A Cosmic-Scale Benchmark for Symmetry-Preserving Data Processing

Julia Balla, Siddharth Mishra-Sharma,[*] Carolina Cuesta-Lazaro, Tommi Jaakkola, Tess Smidt
Massachusetts Institute of Technology, IAIFI
{jballa, smsharma, cuestalz, jaakkola, tsmidt}@mit.edu

Abstract

Efficiently processing structured point cloud data while preserving multiscale information is a key challenge across domains, from graphics to atomistic modeling. Using a curated dataset of simulated galaxies, represented as a point cloud, we benchmark the ability of graph neural networks to simultaneously capture local clustering environments and long-range correlations. Given the homogeneous and isotropic nature of the Universe, the data exhibits a high degree of symmetry. We thus focus on evaluating the performance of Euclidean symmetry-preserving ($E(3)$-equivariant) graph neural networks, showing that they can outperform non-equivariant counterparts and domain-specific information extraction techniques in downstream performance as well as simulation-efficiency.

1 Introduction

Point clouds are discrete elements in a coordinate system, defined by spatial coordinates and optional attributes like velocities. Their unordered structure often reveals intricate geometric patterns that span multiple scales, from local neighborhoods to global distributions. Extracting meaningful insights from such data motivates the development of new machine learning algorithms that are capable of capturing and exploiting these multiscale features [1]. Scientific datasets provide an ideal setting for stress-testing these algorithms, as they often exhibit highly structured, yet low-dimensional latent representations despite their complex, high-dimensional observational data.

Cosmology is a prime example of this, since the laws driving the Universe's origin, structure, and evolution are amenable to relatively 'simple' descriptions, allowing scientific data to be characterized using low-dimensional summaries. A particular, flagship observation in cosmology is *galaxy clustering*, where the positions and associated properties of galaxies are measured by large-scale surveys Kravtsov and Borgani [2]. The spatial distribution of these galaxies offers deep insights into the underlying structure of the Universe, helping to answer questions about dark matter, the Universe's expansion, and its overall evolution. As next-generation cosmological surveys are set to deliver petabytes of data, there is a growing need for new, simulation-efficient algorithms to handle this data deluge and enable effective processing and compression, without relying on hand-crafted, lossy features Dvorkin et al. [3].

Cosmological datasets and associated tasks exhibit several distinguishing features that make them a valuable benchmark for stress-testing and developing novel machine learning algorithms, like graph neural networks (GNNs), for processing point cloud data. Examples of these features include:

- **Point cloud cardinality**: The datasets under consideration, described below, are larger than those commonly encountered in other scientific domains where graph processing is used, such as the study of atomistic systems, with $\mathcal{O}(10-100)$ points. This presents unique challenges when it comes to scalability and processing information across the point cloud.
- **Information across scales**: Gravitational forces cause matter to cluster, leading to strong small-scale correlations. On the other hand, growth of structures that were initially in causal contact

[*]Currently at Anthropic; work performed while at MIT/IAIFI.

J. Balla et al., A Cosmic-Scale Benchmark for Symmetry-Preserving Data Processing (Extended Abstract). Presented at the Third Learning on Graphs Conference (LoG 2024), Virtual Event, November 26–29, 2024.

but are spatially separated at present times induces long-range correlations. This multiscale nature necessitates the use of algorithms that can capture both local and global information.

- **Symmetry structure**: The Universe is homogeneous and isotropic – its properties are spatially uniform.This implies that the distribution of galaxies and other cosmic structures should exhibit Euclidean symmetry (i.e., invariance to translations, rotations, and reflections).

Neural networks that use physically-informed inductive biases, e.g. symmetries [4], have been shown to be effective in a variety of domains, from particle physics to materials science [5–8]. Conversely, some recent works question the need to incorporate symmetries for downstream performance [9, 10].

In this paper, we use a cosmological dataset of galaxy positions and properties to systematically study the performance of GNNs on downstream tasks that are sensitive to both local and global correlations, with a particular focus on those that incorporate relevant symmetries. We compare the performance of these models against traditional domain-informed summary statistics to assess the potential of machine learning methods to automate and improve upon existing data analysis techniques in cosmology. Our benchmark aims to stress test existing symmetry-sensitive architectures in a novel, challenging setting.

Related Work. Previous works have considered point cloud-based approaches to information extraction from galaxy distributions. Makinen et al. [11] used GNNs to extract cosmological parameters from galaxy positions, restricting themselves to a specific architecture and relatively small point clouds with $\mathcal{O}(100)$ points. Villanueva-Domingo and Villaescusa-Navarro [12] considered $\mathcal{O}(1000)$ galaxy points and again with limited architectural variation. Anagnostidis et al. [13] considered a non-graph based approach using the PointNet++ architecture [14, 15]. Our goal instead is to systematically study the performance and simulation-efficiency of a range of GNNs, focusing for the first time on symmetry-preserving architectures, on more challenging datasets with thousands of points. Benchmarking GNNs is an active research area, with recent studies highlighting the difficulty of capturing long-range correlations due to oversmoothing and oversquashing effects [16, 17]. Dwivedi et al. [16] introduced the Long Range Graph Benchmark (LRGB) datasets, showing that graph transformers significantly outperform vanilla message-passing GNNs on tasks requiring long-range correlations. Unlike the LRGB and other geometric benchmarks [18, 19], which are limited to $\sim$500 nodes, ours requires processing long-range information across a larger set of points.

