Multi-View Graph Learning with Graph-Tuple

Shiyu Chen
Department of Applied Mathematics and Statistics
Johns Hopkins University
schen355@jh.edu

Ningyuan (Teresa) Huang
Flatiron Institute
thuang@flatironinstitute.org

Soledad Villar
Department of Applied Mathematics and Statistics
Johns Hopkins University
and Flatiron Institute
soledad.villar@jhu.edu

Abstract

Graph Neural Networks (GNNs) typically scale with the number of graph edges, making them well suited for sparse graphs but less efficient on dense graphs, such as point clouds or molecular interactions. A common remedy is to sparsify the graph via similarity thresholding or distance pruning, but this forces an arbitrary choice of a single interaction scale and discards crucial information from other scales. To overcome this limitation, we introduce a multi-view graph-tuple framework. Instead of a single graph, our graph-tuple framework partitions the graph into disjoint subgraphs, capturing primary local interactions and weaker, long-range connections. We then learn multi-view representations from the graph-tuple via a heterogeneous message-passing architecture inspired by the theory of non-commuting operators, which we formally prove is strictly more expressive and guarantees a lower oracle risk compared to single-graph message-passing models. We instantiate our framework on two scientific domains: molecular property prediction from feature-scarce Coulomb matrices and cosmological parameter inference from geometric point clouds. On both applications, our multi-view graph-tuple models demonstrate better performance than single-graph baselines, highlighting the power and versatility of our multi-view approach.

1 Introduction

Graph neural networks (GNNs) have demonstrated remarkable success in learning from structured data [1], achieving state-of-the-art results across diverse fields such as social network analysis, recommendation systems, drug discovery, and materials science [2–6]. The power of GNNs stems from their ability to learn rich representations of nodes and entire graphs by iteratively passing and aggregating messages over a relational structure [5, 7]. This core mechanism endows them with a strong relational inductive bias [8]: the inherent assumption that an object's properties are determined by its connections and local context. This bias is precisely why GNNs are so effective for tasks on graph-structured data [2].

Computationally, GNNs typically scale with the number of graph edges, making them efficient for sparse graphs. However, this efficiency degrades as the graphs become denser. This poses a particular challenge on large, dense graphs such as fully-connected distance graphs derived from point clouds. To make GNN training efficient on dense graphs, a common way is to sparsify it by applying similarity thresholding or distance pruning [9]. Yet, this often reduces information and results in graph representations based on a single fixed scale of interaction. For example, a

1st Conference on Topology, Algebra, and Geometry in Data Science (TAG-DS 2025).

high threshold applied to a molecule's Coulomb matrix retains only strong chemical bonds at the expense of losing important weaker connections. Conversely, a low threshold preserves these weaker connections, but also introduces significant noise. Alternatively, invariant feature models [10] reduce the computational cost to linear in the number of points, by exploiting the low-rank structure of the point cloud and allowing exact reconstruction of the full adjacency matrix from a small submatrix and anchor points.

A complementary line of works avoids graph sparsification and low-rank assumptions, by decomposing a single dense graph into multiple sparser graphs and then learning them in parallel, such as multi-view methods [11–13] and heterogeneous GNNs [14, 15]. These approaches preserve diverse interaction ranges while being computationally tractable, yet are typically designed for heterogeneous graphs with multiple node types or edges types, not directly applicable on homogeneous graphs with continuous edge features.

To tackle these challenges, we propose a multi-view graph representation that captures both fine-grained and contextual interactions. Instead of a single graph, we construct a graph tuple over the same nodes by explicitly partitioning edges according to interaction strength (e.g., distance or Coulomb energy): a strong-connection graph retaining the strongest local interactions and a complementary weak-connection graph providing broader context. Inspired by the theoretical insights of the GtNN framework [16], we then explicitly integrate multiple message-passing operations in a single layer: intra-scale operations (within each graph view) and, crucially, inter-scale operations that model the distinct operator orderings (across different graph views). This yields an interpretable and physically grounded mechanism that links local topology to global effects. We prove that under mild assumptions, this heterogeneous message-passing architecture is more expressive than single-graph models and guaranteed to achieve a lower or equal oracle prediction risk.

