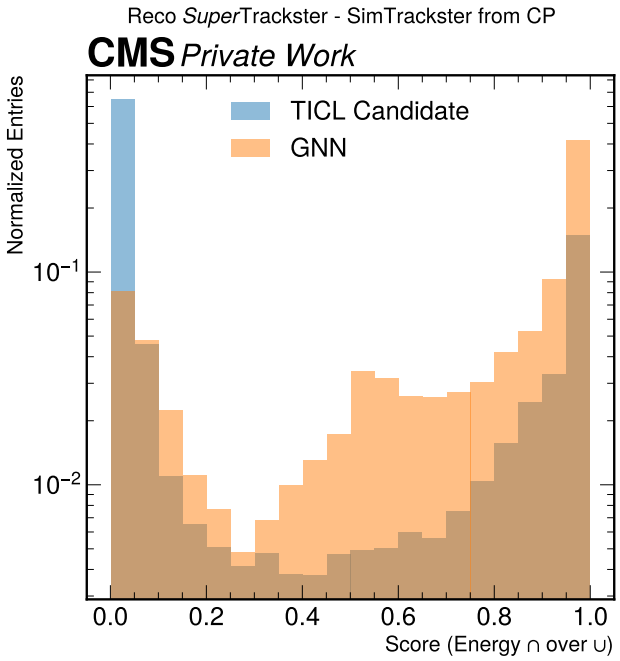
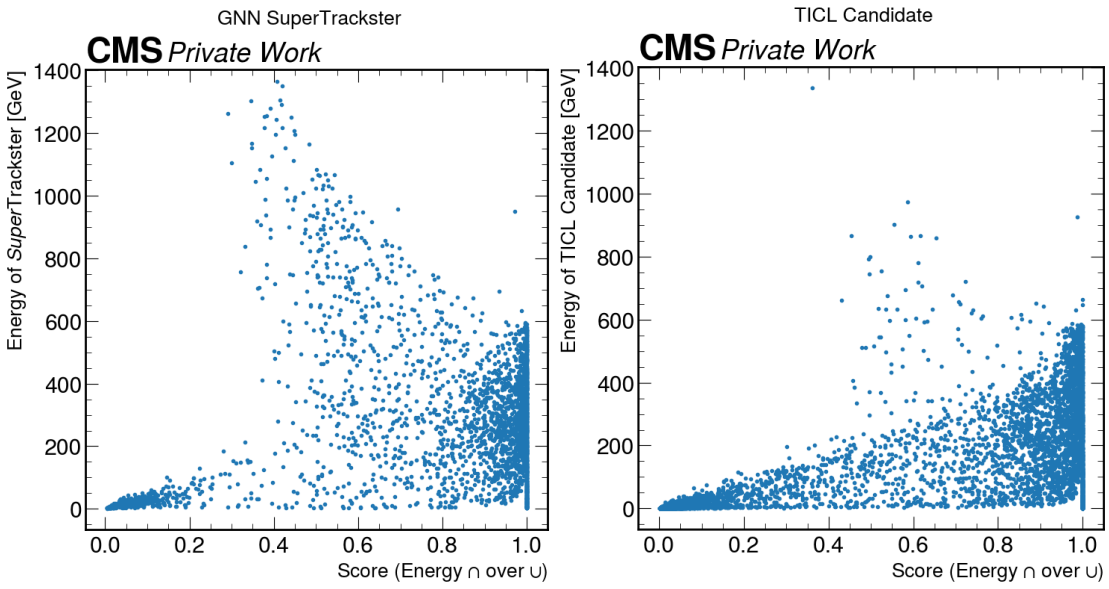


Figure 5.9: The histogram of EIoU scores of SuperTracksters obtained from the GNN approach compared with that of the merged tracksters from the TICLCandidates obtained using the geometric linking – with 2000 events of 10 randomly chosen particles shot in front of the detector. Each histogram is separately normalized by its total number of entries.

