Particle Graph Autoencoders and Differentiable, Learned Energy Mover's Distance

Steven Tsan, Raghav Kansal, Anthony Aportela, Daniel Diaz,
Javier Duarte, Sukanya Krishna, Farouk Mokhtar
University of California San Diego
La Jolla, CA 92093, USA

Jean-Roch Vlimant
California Institute of Technology
Pasadena, CA 91125, USA

Maurizio Pierini
European Organization for Nuclear Research (CERN)
CH-1211 Geneva 23, Switzerland

Abstract

Autoencoders have useful applications in high energy physics in anomaly detection, particularly for jets—collimated showers of particles produced in collisions such as those at the CERN Large Hadron Collider. We explore the use of graph-based autoencoders, which operate on jets in their "particle cloud" representations and can leverage the interdependencies among the particles within a jet, for such tasks. Additionally, we develop a differentiable approximation to the energy mover's distance via a graph neural network, which may subsequently be used as a reconstruction loss function for autoencoders.

1 Introduction

One of the primary motivations behind the development of the CERN Large Hadron Collider (LHC) was to search for new physics that may explain experimental observations left unaddressed by the standard model (SM) and expand our understanding of phenomena such as gravity and dark matter. The search for beyond the SM (BSM) physics has had limited success at the LHC possibly because current methods rely too heavily on hypothesized BSM signatures that may not reflect the true nature of the new physics. To address this, there has been a growing interest in employing unsupervised machine learning (ML) models that can search for new physics independent of underlying signature assumptions. For example, autoencoders, ML models that learn to map data down to a compressed encoding of its most salient features and then reverse such encodings back to their original form, have been employed for unsupervised anomaly detection [1–6]. Autoencoders learn to accurately reconstruct data similar to what is seen during its training; however, anomalous signals rare or absent in the training data may not be accurately reconstructed—a property that can be used to detect them.

We propose particle graph autoencoders (PGAEs) based on graph neural networks (GNNs) [7, 8] for unsupervised detection of new physics in multijet final states at the LHC. By embedding particle jet showers as a graph, GNNs are able to exploit particle-to-particle relationships to efficiently encode and reconstruct particle-level information within jets. We posit that this can improve the capacity of autoencoders to learn a compressed representation of a jet and consequently help identify anomalous BSM multijet signal events from LHC data. We also develop and validate a differentiable, learned approximation to an important distance metric, the energy mover's distance [9], using a GNN, dubbed EMD-GNN, which has the potential to be used as both a loss function to train a PGAE as well as a metric by which to judge how anomalous a jet is.

Fourth Workshop on Machine Learning and the Physical Sciences (NeurIPS 2021).

2 Related Work

Autoencoders in HEP A number of different autoencoder models have been studied for application in searching for new physics at the LHC [1–4]. One major drawback of many of these studies is the use of vector- or image-based representations of HEP data, which aren't well-suited to the sparsity and irregular geometry typical data produced at the LHC. We propose instead to use the more natural set-based "particle cloud" [10] representation for particles in a jet, which is inherently sparse and agnostic to the underlying geometry, and operate on this representation using GNNs.

Graph networks GNNs are powerful, expressive networks that can operate on particle clouds and respect permutation invariance (for graph-level outputs) and covariance (for edge- and node-level outputs) [11]. Due to this they have been steadily gaining prominence in HEP [7]. Notable examples include the dynamic graph convolutional neural network (DGCNN) [12], which has been used for calorimetry in a high-granularity calorimeter [13] and jet identification [10], as well as the interaction network [14] and its generalization to "graph networks" [15], which have been used for jet identification [16, 17] and particle tracking [13, 18]. Other network architectures, GravNet and GarNet, have been studied for calorimetry [19, 20].

GNNs for anomaly detection in HEP have not yet been fully explored. However, recent work [6] develops an autoencoder-based strategy to facilitate anomaly detection for boosted jets, using a symmetric decoder that simultaneously reconstructs edge features and node features. Latent-space discriminators are used isolate W bosons, top quarks, and exotic hadronically-decaying exotic scalar bosons from QCD multijet background. This work expands on that by performing a realistic resonance search using the PGAE model.