We instantiate our framework on two scientific domains. For molecular property prediction on the QM7b dataset, we develop GINE-Gt, a specialized architecture that uses the powerful Graph Isomorphism Network with Edge Features(GINE) [17] as the backbone. Second, for cosmological parameter inference from point cloud data, we develop EGNN-Gt based on Equivariant Graph Neural Network (EGNN) [18], a powerful GNN architecture that guarantees equivariance to rotations, translations, and reflections.

The empirical results demonstrate the efficacy of our framework. In QM7b, GINE-Gt outperforms invariant-feature models and a suite of single-graph GNN baselines in most prediction targets. In the cosmological simulations from the CAMELS suite, EGNN-Gt demonstrates superior overall performance over its corresponding single-graph counterparts across a wide range of interaction radii. These results not only highlights the power of our multi-view strategy but also demonstrates the potential of our multi-view graph tuple framework for a broader range of applications involving continuous relational data. The source code is available on Github[1].

2 Related Work

Our work lies at the intersection of heterogeneous graph learning that process graphs with different typed nodes or edges, and multi-view representation learning to extract features from different scales.

Heterogeneous graph learning. Early heterogeneous GNNs such as R-GCN [14] and HAN [15] demonstrated the benefit of relation-specific message-passing design, but they assume pre-defined, *discrete* node and edge types (e.g., knowledge graphs or bibliographic networks). To go beyond pre-defined relations, Graph Transformer Networks (GTN) [11] proposed to softly select relation-specific adjacency matrices and then generate new graphs by their matrix products. We generalize this heterogeneous graph learning paradigm to homogeneous graphs, by inducing different relations via partitioning *continuous* edge features (e.g., physical distances or chemical interactions).

Multi-view graph learning. A recent line of work constructs multiple relational views from a single graph, motivated from self-supervising learning (e.g., contrastive multi-view learning [12]) or community detection (e.g., variational edge partition model [19]). In contrast to these prior works, we are motivated to construct multiple views based on a physical measure of interaction strength from scientific applications (e.g., the Coulomb matrix in the molecular domain, and the Euclidean distance matrix in the cosmological applications).

[1] https://github.com/chenshy202/Multi-View-Graph-Learning/tree/main

Multi-scale GNNs. Our approach is architecturally most related to multi-scale GNNs that learn a hierarchical representation from a graph, such as FraGAT [20] designed for molecular property prediction,and MultiScale MeshGraphNets [21] for physics simulation. But these methods typically build multiple graphs on different node sets (e.g., atoms vs. fragments and fine vs. coarse mesh), which requires non-trivial cross-level alignment and transfer operators. In contrast, our framework defines multiple graphs over the same node set by partitioning a continuous interaction strength, enabling simple and interpretable within- and between-graph message passing.

3 Preliminaries

In this work, we consider graphs denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, \ldots, v_n\}$ is a set of n nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is a set of edges. Each node v_i is associated with an initial feature vector, and each edge (i, j) with an initial feature vector. Before being processed by any network layers, initial node and edge features are projected into a hidden space via learnable encoders. For notational simplicity, we let $h_i^{(l)}$ denote the encoded feature vector for node i at layer l, and we let e_{ij} denote the encoded feature for the edge (i, j). We collectively represent all node features at a given layer l as a matrix $H^{(l)}$ and all edge features as a matrix E.

3.1 Graph Isomorphism Network with Edge Features (GINE)

GINE [17] extends Graph Isomorphism Network (GIN) by incorporating edge features into its message passing procedure:

$$h_i^{(l+1)} = MLP^{(l)}\Big((1+\varepsilon)\, h_i^{(l)} \; + \sum_{j \in \mathcal{N}(i)} \text{ReLU}\left(h_j^{(l)} + e_{ij}\right)\Big) \tag{1}$$

The entire single-layer update process can then be concisely expressed as:

$$H^{(l+1)} = \text{GINEConv}\left(H^{(l)}, \mathcal{E}, E\right) \tag{2}$$

3.2 Equivariant Graph Neural Networks (EGNN)

Equivariant Graph Neural Networks (EGNNs) [18] incorporate geometric information by endowing each node with Euclidean coordinates $x_i \in \mathbb{R}^d$. They jointly update node features and coordinates in a way that is equivariant to node permutations and Euclidean isometries, i.e., translations, rotations (and reflection). An EGNN convolution layer (EGCL) is defined as follows:

$$m_{ij} = \phi_e\left(h_i^{(l)},\, h_j^{(l)},\, \|x_i^{(l)} - x_j^{(l)}\|^2,\, a_{ij}\right), \tag{3}$$

$$x_i^{(l+1)} = x_i^{(l)} + C \sum_{j \in \mathcal{N}(i)} (x_i^{(l)} - x_j^{(l)})\, \phi_x(m_{ij}), \tag{4}$$

$$h_i^{(l+1)} = \phi_h\left(h_i^{(l)},\, \sum_{j \in \mathcal{N}(i)} m_{ij}\right). \tag{5}$$