Fig. 5.10 shows the energy versus score distributions for the SuperTracksters and the merged tracksters with the multiparticle sample. The number of events used is however 400, both to keep a parity with the total number of particles in Fig. 5.7 and for a cleaner

[1]Showers can be more separated in the multiparticle sample, but this effect was also observed in double pions when comparing the two versions of linking.



(a) (b)

FIGURE 5.10: Energy vs. EIoU score distributions of (a) SuperTracksters and (b) merged tracksters from TICLCandidates, plotted for 400 events of 10 randomly chosen particles shot in front of the detector.

visualization (every event has five times the number of particles now). In addition to low energy clusters contributing to the low scores, one can observe a significant number of clusters between scores of around 0.4 and 0.8 that have energies larger than 600 GeV, the maximum energy a single particle can have in the sample. This indicates some merging of clusters and is also consistent with the observed, on average, ∼7.7 SuperTrackters produced per event. For comparison, there were around 32.6 merged tracksters and 80.7 tracksters produced on average in an event.

Similarly, the distributions of the event-wise fraction of energy clustered correctly, over 2000 events, in SuperTracksters and in merged tracksters are compared in Fig. 5.11. Even though there are more events where close to the entire energy is clustered correctly with the GNN approach, there is also an appreciable number of events when this approach fails to correctly cluster a significant chunk of the available energy – most probably for the instances when the GNN approach merges clusters. However, this tail of events, with a low fraction of energy correctly clustered, is not as pronounced with the geometric linking.

To summarize the performance on the multiparticle sample, the clusters from the GNN approach are in general better in quality, when compared to the merged tracksters obtained from the geometric linking. But there is also some merging of clusters observed and this

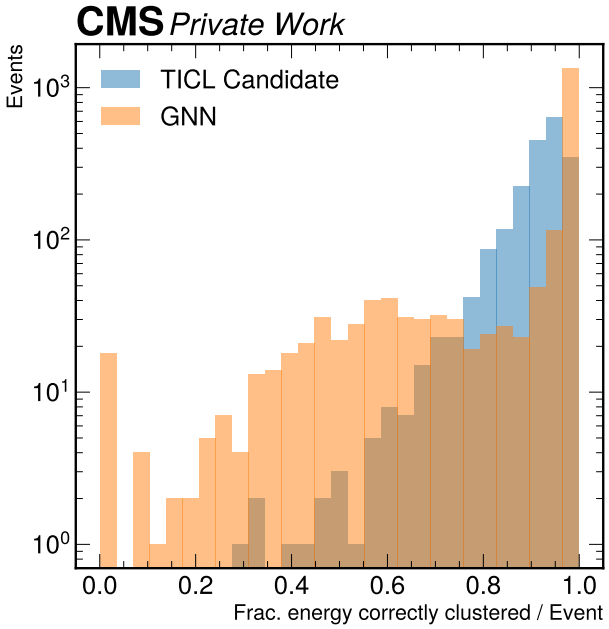


Figure 5.11: Histograms of the fraction of energy correctly clustered in an event using the GNN approach and in the merged tracksters from TICLCandidates, for 2000 events of 10 randomly chosen particles shot in front of the detector.

possibly also contributes to the instances when an appreciable fraction of the energy in an event is not clustered correctly. The threshold (currently 0.5) applied to the edge scores is a parameter that affects this and with a higher threshold, the number of clusters inferred using the eigengap heuristic increases – for example to around 10.7 per event with 0.6. But this also results in clusters that are not as good, presumably due to the additional loss of information.

6

Conclusion and Outlook

Reconstruction for the CMS High Granularity Calorimeter, the Phase-2 upgrade for the calorimeter endcaps, is a challenging task. The high expected pile-up and the novelty of the detector contribute to the challenges. The work on developing a reconstruction that can handle the challenges and turn the novelty of the detector into its strength is currently underway. The granular, 5-Dimensional information obtained from the HGCAL makes it an ideal candidate for Particle Flow calorimetry.

