A Cosmic-Scale Benchmark for Symmetry-Preserving Data Processing

Editors: List of editors' names

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 galaxy positions and properties, represented as point clouds, 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 therefore 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.

Keywords: Cosmology, Point Cloud, Graph Neural Networks, Equivariance

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. 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 – the study of the Universe's origin, structure, and evolution – 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 type of flagship observation in cosmology is *galaxy clustering*, where the positions and associated properties of galaxies are measured by cosmological surveys. Different spatial distributions of these galaxies can provide insights into the underlying structure of the Universe, targeting questions like the distribution and nature of dark matter and the expansion history of the Universe. Reliably extracting information from these surveys is a challenging task that is becoming increasingly timely as next-generation cosmological surveys like the Dark Energy Spectroscopic Instrument (DESI) (Levi et al., 2019) are set to deliver petabytes of data in the form of point clouds consisting of over $\mathcal{O}(10^6)$ points, necessitating the development of new data-processing and compression algorithms based on simulations to fully realize their scientific goals. Furthermore, the simulations are extremely computationally expensive, making simulation-efficiency a key consideration in the development of new algorithms for this domain.

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, for processing point cloud data. Examples include:

- **Point cloud cardinality**: The datasets under consideration 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 point clouds.
- **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 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).

Although the data is not inherently graph-structured (rather, it is a set of points without specified ordering or interconnectedness), graph neural networks (GNNs) provide an efficient way to process this data, and we focus on benchmarking these here.

Primary contributions Our primary contributions are as follows. First, we curate a dataset from existing cosmological N-body simulations tailored towards benchmarking point cloud processing algorithms and introduce an easy-to-use interface for accessing the dataset, without the need for specialized domain expertise. Simultaneously, we introduce a Jax-based code repository implementing common equivariant neural networks customized for processing point cloud data with associated physical features and using them for inference tasks. Second, we study the effect of various architectural choices on the performance of graph neural networks on downstream tasks using this dataset, with a particular focus on those that incorporate the relevant symmetries. Finally, we compare the performance of these models against traditional domain-informed summary statistics, with the goal of assessing the potential of machine learning methods to automate and improve upon existing data analysis techniques in cosmology.

2. Background and Related Work

Cosmology with point clouds Previous works have explored point cloud-based approaches to extracting information from galaxy distributions. Makinen et al. (2022) used GNNs to extract cosmological parameters from galaxy positions, focusing on a specific architecture and relatively small point clouds with $\mathcal{O}(100)$ points. Villanueva-Domingo and Villaescusa-Navarro (2022) again considered $\mathcal{O}(1000)$ galaxy points with limited architectural variation. Anagnostidis et al. (2022) used a non-graph-based approach with PointNet++ (Qi et al., 2017; Qian et al., 2022). Our goal is to systematically study the performance and efficiency of various GNNs, focusing for the first time on symmetry-preserving architectures for larger datasets with $\mathcal{O}(10^4)$ points. While conventional 3D computer vision datasets (Chang et al., 2015; Wu et al., 2015) often have high cardinality, they emphasize structured shapes, whereas our dataset requires models to capture multi-scale structures across entire point clouds.

Long-range graph benchmarks Benchmarking GNNs is an active research area, with recent studies highlighting the difficulty of capturing long-range correlations due to oversmoothing and oversquashing effects (Dwivedi et al., 2022; Tönshoff et al., 2023). Dwivedi et al. (2022) introduced the Long Range Graph Benchmark (LRGB), showing that graph transformers significantly outperform vanilla message-passing GNNs on tasks requiring long-range correlations. However, unlike the LRGB datasets, which are limited to 500 nodes, our benchmark requires processing long-range information across a larger set of points.