2.1 Reconstruction loss functions

Since the inputs and outputs are sets, a reconstruction loss needs to address the assignment problem, i.e. find a one-to-one correspondence between the two sets of nodes. For a permutation-equivariant model, such as the presently considered GNN, the mean-squared error (MSE) is a standard choice because the order is preserved between the inputs and outputs. The Chamfer loss [21–23] is permutation invariant, but has been found to be suboptimal [5]. Finally, the energy mover's distance (EMD) [9], related to the Earth mover's distance [24–26], is a desirable loss, which quantifies the difference between jets through optimal transport as the minimum "work" required to rearrange one jet into another by movements of transverse momentum between the particles in each jet. Finding the EMD is a linear program [25], the exact solution to which is not efficiently differentiable, which limits its use directly as a loss function for training with backpropagation. Thus, instead we develop a GNN-based approximation of the EMD, "EMD-GNN", which may be used in the future as a differentiable loss function. Others have studied alternative approximations to the Earth or energy mover's distance to improve computability [27–29].

3 Network Architectures

PGAE In the PGAE model, we represent input jets as fully-connected graphs where each constituent particle is represented as a node, and with edges between each pair of nodes. When encoding and decoding, the graph structure of the data remains the same, but the node features, initially the particle's three-momentum $\boldsymbol{p} = (p_x, p_y, p_z)$, have their dimensionality reduced during the encoding phase. We note the model can be expanded to consider additional particle-level information, such as particle type, electromagnetic charge, and pileup probability weight [30]. For the encoder and decoder, we use the edge convolution layer from Ref. [12], which performs message passing along the edges and aggregation of messages at the nodes of the graphs.

The PGAE model is constructed using the PyTorch Geometric library [31]. The node features inputted to the encoder are first processed by a batch normalization layer [32]. The encoder itself is a single DGCNN layer, built from a fully connected neural network ϕ_e with layers of sizes $(32, 32, 2)$ and rectified linear activation unit (ReLU) activation functions [33]. The network takes in as input $(\boldsymbol{p}_i, \boldsymbol{p}_j - \boldsymbol{p}_i)$, where $\boldsymbol{p}_i$ $(\boldsymbol{p}_j)$ is the three-momentum for particle i (j) and $i \neq j$. The final layer produces a two-dimensional message vector from each pair of distinct particles. These message vectors are aggregated (using a mean function) for each receiving particle using

$$\boldsymbol{h}_i = \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} \phi_{\mathrm{e}}(\boldsymbol{p}_i, \boldsymbol{p}_j - \boldsymbol{p}_i) \,, \tag{1}$$

where $\mathcal{N}(i)$ is the neighborhood of particles connected to the i-th particle, which corresponds to all other particles in this case. This summed message $\vec{h}_i$ is the bottleneck or encoded representation for the i-th particle. The decoder is also a DGCNN layer, containing a network ϕ_{d} with layers of sizes $(32, 32, 3)$ and ReLU activation functions after all but the final layer. The input is a 3-dimensional vector representing $(\boldsymbol{h}_i, \boldsymbol{h}_j - \boldsymbol{h}_i)$ and the output is intended to reconstruct each particle's momentum. We note that the architecture itself is covariant under permutations of the input particles and applicable to variable-size jets.

EMD-GNN The input to the EMD network is a pair of jets, represented in a single graph in a similar format to the PGAE's input, but with an extra binary channel per node to differentiate which jet it belongs to: $+1$ (-1) for the first (second) jet.

The EMD network itself is a GNN that utilizes three DGCNN layers, each one using two-layered fully connected networks with ReLU activations and batch normalization. For each DGCNN layer the graph structure is dynamically recomputed with edges directed to each node from its 16-nearest-neighbors in feature space. A softplus activation is applied to the final output.