As part of this thesis, a new geometric linking algorithm was developed and deployed in CMSSW, in the recently upgraded version 4 of the TICL reconstruction framework. The linking of clustered energy deposits in the calorimeter, to themselves and to off-calorimeter objects, is essential for a Particle Flow interpretation of the event. The new linking algorithm was introduced along with a plug-in system where several such algorithms could reside simultaneously and be switched as required. The developed algorithm is based on the idea of finding geometrically close objects to be linked after projecting tracks and tracksters onto the same surface. The graph of such geometrically close by objects is then explored, aided with considerations of energy and time compatibility, to build the final objects. An excellent reconstruction efficiency was observed for electromagnetic objects and an overall good efficiency for reconstruction of charged pions, which have notoriously fluctuating shower profiles and produce separated reconstructed clusters of energy in the

detector. The linking algorithm could successfully accumulate most of those. With the new energy density based pattern recognition algorithm, CLUE3D, that the new version of TICL is built around, improvements in the reconstruction of higher level objects were also observed, namely in the jet response and efficiency. The time and memory consumption of the developed algorithm were observed to be under control even in the scenario with 200 pile-up.

There are several important parameters of the algorithm, like the window sizes for geometrical compatibility, that have not been rigorously tuned yet and can lead to further improvements in performance. Moreover, the tuning will also lead to a better understanding of how the parameters affect the linking performance, especially in the presence of high pile-up. Furthermore, the availability of truth matching of reconstructed tracks will be important in quantifying and therefore understanding the track-to-trackster linking performance better. To sum up, the algorithm presented is a step in the way of exploring and improving similarly motivated solutions for the linking problem.

A novel approach for linking tracksters using Graph Neural Networks and graph clustering was explored in the other part of this thesis. The segmentation/clustering task was framed as an edge classification problem. The static graph EdgeConv model was trained to learn the structure of showers on a loose graph of tracksters – predicting the probabilities of edges connecting two tracksters that should be linked. The similarity graph thus generated was then used to find strongly connected communities using spectral clustering. The model was trained and tested separately on events containing two close by pions and ten particles randomly chosen from a list containing electrons, photons, charged and neutral hadrons. This approach was observed to perform very well for two pions, managing to assign even distant and small tracksters to the correct clusters. The low quality clusters produced were of low energy and the approach could correctly cluster close to the entire energy available in the event in a significantly higher number of events, when compared to the geometric linking. The performance in the case of multiparticles was not as good, with some merging of clusters being observed. However, in both the cases, the clusters of tracksters produced by this approach were better matched to the truth than the ones obtained from the geometric linking algorithm.

The situations where the clustering was unsatisfactory could be separately studied in more detail to better understand the point(s) of failure. For example, it is not clear if in some cases the showers were overlapping – and hence could not possibly be separately clustered – or not. The implementation of the eigengap heuristic, responsible for inferring the number of clusters, is a weak point. The logic used for that can be improved to better take care of the edge cases. It was found that the model trained on the multiparticle

sample does also work for double pions; it would indeed be interesting to repeat the exercise with more complicated event topologies, e.g. containing jets – to also understand if this approach is feasible for global linking or more profitable for use in local reconstruction, e.g. Bremsstrahlung recovery around electrons, or accumulating separate components of hadronic showers. Moreover, only the linking of tracksters has been considered so far, and it is interesting to extend it to also include tracks – perhaps using its propagated state to the HGCAL front as an additional node in the graph.

Nevertheless, this approach demonstrates that the linking problem can be solved starting from a sparse graph of already clustered energy deposits, benefiting from a reduction of $\mathcal{O}(10^2)$ in problem size compared to starting from the hits, and learning structures on it. The very small Graph Neural Network model of only around 42,000 parameters provides fast inference, and efficient methods exist for computing the first eigenvalues of a sparse matrix. Efficient linear algebra libraries are also available in C++ that can be used for the eigenvalue calculation, e.g., Eigen [45], Spectra [46]. All things considered, this approach of combining a Graph Neural Network and graph clustering can have fast implementations, potentially making it a viable alternative.