Equivariant graph neural networks Neural networks that incorporate physically-informed inductive biases, e.g. symmetries (Geiger and Smidt, 2022), have been demonstrated to be effective in a variety of domains, from particle physics to materials science (Batatia et al., 2023; Batzner et al., 2022; Batatia et al., 2022; Jumper et al., 2021). These networks can be more simulation-efficient, a requirement in many data-poor settings where data may be expensive to generate, and can also guarantee the validity of output configurations when these contain strict symmetry requirements. Equivariant neural networks have also been shown to scale more favorably with training set size (Batzner et al., 2022). On the other hand, some recent applications question the need to incorporate symmetries for downstream performance (Abramson et al., 2024; Wang et al., 2023). Equivariant neural networks have previously been benchmarked in other physical sciences domains, e.g. fluid mechanics (Toshev et al., 2024). Our benchmark aims to stress-test existing symmetry-sensitive architectures in a novel, challenging setting.

3. Dataset and Benchmark Tasks

3.1. Galaxy Clustering Dataset

Simulation and processing details Our dataset is derived from Sobol sequence set of the *Quijote* suite of N-body simulations (Villaescusa-Navarro et al., 2020).[1] These simulations model the evolution of 512^3 dark matter particles in a periodic comoving volume of ~ 1 Gigaparsec. Each simulation varies the cosmological parameters and the random phases of the initial conditions simultaneously, providing a diverse range of possible universes to learn from and test on. The simulations employ periodic boundary conditions, which ensure that the Universe is modeled as statistically homogeneous and isotropic on large scales. This treatment allows for the accurate modeling of the large-scale structure without introducing artificial boundary effects.

Dark matter halos, which are gravitationally bound structures that host galaxies, are identified in the simulations using a halo-finding algorithm. We do not distinguish between dark matter halos and galaxies in this work, neglecting the details of the connection between galaxies and dark matter, which are not significant for benchmarking purposes. These simulations are computationally expensive to run, with over 35 million CPU hours required to generate 44,100 simulations for the initial suite (Villaescusa-Navarro et al., 2020). This computational cost highlights the need for simulation-efficient methods to extract information from the resulting datasets, which is one of the primary motivations for our work. Further details of the simulation are given in App. A.

[1] The *Quijote* simulation dataset and, in particular, the Big Sobol Sequence (BSQ) is available at https://quijote-simulations.readthedocs.io/en/latest/bsq.html.

Dataset description The resulting dark matter halo coordinates are represented as a 3D point cloud. We select the 5000 most massive halos by halo mass in each simulation to construct our dataset. This selection by number of objects, rather than by a minimum halo mass threshold, mimics the constraints of observational data where masses are unknown. The final dataset consists of point clouds of shape $\mathbb{R}^{5000\times3}$, with each point representing the 3D position of a dark matter halo. In addition to the halo coordinates, the data also contains other attributes, namely halo mass, as well the velocity and angular momentum vectors for each halo. In the current benchmarks, we utilize the velocity attributes for a subset of our experiments, resulting in point clouds of shape $\mathbb{R}^{5000\times6}$. A total of 12,384 simulations are available, from which a subset is split into a training set of size 1248 and validation and test sets of size 512 for benchmarking. We show how the different architectures scale with training set size in Section 5.4. Examples of point cloud samples are visualized in Fig. 1.

Two-point correlation function (2PCF) A key statistical measure of the large-scale structure is the isotropic two-point correlation function (2PCF), which quantifies the excess probability of finding pairs of galaxies at a given separation compared to a random distribution. The 2PCF is an efficient summary statistic because it encodes information about the clustering of galaxies at different scales, can be accurately measured from observational data, and is well-understood theoretically, making it a powerful tool for constraining cosmological models. In this work, we use the 2PCF as a baseline to compare against our models, highlighting their potential to extract more information from the point cloud data than traditional summary statistics. We include the 2PCF as part of the dataset, computed as a 24-dimensional vector where each dimension represents a logarithmically spaced bin in radial distance. 2PCFs corresponding to example point clouds are visualized in Fig. 1.