To ensure a symmetric distance metric, the network is inputted both permutations of the input jets and the predicted EMD value is the average of the network outputs. We also utilize a symmetric loss function consisting of the MSE between the predicted and true EMD value, plus the MSE between the predicted EMDs for the two inputs.

4 Experiments

Datasets The dataset [34] comes from the LHC Olympics (LHCO) 2020 challenge and consists of a collection of simulated particle collisions divided up across three "black boxes" (BB) and one background QCD dijet events sample, each with one million particle collision events. Two of the black boxes (1 and 3) were injected with anomalous signals, while one (2) had no anomalous signals injected. In addition, we also use a R&D dataset from the LHCO [35], which has similar QCD events plus an additional 100,000 injected signal events with labels[1].

For input to the PGAE, we process the events using pyjet [36] to cluster $R = 1$ anti-k_{T} jets [37, 38], selecting only the leading two jets by transverse momentum per event, and then representing each jet as a vector of its constituents' three-momenta $p = (p_x, p_y, p_z)$, with array format $(N_{\mathrm{particles}}, 3)$. We also require each jet to have $p_{\mathrm{T}} > 200\,\mathrm{GeV}$. We train the autoencoders on the processed background dataset, and then evaluate them on the black boxes. For the EMD-GNN, we use the same LHCO background data for training, but represent each particle in a jet by its relative $(p_{\mathrm{T}}, \eta, \phi)$, forming all possible unique pairs of jets from 1,000 total events. The true EMD value is computed with the EnergyFlow library [9], which bases its computation on the Python Optimal Transport library [26]. The dataset is randomly partitioned into training (80%), validation (10%), and testing samples (10%).

PGAE+MSE Results Training details for both the PGAE and EMD networks can be found in App. A. Figure 1 (left) shows a comparison of input and reconstructed features for the models trained with PGAE evaluated on a test set. We see that the PGAE trained using a MSE loss performs well at reconstructing particles features. Although the model does not perfectly reconstruct the (double) peak in the center of the feature distributions, for the purpose of anomaly detection, this may not be a problem as long as non-outliers are reconstructed well enough that they have a lower reconstruction loss compared to actual outliers.

For anomaly detection we first study our algorithm on the R&D dataset. As the truth information is provided, we construct a receiver operating characteristic (ROC) curve to determine the effectiveness of the PGAE to identify a signal ($\mathrm{W}' \rightarrow \mathrm{XY}$, $\mathrm{X} \rightarrow \mathrm{qq}$, and $\mathrm{Y} \rightarrow \mathrm{qq}$ with $m_{\mathrm{W}'} = 3.5\,\mathrm{TeV}$, $m_{\mathrm{X}} = 500\,\mathrm{GeV}$, and $m_{\mathrm{Y}} = 100\,\mathrm{GeV}$) that it did not observe during training. As seen in Fig. 1 (center), the PGAE is capable of correctly identifying anomalies.

[1] Both datasets have been released under the CC-BY 4.0 license.