Acknowledgments

First of all, I would like to express my deepest appreciation for Prof. Alexander Schmidt and Dr. Felice Pantaleo for their patient guidance and supervision, academic or otherwise, and for giving me this opportunity to work on a very interesting problem for my master's thesis. I am grateful to Prof. Johannes Erdmann for agreeing to be the second examiner.

I would also like to extend my deepest gratitude to Dr. Marco Rovere, of the HGCAL DPG, for providing many valuable inputs from the physics side of reconstruction in our weekly meetings and maintaining a jovial work environment. This thesis could not have been completed in a year if not for the innumerable times Wahid Redjeb answered to my calls for help. He also provided the datasets, wrote the code for the beautiful visualization of the graphs and helped with the development of the graph neural network approach during the TICL ML hackathon. I could not have imagined a better mentor, thank you for everything!

I would like to thank Prof. Alexander Schmidt again and the Institute IIIA for supporting my travel to CERN. I must also thank Dr. Shamik Ghosh, Dr. Benedikt Maier, Dr. Davide Valsecchi, Dr Huilin Qu at CERN, and almost everyone else present at the TICL ML Hackathon for the helpful discussions, and the many people involved in TICL for their insightful comments during the meetings.

The world has been a challenging place to live in the last few years. I must thank the people who made it slightly less so: Vasilis, Alena (for also being the best office mate), Andrey, Ming-Yan, Purna, Bernardo, Andrzej at Aachen; Toni, Nikos, Juan, Giorgio, Nicolo, Yassine, Luca, Aurora at CERN and Ashutosh, Varun at neither of those two places – Thank you!

Last, but definitely not the least, I thank my parents for all their love, support and encouragement – and maybe also for teaching me to ask questions and be curious in the first place.

References

[1] Lyndon Evans and Philip Bryant. LHC Machine. *Journal of Instrumentation*, 3(08): S08001, aug 2008. doi: 10.1088/1748-0221/3/08/S08001. URL https://dx.doi.org/10.1088/1748-0221/3/08/S08001. 1, 3

[2] O. Aberle, I Béjar Alonso, O Brüning, et al. *High-Luminosity Large Hadron Collider (HL-LHC): Technical design report*. CERN Yellow Reports: Monographs. CERN, Geneva, 2020. doi: 10.23731/CYRM-2020-0010. URL https://cds.cern.ch/record/2749422. 1, 8

[3] CMS Collaboration. The CMS experiment at the CERN LHC. *Journal of Instrumentation*, 3(08):S08004, aug 2008. doi: 10.1088/1748-0221/3/08/S08004. URL https://dx.doi.org/10.1088/1748-0221/3/08/S08004. 1, 3

[4] The ATLAS Collaboration. The ATLAS Experiment at the CERN Large Hadron Collider. *Journal of Instrumentation*, 3(08):S08003, aug 2008. doi: 10.1088/1748-0221/3/08/S08003. URL https://dx.doi.org/10.1088/1748-0221/3/08/S08003. 1

[5] D Contardo, M Klute, J Mans, L Silvestris, and J Butler. Technical Proposal for the Phase-II Upgrade of the CMS Detector. Technical report, Geneva, 2015. URL https://cds.cern.ch/record/2020886. 1, 9

[6] The Phase-2 Upgrade of the CMS Endcap Calorimeter. Technical report, CERN, Geneva, 2017. URL https://cds.cern.ch/record/2293646. vii, 1, 10, 13, 21

[7] Felice Pantaleo and Marco Rovere. The Iterative Clustering framework for the CMS HGCAL Reconstruction. Technical report, CERN, Geneva, 2022. URL http://cds.cern.ch/record/2806234. viii, 2, 18

[8] CMS Collaboration. Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC. *Physics Letters B*, 716(1):30–61, 2012. ISSN 0370-2693. doi: https://doi.org/10.1016/j.physletb.2012.08.021. URL https://www.sciencedirect.com/science/article/pii/S0370269312008581. 3

[9] G L Bayatian et al. *CMS Physics: Technical Design Report Volume 1: Detector Performance and Software*. Technical design report. CMS. CERN, Geneva, 2006. URL https://cds.cern.ch/record/922757. 3