2 Dataset and Benchmark Tasks

2.1 Description of Dataset

Our dataset is derived from the high-resolution *Quijote* suite of N-body simulations [20]. These simulations are computationally expensive to run, with over 35 million CPU hours required to generate 44,100 simulations for the initial suite. This computational cost highlights the need for simulation-efficient methods. The final dataset consists of point clouds $\mathbf{X} \in \mathbb{R}^{5000\times 3}$, with each point representing the 3D position of a dark matter clump; we refer to these as galaxies for simplicity, ignoring the details of the dark matter-galaxy connection. In addition to galaxy coordinates, we utilize the galaxy velocities for a subset of our experiments. A total of 12,384 simulations are available, from which a subset is split into a training set of size 2048 and validation and test sets of size 512 for benchmarking. The full dataset is available at https://doi.org/10.5281/zenodo.11479419.

A key statistical measure describing the data is the two-point correlation function (2PCF), which quantifies the excess probability of finding pairs of galaxies at a given separation compared to random. The 2PCF is an efficient summary statistic because it encodes information about the clustering of galaxies at different scales, which is sensitive to the underlying cosmology. In this work, we use the 2PCF as a baseline, highlighting the potential of machine learning models to extract more information from the point cloud data than traditional summary statistics.

2.2 Benchmark Tasks

We consider two benchmark tasks to evaluate the performance of our models: a graph-level prediction task and a node-level prediction task. A visualization of both tasks is provided in App. A .

Graph-level prediction. The graph-level prediction task is a regression problem where the goal is to infer two key cosmological parameters from an input point cloud. Specifically, given a point

cloud $\mathbf{X} \in \mathbb{R}^{5000\times 3}$, representing 5000 galaxy positions, the task is to predict two scalar values $f : \mathbb{R}^{5000\times 3} \to \mathbb{R}^2$: the matter density ($\Omega_m$) and the root-mean-square matter fluctuation averaged over a sphere of radius ~ 8 Mpc (σ_8), which indicates the degree of inhomogeneity in the matter distribution on these scales. These parameters are fundamental to describing the structure of the Universe and are primary targets of current and upcoming cosmological surveys. Ω_m tends to depend sensitively on the nature of long-range correlations, while σ_8 captures information about local correlations. We train the models using the mean squared error (MSE) loss between the predicted output and true target parameters.

Node-level prediction. The node-level prediction task is a regression problem designed to test the ability of the models to capture local information and dependencies within the point cloud, while outputting a more manifestly "geometric" quantity – a velocity vector. The input to the model is again a point cloud $\mathbf{X} \in \mathbb{R}^{5000\times 3}$, where each row corresponds to a galaxy. The output is a tensor $\mathbf{Y} \in \mathbb{R}^{5000\times 3}$ representing the predicted velocity components for all points; $f : \mathbb{R}^{5000\times 3} \to \mathbb{R}^{5000\times 3}$. We train the models using the MSE loss on the predicted velocities.

3 Architectures and Baselines

The graph neural networks we utilize follow the general message-passing framework based on Battaglia et al. [21]. A local k-nearest neighbors graph is constructed using the Euclidean distance between coordinates as the distance metric, accounting for periodic boundary conditions across the box edges. The graph is represented by node features x_i (positions) and edge features e_{ij} (relative distances). We project relative distances onto a basis of radial Bessel functions with a radial cutoff of 0.6 on the Z-scored positions, which was found to be crucial for downstream performance in the graph-level prediction task to predict σ_8, the parameter most affected by short range correlations.

Specific implementations differ in the choice of edge/node update functions and features used in message passing. The GNN closely follows the general framework outlined above, using MLPs for the edge and node update functions ϕ_e^l and ϕ_x^l. $E(n)$ Equivariant Graph Neural Network (EGNN) [22] designs the edge and node updates such that the message-passing operation is equivariant to $E(n)$ transformations. Steerable $E(3)$ Equivariant Graph Neural Network (SEGNN) [23] utilizes steerable feature representations, allowing the node and edge features to be covariant geometric tensors of arbitrary order (e.g. vectors, higher-order tensors) rather than just invariant scalars. Neural Equivariant Interatomic Potential (NequIP) [6] also uses steerable feature representations, constructing equivariant message passing layers using Clebsch-Gordan tensor products and spherical harmonics. We compare these MPNN-based methods with PointNet++ [15], which processes a set of points sampled in a metric space in a hierarchical fashion. See App. B for a detailed description of each architecture.

4 Experiments

Models were trained for 5000 steps using the AdamW optimizer Kingma and Ba [24], Loshchilov and Hutter [25] and a cosine decay schedule. The baseline is given by training an MLP on 24-dimensional 2PCF vectors. The checkpoint corresponding to the lowest validation loss is used for evaluation.

Graph- and node-level prediction. Table 1 compares the test-set performance of different models on the two tasks, along with the number of parameters for the best-performing model. We see that the equivariant models (SEGNN and NequIP) outperform the non-equivariant models (GNN and PointNet) for both tasks. Higher spherical harmonic orders $\ell_{\max}$ provide benefit for the velocity prediction task for SEGNN, but not in the other cases. EGNN does not perform competitively, likely to do its limited expressivity. Equivariant models also show faster convergence than the non-equivariant counterpart, as shown in Fig. 3 in App. D. Additionally, the domain-informed 2PCF summary shows superior performance in extracting Ω_m, which requires capturing long-range correlation. We analyze the impact of incorporating the 2PCF as a global input feature in App. C, showing that the best performing GNN model is unable to capture crucial information in the middle–long range scales.

