Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Scalable neural network models and terascale datasets for particle-flow reconstruction

Joosep Pata (joosep.pata@cern.ch)
National Institute of Chemical Physics and Biophysics https://orcid.org/0000-0002-5191-5759
Eric Wulff
CERN
Farouk Mokhtar
University of California San Diego
David Southwick
CERN
Mengke Zhang
University of California San Diego
Maria Girone
CERN
Javier Duarte
University of California San Diego

Article

Keywords:

Posted Date: November 16th, 2023

DOI: https://doi.org/10.21203/rs.3.rs-3466159/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: There is **NO** Competing Interest.

Scalable neural network models and terascale datasets for particle-flow reconstruction

Joosep Pata[1*], Eric Wulff[2], Farouk Mokhtar[3], David Southwick[2], Mengke Zhang[3], Maria Girone[2], Javier Duarte[3]

[1*]National Institute of Chemical Physics and Biophysics (NICPB), Rävala pst 10, 10143 Tallinn, Estonia.
[2]European Center for Nuclear Research (CERN), CH 1211, Geneva 23, Switzerland.
[3]University of California San Diego, La Jolla, CA 92093, USA.

*Corresponding author(s). E-mail(s): joosep.pata@cern.ch;
Contributing authors: eric.wulff@cern.ch; fmokhtar@ucsd.edu; david.southwick@cern.ch; mezhang@ucsd.edu; maria.girone@cern.ch; jduarte@ucsd.edu;

Abstract

We study scalable machine learning models for full event reconstruction in high-energy electron-positron collisions based on a highly granular detector simulation. Particle-flow (PF) reconstruction can be formulated as a supervised learning task using tracks and calorimeter clusters or hits. We compare a graph neural network and kernel-based transformer and demonstrate that both avoid quadratic memory allocation and computational cost while achieving realistic PF reconstruction. We show that hyperparameter tuning on a supercomputer significantly enhances the physics performance of the models, improving the jet transverse momentum resolution by up to 50% compared to the baseline. The resulting model is highly portable across hardware processors, supporting Nvidia, AMD, and Intel Habana cards. Finally, we demonstrate that the model can be trained on highly granular inputs consisting of tracks and calorimeter hits, resulting in a competitive physics performance with the baseline. Datasets and software to reproduce the studies are published following the findable, accessible, interoperable, and reusable (FAIR) principles.

1 Main

One of the main approaches for event reconstruction at the Large Hadron Collider (LHC) is currently based on the particle-flow (PF) algorithm [1–13], which combines measurements from different subdetectors to produce a holistic particle-based description of the entire event. Given the planned High Luminosity LHC (HL-LHC) program, as well as possible future experimental programs such as the Future Circular Collider (FCC), it is necessary to develop and evaluate a computationally efficient and physically optimal evolution of PF reconstruction.

Among various approaches, there has been considerable interest and development of machine learning (ML)-based reconstruction methods, including for global event description. To summarize recent developments, a recipe for a per-particle loss was given in Ref. [14]. In Ref. [15], a graph neural network (GNN)-based approach was demonstrated to reconstruct events with high particle multiplicity accurately. This approach was successfully applied in the Compact Muon Solenoid (CMS) experiment [16, 17]. Recently, Ref. [18], demonstrated that hypergraph networks can improve jet reconstruction. In parallel, clustering using ML has been demonstrated for high-granularity calorimeter reconstruction [19].

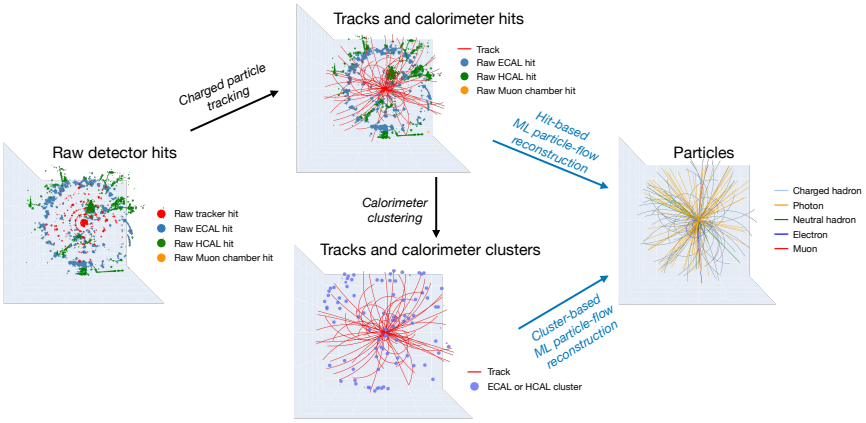

Fig. 1 A conceptual overview of the machine-learned particle flow (MLPF) approach based on tracks, hits and clusters.

One of the critical challenges for PF reconstruction is the highly granular nature of the data: events can comprise hundreds of thousands of heterogeneous measurements in various detector subsystems. This motivates studying models that can scale to large input multiplicities and efficiently process full events or batches of events. To support such studies, it is beneficial to establish open realistic simulated datasets with sufficient granularity to test various approaches.

In this paper, we describe an open dataset of electron-positron (e^+e^-) collision events at a center of mass energy $\sqrt{s} = 380\,\text{GeV}$ with full GEANT4 simulation, suitable for PF reconstruction, available in the EDM4HEP [20] format for future studies. Furthermore, we test two types of scalable ML models as benchmarks that

can process full events consisting of tens of thousands of measurements, while avoiding memory allocation or computations that scale quadratically with the input size. Using an input of charged particle tracks and clusters of calorimeter energy deposits, we minimize a particle-based loss function and monitor physics metrics that quantify event reconstruction performance during training. We report the results of an extensive hyperparameter optimization (HPO) of the GNN-based model performed using high performance computing (HPC) resources. We then evaluate the model's physics and computational performance, as well as its portability and quantization compatibility. We find that the GNN-based model can reconstruct physics events with a higher degree of fidelity compared to the baseline, and its computational cost scales linearly with the size of the input event. Moreover, we demonstrate the portability of the model to several computational hardware processors from Nvidia, AMD, and Intel Habana. The model demonstrates high-fidelity physics reconstruction performance when trained on a lower-level dataset consisting of tracks and raw calorimeter hits, providing further evidence that such heterogeneous and granular datasets can be reconstructed at scale using ML. Finally, we identify steps for future development and publish the code and datasets following the findable, accessible, interoperable, and reusable (FAIR) principles [21–23] for reproducibility and future development.

The setup follows the MLPF approach previously used in Refs. [15–17], and applies a physics-inspired ansatz to simplify the event-based reconstruction loss to a particle-based classification and regression loss. The approach is described in more detail in the Methods section.

We note that in the recent hypergraph-based reconstruction method [18], the strict particle-to-element association is completely avoided, and the associations are instead made an optimization target for an intermediate model. This hypergraph reconstruction approach shows excellent physics performance on small jet-based samples, but further work is needed to extend it to full events with large particle multiplicities and to demonstrate computational feasibility on realistic datasets.

We have not yet specified the specific neural network (NN) structure of the reconstruction model $\Phi(X)$. It could be implemented with a simple feedforward network over the individual feature vectors of each input element, i.e. $\phi(x) \to y'$. However, such a model would be unable to consider correlations between related input elements, such as tracks and calorimeter clusters (or hits) arising from a single particle.