[10] Tai Sakuma. Cutaway diagrams of CMS detector. 2019. URL https://cds.cern.ch/record/2665537. vii, 5

[11] The European Strategy for Particle Physics Update 2013. La stratégie européenne pour la physique des particules Mise à jour 2013. 16th Session of European Strategy Council. 2013. URL https://cds.cern.ch/record/1567258. 8

[12] The European Strategy Group. Deliberation document on the 2020 Update of the European Strategy for Particle Physics. Technical report, Geneva, 2020. URL https://cds.cern.ch/record/2720131. 8

[13] Building for Discovery: Strategic Plan for U.S. Particle Physics in the Global Context; May 2014. 5 2014. URL https://www.osti.gov/biblio/1320565. 8

[14] The Phase-2 Upgrade of the CMS Tracker. Technical report, CERN, Geneva, 2017. URL https://cds.cern.ch/record/2272264. 9

[15] The Phase-2 Upgrade of the CMS Muon Detectors. Technical report, CERN, Geneva, 2017. URL https://cds.cern.ch/record/2283189. 10

[16] CMS Collaboration. A MIP Timing Detector for the CMS Phase-2 Upgrade. Technical report, CERN, Geneva, 2019. URL https://cds.cern.ch/record/2667167. 10

[17] The Phase-2 Upgrade of the CMS Barrel Calorimeters. Technical report, CERN, Geneva, 2017. URL https://cds.cern.ch/record/2283187. 10

[18] The Phase-2 Upgrade of the CMS Level-1 Trigger. Technical report, CERN, Geneva, 2020. URL https://cds.cern.ch/record/2714892. 10

[19] CMS Collaboration. The Phase-2 Upgrade of the CMS Data Acquisition and High Level Trigger. Technical report, CERN, Geneva, 2021. URL https://cds.cern.ch/record/2759072. 10

[20] J-C. Brient and H. Videau. The calorimetry at the future e+ e- linear collider. 2002. doi: 10.48550/ARXIV.HEP-EX/0202004. URL https://arxiv.org/abs/hep-ex/0202004. 15, 21

[21] A.M. Sirunyan et al. Particle-flow reconstruction and global event description with the CMS detector. *Journal of Instrumentation*, 12(10):P10003, oct 2017. doi: 10.1088/1748-0221/12/10/P10003. URL https://dx.doi.org/10.1088/1748-0221/12/10/P10003. 15

[22] Marco Rovere, Ziheng Chen, Antonio Di Pilato, Felice Pantaleo, and Chris Seez. CLUE: A Fast Parallel Clustering Algorithm for High Granularity Calorimeters in High-Energy Physics. *Frontiers in Big Data*, 3, 2020. ISSN 2624-909X. doi: 10.3389/fdata.2020.591315. URL https://www.frontiersin.org/articles/10.3389/fdata.2020.591315. vii, 16

[23] Alex Rodriguez and Alessandro Laio. Clustering by fast search and find of density peaks. *Science*, 344(6191):1492–1496, 2014. doi: 10.1126/science.1242072. URL https://www.science.org/doi/abs/10.1126/science.1242072. 16

[24] CLUE: a clustering algorithm for current and future experiments. 2022. URL http://cds.cern.ch/record/2805639. 17, 20

[25] Matteo Cacciari, Gavin P. Salam, and Gregory Soyez. FastJet user manual. *The European Physical Journal C*, 72(3):1896, Mar 2012. ISSN 1434-6052. doi: 10.1140/epjc/s10052-012-1896-2. URL https://doi.org/10.1140/epjc/s10052-012-1896-2. 18, 19

[26] The TICL (v4) reconstruction at the CMS Phase-2 High Granularity Calorimeter Endcap. 2022. URL http://cds.cern.ch/record/2839740. 20

[27] Trackster ID and cleaning for EM iterations. 2022. URL https://cds.cern.ch/record/2805638. 23