Here, ϕ_e, ϕ_x, ϕ_h are learnable functions (e.g., MLPs), C is a scalar and a_{ij} represents optional edge attributes. In this work, we simply use edge features, i.e., $a_{ij} = e_{ij}$. We denote the computation of the EGNN convolution layer as

$$(H^{(l+1)}, X^{(l+1)}) = \text{EGCL}(H^{(l)}, X^{(l)}, \mathcal{E}, E). \tag{6}$$

4 Method

Our work introduces a multi-view graph-tuple framework for learning from complex relational systems where interactions occur at different scales. The core principle is to model these interactions and learn how information flows both within and between these scales. This is achieved by decomposing a graph $\mathcal{G}$ into a graph tuple, $(\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_k)$, that provides distinct yet complementary views of

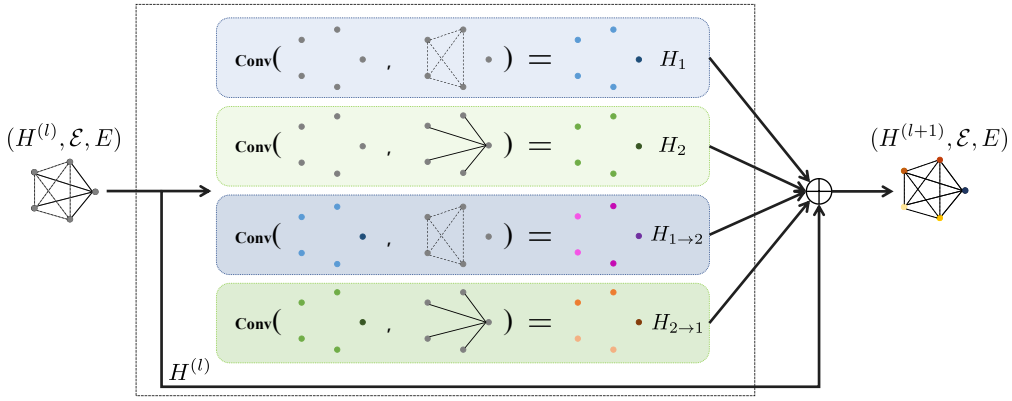


Figure 1: An illustration of our heterogeneous message-passing architecture for $k = 2$ views. The node embeddings H_1, H_2 are obtained from message-passing within each graph (with the corresponding edges and edge features); The node embeddings $H_{1\to 2}, H_{2\to 1}$ are computed from message-passing across views, one for each direction. These embeddings are then aggregated per 7.

the interaction space, representing fine-grained local structures and broader contextual relationships. This graph-tuple framework can be instantiated using different graph neural network backbones. We present two such instantiations: an edge-aware model for attributed graphs, GINE-Gt based on [17], and an equivariant extension for geometric data, EGNN-Gt [18].

4.1 Multi-view Graph-tuple Representation

We begin by decomposing a single graph into multiple views (subgraphs), represented as a graph tuple $\mathcal{G} = (\mathcal{G}_1, \ldots, \mathcal{G}_k)$. Each graph $\mathcal{G}_i = (\mathcal{V}, \mathcal{E}_i)$ is defined on the same node set, but with disjoint edges sets to capture different interaction scales, namely $\bigcup_{i=1}^{k} \mathcal{E}_i = \mathcal{E}$ and $\mathcal{E}_i \bigcap \mathcal{E}_j = \emptyset$ for $i \neq j$. For example, when $k = 2$, we can decompose $\mathcal{G}$ (with nonnegative edge weights) into a strong-connection graph $\mathcal{G}_1$ containing the largest-magnitude edges (greater than certain threshold τ), and a weak-connection graph $\mathcal{G}_2$ capturing the remaining connections. The precise instantiation of these graphs is detailed in Section 6 and 7.