3.2. Data Access

To facilitate easy access to the dataset used in this work, we provide a high-level Python interface (described in App. A.2) 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.[2] The code repository is included in the supplementary materials and will be made publicly available via GitHub.

3.3. Benchmark Tasks

We consider two benchmark tasks to evaluate the performance of our models: a graph-level and a node-level prediction task. These tasks are designed to test the ability of the models to learn global properties of the point cloud as well as leverage correlations in the local environment, respectively. We note that we don't model the dynamics of the system, which is otherwise a common task for benchmarking (equivariant) GNNs (Toshev et al., 2024).

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 $\mathbf{X} \in \mathbb{R}^{5000\times3}$, representing 5000 galaxy positions, the task is to predict the values of the matter density (Ω_m) and the root-mean-square matter fluctuation averaged

[2] The full dataset is available at https://doi.org/10.5281/zenodo.11479419.

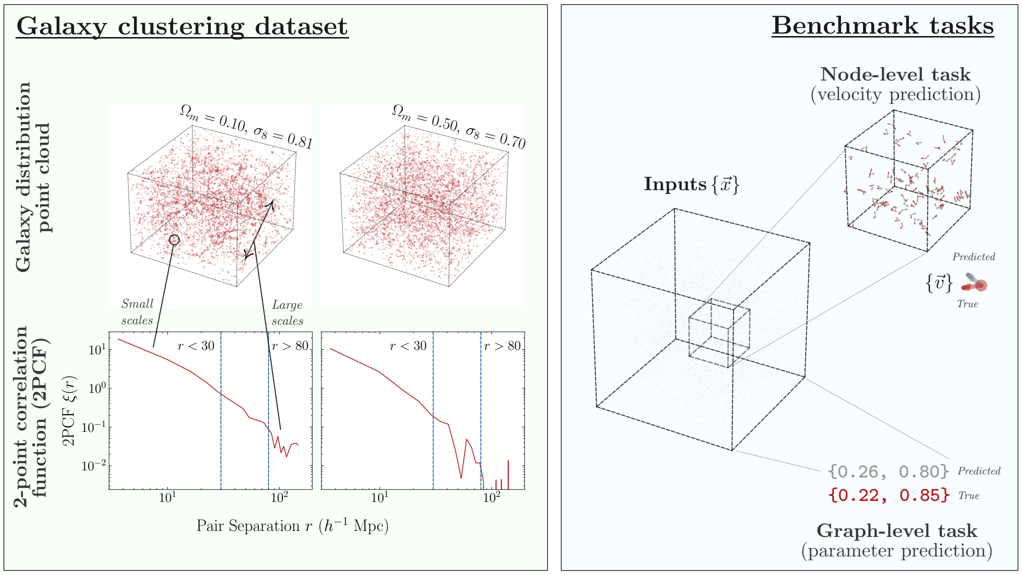

Figure 1: (Left) Exemplary point clouds from the training set and their corresponding 2-point correlation functions. (Right) An illustration of the benchmark tasks.

over a sphere of radius ~ 8 Mpc (σ_8), the degree of inhomogeneity in the matter distribution on these scales; $f : \mathbb{R}^{5000\times3} \to \mathbb{R}^2$. 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 dependencies within the point cloud, while outputting a more manifestly "geometric" quantity – a velocity vector for each node. The input is again a point cloud $\mathbf{X} \in \mathbb{R}^{5000\times3}$, where each row corresponds to a galaxy. The output is a tensor $\mathbf{Y} \in \mathbb{R}^{5000\times3}$ representing the predicted velocity components for all points; $f : \mathbb{R}^{5000\times3} \to \mathbb{R}^{5000\times3}$. We train the models using the MSE loss on the predicted velocities. Note that this is a very challenging task when only considering the heaviest 5000 galaxies.