An early approach proposed for full-event reconstruction previously used a GNN with a learnable graph structure between the input nodes [15]. This is conceptually similar to a transformer with full self-attention, which has also recently been thoroughly investigated [18].

However, it is well known that the evaluation of full self-attention can be computationally demanding and prohibitive for input multiplicities $\geq 10^4$. Such input multiplicities can be easily reached when considering all tracks and calorimeter clusters in future high luminosity collider events, or if calorimeter clustering is avoided by using tracks and calorimeter hits even in the case of a future lepton collider, as shown below. It is possible that the local nature of the problem can be exploited directly by first pre-partitioning the event and running the full attention-based model

only on subsets. However, this requires careful implementation to avoid artifacts from partitioning and subsequent stitching.

Here, we study models that can naturally scale to large input multiplicities by avoiding pairwise memory allocations or computation. Increasing the context lengths for ML models also receives considerable attention in the literature because of its wide range of applications [24, 25], and our approaches are based on existing research:

- The first model uses a dynamically learned graph structure [15], but avoids a full quadratic allocation or computation by using a learnable binning based on locality sensitive hashing (LSH) in each graph building layer [16, 17], inspired by the reformer architecture [26].
- The alternative model is a transformer in which the softmax self-attention layer is approximated using *fast attention via positive orthogonal random features* [27]. The complete model for PF reconstruction is then implemented based on layers of self-attention and feedforward networks, with the number of layers being a hyperparameter.

The models can be implemented using matrix operations in native TensorFlow.

Having defined NN models for PF reconstruction that avoid $\mathcal{O}(N^2)$ memory allocation and computational scaling while retaining the ability to treat large input sets and consider associations between all input elements, we train and test them on a realistic dataset. We generate e^+e^- collision events with Pythia8 (v8.306) [28] and carry out a complete detector simulation with GEANT4 (v11.0.2) using the `Key4HEP` framework (v2023-01-15) [29]. In particular, we use the Compact Linear Collider (CLIC) detector model [30, 31], along with the `Marlin` reconstruction code [32], and the `Pandora` package [33–35] for a baseline PF implementation. The CLIC detector model is chosen because it is publicly available, well documented and realistic.

The CLIC detector model is based on the CMS detector at CERN. It features a superconducting solenoid with an internal diameter of 7 m, providing a magnetic field of 4 T in the center of the detector. Silicon pixel and strip trackers, the electromagnetic (ECAL) and hadron calorimeters (HCAL) are embedded within the solenoid. Each subdetector is divided into a barrel and two endcap sections. The ECAL is a highly granular array of 40 layers of silicon sensors and tungsten plates. The HCAL is built from 60 layers of plastic scintillator tiles, read out by silicon photomultipliers, and steel absorber plates. The muon system surrounding the solenoid consists of 6 and 7 layers of resistive plate chambers interleaved with yoke steel plates in the endcap and barrel respectively. Two smaller electromagnetic calorimeters, LumiCal and BeamCal, cover the very forward region of the detector on either side of the interaction point [30, 31].

Collision events are generated with different physics processes to test the out-of-distribution performance of the model. In particular, we use Pythia8 to generate $\simeq 10^6$ $\mathrm{t\bar{t}}$, ZH, WW each, and $\simeq 2 \times 10^6$ $\mathrm{q\bar{q}}$ events. Furthermore, we generate $\simeq 7 \times 10^5$ of $\mathrm{t\bar{t}}$ PU10 events with a beam-induced hadronic overlay $\mathrm{gg} \to \mathrm{q\bar{q}}$ interactions, corresponding to an average of 10 additional interactions per event, known as pileup (PU), to test the stability under varying conditions. The datasets are augmented with single $e^{\pm}$, $\mu^{\pm}$, K^0_L, π^0, $\pi^{\pm}$, neutron, and photon particle gun samples, with a uniform energy distribution in $E \in [1, 100]$ GeV, generated using the ddsim package, $\simeq 10^6$

events each. The $q\bar{q}$, $t\bar{t}$ and single particle gun datasets were split in an 80/20 ratio to form train/test samples, while the ZH, WW and $t\bar{t}$ PU10 datasets were never used in training.

The datasets with generator particles; reconstructed tracks, hits and calorimeter clusters; as well as reconstructed particles from the baseline `Pandora` algorithm are saved in the `EDM4HEP` format, including all the relevant associations. Overall, the size of the dataset is approximately 2.5 TB before preprocessing to the ML-specific format using the TFDS library [36]. The raw datasets in `EDM4HEP` format, along with the scripts and configurations to generate the data, are available at [37].

There are about 50–150 tracks or calorimeter clusters per event, while the multiplicity for raw calorimeter hits is considerably larger at $5\text{–}15 \times 10^3$ per event. We analyze the datasets at the level of the tracks and calorimeter clusters to study the physics performance of models, while foregoing the `Pandora`-based clustering allows us to test the scalability of the model to high input multiplicities, as reconstructing events from raw hits is a much more challenging task with an *ab-initio* ML-based model. The input elements and target particles have different underlying signatures in different samples, depending on the physics process, therefore the models will have to demonstrate out-of-distribution generalization.

Jets are defined by clustering particles with the generalized k_T algorithm ($R = 0.7, p = -1$) for e^+e^- colliders [38, 39]. We also evaluate the $\vec{p}_\mathrm{T}^{\,\mathrm{miss}}$ and total 3D momentum of the reconstructed and generated particles. The missing transverse momentum vector $\vec{p}_\mathrm{T}^{\,\mathrm{miss}}$ is calculated as the negative vector sum of the transverse momenta of all particles in an event, and its magnitude is denoted as $p_\mathrm{T}^{\mathrm{miss}}$. We use these quantities to assess the performance of MLPF and the baseline PF reconstruction with respect to the ground truth defined by the generator particles.

The input features for the tracks consist of the track p_T, the pseudorapidity of the particle momentum (track tangent) η, the azimuthal angle of the particle momentum ϕ, the track goodness-of-fit χ^2 and the number of degrees of freedom N_dof, the slope of the track $\tan\lambda = p_z/\sqrt{p_x^2 + p_y^2}$, the signed impact parameter D_0 in the xy plane, the curvature of the track $\Omega = 1/R$, the position of the track along the z-axis Z_0 with respect to the reference point at $(0, 0, 0)$ [40]. For calorimeter clusters, the features consist of E_T, η, ϕ, E_ECAL, E_HCAL, Cartesian spatial coordinates x, y, z; the number of hits and the cluster size, measured in standard deviations of the hit sets in x, y, and z.