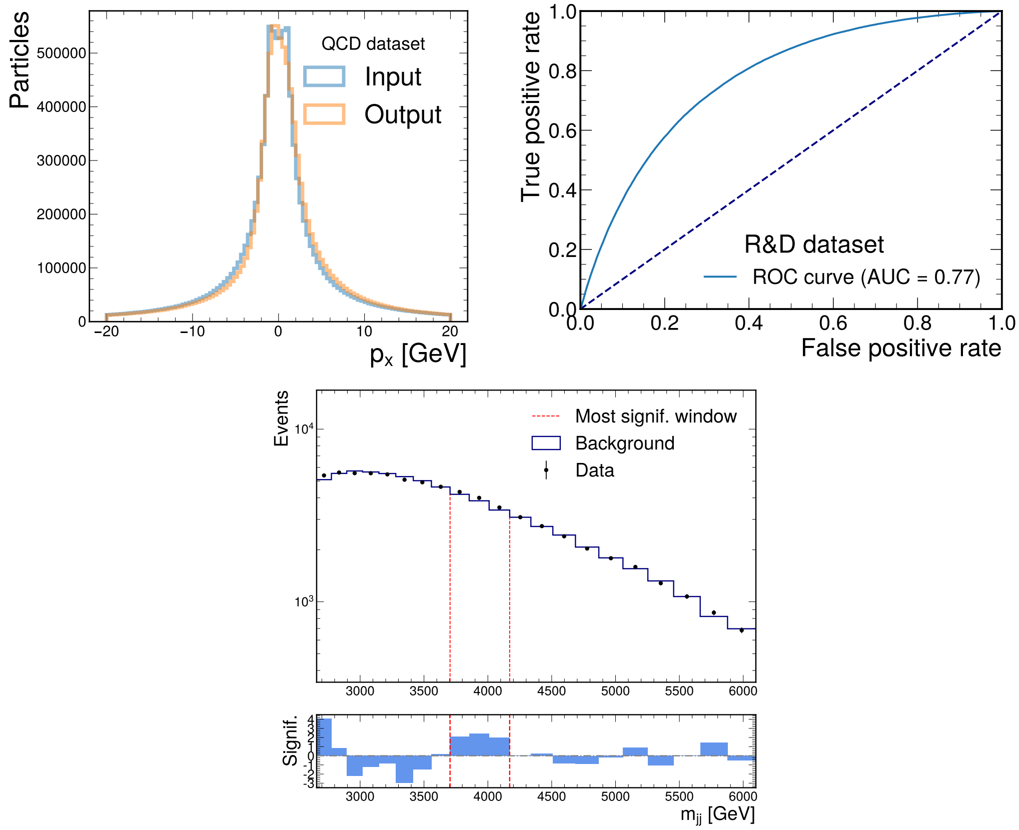

Figure 1: Comparison of input and reconstructed distributions of a sample particle feature p_x (top left) for the models trained with PGAE, evaluated on a test set. ROC curve on the R&D dataset for the PGAE model (top right), and the result of a resonance search using the dijet invariant mass performed on BB 1 (bottom). For the search on BB 1, outlier jets have a reconstruction loss in the top 30% and outlier events are required to have two outlier jets.

To evaluate the model's performance on unlabeled data, we perform a resonance search in the dijet invariant mass m_{jj}, computed from the two leading jets per event, using a variable-width mass binning [39] in the range from 2659 GeV to 6099 GeV. We perform this dijet resonance search in BB 1, which contains a resonant dijet signal at $m_{\mathrm{jj}} \approx 3.8$ TeV. We require both of the jets to be "outliers," which we define as having a reconstruction loss exceeding a threshold corresponding to the 70% quantile of the loss distribution on the evaluation dataset. We note that because our algorithm is jet-focused, it is straightforward to generalize this search to multijet events. To predict the background in the signal-enriched outlier region, we use the shape of the data in the background-enriched nonoutlier region. We perform a maximum-likelihood fit to the ratio of the nonoutlier-to-outlier dijet mass distribution with a fourth-order polynomial to derive a transfer factor (TF) and take the nonoutlier data distribution weighted by the TF as an estimate of the expected background in the outlier region. To derive the observed significance with the simplified background prediction, we use the bump hunter (BH) algorithm [40, 41] to look for resonances in windows spanning two to five bins. With the MSE model in BB 1, we identify a possible resonance around 3.7 TeV with a global significance of $2.8\,\sigma$, which is consistent with the injected dijet resonance mass.

Comparison to other LHCO Contributions Many algorithms proposed for the 2020 LHCO [5] performed similar evaluations on the R&D and BB 1 dataset. Bump hunting in latent space [42] implemented a VAE that incorporates an invariant mass embedding in its latent space. This method obtained an AUC of 0.915 for an event-level discriminant on the R&D dataset. Another algorithm combined a generative adversarial network (GAN) based autoencoder (AE) with the BH algorithm and achieved an AUC of about 0.90 on the R&D dataset. They also performed a dijet resonance search on BB 1, estimating a signal in the 3000–3600 GeV range. Another contribution used regularized