The ablation study in App. E shows that using attention-based aggregation in the readout layer GNN leads to better performance on the graph-level tasks. We leave the exploration of whether this holds for the other models as future work. When node velocities are included as an input (capturing additional information about the local density field), both the GNN and SEGNN show significant

Inputs	Model	Graph task			Node task	
		Ω_m	σ_8	Params.	$\vec{v}$	Params.
$\vec{x}$	2PCF	$\mathbf{2.03 \pm 0.02}$	4.66 ± 0.06	56k	–	–
	GNN	2.77 ± 0.41	4.84 ± 2.90	1441k	2.94 ± 0.03	463k
	EGNN	13.33 ± 0.00	13.37 ± 0.00	342k	–	–
	NequIP ($\ell_{\max} = 1$)	2.88 ± 0.15	5.05 ± 1.08	439k	2.28 ± 0.00	154k
	NequIP ($\ell_{\max} = 2$)	3.07 ± 0.18	4.80 ± 0.49	450k	2.44 ± 0.00	163k
	SEGNN ($\ell_{\max} = 1$)	2.31 ± 0.03	$\mathbf{2.34 \pm 0.08}$	1015k	2.06 ± 0.00	280k
	SEGNN ($\ell_{\max} = 2$)	2.37 ± 0.06	2.36 ± 0.22	1458k	$\mathbf{2.04 \pm 0.00}$	401k
	PointNet++	2.87 ± 0.07	9.00 ± 3.94	1354k	2.92 ± 0.00	463k
$\vec{x}, \vec{v}$	GNN	1.10 ± 0.02	1.96 ± 0.04	702k	–	–
	SEGNN ($\ell_{\max} = 1$)	1.16 ± 0.02	1.65 ± 0.02	654k	–	–
	SEGNN ($\ell_{\max} = 2$)	1.13 ± 0.03	1.76 ± 0.07	876k	–	–
	SEGNN ($\ell_{\max} = 1$, steerable $\vec{v}$)	0.99 ± 0.03	1.86 ± 0.04	654k	–	–
	SEGNN ($\ell_{\max} = 2$, steerable $\vec{v}$)	$\mathbf{0.84 \pm 0.01}$	$\mathbf{1.42 \pm 0.02}$	876k	–	–

Table 1: Comparison of different models on the graph- and node-level tasks. Two sets of results are shown: those where *(1)* the input point cloud consists of just position coordinates, and *(2)* where the point clouds additionally include a velocity vector for each galaxy. All mean-squared error values of Ω_m and σ_8 are in units of 10^{-3}. The best results for each section are shown in **bold**.

improvements on graph-level tasks. The key gain for the SEGNN comes from using velocities as steerable attributes, allowing it to outperform the GNN when $\ell_{\max} = 2$.

Scaling with dataset size. Figure 1 shows the test-set performance of various models as a function of the number of samples in the training dataset, for the graph-level task. The equivariant SEGNN models, in particular, show better performance at all training sample sizes, while being more simulation-efficient.

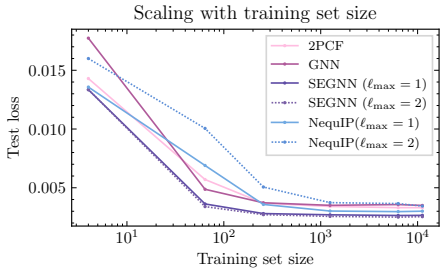

Figure 1: Scaling of the test loss as a function of dataset size, for various models considered, for the graph task.

5 Conclusion

We investigated the ability of graph neural network architectures, with a focus on symmetry-preserving variants, to extract short- and long-range information from point cloud data using cosmology data. The benchmark dataset consists of positions of simulated galaxies, whose spatial distribution is informative of the underlying cosmological model. We showed that both graph-level and node-level prediction tasks can benefit from the use of equivariant models, which were also found to be more simulation-efficient. This is particularly relevant for the domain under study, where producing new simulations is compute-intensive. Equivariant models can therefore enable practitioners to do more with available simulations.

However, we also found that the domain-specific two-point correlation function (2PCF) summary statistic outperformed the graph neural networks in inferring the cosmological parameter Ω_m, which is sensitive to long-range correlations. Message-passing GNNs are known to struggle with long-range correlations [26, 27], and this dataset provides a benchmark to probe their ability to effectively leverage these. The present benchmark would be a good target for methods that aim to mitigate issues associated with long-range information preservation through graphs [28–30]. The equivariant architectures we studied were either general-purpose in nature (e.g., SEGNN) or designed for a specific domain applications (e.g., NequIP for atomistic systems). Our results motivate the development of specialized architectures tailored to cosmology data, which would be sensitive to the local gravitational clustering environment as well as the nature of long-range correlations in galaxy fields.