2 Results

An extensive HPO was performed for both the GNN- and transformer-based model using the JURECA supercomputer [41]. JURECA is a pre-exascale modular supercomputer operated by Jülich Supercomputing Centre at Forschungszentrum Jülich. The system consists of a flexible Data Centric (DC) module, based on the Atos BullSequana XH2000. It has, among others, 192 accelerated compute nodes with four NVIDIA A100 GPUs and two AMD EPYC 7742 CPUs each. We employ a similar approach to HPO as in Ref. [42], using Bayesian Optimization in combination with the ASHA [43] algorithm. The optimization was performed in a distributed manner on 96

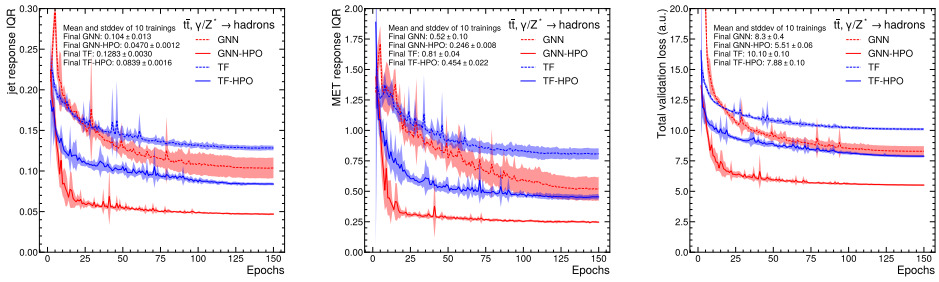

Fig. 2 The validation loss, jet and $p_{\mathrm{T}}^{\mathrm{miss}}$ resolution of the GNN model before and after hypertuning, evaluated using 10 trainings, each trained on four Nvidia A100 devices in a distributed data-parallel manner. We show the evolution of the validation loss as well as the jet and $p_{\mathrm{T}}^{\mathrm{miss}}$ resolutions during the full training process, where only the random seed differs in each run. The shaded regions show the standard deviation of the metrics across the runs.

GPUs spread over 24 compute nodes, consuming roughly 7000 and 5000 GPU hours for the optimization of the transformer and the GNN model respectively. The performance improvements achieved from HPO are presented in Fig. 2. We find that the optimized versions of both the GNN and the transformer significantly outperform the unoptimized versions. The optimized version of the GNN significantly outperforms the transformer, although both use a similar number of trainable parameters ($\simeq 5 \times 10^6$).

The final GNN model is trained separately in two different input modes, visualized on Fig. 1:

(1) using charged particle tracks and calorimeter clusters—this tests the physics performance of the model under a typical PF scenario, where a form of clustering is applied beforehand—and
(2) using charged particle tracks and raw calorimeter hits—this tests the scalability of the model in extreme cases of high input multiplicity when an explicit clustering step is omitted.

Due to the computational cost, the hyperparameters are identified using the cluster-based model (1) and reused for the hit-based model (2). We plan to follow up on hypertuning of model (2) in a subsequent work. The relevant evaluation criterion for the model is the reconstruction of jets and $p_{\mathrm{T}}^{\mathrm{miss}}$ compatible with those at the generator level, a target that is not explicitly trained for, but can be reached by minimizing the particle-level loss. The physics performance in event-level quantities is summarized in Fig. 3.

We use ZH, WW and $\mathrm{t\bar{t}}$ PU10 events to evaluate out-of-distribution performance. We evaluate the jet response by clustering reconstructed particles into jets as described above, by matching the reconstructed and generator-level jets, and computing the ratio of reconstructed to generator-level jet p_{T}. In all samples, the fraction of reconstructed jets from MLPF is the same or higher than for PF, with generally an improved jet response width, quantified by the interquartile range (IQR) and median compared to the rule-based baseline. This improvement over the baseline was not observed before hyperparameter tuning. We also evaluate the total 3-momentum response for either

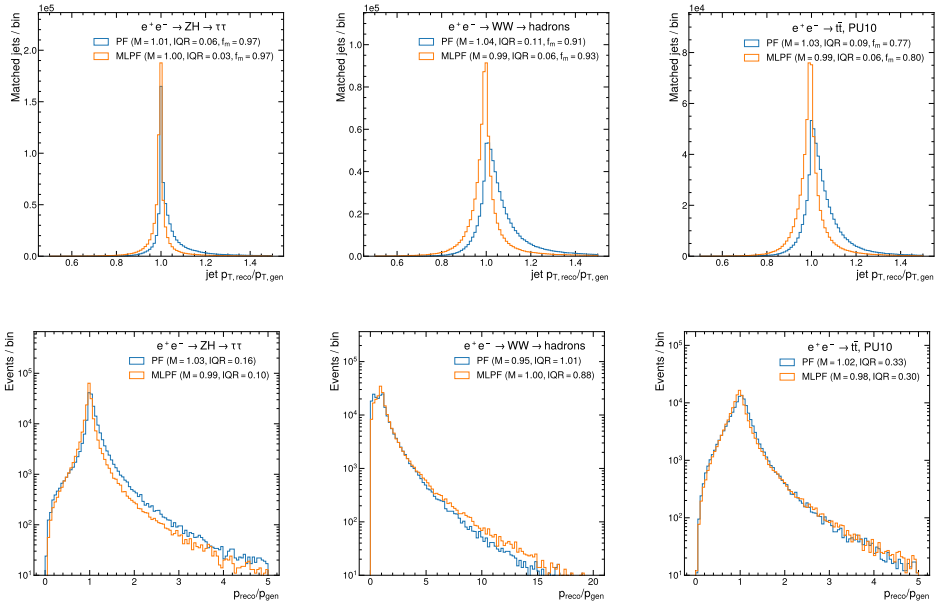

Fig. 3 Overall jet (top) and total 3-momentum (bottom) response in the ZH, WW and $t\bar{t}$ PU10 validation samples that were never used in training.

PF or MLPF particles, and find that the MLPF model improves both the median and IQR of the total 3-momentum response distributions for all samples.

To compare the momentum resolutions between the different approaches, while accounting for the differences in the momentum scales, we evaluate the metric of the IQR divided by the median of the response distributions. We quantify the evolution of this metric for the jet p_T (total event 3-momentum) in bins of generator-level jet p_T (generated total event 3-momentum) on the $t\bar{t}$ with 10 PU interactions ($t\bar{t}$ PU10) sample in Fig. 4. The baseline PF and proposed MLPF algorithms behave qualitatively similarly, with improved response IQR over median values at higher generator-level jet p_T (total 3-momentum), while the MLPF algorithm consistently outperforms the baseline PF on this sample by up to 50%.

We train version (2) of the model with significantly higher input element multiplicity, where each event consists of $\simeq 10^4$ tracks and raw calorimeter hits. This is to test the computational scalability of the approach, and to study the physics performance using lower-level inputs. To improve the physics response, single-particle gun samples are now included in the training set. The training is carried out on a single node at the LUMI HPC center on four AMD MI250X devices with two accelerator processors each. The model is currently trained for 10 epochs with the same procedure as described above, with the results shown in Fig. 5.