Relevance to model performance interpretation The different tasks and target parameters probe different abilities of GNN information extraction, highlighting the usefulness of scientific datasets as benchmarks. The parameter Ω_m tends to depend sensitively on the nature of long-range correlations, while σ_8 captures information about small-scale gravitational clustering. The ability of a model to perform well at inferring these target parameters is hence indicative of their ability to leverage long- and short-range correlations, respectively. Inferring velocities (the node task) on the other hand requires leveraging the structure of the local environment around a point, without averaging over the point cloud, but also learning to capture long range correlations that introduce large scale motions.

4. Architectures and Baselines

The graph neural networks we utilize follow the general message-passing framework based on Battaglia et al. (2018). 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. Graph Neural Network (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) (Satorras et al., 2021) 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) (Brandstetter et al., 2021) 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) (Batzner et al., 2022) also uses steerable feature representations, constructing equivariant message passing layers using Clebsch-Gordan tensor products and spherical harmonics. 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.

We compare the above MPNN-based methods with PointNet++ (Qian et al., 2022), which processes a set of points sampled in a metric space in a hierarchical fashion. See App. B for a more detailed description of each architecture.

5. Experiments and Discussion

5.1. Training details

Models were trained for 5000 steps with 5-fold cross validation using the AdamW optimizer (Loshchilov and Hutter, 2017) and a cosine decay schedule. The checkpoint corresponding to the lowest validation loss is used for evaluation. Training was performed on 4 NVIDIA A100 GPUs and took approximately 14–44 minutes per training run.

5.2. Graph- and node-level prediction

Figure 2 shows the losses over the course of training for the graph-level prediction task and the node-level prediction task. The final test losses are shown as horizontal lines. 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. Three sets of results are shown, separated by horizontal lines – those where *(1)* the input point cloud consists of just position coordinates, *(2)* where the point clouds additionally include a velocity vector for each galaxy, and *(3)* where the two-point correlation function is used as an additional global context input (see Sec. 5.3 below).

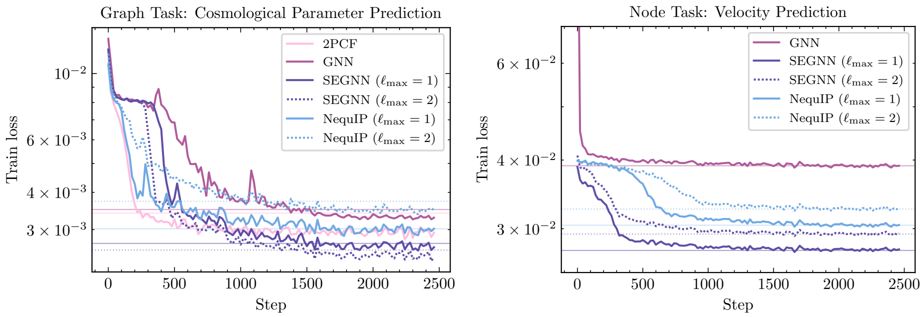

Figure 2: Training losses over the course of training for various models considered, for the graph-level prediction task (left) and the node-level prediction task (right). The final test loss is shown as a horizontal dashed line.

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. While equivariant models beat the non-equivariant ones in graph-level parameter prediction, the domain-informed 2PCF summary shows superior performance in extracting Ω_m, which requires capturing long-range correlations; this indicates that there is yet information left on the table when using message-passing GNNs, in particular when it comes to long-range correlations. Equivariant models also show faster convergence than the non-equivariant counterpart. When node velocities are provided as an additional input feature, both the GNN and SEGNN show significant improvements on graph-level tasks. However, when velocities are used solely as node features in the SEGNN, its performance is worse than the GNN, likely due to a more complex loss landscape that requires additional hyperparameter tuning (Flinth and Ohlsson, 2023). The key gain for the SEGNN comes from using velocities as steerable attributes, allowing it to outperform the GNN when $\ell_{\max} = 2$.