[28] TICL trackster energy regression and particle identification - new CNN plots. https://rovere.web.cern.ch/rovere/TICL_ML_Hackathon/HackathonRetrain/10_600_Energies/trainedFixTracksterIndicesOnlyTracksterTargetRegressed/, Oct 2022. Presented in TICL Reconstruction Meeting; accessed: 22-12-2022. viii, 27

[29] Marcel Rieger, Ziheng Chen, Adriano Di Florio, Antonio Di Pilato, Abdulla Mohamed, Hasib Muhammad, Felice Pantaleo, and Marco Rovere. Energy Regression and Particle ID in `RecoHGCal/TICL`, Aug 2019. URL https://indico.cern.ch/event/841640/#225-hgcal-energy-regression-an. Presented at Reconstruction and Analysis Tools meeting, CERN. viii, 28

[30] Daniele Bertolini, Philip Harris, Matthew Low, and Nhan Tran. Pileup per particle identification. *Journal of High Energy Physics*, 2014(10):59, Oct 2014. ISSN 1029-8479. doi: 10.1007/JHEP10(2014)059. URL https://doi.org/10.1007/JHEP10(2014)059. 34

[31] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry, 2017. URL https://arxiv.org/abs/1704.01212. 37

[32] Peter W. Battaglia et al. Relational inductive biases, deep learning, and graph networks. *CoRR*, abs/1806.01261, 2018. URL http://arxiv.org/abs/1806.01261. ix, 38, 39

[33] Michael M. Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric Deep Learning: Going beyond Euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017. doi: 10.1109/MSP.2017.2693418. 37

[34] Frank Rosenblatt. *Principles of neurodynamics; perceptrons and the theory of brain mechanisms.* Spartan Books, Washington, 1962. URL https://catalog.hathitrust.org/Record/000203591. 37

[35] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, April 1980. ISSN 0340-1200, 1432-0770. doi: 10.1007/BF00344251. URL http://link.springer.com/10.1007/BF00344251. 37

[36] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation*, 1(4):541–551, December 1989. ISSN 0899-7667, 1530-888X. doi: 10.1162/neco.1989.1.4.541. URL https://direct.mit.edu/neco/article/1/4/541-551/5515. 37

[37] Jeffrey L. Elman. Finding Structure in Time. *Cognitive Science*, 14(2):179–211, March 1990. ISSN 03640213. doi: 10.1207/s15516709cog1402_1. URL http://doi.wiley.com/10.1207/s15516709cog1402_1. 37

[38] Yue Wang et al. Dynamic Graph CNN for Learning on Point Clouds, 2018. URL https://arxiv.org/abs/1801.07829. 39, 40

[39] Shah Rukh Qasim, Jan Kieseler, Yutaro Iiyama, and Maurizio Pierini. Learning representations of irregular particle-detector geometry with distance-weighted graph networks. *Eur. Phys. J. C*, 79(7):608, 2019. doi: 10.1140/epjc/s10052-019-7113-9. URL https://cds.cern.ch/record/2666193. 10p, v2: published version. 40

[40] Xiangyang Ju et al. Graph Neural Networks for Particle Reconstruction in High Energy Physics detectors, 2020. URL https://arxiv.org/abs/2003.11603. 40

[41] Fast Graph Representation Learning with PyTorch Geometric, May 2019. URL https://github.com/pyg-team/pytorch_geometric. original-date: 2017-10-06T16:03:03Z. 43

[42] PyTorch: An Imperative Style, High-Performance Deep Learning Library, 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf. original-date: 2016-08-13T05:26:41Z. 43

[43] F. Pedregosa et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. 45

[44] Ulrike von Luxburg. A Tutorial on Spectral Clustering. 2007. doi: 10.48550/ARXIV.0711.0189. URL https://arxiv.org/abs/0711.0189. 47

[45] Gaël Guennebaud, Benoît Jacob, et al. Eigen v3. http://eigen.tuxfamily.org, 2010. 57

[46] Spectra - C++ Library For Large Scale Eigenvalue Problems. URL https://spectralib.org/. 57