Throughout the training, despite using a much lower level training input, the jet and $p_\mathrm{T}^\mathrm{miss}$ IQR and median trend towards the optimal values of 0 and 1 respectively. In Fig. 5, we see that the hit-based MLPF has comparable, currently slightly worse

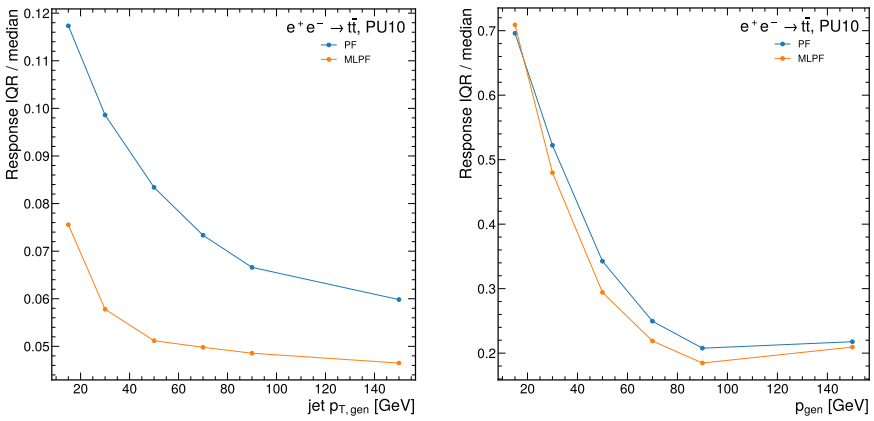

Fig. 4 The response IQR divided by the median for the jet p_T (left) and total event 3-momentum (right) in the $\mathrm{t\bar{t}}$ PU10 sample. This metric is evaluated in bins of the generator-level jet p_T (left) and generated total event 3-momentum (right).

reconstruction performance compared to the baseline PF when using raw calorimeter hits. This can be attributed to not using an optimized set of hyperparameters, and not training for a sufficient number of epochs—both limited by computational throughput of the training and motivating further research effort into high-performance training.

An important factor in the development of scalable ML-based full event reconstruction models is improving the computational throughput in future deployment scenarios. Therefore, we compare the inference scalability of the rule-based PF implementation on CPU and the ML-based implementation on GPU with respect to increasing input multiplicity, with the results summarized in Fig. 6. Inference scaling tests of the GNN model are tested on a small, 8 GB consumer GPU (Nvidia RTX2070S) to emulate a resource-constrained scenario in edge deployment.

The rule-based version is run with 25, 50, 100 and 200 generated π^- particles per event for 100 events, three times each. The relative runtime of the particle-flow module in `Key4HEP` is measured, and compared with the 25-particle case. The runtime of the baseline increases nonlinearly with increasing particle multiplicity, and memory segmentation faults occur for more than approximately 200 particles per event.

The GNN model is run with a varying input size on an 8 GB consumer GPU multiple times to average over random fluctuations. Both the batch size B, i.e., the number of batched events ($B \in \{1, 2, 4, 8, 16\}$) and the event size N, i.e., the number of input elements per event ($N = 256n$ for $n \in [1, 20]$) are varied independently. The relative runtime with respect to the $N = 256, B = 16$ case scales approximately linearly with N. We note that $B \gg 1$ is required to saturate the GPU, but this is highly model and device specific.

We confirm that the ML-based reconstruction based on the LSH-binned GNN for full event reconstruction avoids the quadratic scaling present in a typical rule-based full event reconstruction algorithm. It is likely that additional effort will be able to significantly improve the performance of both algorithms. For example, in CMS, a

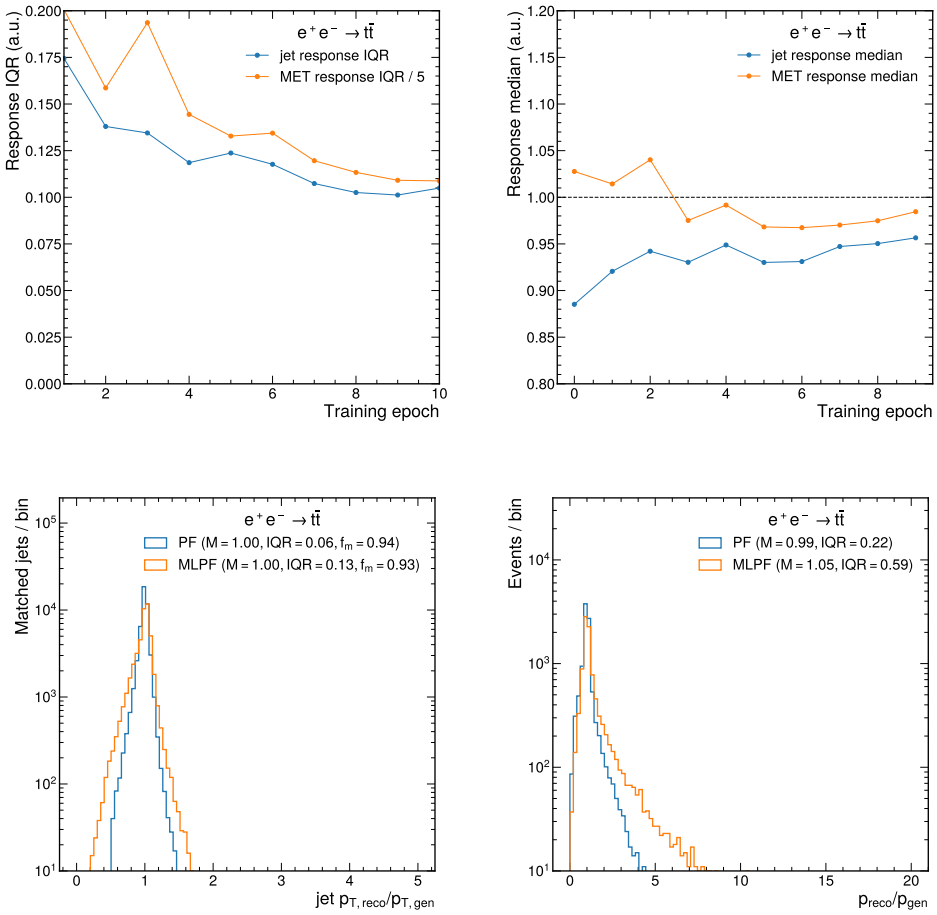

Fig. 5 The evolution of the IQR and median of the jet and $p_{\mathrm{T}}^{\mathrm{miss}}$ response in the $\mathrm{t\bar{t}}$ sample, using tracks and raw calorimeter hits as input, trained on eight AMD MI250X accelerator chips on the LUMI HPC. Also shown are the final jet and total 3-momentum response distributions after 10 epochs of training.

k-dimensional tree [44] is used to avoid quadratic scaling [11]. With aggressive quantization, the throughput of ML models can be significantly improved with a negligible performance degradation [45].

At this point, it is not meaningful to compare the absolute throughput of the rule-based model on a CPU and the ML-based model on a GPU, as neither method is particularly optimized for throughput, and the comparison is strongly affected by the specific choice of hardware.