Acknowledgements

This work is supported by the National Science Foundation under Cooperative Agreement PHY-2019786 (The NSF AI Institute for Artificial Intelligence and Fundamental Interactions, http://iaifi.org/). This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of High Energy Physics of U.S. Department of Energy under grant Contract Number DE-SC0012567. This work was performed in part at the Aspen Center for Physics, which is supported by NSF grants PHY-2210452. The computations in this paper were run on the FASRC Cannon cluster supported by the FAS Division of Science Research Computing Group at Harvard University. This work is also supported by the Department of Defense (DoD) through the National Defense Science & Engineering Graduate (NDSEG) Fellowship Program.

References

[1] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. Deep learning for 3d point clouds: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2020. 1

[2] Andrey V. Kravtsov and Stefano Borgani. Formation of galaxy clusters. *Annual Review of Astronomy and Astrophysics*, 50(1):353–409, September 2012. ISSN 1545-4282. doi: 10.1146/annurev-astro-081811-125502. 1

[3] Cora Dvorkin et al. Machine Learning and Cosmology. In *Snowmass 2021*, 3 2022. 1

[4] Mario Geiger and Tess Smidt. e3nn: Euclidean neural networks. *arXiv preprint arXiv:2207.09453*, 2022. 2

[5] Ilyes Batatia, Philipp Benner, Yuan Chiang, Alin M Elena, Dávid P Kovács, Janosh Riebesell, Xavier R Advincula, Mark Asta, William J Baldwin, Noam Bernstein, et al. A foundation model for atomistic materials chemistry. *arXiv preprint arXiv:2401.00096*, 2023. 2

[6] Simon Batzner, Albert Musaelian, Lixin Sun, Mario Geiger, Jonathan P Mailoa, Mordechai Kornbluth, Nicola Molinari, Tess E Smidt, and Boris Kozinsky. E (3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *Nature communications*, 13(1): 2453, 2022. 3, 9

[7] Ilyes Batatia, David P Kovacs, Gregor Simm, Christoph Ortner, and Gábor Csányi. Mace: Higher order equivariant message passing neural networks for fast and accurate force fields. *Advances in Neural Information Processing Systems*, 35:11423–11436, 2022.

[8] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021. 2

[9] Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J Ballard, Joshua Bambrick, et al. Accurate structure prediction of biomolecular interactions with alphafold 3. *Nature*, pages 1–3, 2024. 2

[10] Yuyang Wang, Ahmed A Elhag, Navdeep Jaitly, Joshua M Susskind, and Miguel Angel Bautista. Generating molecular conformer fields. *arXiv preprint arXiv:2311.17932*, 2023. 2

[11] T Lucas Makinen, Tom Charnock, Pablo Lemos, Natalia Porqueres, Alan Heavens, and Benjamin D Wandelt. The cosmic graph: Optimal information extraction from large-scale structure using catalogues. *arXiv preprint arXiv:2207.05202*, 2022. 2

[12] Pablo Villanueva-Domingo and Francisco Villaescusa-Navarro. Learning cosmology and clustering with cosmic graphs. *The Astrophysical Journal*, 937(2):115, 2022. 2

[13] Sotiris Anagnostidis, Arne Thomsen, Tomasz Kacprzak, Tilman Tröster, Luca Biggio, Alexandre Refregier, and Thomas Hofmann. Cosmology from galaxy redshift surveys with pointnet. *arXiv preprint arXiv:2211.12346*, 2022. 2

[14] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 2, 9

[15] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. *Advances in Neural Information Processing Systems*, 35:23192–23204, 2022. 2, 3, 9

[16] Vijay Prakash Dwivedi, Ladislav Rampášek, Michael Galkin, Ali Parviz, Guy Wolf, Anh Tuan Luu, and Dominique Beaini. Long range graph benchmark. *Advances in Neural Information Processing Systems*, 35:22326–22340, 2022. 2

[17] Jan Tönshoff, Martin Ritzert, Eran Rosenbluth, and Martin Grohe. Where did the gap go? reassessing the long-range graph benchmark. *arXiv preprint arXiv:2309.00367*, 2023. 2

[18] Lowik Chanussot, Abhishek Das, Siddharth Goyal, Thibaut Lavril, Muhammed Shuaibi, Morgane Riviere, Kevin Tran, Javier Heras-Domingo, Caleb Ho, Weihua Hu, Aini Palizhati, Anuroop Sriram, Brandon Wood, Junwoong Yoon, Devi Parikh, C. Lawrence Zitnick, and Zachary Ulissi. Open catalyst 2020 (oc20) dataset and community challenges. *ACS Catalysis*, 11(10):6059–6072, May 2021. ISSN 2155-5435. doi: 10.1021/acscatal.0c04525. URL http://dx.doi.org/10.1021/acscatal.0c04525. 2

[19] Anuroop Sriram, Sihoon Choi, Xiaohan Yu, Logan M. Brabson, Abhishek Das, Zachary Ulissi, Matt Uyttendaele, Andrew J. Medford, and David S. Sholl. The open dac 2023 dataset and challenges for sorbent discovery in direct air capture, 2023. URL https://arxiv.org/abs/2311.00341. 2

[20] Francisco Villaescusa-Navarro, ChangHoon Hahn, Elena Massara, Arka Banerjee, Ana Maria Delgado, Doogesh Kodi Ramanah, Tom Charnock, Elena Giusarma, Yin Li, Erwan Allys, et al. The quijote simulations. *The Astrophysical Journal Supplement Series*, 250(1):2, 2020. 2, 7