5.3. 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, the pooled graph-wise representation 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 matches the performance of the 2PCF baseline on Ω_m prediction when it is equipped with the large-scale 2PCF components. Moreover, it drastically outperforms the baseline when the whole 2PCF vector is used, suggesting that there is crucial information present in the mid-range scales as well. For the task of σ_8 prediction, which relies on capturing local correlations, the small-scale 2PCF information helps as much as the full 2PCF vector.

Inputs	Model	Graph task			Node task	
		Ω_m	σ_8	Params.	$\boldsymbol{v}$	Params.
$\boldsymbol{x}$	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
$\boldsymbol{x}, \boldsymbol{v}$	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 $\boldsymbol{v}$)	0.99 ± 0.03	1.86 ± 0.04	654k	–	–
	SEGNN ($\ell_{\max} = 2$, steerable $\boldsymbol{v}$)	$\mathbf{0.84 \pm 0.01}$	$\mathbf{1.42 \pm 0.02}$	876k	–	–
$\boldsymbol{x}$, 2PCF	SEGNN ($\ell_{\max} = 2$) + 2PCF	$\mathbf{1.66 \pm 0.01}$	2.38 ± 0.07	1543k	–	–
	SEGNN ($\ell_{\max} = 2$) + 2PCF$_{\text{small}}$	2.27 ± 0.01	2.40 ± 0.04	1504k	–	–
	SEGNN ($\ell_{\max} = 2$) + 2PCF$_{\text{large}}$	1.73 ± 0.04	$\mathbf{2.26 \pm 0.09}$	1512k	–	–

Table 1: Comparison of different models on the graph- and node-level tasks. 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**.

5.4. Scaling with dataset size

Figure 3 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. Higher spherical harmonic order help marginally when using SEGNN, but not NequIP. We see that, in addition to performing better in the data-poor regime, the relative advantage of equivariant models is maintained in the asymptotic regime, up to the number of simulations available ($\sim 10^4$).

6. 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.

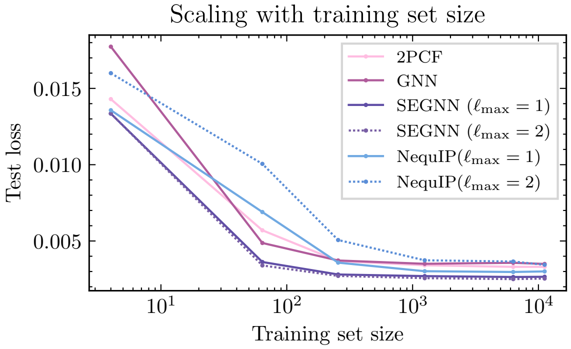

Figure 3: Test loss scaling on the graph task as a function of training set size across models.

However, we also found that the domain-specific two-point correlation function (2PCF) summary statistic outperformed the GNNs in inferring the cosmological parameter Ω_m, which is sensitive to long-range correlations. By incorporating the 2PCF as a global context vector in the SEGNN architecture, we were able to significantly improve its performance, surpassing that of the 2PCF alone, demonstrating the potential of combining domain knowledge with machine learning models. Message-passing GNNs are known to struggle with long-range correlations (Li et al., 2018; Alon and Yahav, 2021), 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 (Jain et al., 2021; Topping et al., 2022; Karhadkar et al., 2023).

Limitations and future work This work does not include many recent architectures, such as GATr (Brehmer et al., 2023), MACE (Batatia et al., 2022), or PΘNITA (Bekkers et al., 2023) as our goal was not to achieve the best possible performance. Instead, we focused on isolating the effects of key architectural components, such as invariance and equivariance, on tasks involving local and long-range correlations. By selecting a few representative models, we aimed for a focused analysis rather than testing a wide range of architectures. Future work can leverage our dataset to benchmark more advanced models. The equivariant architectures we studied were either general-purpose in nature (e.g., SEGNN) or designed for specific domains (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 long-range correlations in galaxy fields.