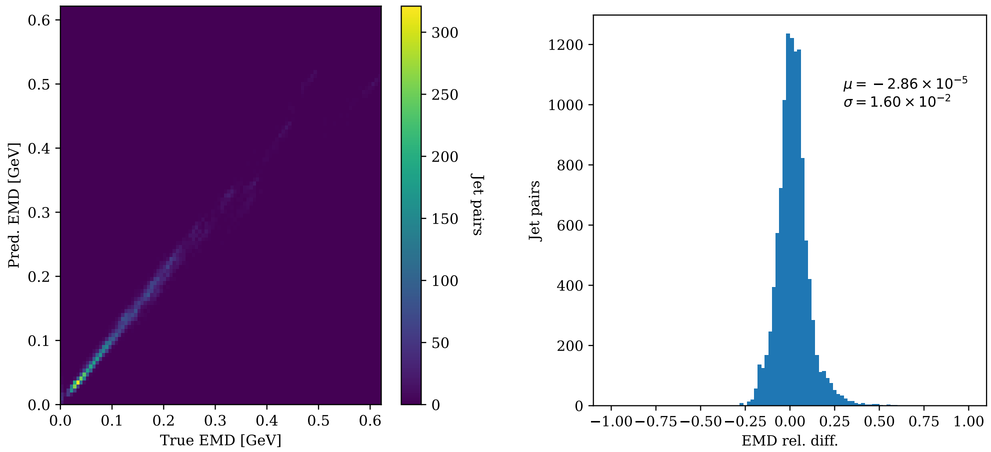

Figure 2: Correlation between EMD-GNN's EMD prediction and the true EMD (left) and relative difference between corresponding true and predicted EMD (right) for the testing dataset composed of random pairs of jets.

likelihoods with manifold-learning flows [43] to construct an anomaly score, which achieved an AUC of 0.7882 for their best performing model. Tag N' Train [44] achieved an AUC of 0.918 on the full R&D dataset, and detected a dijet resonance in BB 1 around 3800 GeV with a local significance of $4\,\sigma$. The deep ensemble anomaly detection method obtained an AUC of 0.96 using boosted decision trees, though we note that they performed semisupervised training on the R&D dataset compared to our fully unsupervised training.

In comparison to our application of the PGAE, many of the other LHCO proposals achieved a higher AUC on the R&D dataset. However, because our discriminant is per jet, the discriminant values from multiple jets may be combined to achieve a better event-level discrimination. Moreover, many of the analysis methodologies are independent of the anomaly detection algorithm itself, thus are complementary to and can be integrated with our PGAE approach.

EMD-GNN Performance As shown in Fig. 2, the EMD-GNN can learn to approximate the EMD between real jets with a very high degree of accuracy, with a -0.003% relative difference between the predicted and true EMD on average, and a standard deviation of 1.6%. This indicates the potential to use this architecture to define a differentiable loss function for particle graph reconstruction.

5 Summary

We demonstrated that particle graph autoencoders (PGAEs) are effective at reconstruction of QCD background jets and, by extension, anomaly detection of anomalous jet signals. Good discrimination between background and signal jets was observed on the LHC Olympics (LHCO) R&D dataset, which provides labels. Moreover, using this algorithm, a dijet resonance was identified in the correct mass range in the LHCO Black Box 1 dataset with a global significance of $2.8\,\sigma$. Additionally, we show that a graph neural network (GNN) can be used to approximate the energy mover's distance (EMD) and therefore potentially be used as a differentiable, permutation-invariant loss function. Future work will investigate optimizing the PGAE with the EMD-GNN as its loss function.

Broader Impact Our PGAE demonstrates the potential of unsupervised anomaly detection through particle graph representations of jets. As jet representations shift towards this particle cloud based format, permutation invariant loss functions such as the EMD become increasingly important. However approximations of EMD are known for their extremely large time complexity, as such, our EMD-GNN serves as an effective and differentiable alternative for approximating EMD. In general, GNNs are becoming increasingly prevalent in many fields, including areas in which they may have harmful impacts on human welfare, to which this work may potentially contribute.