[21] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018. 3

[22] Vıctor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E (n) equivariant graph neural networks. In *International conference on machine learning*, pages 9323–9332. PMLR, 2021. 3, 8

[23] Johannes Brandstetter, Rob Hesselink, Elise van der Pol, Erik J Bekkers, and Max Welling. Geometric and physical quantities improve e (3) equivariant message passing. *arXiv preprint arXiv:2110.02905*, 2021. 3, 9

[24] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3

[25] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 3

[26] Qimai Li, Zhichao Han, and Xiao-ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32 (1), Apr. 2018. doi: 10.1609/aaai.v32i1.11604. 4

[27] Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications. In *International Conference on Learning Representations*, 2021. 4

[28] Paras Jain, Zhanghao Wu, Matthew A. Wright, Azalia Mirhoseini, Joseph E. Gonzalez, and Ion Stoica. Representing long-range context for graph neural networks with global attention. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. 4

[29] Jake Topping, Francesco Di Giovanni, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael M. Bronstein. Understanding over-squashing and bottlenecks on graphs via curvature. In *International Conference on Learning Representations*, 2022.

[30] Kedar Karhadkar, Pradeep Kr. Banerjee, and Guido Montufar. FoSR: First-order spectral rewiring for addressing oversquashing in GNNs. In *International Conference on Learning Representations*, 2023. 4

[31] Peter S. Behroozi, Risa H. Wechsler, and Hao-Yi Wu. The rockstar phase-space temporal halo finder and the velocity offsets of cluster cores. *The Astrophysical Journal*, 762(2):109, December 2012. ISSN 1538-4357. doi: 10.1088/0004-637x/762/2/109. URL http://dx.doi.org/10.1088/0004-637X/762/2/109. 7

A Dataset description

A.1 Details of simulation

We use the Big Sobol Sequence (BSQ) of the *Quijote* simulations [20], a collection of 32,768 N-body simulations designed for machine learning applications. Each simulation models the evolution of the large-scale structure of the Universe by following the dynamics of 512^3 cold dark matter particles in a cubic comoving volume of side ~ 1 Gigaparsec from redshift $z = 127$ to $z = 0$ (present time). Dark matter halos, which are gravitationally bound structures that host galaxies, are identified in the simulations using the *Rockstar* halo finder [31].

The simulations are performed using the TreePM *Gadget-III* code, which efficiently computes gravitational forces using a combination of a short-range tree method and a long-range particle mesh method. Each of these simulations has a different initial random seed and a value of the cosmological parameters arranged in a Sobol sequence with boundaries

$$\begin{aligned} \Omega_{\rm m} &\in [0.10; 0.50] \\ \Omega_{\rm b} &\in [0.02; 0.08] \\ h &\in [0.50; 0.90] \\ n_s &\in [0.80; 1.20] \\ \sigma_8 &\in [0.60; 1.00] \end{aligned}$$

The initial conditions were generated at $z = 127$ using 2LPT, and the simulations have been run using Gadget-III.

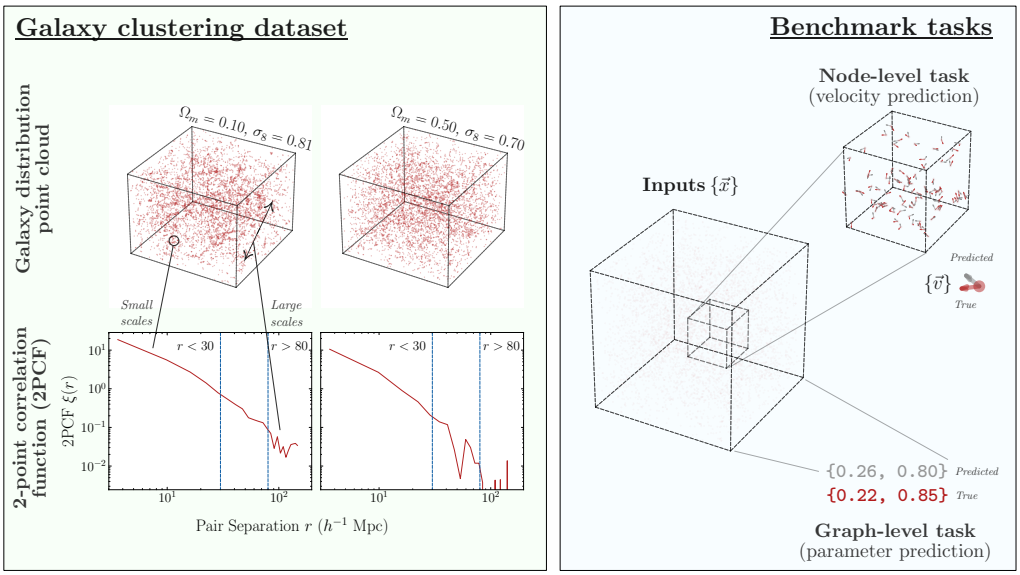

Figure 2: (Left) Exemplary point clouds from the training set and their corresponding 2-point correlation functions. (Right) An illustration of the benchmark tasks.