The benchmark presented is aimed at machine learning researchers as well as domain scientists in cosmology. From the perspective of machine learning, the benchmark provides a challenging and realistic test bed for the development of novel equivariant graph neural network architectures that can handle long-range correlations and scale to large datasets. From the perspective of cosmology, our results demonstrate the potential of machine learning techniques to extract more information from expensive simulations and to automate and improve upon traditional data analysis pipelines. We hope that this work will spur bidirectional collaboration between the two communities.

References

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.

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

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.

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.

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.

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.

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.

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. URL http://dx.doi.org/10.1088/0004-637X/762/2/109.

Erik J Bekkers, Sharvaree Vadgama, Rob D Hesselink, Putri A van der Linden, and David W Romero. Fast, expressive se (n) equivariant networks through weight-sharing in position-orientation space. *arXiv preprint arXiv:2310.02970*, 2023.

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.

Johann Brehmer, Pim de Haan, Sönke Behrends, and Taco Cohen. Geometric algebra transformer. In *Advances in Neural Information Processing Systems*, volume 37, 2023. URL https://arxiv.org/abs/2305.18415.

Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

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.

Axel Flinth and Fredrik Ohlsson. Optimization dynamics of equivariant and augmented neural networks, 2023.

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

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.

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.

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.

Michael E Levi, Lori E Allen, Anand Raichoor, Charles Baltay, Segev BenZvi, Florian Beutler, Adam Bolton, Francisco J Castander, Chia-Hsun Chuang, Andrew Cooper, et al. The dark energy spectroscopic instrument (desi). *The Bulletin of the American Astronomical Society*, 57(7), 2019.

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.

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

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.

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.

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.

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.

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.

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.

Artur Toshev, Gianluca Galletti, Fabian Fritz, Stefan Adami, and Nikolaus Adams. Lagrangebench: A lagrangian fluid mechanics benchmarking suite. *Advances in Neural Information Processing Systems*, 36, 2024.

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.

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

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

Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Computer Vision and Pattern Recognition*, 2015.

Appendix A. Dataset description

A.1. Details of simulation

We use the Big Sobol Sequence (BSQ) of the *Quijote* simulations Villaescusa-Navarro et al. (2020), 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 (Behroozi et al., 2012).

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

$$\Omega_\mathrm{m} \in [0.10; 0.50] \tag{1}$$
$$\Omega_\mathrm{b} \in [0.02; 0.08] \tag{2}$$
$$h \in [0.50; 0.90] \tag{3}$$
$$n_s \in [0.80; 1.20] \tag{4}$$
$$\sigma_8 \in [0.60; 1.00] \tag{5}$$