4.2 Heterogeneous Message-Passing Architecture

To learn the multi-view representations from a graph tuple, we use a heterogeneous message-passing architecture, where each layer updates node representations by computing and integrating information from multiple distinct message-passing operations. The final update is defined as a residual combination of these multi-view representations, governed by learnable scalar weights c_k:

$$H^{(l+1)} = H^{(l)} + \sum_{i=1}^{k} c_i \cdot H_i + \sum_{i \neq j} \left(c_{ij} \cdot H_{i\to j} + c_{ji} \cdot H_{j\to i} \right), \tag{7}$$

where H_i denotes the node embeddings from intra-scale message-passing within each graph $\mathcal{G}_i$, and $H_{i\to j}, H_{j\to i}$ denote the node embeddings from inter-scale message-passing across different graphs $\mathcal{G}_i, \mathcal{G}_j$. These representations are designed to capture distinct patterns. For example, given the graph-tuple with a strong-connection graph $\mathcal{G}_1$ and a weak-connection graph $\mathcal{G}_2$, the intra-scale representations (H_1, H_2) extract local interactions from $\mathcal{G}_1$ and broader, long-range information from $\mathcal{G}_2$, respectively. Then these representations are fused in $H_{i\to j}, H_{j\to i}$ via inter-scale message-passing, which capture relational information that is sensitive to the order of operations.

Figure 1 provides an overview of our framework. We note that if the original edge set $\mathcal{E}$ contains *typed* edges, it can be naturally partitioned into homogeneous edge sets, one for each type, as done in R-GCN [14]. Our framework extends R-GCN from heterogeneous graphs to homogeneous graphs, replacing discrete edge types with partitions induced by continuous edge features. The architectures proposed below also extend the Graph Tuple Neural Network framework from [16].

4.2.1 GINE-Gt

For general graphs with node features and edge attributes, we implement our framework using the GINE layer [17]. The intra-scale message-passing for $i = 1, \ldots, k$ are computed as

$$H_i = \text{GINEConv}_i \left(H^{(l)}, \mathcal{E}_i, E_i \right). \tag{8}$$

The inter-scale message-passing are then computed by

$$H_{i \to j} = \text{GINEConv}_{ij} \left(H_i, \mathcal{E}_j, E_j \right), \quad H_{j \to i} = \text{GINEConv}_{ji} \left(H_j, \mathcal{E}_i, E_i \right), \tag{9}$$

where $i \neq j$. Each GINEConv_k is a distinct function with its own parameter weights, allowing the model to learn specialized functions for each interaction type.

4.2.2 EGNN-Gt

For geometric data where node features represent point positions in $\mathbb{R}^d$, we provide an $E(d)$-equivariant implementation of our framework using the EGCL layer [18]. The overall feature update follows Eq. 7, while the position feature update is analogously defined as:

$$X^{(l+1)} = X^{(l)} + \sum_{i=1}^{k} c_i \Delta X_i + \sum_{i \neq j} \left(c_{ij} \Delta X_{i \to j} + c_{ji} \Delta X_{j \to i} \right). \tag{10}$$

The representations $(H_k, \Delta X_k)$, which contain both feature updates and coordinate displacements, are all computed from a single, shared EGCL layer.

The intra-scale representations are computed as

$$(H_i, \Delta X_i) = \text{EGCL} \Big(H^{(l)}, X^{(l)}, \mathcal{E}_i, E_i \Big), \quad \text{for } i = 1, \ldots, k. \tag{11}$$

Subsequently, the inter-scale representations are obtained using these intermediate outputs: for $i \neq j$,

$$(H_{i \to j}, \Delta X_{i \to j}) = \text{EGCL} \left(H_i, X^{(l)} + \Delta X_i, \mathcal{E}_j, E_j \right); \tag{12}$$

$$(H_{j \to i}, \Delta X_{j \to i}) = \text{EGCL} \left(H_j, X^{(l)} + \Delta X_j, \mathcal{E}_i, E_i \right). \tag{13}$$

This formulation allows the EGNN-Gt layer to learn geometrically-aware representations from the multi-view interaction patterns.