The training scalability is tested, with results presented in the rightmost plot in Fig. 6, on three different HPC centers with different accelerator hardware: Nvidia H100 GPUs from Flatiron Institute's CoreSite cluster [46], AMD MI250 GPUs from

the LUMI supercomputer [47], and Intel Habana Gaudi HPUs from the Voyager supercomputer [48]. The LUMI supercomputer features GPU nodes with 64-core AMD Trento CPUs and four AMD MI250X cards, each card consisting of two accelerator chips. Voyager is an NSF-funded supercomputer with 42 first-generation Intel Habana Gaudi (training) nodes, each with eight cards, two first-generation Intel Habana Goya (inference) nodes, a 400 GbE Arista switch, and 3 PB of Ceph file system available at the San Diego Supercomputer Center located at the University of California San Diego. Intel Habana also provided additional access to eight Gaudi2 nodes in an HLS-Gaudi2 Deep Learning Server [49]. For the training tests, there are some differences in the configuration on the HPCs. For the AMD processors, multi-card training was implemented with a mirrored worker configuration, while for the Nvidia and Habana processors, Horovod [50] was used. Events were zero-padded to a regular size of 512 elements per events. A batch size of 250 events per device was used for the Nvidia, AMD, and Habana Gaudi2 processors, while a batch size of 100 per device was used for the Habana Gaudi processors. We observe nearly linear scaling or better for all processors. The scaling for the Habana Gaudi1 (Gaudi2) processors is enhanced by the all-to-all non-blocking intra-node network connection, where each processor has a 100 (300) Gb network connection to every other processor [48, 49].

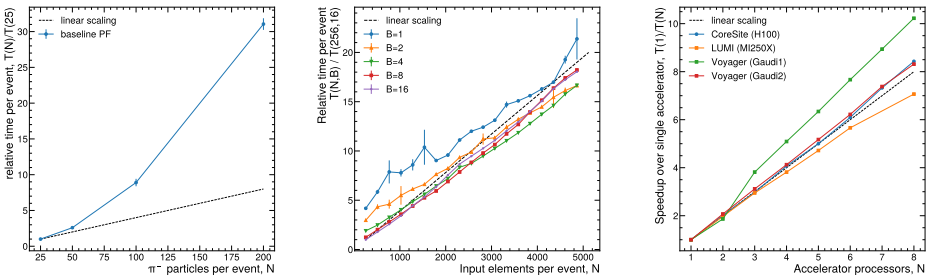

Fig. 6 Relative timing of the baseline PF (left) and MLPF algorithms (middle), illustrating the scaling with respect to input particle or element (track, cluster, or hit) multiplicity. The absolute processing time of the baseline algorithm on a single CPU thread is approximately 1 s / event at the reference point of 100 charged pions, which corresponds to approximately 96 ± 3 tracks and 170 ± 20 clusters. The runtime of the MLPF algorithm on the RTX2060S at the reference point of $N = 256, B = 16$ is 2 ms / event. This should not be interpreted as a complete, exhaustive and final computational benchmark, nor should it be used to claim 500-fold improvement in throughput from MLPF, as the absolute timing of any algorithm is heavily dependent on optimizations and hardware. In [17], the runtime of a heavily optimized version of the baseline algorithm was measured at 9ms / event, whereas the MLPF model at 320ms / event, both on a single CPU thread. We also demonstrate the scaling of the training performance across multiple devices on a single machine (right) on Nvidia, AMD, and Habana processor cards from the CoreSite, LUMI, and Voyager supercomputers, respectively. For the multi-device scaling test, Horovod was used on CoreSite and Voyager, while the TensorFlow `MirroredStrategy` was used on LUMI. The batch size was adjusted to fit a single device and the dataset was fully cached in RAM.

3 Discussion

In this paper, we proposed a new multi-terabyte dataset of high-energy electron-positron collisions in a future detector concept, simulated with GEANT4 and stored in EDM4HEP format, featuring increased realism and granularity for particle-flow (PF) reconstruction studies. We stressed the question of model scalability on large-scale datasets, and investigated and compared two alternative models featuring approximately linear scaling. Through extensive hyperparameter tuning, we found that a graph neural network (GNN)-based model outperforms a transformer-based one, as well as the rule-based baseline PF algorithm. We evaluated the physics performance with event-level metrics and found that both the interquartile range (IQR) and median of the jet and total 3-momentum response distributions are improved by the GNN model. In particular, the jet p_{T} resolution, quantified by the response IQR divided by the median, improved by up to 50%. This improvement has important implications for the sensitivity of key measurements at future colliders, such as those involving Higgs bosons that decay to bottom quark-antiquark pairs [51].

We demonstrated that the model supports and benefits from mixed-precision training. Furthermore, we find that the runtime of the GNN-based model scales approximately linearly with increasing input size, while the rule-based baseline PF algorithm scales nonlinearly. Finally, we demonstrated that training the GNN model on a dataset consisting of tracks and raw calorimeter hits is feasible and results in physics performance comparable to the baseline, confirming that explicit clustering can in principle be avoided in favor of direct particle reconstruction. This training, carried out on the LUMI HPC featuring AMD GPUs, and other scaling tests done on Nvidia GPUs and Intel Habana processing units demonstrate effective portability across vendors.

Further research is possible in several directions. First, it would be useful to repeat this exercise using the simulation from detectors that are taking data in Run 3 of the Large Hadron Collider (LHC), e.g. CMS, to study the performance of the reconstruction in real data. Second, we are currently using a simple particle-based loss, while the use of contrastive-adversarial learning methods may allow to account for event-level discrepancies more effectively [52]. Third, the hypergraph model [18] shows promising physics performance but currently only supports small input sequences. It may be interesting to extend the hypergraph construction over dynamically binned events using the locality sensitive hashing (LSH) approach. It may be interesting to construct features using semi-supervised learning from real data, to reduce the reliance on simulated datasets for supervised learning. Furthermore, large-context models are continuously improving, and it may be interesting to apply the latest developments such as FlashAttention [53, 54] on the models in this paper. For improving throughput on e.g. CPUs, quantization has shown promise, and it may be interesting to investigate if this can be repeated for the type of models proposed for PF reconstruction. Finally, it is important to integrate the proposed machine learning (ML)-based reconstruction models into reconstruction frameworks such as CMSSW and Key4HEP.

4 Methods

The input to the model is a set of input detector elements: charged tracks and calorimeter clusters (or alternatively, raw calorimeter hits), each described by a feature vector $\mathbf{x} \in X$. The target of the model is a set of stable particles, each described by a particle type, charge, and continuous four-momentum values, combined to form a target feature vector $\mathbf{y} \in Y$. The goal of the model is, for each event, to reconstruct the set Y, given the set X, i.e. $\Phi(X) \to Y' \simeq Y$. To make the set-to-set translation tractable with ML, and to be able to treat particle multiplicities in the range of $|X|, |Y| \simeq 10^4$ with a single model pass, each target particle is associated with a single unique input element. The target particles are assigned to input elements by an injective, non-surjective function based on a physics prior. The general approach is based on the object condensation (OC) [14], which includes a potential between inputs and outputs. Here, we only associate charged particles to tracks and neutral particles to the highest-energy calorimeter cluster (or hit), as a tradeoff between expressiveness and computational cost. The event reconstruction loss can be written as a sum over the input elements in each event

$$L(Y, Y') = \sum_i L_{\text{cls}}(y_i, y_i') + L_{\text{reg}}(y_i, y_i'). \tag{1}$$

Here, $L_{\text{cls}}(y_i, y_i')$ is a classification loss between the predicted and target particle type and charge labels. The classification task is imbalanced, so we use the focal loss for particle type classification, which assigns greater weight to samples that are difficult to classify. Similarly, $L_{\text{reg}}(y_i, y_i')$ is a regression loss for the momentum components, where we use the Huber loss to reduce the effect of outliers.