Acknowledgments and Disclosure of Funding

We thank the University of California San Diego Triton Research and Experiential Learning Scholars (TRELS) program for supporting this research in its initial stages. R. K., A. A., D. D., J. D., and F. M. are supported by the US Department of Energy (DOE), Office of Science, Office of High Energy Physics Early Career Research program under Award No. DE-SC0021187 and by the DOE, Office of Advanced Scientific Computing Research under Award No. DE-SC0021396 (FAIR4HEP). F. M. is also supported by an Halıcıoğlu Data Science Institute (HDSI) fellowship. R. K. is also supported by the LHC Physics Center at Fermi National Accelerator Laboratory, managed and operated by Fermi Research Alliance, LLC under Contract No. DE-AC02-07CH11359 with the U.S. Department of Energy (DOE). J-R. V. is supported by the DOE, Office of Science, Office of High Energy Physics under Award No. DE-SC0011925, DE-SC0019227, and DE-AC02-07CH11359. M. P. is supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (Grant Agreement No. 772369). J-R. V. is additionally supported by the same ERC grant as M. P. This work was performed using the Pacific Research Platform Nautilus HyperCluster supported by NSF awards CNS-1730158, ACI-1540112, ACI-1541349, OAC-1826967, the University of California Office of the President, and the University of California San Diego's California Institute for Telecommunications and Information Technology/Qualcomm Institute. Thanks to CENIC for the 100 Gpbs networks.

A PGAE+MSE and EMD-GNN Training

For training the PGAE, we use a batch size of 256, early stopping with a patience of 10 epochs, and an initial learning rate of 0.01. Additionally we employ a learning rate scheduler that lowers the learning rate by a factor of 0.1 after every 4 epochs where the validation loss does not improve, with the minimum learning rate set by this process being 10^{-6}. We use MSE as the loss in our experiments with the PGAE.

To train the EMD-GNN, we use the same training hyperparameters as the PGAE except for the batch size, which we set to 128. Both models were trained on Nvidia GTX 1080Ti GPUs. The PGAE model requires about 5 days to train, and the EMD-GNN model takes about 20 hours.

References

[1] J. Monk, "Deep learning as a parton shower", *J. High Energy Phys.* **12** (2018) 021, doi:10.1007/JHEP12(2018)021, arXiv:1807.03685.

[2] O. Cerri et al., "Variational Autoencoders for New Physics Mining at the Large Hadron Collider", *J. High Energy Phys.* **05** (2019) 036, doi:10.1007/JHEP05(2019)036, arXiv:1811.10276.

[3] M. Farina, Y. Nakai, and D. Shih, "Searching for New Physics with Deep Autoencoders", *Phys. Rev. D* **101** (2020), no. 7, 075021, doi:10.1103/PhysRevD.101.075021, arXiv:1808.08992.

[4] T. Cheng et al., "Variational autoencoders for anomalous jet tagging", arXiv:2007.01850.

[5] G. Kasieczka et al., "The LHC Olympics 2020: A community challenge for anomaly detection in high energy physics", *Rep. Prog. Phys.* (01, 2021) doi:10.1088/1361-6633/ac36b9, arXiv:2101.08320.

[6] O. Atkinson et al., "Anomaly detection with Convolutional Graph Neural Networks", *J. High Energy Phys.* **08** (2021) 080, doi:10.1007/JHEP08(2021)080, arXiv:2105.07988.

[7] J. Shlomi, P. Battaglia, and J.-R. Vlimant, "Graph Neural Networks in Particle Physics", *Mach. Learn.: Sci. Technol.* **2** (2021) 021001, doi:10.1088/2632-2153/abbf9a, arXiv:2007.13681.

[8] J. Duarte and J.-R. Vlimant, "Graph neural networks for particle tracking and reconstruction", in *Artificial Intelligence for Particle Physics*. World Scientific Publishing, 2020. arXiv:2012.01249. Submitted to *Int. J. Mod. Phys. A*. doi:10.1142/12200.