5 Expressivity

We analyze our multi-view graph-tuple framework in a simplified linear setting to establish its expressivity and generalization properties. We consider $k = 2$ and define the shift operators S_1 and S_2 to be the adjacency matrices $\mathcal{G}_1$ and $\mathcal{G}_2$ respectively. We study three classes of linear graph filters: $H_1(m)$ the polynomials of degree m in S_1, $H_0(m)$ the polynomials of degree m in $S_1 + S_2$ the adjacency of the dense graph $\mathcal{G}$, and our multi-view graph-tuple class $H_{\text{Gt}}(m)$ of multivariate polynomials of degree m in (S_1, S_2). See Definition 1 in Appendix A. We show the following (see proofs in Appendix A):

Proposition (Expressivity)**.** *For any degree bound m, $H_1(m) \subseteq H_{\text{Gt}}(m)$ and $H_0(m) \subseteq H_{\text{Gt}}(m)$; if $S_1 S_2 \neq S_2 S_1$ and $m \geq 2$, then the latter inclusion is strict. See Proposition 1 in Appendix A.*

Proposition (Oracle risk dominance)**.** *For any m, $\inf_{g \in H_{\text{Gt}}(m)} R(g) \leq \inf_{q \in H_0(m)} R(q)$ and $\inf_{g \in H_{\text{Gt}}(m)} R(g) \leq \inf_{p \in H_1(m)} R(p)$. Moreover, if the oracle predictor $M^\star$ lies outside the expressivity of the baseline class $H_0(m)$, the advantage is strict, and the performance gap is a strictly positive, quantifiable value. See Proposition 2 in Appendix A.*

Proof Sketch. It is easy to see that the single operator baseline models correspond to a special case of the multi-view graph-tuple class for a specific choice of coefficients. The risk dominance is a direct consequence of this expressivity gap. □

Table 1: Performance comparison of our GINE-Gt with all baselines on the QM7b dataset. The result report the Mean Absolute Error (MAE) $\pm$ standard error over ten folds (lower is better). The best result in each column is highlighted in **bold**, and the second-best is in *italics*. Our GINE-Gt is the top-performing method overall, while GINE-2 is the strongest among the single-graph baselines.

MAE $\downarrow$	Atomization PBE0	Excitation ZINDO	Absorption ZINDO	HOMO ZINDO	LUMO ZINDO	1st excitation ZINDO	Ionization ZINDO
KRR [22]	9.3	1.83	0.098	0.369	0.361	0.479	0.408
DS-CI	12.849 ± 0.757	1.776 ± 0.069	0.086 ± 0.003	0.401 ± 0.017	0.338 ± 0.048	0.492 ± 0.058	0.422 ± 0.012
DTNN [22]	21.5	1.26	0.074	0.192	0.159	0.296	0.214
DS-CI+	*7.650* ± 0.399	1.045 ± 0.030	0.069 ± 0.005	0.172 ± 0.009	0.119 ± 0.005	0.160 ± 0.011	0.189 ± 0.011
GINE-0	12.812 ± 0.372	1.034 ± 0.027	0.064 ± 0.002	0.197 ± 0.007	0.072 ± 0.002	0.143 ± 0.003	0.212 ± 0.005
GINE-0.5	12.171 ± 0.543	1.030 ± 0.016	0.068 ± 0.002	0.207 ± 0.004	0.080 ± 0.002	0.143 ± 0.006	0.240 ± 0.007
GINE-1	11.170 ± 0.337	1.000 ± 0.015	*0.064* ± 0.001	0.177 ± 0.005	0.093 ± 0.005	0.120 ± 0.004	0.200 ± 0.005
GINE-2	10.349 ± 0.590	0.998 ± 0.019	0.067 ± 0.002	*0.147* ± 0.004	**0.063** ± 0.001	*0.116* ± 0.006	*0.176* ± 0.009
GINE-2.5	11.306 ± 0.677	*0.969* ± 0.013	0.067 ± 0.001	0.168 ± 0.006	*0.066* ± 0.002	0.131 ± 0.004	0.193 ± 0.005
GINE-Gt	**6.700** ± 0.183	**0.955** ± 0.011	**0.062** ± 0.001	**0.131** ± 0.005	0.067 ± 0.001	**0.111** ± 0.003	**0.151** ± 0.005