The LSH-based GNN is constructed as follows. We define a graph building layer, which receives a set of input feature vectors $\mathbf{x} \in X$, assigns each input vector to a bin with a learnable function $f(\mathbf{x}) \to b \in B$, such that the number of bins is much smaller than the input multiplicity $|B| \ll |X|$. In each bin of average size $N_B = |X|/|B|$, a fully connected graph is built between all elements of that bin using a learnable function $g(x_i, x_j) \to w_{ij}$. Therefore, the output of the graph building layer is a learnable association matrix $W = w_{bij}$ of size $|B|N_B^2$, avoiding the allocation of a matrix of size $|X|^2$. This matrix can be used for further GNN operations, such as message passing. The graph structure defined by the association matrix consists of $|B|$ disjoint graphs of N_B elements each, which is not a major limitation because the graph building layers can be stacked multiple times, each building different disjoint graphs. From a physics point of view, it is reasonable to expect that nearby (for some learnable definition of neighborhood) inputs should have a stronger association than input elements in highly separated parts of the detector. This graph building layer can be implemented using elementary matrix operations in a fully batched and differentiable way using TensorFlow [55, 56]. This is similar to manually partitioning and restitching the event, but achieved directly in the neural network (NN) model using fully differentiable operations, rather than with a manual heuristic. The GNN is based on stacked layers of graph building and convolution, with the number of layers being a configurable hyperparameter.

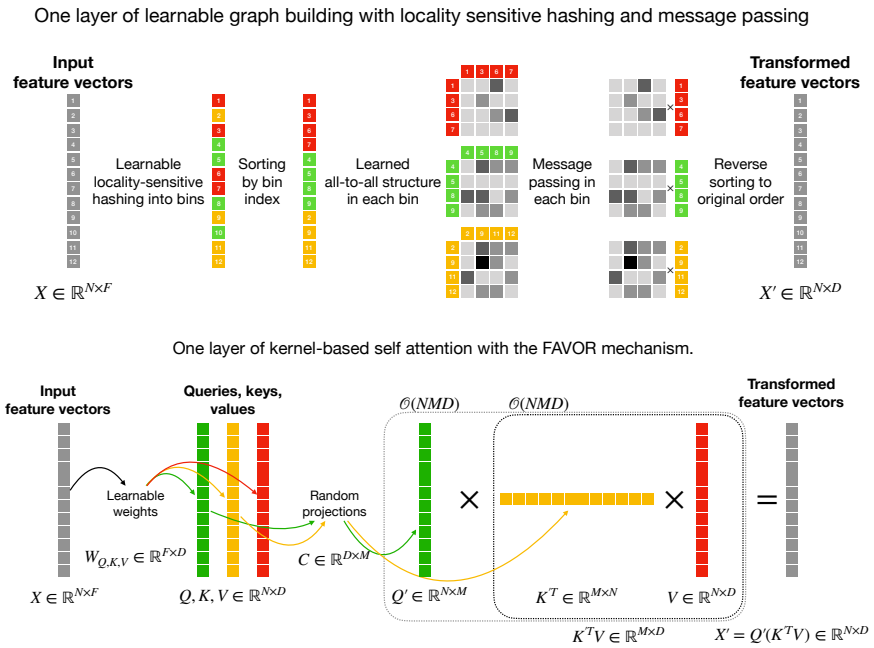

Fig. 7 Top: one layer of the LSH-based GNN. Bottom: one layer of the kernel-based self-attention, figure adapted from [27].

The alternative kernel-based transformer model avoids quadratic scaling using the following approach. For N elements, given queries $Q \in \mathbb{R}^{N\times d_q}$ and keys $K \in \mathbb{R}^{N\times d_k}$, the attention mechanism encodes a value matrix $V \in \mathbb{R}^{N\times d_v}$ as

$$\mathrm{Attn}(Q,K,V) = \mathrm{softmax}\left(\frac{QK^\intercal}{\sqrt{d_k}}\right)V. \tag{2}$$

Here, the $\mathrm{softmax}(QK^\intercal)$ operation creates a full $N\times N$ matrix. As in Ref. [27], we define a transformation $\psi(\mathbf{x}) \to \mathbf{x}'$ that transforms an input feature map $\mathbf{x}$ using predetermined random projections to a new feature space $\mathbf{x}'$. For a sufficiently large number of random projections, attention can be approximated as

$$\mathrm{Attn}(Q,K,V) \simeq Q'(K'^\intercal V) \tag{3}$$

where Q' and K' are the query and key matrices after the random feature mapping ψ, respectively. Allocation of the entire $N\times N$ matrix is avoided, since the order of operations is changed to first multiply keys with values and then subsequently with queries. In the special case of self-attention, Q, K, and V are all derived from X through a linear layer, and the self-attention mechanism can be seen as an analogy to graph building and message passing. A visual overview of the supervised learning setup, as well as the two alternative model configurations, is shown in Fig. 7.

Several steps were taken to ensure efficient training on large-scale datasets. All relevant matrix operations for the GNN implementation support efficient batched application using native TensorFlow kernels. Given the varying size of events, the model implements variable-sized batching, supporting batching events at fixed batch size increments, with the batch size inversely proportional to the size of the events in the batch. This was necessary to reduce the amount of zero-padding, and to enable automatic kernel compilation, so that a limited number of kernels have to be compiled for each bucket size increment. Since the total size of the datasets can reach several terabytes, efficient larger-than-RAM training is enabled through dynamic dataset loading and interleaving using the TensorFlow datasets library. The model supports mixed-precision training in bfloat16 and float16. The training in bfloat16 results in loss values that are stable and largely compatible with that of float32, while we find that the dynamic loss scaling for float16 results in NaN gradients and does not converge. At the time of writing, software support for bfloat16 in TensorFlow is limited, such that the operations are not placed on tensor cores and the use of bfloat16 does not result in increased training throughput, while training with float16 results in moderate speedups of 30–40% for the GNN model. Improving training throughput using bfloat16 is the focus of future work.

Data and software availability

The datasets are available in [37], the results in [57], and the software used for analysis in [58].

Authorship statement

The authors contributed according to the contributor roles taxonomy (CRediT) categories as follows. **Joosep Pata**: conceptualization, methodology, software, validation, data curation, writing (original draft), funding acquisition, resources. **Eric Wulff**: methodology, software, data curation, writing (review and editing), resources. **Farouk Mokhtar**: methodology, software, investigation, supervision. **David Southwick**: software, investigation. **Mengke Zhang**: validation, investigation. **Maria Girone**: funding acquisition, resources. **Javier Duarte**: conceptualization, software, resources, funding acquisition, supervision, writing (review and editing).

Acknowledgements

Eric Wulff and David Southwick were supported by CoE RAISE. The CoE RAISE project has received funding from the European Union's Horizon 2020 – Research and Innovation Framework Programme H2020-INFRAEDI-2019-1 under grant agreement no. 951733.

Joosep Pata was supported by the grant PSG864 of the Estonian Research Council.