A.2 Data Access

To facilitate easy access to the dataset used in this work, we provide a high-level Python interface for loading and preprocessing the point cloud data derived from the processed simulation data. The raw data is stored in the TFRecord format, allowing for efficient storage and retrieval. The full dataset is available at https://doi.org/10.5281/zenodo.11479419. The code repository is included in the supplementary materials and will be made publicly available via GitHub.

```python
from benchmarks.galaxies.dataset import get_halo_dataset

features = ['x', 'y', 'z', 'v_x', 'v_y', 'v_z', 'M200c']
params = ['Omega_m', 'sigma_8']

dataset, num_total = get_halo_dataset(batch_size=32 num_samples=2048, split='
    train', standardize=True, return_mean_std=False, seed=42, features=
    features, params=params, include_tpcf=True)

iterator = iter(dataset)
for _ in range(num_total // batch_size):
    x, params, tpcf = next(iterator)  # Load a batch of data

print(x.shape, params.shape, tpcf.shape)
>> (TensorShape([32, 5000, 7]), TensorShape([32, 2]), TensorShape([32, 24]))
```

The `get_halo_dataset` function loads the dataset with the specified batch size, number of samples, data split, and a list of desired features and cosmological parameters. The loaded data can be easily iterated over in batches, with each batch containing the point cloud features (spatial coordinates, velocities, and halo masses in this case) and the corresponding cosmological parameters (Ω_m and σ_8, in this case). There is also an option to include the pre-computed 2PCF vectors as a third output.

B Details of neural network architectures

Below, we describe the differences between the message-passing functions of all graph neural network models used in our study. The training hyperparameters are provided in Tab. 2. In all equations below, we use the following notation to denote the relative distance vectors and their projections onto a basis of radial Bessel functions of order $n = 64$ and radial cutoff $c = 0.6$:

$$\vec{r_{ij}}^{\,l} = \vec{x_i}^{\,l} - \vec{x_j}^{\,l} \tag{1}$$

$$R_{ij}^l = B_n(\|\vec{r}_{ij}^{\,l}\|^2, c). \tag{2}$$

MLP on 2PCF. When using the 2-point correlation function summary instead of the full point cloud, an MLP with 3 hidden layers of dimension 128 and GELU activations was used on the 24-dimensional 2PCF vectors.

GNN. (Message-Passing Graph Neural Network) consists of the following edge and node update functions in one message-passing layer,

$$e_{ij}^{l+1} = \phi_e^l\left(h_i^l, h_j^l, e_{ij}^l\right) \tag{3}$$

$$h_i^{l+1} = \phi_h^l\left(h_i^l, \square_{j\in\mathcal{N}(i)} e_{ij}^{l+1}\right) \tag{4}$$

where all input vectors are concatenated before being fed into 3-layer MLPs ϕ_e^l and ϕ_h^l. Additionally, $\square$ denotes a permutation-invariant message-passing aggregation function over the neighboring edges $\mathcal{N}(i)$ of node i. We select $\square$ to be defined as the mean for all of our models.

EGNN. ($E(n)$ Equivariant Graph Neural Network) [22] edge, position, and node representation update functions are defined as

$$e_{ij}^{l+1} = \phi_e^l\left(h_i^l, h_j^l, R_{ij}^l\right) \tag{5}$$

$$\vec{x_i}^{\,l+1} = \vec{x_i}^{\,l} + C\sum_{j\neq i} \vec{r_{ij}}^{\,l} \cdot \phi_x^l\left(e_{ij}^{l+1}\right) \tag{6}$$

$$h_i^{l+1} = \phi_h^l\left(h_i^l, \square_{j\in\mathcal{N}(i)} e_{ij}^{l+1}\right) \tag{7}$$

The edge update operation is invariant, depending only on the absolute distances. The node position updates are equivariant, depending linearly on the relative position vectors, gated with a nonlinear function of the invariant edge features.

SEGNN. (Steerable $E(3)$ Equivariant Graph Neural Network) [23] extends the EGNN by implementing the node and edge update functions as O(3) steerable MLPs ϕ, consisting of steerable linear layers conditioned on a steerable feature $\tilde{a} \in V_0 \oplus \ldots \oplus V_{\ell_{\max}}$ (e.g., positions and/or velocities),

$$\sigma\left(W_{\tilde{a}}\tilde{h}^l\right) := \sigma\left(\tilde{h}^l \otimes_{cg}^{W} \tilde{a}\right) \tag{8}$$

where σ is a gated non-linearity, $W_{\tilde{a}}$ is a linear transformation matrix conditioned on $\tilde{a}$, and $\otimes_{cg}^{W}$ is the Clebsch-Gordan tensor product that is parametrized by a collection of weights. Thus, the steerable edge and node feature updates are given by

$$\tilde{e_{ij}}^{l+1} = \phi_e^l\left(\tilde{h_i}^l, \tilde{h_j}^l, R_{ij}^l\right) \tag{9}$$

$$\tilde{h_i}^{l+1} = \phi_h^l\left(\tilde{h_i}^l, \square_{j\in\mathcal{N}(i)}\tilde{e_{ij}}^{l+1}, \tilde{a}_i\right) \tag{10}$$

while the node position updates remain unchanged from eq. (11) above. In our experiments, $\tilde{a}_i$ is optionally defined via the node velocities.

The steerable MLPs allow the network to leverage richer geometric information and express anisotropic interactions. The updates are conditioned on steerable node and edge *attributes*, which can inject additional physical information about the local environment into the updates while maintaining end-to-end equivariance.