MAE $\downarrow$	Affinity ZINDO	HOMO KS	LUMO KS	HOMO GW	LUMO GW	Polarizability PBE0	Polarizability SCS
KRR [22]	0.404	0.272	0.239	0.294	0.236	0.225	0.116
DS-CI	0.404 ± 0.047	0.302 ± 0.009	0.225 ± 0.010	0.329 ± 0.016	0.213 ± 0.008	0.255 ± 0.015	0.114 ± 0.008
DTNN [22]	0.174	*0.155*	0.129	0.166	0.139	0.173	0.149
DS-CI+	*0.122* ± 0.002	0.169 ± 0.007	0.135 ± 0.007	0.183 ± 0.005	0.139 ± 0.004	0.139 ± 0.005	*0.088* ± 0.004
GINE-0	0.082 ± 0.002	0.184 ± 0.008	0.109 ± 0.005	0.198 ± 0.008	0.116 ± 0.004	0.170 ± 0.006	0.094 ± 0.003
GINE-0.5	0.087 ± 0.002	0.207 ± 0.007	0.103 ± 0.003	0.234 ± 0.010	0.129 ± 0.007	0.189 ± 0.004	0.102 ± 0.002
GINE-1	0.088 ± 0.003	0.176 ± 0.005	0.096 ± 0.002	0.201 ± 0.004	0.118 ± 0.003	0.171 ± 0.004	0.104 ± 0.003
GINE-2	**0.067** ± 0.002	0.163 ± 0.006	**0.080** ± 0.002	*0.166* ± 0.003	*0.106* ± 0.002	*0.135* ± 0.003	0.092 ± 0.005
GINE-2.5	*0.070* ± 0.002	0.162 ± 0.006	0.086 ± 0.004	0.180 ± 0.005	0.112 ± 0.002	0.142 ± 0.004	*0.087* ± 0.003
GINE-Gt	0.073 ± 0.002	**0.133** ± 0.002	*0.084* ± 0.001	**0.148** ± 0.003	**0.101** ± 0.002	**0.098** ± 0.002	**0.071** ± 0.002

6 Molecular Property Prediction

We consider the QM7b benchmark [23, 24], which contains 7 211 molecules with 14 regression targets. Each molecule is encoded by a $n \times n$ Coulomb matrix X whose entries depend only on nuclear charges $Z_i \in \mathbb{R}$ and 3D coordinates $R_i \in \mathbb{R}^3$:

$$X_{ij} = \begin{cases} 0.5\, Z_i^{2.4}, & i = j, \\ \dfrac{Z_i Z_j}{\|R_i - R_j\|}, & i \neq j, \end{cases} \tag{14}$$

Then we build the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $\mathcal{V} = \{1, \ldots, n\}$ and $\mathcal{E} = \{(i, j) \mid i \neq j\}$, i.e. all atom pairs are connected while self–loops (i, i) are removed. Notably, since its off-diagonal entries are computed from inter-atomic distances, the Coulomb matrix is invariant to rotations and translations (i.e., E(3)-invariant) by construction.

6.1 Graph Construction

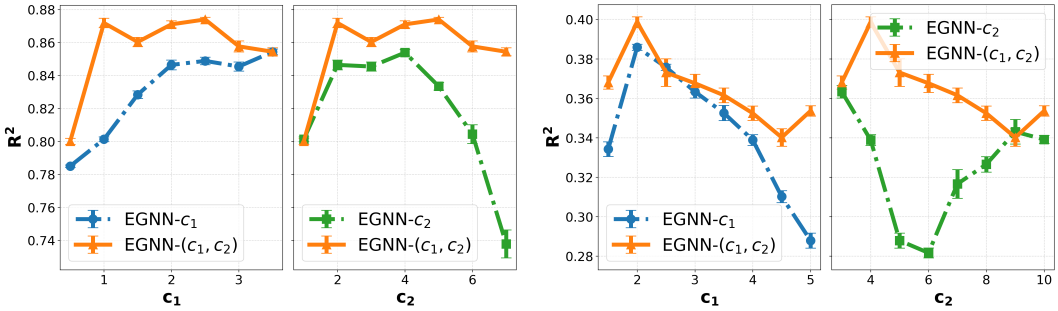
Graphs are constructed from the molecule's Coulomb matrix. The baseline models, denoted GINE-c, operate on a single graph formed by applying a threshold c, where edges are all pairs (i, j) with an interaction strength $X_{ij} \geq c$. This method discards all information below the threshold.