Javier Duarte, Farouk Mokhtar, and Mengke Zhang were supported by U.S. Department of Energy (DOE) Award Nos. DE-SC0021187 and DE-SC0021396 (FAIR4HEP) and U.S. National Science Foundation (NSF) Cooperative Agreement

OAC-2117997 (A3D3). Farouk Mokhtar was also supported by a UCSD HDSI fellowship and an IRIS-HEP fellowship through NSF Cooperative Agreement OAC-1836650. The use of Intel Habana processors and additional support for Javier Duarte was facilitated by NSF Award No. 2005369 (Voyager). We also thank Amit Majumdar, Mahidhar Tatineni, Paul Rodriguez, and Marty Kandes at the San Diego Supercomputer Center and Leonid Goldgeisser, Charles Chen, Chen Levkovich, Leon Si Tran, and Chenna Bayapureddy at Intel Habana for technical support and assistance in producing results for the Gaudi2 processors.

The authors gratefully acknowledge the computing time granted through JARA on the supercomputer JURECA [41] at Forschungszentrum Jülich. We acknowledge the Estonian Scientific Compute Infrastructure and NICPB for awarding this project access to the LUMI supercomputer, owned by the EuroHPC Joint Undertaking, hosted by CSC (Finland) and the LUMI consortium through Estonian Scientific Compute Infrastructure.

Datasets and software published following the findable, accessible, interoperable, and reusable (FAIR) principles support open, transparent, and reusable research, and therefore we thank our colleagues at the relevant collaborations for releasing the `Key4HEP`, `Marlin`, `EDM4HEP` and `CLICdp` software to the wider community, and encourage the wider adoption of the FAIR principles.

References

[1] CELLO Collaboration, "An analysis of the charged and neutral energy flow in e^+e^- hadronic annihilation at 34 GeV, and a determination of the QCD effective coupling constant", *Phys. Lett. B* **113** (1982) 427, doi:10.1016/0370-2693(82)90778-X.

[2] ALEPH Collaboration, "Performance of the ALEPH detector at LEP", *Nucl. Instrum. Meth. A* **360** (1995) 481, doi:10.1016/0168-9002(95)00138-7.

[3] DELPHI Collaboration, "Performance of the DELPHI detector", *Nucl. Instrum. Meth. A* **378** (1996) 57, doi:10.1016/0168-9002(96)00463-9.

[4] ZEUS Collaboration, "Measurement of the diffractive structure function F2(D(4)) at HERA", *Eur. Phys. J. C* **1** (1998) 81, doi:10.1007/s100520050063, arXiv:hep-ex/9709021.

[5] ZEUS Collaboration, "Measurement of the diffractive cross-section in deep inelastic scattering using ZEUS 1994 data", *Eur. Phys. J. C* **6** (1999) 43, doi:10.1007/PL00021606, arXiv:hep-ex/9807010.

[6] A. Bocci, S. Lami, S. Kuhlmann, and G. Latino, "Study of jet energy resolution at CDF", *Int. J. Mod. Phys. A* **16S1A** (2001) 255, doi:10.1142/S0217751X01006632.

[7] A. L. Connolly, "A Search for Supersymmetric Higgs Bosons in the Di-tau Decay Mode in $p\bar{p}$ Collisions at 1.8 TeV". PhD thesis, UC Berkeley, 2003. doi:10.2172/

15017134.

[8] CDF Collaboration, "Measurement of $\sigma(p\bar{p} \to Z).\mathrm{Br}(Z \to 2\tau)$ in $p\bar{p}$ collisions at $\sqrt{s} = 1.96$ TeV", *Phys. Rev. D* **75** (2007) 092004, doi:10.1103/PhysRevD.75.092004.

[9] D0 Collaboration, "Measurement of $\sigma(p\bar{p} \to Z + X)$ $\mathrm{Br}(Z \to \tau^+\tau^-)$ at $\sqrt{s} = 1.96\,\mathrm{TeV}$", *Phys. Lett. B* **670** (2009) 292, doi:10.1016/j.physletb.2008.11.010, arXiv:0808.1306.

[10] CMS Collaboration, "The CMS Experiment at the CERN LHC", *JINST* **3** (2008) S08004, doi:10.1088/1748-0221/3/08/S08004.

[11] CMS Collaboration, "Particle-flow reconstruction and global event description with the CMS detector", *JINST* **12** (2017), no. 10, P10003, doi:10.1088/1748-0221/12/10/P10003, arXiv:1706.04965.

[12] ATLAS Collaboration, "Jet reconstruction and performance using particle flow with the ATLAS detector", *Eur. Phys. J. C* **77** (2017) 466, doi:10.1140/epjc/s10052-017-5031-2, arXiv:1703.10485.

[13] H1 Collaboration, "Measurement of charged particle multiplicity distributions in DIS at HERA and its implication to entanglement entropy of partons", *Eur. Phys. J. C* **81** (2021), no. 3, 212, doi:10.1140/epjc/s10052-021-08896-1, arXiv:2011.01812.

[14] J. Kieseler, "Object condensation: one-stage grid-free multi-object reconstruction in physics detectors, graph and image data", *Eur. Phys. J. C* **80** (2020), no. 9, 886, doi:10.1140/epjc/s10052-020-08461-2, arXiv:2002.03605.

[15] J. Pata et al., "MLPF: Efficient machine-learned particle-flow reconstruction using graph neural networks", *Eur. Phys. J. C* **81** (2021), no. 5, 381, doi:10.1140/epjc/s10052-021-09158-w, arXiv:2101.08578.

[16] CMS Collaboration, J. Pata et al., "Machine Learning for Particle Flow Reconstruction at CMS", in *20th International Workshop on Advanced Computing and Analysis Techniques in Physics Research*, L. Teodorescu, J. Goh, E. Brondolin, and J. Ngadiuba, eds., volume 2438, p. 012100. 2023. arXiv:2203.00330. doi:10.1088/1742-6596/2438/1/012100.

[17] F. Mokhtar et al., "Progress towards an improved particle flow algorithm at CMS with machine learning", in *21th International Workshop on Advanced Computing and Analysis Techniques in Physics Research: AI meets Reality*. 2023. arXiv:2303.17657. doi:10.48550/arXiv.2303.17657.

[18] F. A. Di Bello et al., "Reconstructing particles in jets using set transformer and hypergraph prediction networks", *Eur. Phys. J. C* **83** (2023), no. 7, 596, doi:10.

1140/epjc/s10052-023-11677-7, arXiv:2212.01328.

[19] CMS Collaboration, S. Bhattacharya et al., "GNN-based end-to-end reconstruction in the CMS Phase 2 High-Granularity Calorimeter", in *20th International Workshop on Advanced Computing and Analysis Techniques in Physics Research*, L. Teodorescu, J. Goh, E. Brondolin, and J. Ngadiuba, eds., volume 2438, p. 012090. 2023. arXiv:2203.01189. doi:10.1088/1742-6596/2438/1/012090.

[20] F. Gaede et al., "Edm4hep and podio-the event data model of the key4hep project and its implementation", in *Eur. Phys. J. Web Conf.*, C. Biscarat et al., eds., volume 251, p. 03026. 2021. doi:10.1051/epjconf/202125103026.

[21] M. D. Wilkinson et al., "The FAIR guiding principles for scientific data management and stewardship", *Sci. Data* **3** (2016) 160018, doi:10.1038/sdata.2016.18.