NequIP. (Neural Equivariant Interatomic Potential) [6] also utilizes steerable features in the message-passing updates. In the edge updates, NequIP applies a linear transformation to the incoming node features and computes the spherical harmonic projections of the normalized relative position vectors, which are combined using a tensor product. This is modulated by a nonlinear radial function implemented as an MLP acting on the relative distances. The node updates combine the aggregated messages with the previous node features using a gated nonlinearity.

The edge and node updates are thus defined as

$$a_{ij}^l = \text{Norm}\left(Y_m^{(\ell)}\left(\vec{r_{ij}}^l\right)\right) \tag{11}$$

$$e_{ij}^{l+1} = \phi_e^l(R_{ij}^l) \cdot [W_i^l h_i^l, a_{ij}^l] \otimes a_{ij}^l \tag{12}$$

$$h_i^{l+1} = \phi_h^l\left(\frac{\square_{j\in\mathcal{N}(i)} e_{ij}^{l+1}}{\sqrt{|E|}}\right). \tag{13}$$

$$\tag{14}$$

where the spherical harmonics are normalized via the integral norm such that $\int_{S^2} Y_m^\ell(x)^2 dx = 1$.

While both are $E(3)$ equivariant, SEGNN uses a more expressive steerable MLP conditioned on the spherical harmonic embedding of the relative position vectors, while NequIP uses a simpler nonlinear radial function to gate linear and spherical harmonic projections of the input features. These choices reflect a trade-off between expressiveness and computational efficiency, with NequIP prioritizing the latter, tailored for its original purpose of efficiently learning interatomic potentials where angular and radial dependencies are crucial and separable. SEGNN and NequIP use a hyperparameter $\ell_{\max}$ to control the maximum degree of the spherical harmonics used in the steerable feature representations, with higher values allowing the models to capture more complex angular dependencies at computational cost.

PointNet++. PointNet++ [15] is an extension of the original PointNet architecture [14] designed to capture both local and global structures in point clouds. Unlike the original PointNet, which treats each point independently, PointNet++ introduces a hierarchical learning framework. It recursively applies PointNet at multiple scales, progressively downsampling the point cloud and learning features at different levels of granularity. This enables the network to capture fine-grained local features as well as broader contextual information.

The PointNet++ architecture employs farthest point sampling (FPS) to select a subset of representative points and groups neighboring points within a defined radius for local feature extraction. These

hierarchical groupings are followed by feature pooling and graph-based operations to coarsen the point cloud representation. The combination of these steps allows PointNet++ to learn spatially aware representations that are crucial for processing 3D point clouds.

In each downsampling layer $i \in \{1, \ldots, \text{n_downsamples}\}$, the number of nodes is reduced from n_{nodes} to $n_{\text{nodes_downsampled}}$ by dividing the original node set by a predefined downsampling factor:

$$n_{\text{nodes downsampled}} = \frac{n_{\text{nodes}}}{\text{downsampling factor}}. \tag{15}$$

At each layer, a Graph Neural Network (GNN) is applied to the graph to obtain updated node embeddings z, such that

$$z_i^{l+1} = \text{GNN}(h_i^l, x_i^l, \{e_{ij}^l\}_{j \in \mathcal{N}(i)}), \tag{16}$$

where $z_i^{l+1} \in \mathbb{R}^{n_{\text{nodes}} \times d}$ represents the new node embeddings at this layer. After applying the GNN, we perform a *sample and group* operation to downsample the set of nodes and create a hierarchical representation. This operation consists of two steps: sampling representative points (centroids) and grouping the remaining points around these centroids.

First, the centroids are selected from the set of node positions $\mathbf{X} \in \mathbb{R}^{n_{\text{nodes}} \times d}$. The sampling is performed using Farthest Point Sampling (FPS), which selects $n_{\text{centroids}}$ representative points:

$$\mathbf{X}_{\text{centroids}} = \text{FPS}(\mathbf{X}, n_{\text{centroids}}), \tag{17}$$

where $\mathbf{x}_{\text{centroids}} \in \mathbb{R}^{n_{\text{centroids}} \times d}$ represents the set of centroids chosen from the original node positions $\mathbf{x}$.

After selecting the centroids, each point $\mathbf{x}_i$ from the original set is grouped with the nearest centroid. The distance matrix $\mathbf{D} \in \mathbb{R}^{n_{\text{nodes}} \times n_{\text{centroids}}}$ is then transformed into an assignment matrix $\mathbf{S} \in \mathbb{R}^{n_{\text{nodes}} \times n_{\text{centroids}}}$ using a row-wise softmax:

$$S_{ij} = \frac{\exp(-d_{ij})}{\sum_{k=1}^{n_{\text{centroids}}} \exp(-d_{ik})}. \tag{18}$$

The assignment matrix S represents the association between nodes and centroids, where each entry S_{ij} indicates the probability that node i is assigned to centroid j. The matrix S is subsequently used to pool features, coarsen the graph, or aggregate information for hierarchical graph processing.