[9] P. T. Komiske, E. M. Metodiev, and J. Thaler, "Metric space of collider events", *Phys. Rev. Lett.* **123** (2019), no. 4, 041801, doi:10.1103/PhysRevLett.123.041801, arXiv:1902.02346.

[10] H. Qu and L. Gouskos, "ParticleNet: Jet tagging via particle clouds", *Phys. Rev. D* **101** (2020) 056019, doi:10.1103/PhysRevD.101.056019, arXiv:1902.08570.

[11] M. M. Bronstein, J. Bruna, T. Cohen, and P. Veličković, "Geometric deep learning: Grids, groups, graphs, geodesics, and gauges", (2021). arXiv:2104.13478. Preprint.

[12] Y. Wang et al., "Dynamic graph CNN for learning on point clouds", *ACM Trans. Graph.* **38** (2019) doi:10.1145/3326362, arXiv:1801.07829.

[13] X. Ju et al., "Graph neural networks for particle reconstruction in high energy physics detectors", in *Machine Learning and the Physical Sciences Workshop at the 33rd Annual Conference on Neural Information Processing Systems*. 2019. arXiv:2003.11603.

[14] P. W. Battaglia et al., "Interaction Networks for Learning about Objects, Relations and Physics", in *Advances in Neural Information Processing Systems 29*, NIPS 2016, p. 4502. 2016. arXiv:1612.00222.

[15] P. W. Battaglia et al., "Relational inductive biases, deep learning, and graph networks", (2018). arXiv:1806.01261. Preprint.

[16] E. A. Moreno et al., "JEDI-net: a jet identification algorithm based on interaction networks", *Eur. Phys. J. C* **80** (2020) 58, doi:10.1140/epjc/s10052-020-7608-4, arXiv:1908.05318.

[17] E. A. Moreno et al., “Interaction networks for the identification of boosted $H \to b\bar{b}$ decays”, *Phys. Rev. D* **102** (2020) 012010, doi:10.1103/PhysRevD.102.012010, arXiv:1909.12285.

[18] G. Dezoort et al., “Charged particle tracking via edge-classifying interaction networks”, *Comput. Softw. Big Sci.* **5** (11, 2021) 26, doi:10.1007/s41781-021-00073-z, arXiv:2103.16701.

[19] S. R. Qasim, J. Kieseler, Y. Iiyama, and M. Pierini, “Learning representations of irregular particle-detector geometry with distance-weighted graph networks”, *Eur. Phys. J. C* **79** (2019) 608, doi:10.1140/epjc/s10052-019-7113-9, arXiv:1902.07987.

[20] Y. Iiyama et al., “Distance-weighted graph neural networks on FPGAs for real-time particle reconstruction in high energy physics”, *Front. Big Data* **3** (1, 2021) 44, doi:10.3389/fdata.2020.598927, arXiv:2008.03601.

[21] H. G. Barrow, J. M. Tenenbaum, R. C. Bolles, and H. C. Wolf, “Parametric correspondence and Chamfer matching: Two new techniques for image matching”, in *Proceedings of the 5th International Joint Conference on Artificial Intelligence (KJCAI)*, volume 2, p. 659. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1977.

[22] H. Fan, H. Su, and L. J. Guibas, “A point set generation network for 3D object reconstruction from a single image”, in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, p. 2463. 6, 2017. arXiv:1612.00603. doi:10.1109/CVPR.2017.264.

[23] Y. Zhang, J. Hare, and A. Prügel-Bennett, “FSPool: Learning set representations with featurewise sort pooling”, in *8th International Conference on Learning Representations*. 2020. arXiv:1906.02795.

[24] Y. Rubner, C. Tomasi, and L. J. Guibas, “The Earth mover’s distance as a metric for image retrieval”, *Int. J. Comput. Vis.* **40** (2000), no. 2, 99, doi:10.1023/A:1026543900054.

[25] G. Peyré and M. Cuturi, “Computational optimal transport: With applications to data science”, *Foundations and Trends® in Machine Learning* **11** (2019), no. 5-6, 355–607, doi:10.1561/2200000073, arXiv:1803.00567.