Our main model, GINE-Gt, operates on a multi-view graph tuple $(\mathcal{G}_1, \mathcal{G}_2)$ derived by partitioning the interaction space at a boundary of $c = 2$ selected over a validation set (see Table 1). The strong-connection graph ($\mathcal{G}_1$) is thus composed of edges where $X_{ij} \geq 2$, while the weak-connection graph ($\mathcal{G}_2$) comprises all remaining edges. This threshold effectively identifies the primary interaction backbone for the strong-connection graph ($\mathcal{G}_1$) while assigning the remaining contextual interactions to the weak-connection graph ($\mathcal{G}_2$).

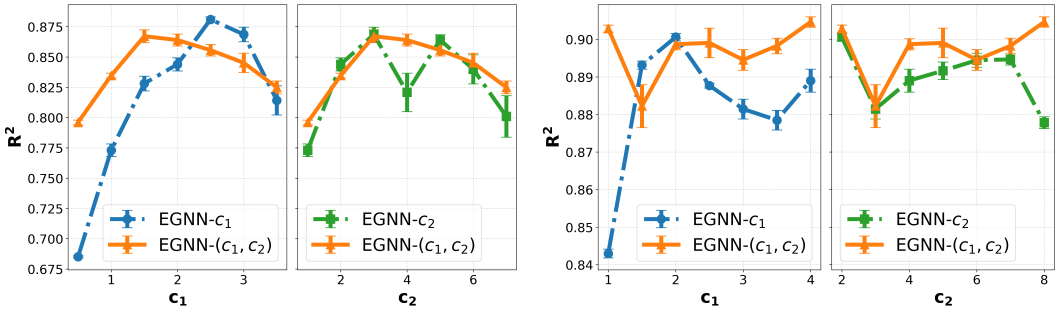
The percentage of retained edges, along with full implementation details such as feature construction, model configurations, and training protocols, are provided in Appendix B.1.

6.2 Results and Analysis

Table 1 presents the performance comparison of our GINE-Gt model against multiple baselines. These include a series of single-graph GINE-c models as well as several non-graph-based methods: Kernel Ridge Regression (KRR), Deep Tensor Neural Network (DTNN) [25], and the state-of-the-art invariant feature model, DS-CI+ [10]. Results for KRR and DTNN are taken from prior work [22].



(a) CAMELS: Ω_{m} (b) CAMELS: σ_8



(c) CAMELS-SAM: Ω_{m} (d) CAMELS-SAM: σ_8

Figure 2: Performance comparison on the CAMELS and CAMELS-SAM datasets for cosmological parameter prediction. We plot the coefficient of determination (R^2, higher is better) for our multi-view EGNN-Gt model against two single-graph baselines. The left plot in each subfigure compares against the strong-connection baseline (EGNN-c_1), while the right plot compares against the dense-graph baseline (EGNN-c_2). All results are reported as the mean and standard error over 10 runs.

Our proposed GINE-Gt model is the top-performing method, achieving the best Mean Absolute Error (MAE) on 11 of the 14 prediction targets. The best single-graph model, GINE-2, demonstrates the importance of focusing on the strong interactions compared to the full-graph GINE-0. However, our results suggest that weak interactions are also relevant.

While our multi-view approach GINE-Gt outperforms these baselines, we note that GINE-2 achieves better performance on three targets. This suggests that these specific properties are predominantly governed by strong, short-range interactions. In such cases, the global context provided by weaker interactions offers limited benefit and may introduce a small amount of non-essential information. This highlights the potential for a more adaptive partitioning mechanism that a flexible, learnable threshold, rather than our current fixed one, could allow the model to dynamically balance the two views and further improve the multi-view graph-tuple framework's performance.

7 Cosmological Parameter Prediction

We evaluate our EGNN-Gt model on the CosmoBench benchmark [9], specifically using the CAMELS (TNG) and the CAMELS-SAM cosmological point cloud datasets. Each sample in these datasets is a cosmological simulation cloud where nodes represent dark matter halos or galaxies. The model input is the matrix of 3D halo (galaxy) positions representing their present-day configuration, $X \in \mathbb{R}^{N\times 3}$, where N is the number of halos in the point cloud.

The primary task is a cloud-level regression problem, where the model infer from the present-day positions X about the cosmological parameters $y = (\Omega_m, \sigma_8)$ that control the evolution of halos (galaxies). The performance on this task is measured using the coefficient of determination (R^2), defined as:

$$R^2 = 1 - \frac{\sum_{i=1}^{n_{\text{test}}} (f(X_i) - y_i)^2}{\sum_{i=1}^{n_{\text{test}}} (\bar{y} - y_i)^2}, \tag{15}$$

where R^2 evaluates the cosmology parameter prediction and higher R^2 indicates better model fit.