Hyperparameter	**MLP**	**GNN**	**SEGNN**	**EGNN**	**NequIP**
`d_hidden`	128	128	128	128	128
`n_layers`	3	3	3	3	3
`message_passing_steps`	–	3	3	3	3
`message_passing_agg`	–	mean	mean	mean	mean
`readout_agg`	–	mean	mean	mean	mean
`mlp_readout_widths`	(4, 2, 2)	(4, 2, 2)	(4, 2, 2)	(4, 2, 2)	(4, 2, 2)
`residual`	–	True	True	True	True
`scalar_activation`	gelu	gelu	gelu	gelu	gelu
`gate_activation`	–	–	sigmoid	–	–
`spherical_harmonic_norm`	–	–	–	–	integral

Table 2: Hyperparameters for each model on all tasks.

C Two-point correlation function as global information

To gain insight into the information captured by the 2PCF that is not captured by any of the graph neural network models (as evidenced by their worse performance when predicting Ω_m), we evaluate the effects of adding the 2PCF as a global input feature. In particular, we select the best-performing model on both tasks: SEGNN with $\ell_{\max} = 2$. After the final message-passing layer, once all of the

node representations are pooled into a graph-wise representation, the pooled vector is concatenated with the 2PCF before being fed into the readout MLP. We also compare the effects of using only subsections of the 2PCF that correspond to small-scale ($r < 30\ h^{-1}$ Mpc) and large-scale information ($r > 80\ h^{-1}$ Mpc). These are shown in the last set of runs in Tab. 1.

The SEGNN outperforms the 2PCF baseline on Ω_m prediction when it is equipped with the large-scale 2PCF components, and even moreso with the full 2PCF vector. This indicates that there is crucial information present in the middle to long range scales. For the task of σ_8 prediction, which relies on capturing local correlations, the difference between the models with full and small-scale 2PCF information are not statistically significant. This suggests that the SEGNN might already be capturing the short-range correlations in the data.

Model	Ω_m	σ_8	Params.
SEGNN ($\ell_{\max} = 2$) + 2PCF	$\mathbf{1.66 \pm 0.01}$	2.38 ± 0.07	1543k
SEGNN ($\ell_{\max} = 2$) + $\text{2PCF}_{\text{small}}$	2.27 ± 0.01	2.40 ± 0.04	1504k
SEGNN ($\ell_{\max} = 2$) + $\text{2PCF}_{\text{large}}$	1.73 ± 0.04	$\mathbf{2.26 \pm 0.09}$	1512k

Table 3: Comparison of different models on the graph-level tasks where different components of the two-point correlation function are used as an additional global context input. The best results for each task are shown in **bold**.

D Training loss curves for position and velocity prediction

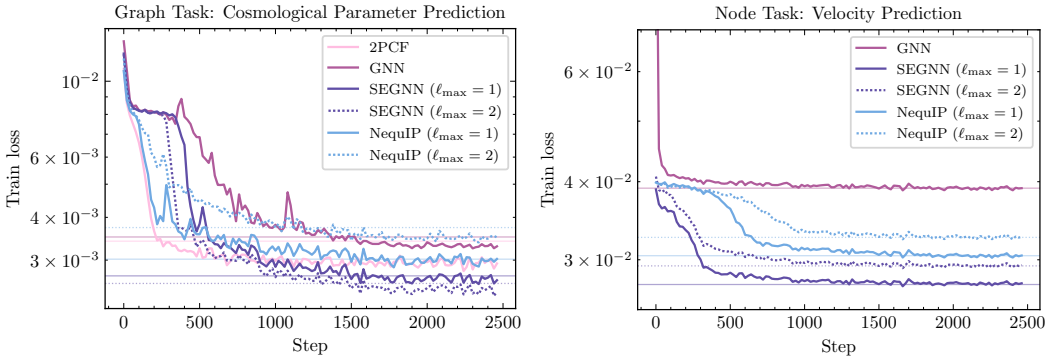

Figure 3: Training losses over the course of training for various models considered, for the graph-level prediction task (left) and the node-level prediction task (right). The final test loss is shown as a horizontal dashed line.

E Ablation study of attention-based aggregation

Given the demonstrated effectiveness of transformer models in capturing long-range dependencies through attention mechanisms, we include an ablation study where we replace the standard aggregation mechanism within the GNN layers with attention-based local and global aggregation. In particular, we modify the GNN layers to incorporate multi-head self-attention mechanisms in place of traditional neighborhood aggregation. Rather than relying solely on proximity-based neighbors, the edge weights are modulated by an attention score computed from other message passing components. These weights dynamically adjust the importance of both local and distant nodes, allowing the model to better capture complex relationships across the graph.

The results demonstrate that incorporating attention-based mechanisms into the readout layer of the GNN significantly improves performance. However, the model that combines both local and global attention does not show the same improvement and even results in increased error. Similarly, the invariant attention model performs worse than both the standard and global attention-based models.

Model	Ω_m	σ_8	Params.
GNN (mean agg.)	2.77 ± 0.41	4.84 ± 2.90	1441k
GNN (global attn. agg.)	$\mathbf{2.60 \pm 0.06}$	$\mathbf{2.84 \pm 0.19}$	915k
GNN (local + global attn. agg.)	3.00 ± 0.14	8.82 ± 3.15	913k
GNN (invariant attn.)	3.62 ± 0.01	13.33 ± 0.01	967k

Table 4: Comparison of different types of local and global aggregation on the graph-level tasks. The best results for each task are shown in **bold**.