[26] R. Flamaryet al., “POT: Python optimal transport”, *J. Mach. Learn. Res.* **22** (2021), no. 78, 1.

[27] T. Cai, J. Cheng, N. Craig, and K. Craig, “Linearized optimal transport for collider events”, *Phys. Rev. D* **102** (2020), no. 11, 116019, doi:10.1103/PhysRevD.102.116019, arXiv:2008.08604.

[28] H. Fan, H. Su, and L. Guibas, “A point set generation network for 3D object reconstruction from a single image”, in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, p. 2463. 2017. arXiv:1612.00603. doi:10.1109/CVPR.2017.264.

[29] M. Cuturi, “Sinkhorn distances: Lightspeed computation of optimal transport”, in *Advances in Neural Information Processing Systems*, C. J. C. Burges et al., eds., volume 26. Curran Associates, Inc., 2013. arXiv:1306.0895.

[30] D. Bertolini, P. Harris, M. Low, and N. Tran, “Pileup per particle identification”, *J. High Energy Phys.* **10** (2014) 059, doi:10.1007/JHEP10(2014)059, arXiv:1407.6013.

[31] M. Fey and J. E. Lenssen, “Fast graph representation learning with PyTorch Geometric”, in *ICLR Workshop on Representation Learning on Graphs and Manifolds*. 2019. arXiv:1903.02428.

[32] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift”, in *Proceedings of the 32nd International Conference on Machine Learning*, F. Bach and D. Blei, eds., volume 37, p. 448. PMLR, 2015. arXiv:1502.03167.

[33] A. F. Agarap, “Deep learning using rectified linear units (ReLU)”, (2018). arXiv:1803.08375. Preprint.

[34] G. Kasieczka, B. Nachman, and D. Shih, “Official Datasets for LHC Olympics 2020 Anomaly Detection Challenge”, 2019. doi:10.5281/zenodo.3596919.

[35] G. Kasieczka, B. Nachman, and D. Shih, “R&D Dataset for LHC Olympics 2020 Anomaly Detection Challenge”, 2019. doi:10.5281/zenodo.2629073.

[36] E. Rodrigues, “The Scikit-HEP Project”, *EPJ Web Conf.* **214** (2019) 06005, doi:10.1051/epjconf/201921406005, arXiv:1905.00002.

[37] M. Cacciari, G. P. Salam, and G. Soyez, “The anti-k_{T} jet clustering algorithm”, *J. High Energy Phys.* **04** (2008) 063, doi:10.1088/1126-6708/2008/04/063, arXiv:0802.1189.

[38] 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.

[39] CMS Collaboration, “Search for narrow and broad dijet resonances in proton-proton collisions at $\sqrt{s} = 13$ TeV and constraints on dark matter mediators and other new particles”, *J. High Energy Phys.* **08** (2018) 130, doi:10.1007/JHEP08(2018)130, arXiv:1806.00843.

[40] G. Choudalakis, “On hypothesis testing, trials factor, hypertests and the BumpHunter”, in *PHYSTAT 2011*. 2011. arXiv:1101.0390.

[41] L. Vaslin, J. Donini, and H. Schreiner, “pyBumpHunter”, 2020. https://github.com/lovaslin/pyBumpHunter.

[42] B. Bortolato, B. M. Dillon, J. F. Kamenik, and A. Smolkovič, “Bump hunting in latent space”, (3, 2021). arXiv:2103.06595.

[43] J. Brehmer and K. Cranmer, “Flows for simultaneous manifold learning and density estimation”, in *Advances in Neural Information Processing Systems*, H. Larochelle et al., eds., volume 33, pp. 442–453. Curran Associates, Inc., 2020. arXiv:2003.13913.

[44] O. Amram and C. M. Suarez, “Tag N’ Train: a technique to train improved classifiers on unlabeled data”, *JHEP* **01** (2021) 153, doi:10.1007/JHEP01(2021)153, arXiv:2002.12376.