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

A.2. Data access

```
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=1248, 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.

Appendix 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 $c = 0.6$:

$$\boldsymbol{r_{ij}}^l = \boldsymbol{x_i}^l - \boldsymbol{x_j}^l \tag{6}$$
$$R^l_{ij} = B_n(\|\boldsymbol{r}_{ij}{}^l\|^2, c). \tag{7}$$

The Bessel basis functions are defined as $B_n(r,c) = \sqrt{\frac{2}{c}}\frac{\sin(\frac{n\pi}{c}r)}{r}$ for $r \neq 0$, and $B_n(0,c) = \sqrt{\frac{2}{c}}\frac{n\pi}{c}$.

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,

$$e_{ij}^{l+1} = \phi_e^l\left(h_i^l, h_j^l, e_{ij}^l\right) \tag{8}$$

$$h_i^{l+1} = \phi_h^l\left(h_i^l, \square_{j\in\mathcal{N}(i)} e_{ij}^{l+1}\right) \tag{9}$$

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) Satorras et al. (2021) edge, position, and node representation update functions are defined as

$$e_{ij}^{l+1} = \phi_e^l\left(h_i^l, h_j^l, R_{ij}^l\right) \tag{10}$$

$$\boldsymbol{x_i}^{l+1} = \boldsymbol{x_i}^l + C\sum_{j\neq i} \boldsymbol{r_{ij}}^l \cdot \phi_x^l\left(e_{ij}^{l+1}\right) \tag{11}$$

$$h_i^{l+1} = \phi_h^l\left(h_i^l, \square_{j\in\mathcal{N}(i)} e_{ij}^{l+1}\right) \tag{12}$$

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) Brandstetter et al. (2021) 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),

$$\sigma\left(W_{\tilde{a}}\tilde{h}^l\right) := \sigma\left(\tilde{h}^l \otimes_{cg}^W \tilde{a}\right) \tag{13}$$

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

$$\tilde{e_{ij}}^{l+1} = \phi_e^l\left(\tilde{h_i}^l, \tilde{h_j}^l, R_{ij}^l\right) \tag{14}$$

$$\tilde{h_i}^{l+1} = \phi_h^l\left(\tilde{h_i}^l, \square_{j\in\mathcal{N}(i)}\tilde{e_{ij}}^{l+1}, \tilde{a}_i\right) \tag{15}$$

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) Batzner et al. (2022) 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

$$a_{ij}^{l} = \mathrm{Norm}\left(Y_m^{(\ell)}\left(\boldsymbol{r_{ij}}^{l}\right)\right) \tag{16}$$

$$e_{ij}^{l+1} = \phi_e^l(R_{ij}^l) \cdot [W_i^l h_i^l, a_{ij}^l] \otimes a_{ij}^l \tag{17}$$

$$h_i^{l+1} = \phi_h^l\left(\frac{\square_{j\in\mathcal{N}(i)} e_{ij}^{l+1}}{\sqrt{|E|}}\right). \tag{18}$$

$$\tag{19}$$

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.

PointNet++ PointNet++ Qian et al. (2022) is an extension of the original PointNet architecture Qi et al. (2017) 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:

$$n_{\text{nodes downsampled}} = \frac{n_{\text{nodes}}}{\text{downsampling factor}}. \tag{20}$$

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

$$z_i^{l+1} = \text{GNN}(h_i^l, x_i^l, \{e_{ij}^l\}_{j \in \mathcal{N}(i)}), \tag{21}$$

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:

$$\mathbf{X}_{\text{centroids}} = \text{FPS}(\mathbf{X}, n_{\text{centroids}}), \tag{22}$$

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:

$$S_{ij} = \frac{\exp(-d_{ij})}{\sum_{k=1}^{n_{\text{centroids}}} \exp(-d_{ik})}. \tag{23}$$

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.

Appendix C. Model hyperparameters

Table 2 contains the hyperparameter values used for each experiment across all models. For the remaining hyperparameters, we conducted an extensive sweep using a grid search method. The hyperparameter ranges are outlined in Table 3.

Hyperparameter	Value
d_hidden	128
n_layers	3
mlp_readout_widths	(4, 2, 2)
residual	True
scalar_activation	gelu
gate_activation	sigmoid
spherical_harmonic_norm	integral

Table 2: Shared hyperparameters for each model (where relavant) on all tasks.

Hyperparameter	Values
learning_rate	{0.001, 0.0001, 0.00001}
decay	{0.0001, 0.00001}
n_radial_basis	{32, 64}
k	{5, 10}
message_passing_agg	{Sum, Mean, Max}
readout_agg	{Sum, Mean, Max}
message_passing_steps	{2, 4, 6}
n_downsamples	{2, 3, 4}
d_downsampling_factor	{2, 5, 10}
k_downsample	{10, 20}
r_downsample	{0.05, 0.2}
combine_hierarchies_method	{Mean, Concat}

Table 3: Hyperparameter ranges used for grid search. The parameters below the midrule were only use for the PointNet++ architecture.

Appendix D. 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	2.77 ± 0.41	4.84 ± 2.90	1441k
GNN (global attn. agg.)	**2.60 ± 0.06**	**2.84 ± 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**.