[22] Y. Chen et al., "A FAIR and AI-ready Higgs boson decay dataset", *Sci. Data* **9** (2022) 31, doi:10.1038/s41597-021-01109-0, arXiv:2108.02214.

[23] J. Duarte et al., "FAIR AI Models in High Energy Physics", 2022. arXiv:2212.05081. Submitted to *Mach. Learn.: Sci. Technol.*

[24] T. Brownet al., "Language models are few-shot learners", in *Advances in Neural Information Processing Systems*, H. Larochelle et al., eds., volume 33, p. 1877. Curran Associates, Inc., 2020. arXiv:2005.14165.

[25] H. Touvron et al., "LLaMA: Open and efficient foundation language models", 2023. arXiv:2302.13971.

[26] N. Kitaev, Łukasz Kaiser, and A. Levskaya, "Reformer: The efficient transformer", in *ICLR 2020*. 2020. arXiv:2001.04451.

[27] K. Choromanski et al., "Rethinking attention with performers", in *ICLR 2021*. 2020. arXiv:2009.14794.

[28] C. Bierlich et al., "A comprehensive guide to the physics and usage of PYTHIA 8.3", *SciPost Phys. Codeb.* (2022) doi:10.21468/SciPostPhysCodeb.8, arXiv:2203.11601.

[29] G. Ganis, C. Helsens, and V. Völkl, "Key4hep, a framework for future HEP experiments and its use in FCC", *Eur. Phys. J. Plus* **137** (2022), no. 1, 149, doi:10.1140/epjp/s13360-021-02213-1, arXiv:2111.09874.

[30] CLIC Collaboration, "CLICdet: The post-CDR CLIC detector model", CLIC Detector and Physics Study Note, 2017.

[31] CLICdp Collaboration, "A detector for CLIC: main parameters and performance", 2018. arXiv:1812.07337.

[32] F. Gaede, "Marlin and LCCD: Software tools for the ILC", *Nucl. Instrum. Meth. A* **559** (2006) 177, doi:10.1016/j.nima.2005.11.138.

[33] J. S. Marshall and M. A. Thomson, "The Pandora software development kit for particle flow calorimetry", *J. Phys. Conf. Ser.* **396** (2012) 022034, doi:10.1088/1742-6596/396/2/022034.

[34] J. S. Marshall, A. Münnich, and M. A. Thomson, "Performance of particle flow calorimetry at CLIC", *Nucl. Instrum. Meth. A* **700** (2013) 153, doi:10.1016/j.nima.2012.10.038, arXiv:1209.4039.

[35] J. S. Marshall and M. A. Thomson, "The Pandora software development kit for pattern recognition", *Eur. Phys. J. C* **75** (2015) 439, doi:10.1140/epjc/s10052-015-3659-3, arXiv:1506.05348.

[36] "TensorFlow Datasets, a collection of ready-to-use datasets". https://www.tensorflow.org/datasets.

[37] J. Pata et al., "Simulated datasets for detector and particle flow reconstruction: CLIC detector", 2023. doi:10.5281/zenodo.8260741.

[38] M. Cacciari, G. P. Salam, and G. Soyez, "The anti-k_{T} jet clustering algorithm", *JHEP* **04** (2008) 063, doi:10.1088/1126-6708/2008/04/063, arXiv:0802.1189.

[39] M. Cacciari, G. P. Salam, and G. Soyez, "FastJet user manual", *Eur. Phys. J. C* **72** (2012) 1896, doi:10.1140/epjc/s10052-012-1896-2, arXiv:1111.6097.

[40] T. Kramer, "Track parameters in LCIO", Technical Report LC-DET-2006-004, 2006. doi:10.3204/PHPPUBDB-2955.

[41] Jülich Supercomputing Centre, "JURECA: Data Centric and Booster Modules implementing the Modular Supercomputing Architecture at Jülich Supercomputing Centre", *J. Large-Scale Res. Facilities* **7** (2021), no. A182, doi:10.17815/jlsrf-7-182.

[42] E. Wulff, M. Girone, and J. Pata, "Hyperparameter optimization of data-driven AI models on HPC systems", *J. Phys. Conf. Ser.* **2438** (2023), no. 1, 012092, doi:10.1088/1742-6596/2438/1/012092, arXiv:2203.01112.

[43] L. Liet al., "A system for massively parallel hyperparameter tuning", in *Proceedings of Machine Learning and Systems*, I. Dhillon, D. Papailiopoulos, and V. Sze, eds., volume 2, p. 230. 2020. arXiv:1810.05934.

[44] J. L. Bentley, "Multidimensional binary search trees used for associative searching", *Commun. ACM* **18** (1975) 509, doi:10.1145/361002.361007.

[45] T. Dettmers, M. Lewis, Y. Belkada, and L. Zettlemoyer, "GPT3.int8(): 8-bit matrix multiplication for transformers at scale", in *Advances in Neural Information Processing Systems*, S. Koyejo et al., eds., volume 35, p. 30318. Curran Associates, Inc., 2022. arXiv:2208.07339.

[46] Flatiron Institute, "CoreSite Cluster", 2023. https://www.simonsfoundation.org/2022/11/14/new-flatiron-institute-supercomputer-the-most-power-efficient-ever-built/.

[47] LUMI Consortium, "LUMI Supercomputer", 2023. https://www.lumi-supercomputer.eu/.

[48] R. E. Amaro et al., "Voyager – An Innovative Computational Resource for Artificial Intelligence & Machine Learning Applications in Science and Engineering", in *Practice and Experience in Advanced Research Computing*, R. Sinkovits et al., eds., PEARC '23, p. 278. Association for Computing Machinery, 2023. doi:10.1145/3569951.3597597.

[49] Intel Habana, "HLS-Gaudi2 Deep Learning Server", 2022. https://habana.,ai/wp-content/uploads/2022/09/HLS-Gaudi2-Datasheet-Aug-2022.pdf.

[50] A. Sergeev and M. D. Balso, "Horovod: fast and easy distributed deep learning in TensorFlow", 2018. arXiv:1802.05799.

[51] S. Dawson et al., "Report of the Topical Group on Higgs Physics for Snowmass 2021: The Case for Precision Higgs Physics", in *Snowmass 2021*. 9, 2022. arXiv:2209.07510.

[52] T. Huang et al., "Learning to measure the point cloud reconstruction loss in a representation space", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12208–12217. 2023. doi:10.1109/CVPR52729.2023.01175.

[53] T. Daoet al., "FlashAttention: Fast and memory-efficient exact attention with IO-awareness", in *Advances in Neural Information Processing Systems*, S. Koyejo et al., eds., volume 35, p. 16344. Curran Associates, Inc., 2022. arXiv:2205.14135.

[54] T. Dao, "FlashAttention-2: Faster attention with better parallelism and work partitioning", 2023. arXiv:2307.08691.

[55] M. Abadi et al., "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems", 2015. arXiv:1603.04467.

[56] TensorFlow Developers, "TensorFlow", 2023. doi:10.5281/zenodo.7753622.

[57] J. Pata et al., "MLPF results on the simulated CLIC dataset, v2023.10", September, 2023. doi:10.5281/zenodo.10006037.

[58] J. Pata et al., "jpata/particleflow: MLPF training with CLIC simulation", August, 2023. doi:10.5281/zenodo.8397954.