7.1 Graph Construction

Insights from prior work [9] and our own preliminary experiments suggest that informative halo interactions typically occur within a radius range approximately from 0 to 10 Mpc/h. To establish strong baselines, we evaluated two types of single-graph models. First, we form a series of strong-connection baseline, EGNN-c_1 by connecting halos within a cutoff radius c_1. Second, we create dense-graph baselines, EGNN-c_2 using a larger radius $c_2 > c_1$. For our EGNN-Gt model, we implement the multi-view approach by partitioning the interaction space with two radii: a strong-connection radius c_1 and a weak-connection radius c_2. To explore the benefits of multi-view processing while maintaining a simple and interpretable relationship between the two scales, we enforce a fixed ratio by setting $c_2 = 2c_1$ in all our experiments. This allows the model to simultaneously capture immediate local neighborhoods ($\mathcal{G}_1$) and broader, second-order contextual regions ($\mathcal{G}_2$). The specific ranges for the systematic search over c_1 for each task, along with full implementation details, are provided in Appendix B.2.

7.2 Results and Analysis

Using the search ranges for c_1 established in our experimental setup, we present the performance of our multi-view EGNN-Gt model in Figure 2, with all results shown as mean R^2 values with standard errors. Each row corresponds to a dataset, and each column to a target cosmological parameter. The plots in each subfigure provide a direct comparison: the left plot shows the performance of our multi-view model, EGNN-(c_1, c_2), against its corresponding strong-connection baseline, EGNN-c_1, as a function of the radius c_1. The right plot shows a similar comparison against the dense-graph baseline EGNN-c_2 as a function of the radius c_2.

Across both datasets and target parameters, our multi-view EGNN-Gt model achieve better performance than single-graph baselines in most cases. Specifically, on the CAMELS dataset (Figures 2a and 2b), our multi-view method EGNN-Gt outperforms the corresponding single-graph baselines, i.e., strong-connection and dense-graph baselines (blue and green lines) at nearly every tested radius, with performance being, at worst, comparable in very few instances.

On the CAMELS-SAM dataset (Figures 2c and 2d), our EGNN-Gt model still exhibits better performance at most tested radius. However, there are instances where the single-graph baselines achieve better results. This does not indicate a failure of the multi-view method. Instead, we attribute this to the constraint of our fixed ratio, $c_2 = 2c_1$. The number of edges in a radius graph grows non-linearly with the radius (approximately as R^3 in 3D space). Therefore, a simple linear scaling between the two radii may not always capture the optimal balance of information density from the strong- and weak-connection graphs. This suggests that exploring adaptive or non-linear relationships between c_1 and c_2 is a promising direction for future work.

A second key design parameter is the number of partitions in the graph-tuple. In this work we intentionally instantiate the graph-tuple with $k = 2$ views corresponding to the strong and weak graphs. This two-scale design already captures the main separation between primary and contextual interactions in our dense graphs, while keeping the number of intra- and inter-scale message-passing paths manageable. Allowing more than two partitions would increase computational and tuning complexity by introducing additional operators and paths, but it could provide greater flexibility to capture diverse interactions. Beyond multi-scale analysis, our framework is also naturally suited to heterogeneous graphs, where each of the k views may correspond to a distinct relation type. Systematically exploring these higher-order, multi-relational graph-tuples is another promising avenue for future research.

8 Conclusion

In this work, we introduced the multi-view graph-tuple framework to address a fundamental challenge of applying GNNs to data with continuous relationships. Standard single-graph approaches face a difficult trade-off: either constructing a weak-connection graph via thresholding, which inevitably discards contextual information, or using the full-graph (complete) graph, which often incurs higher computational costs. Our framework resolves this limitation by explicitly partitioning the interaction space into a graph tuple, comprising a strong-connection graph for primary interactions and a weak-connection graph for global context, and performing (heterogenous) message-passing in parallel

to maintain efficiency. We theoretically show the expressivity improvements of our multi-view graph-tuple model over the single-graph models. We also empirically validate our framework through experiments on molecular property prediction and cosmological parameter prediction, showing that our multi-view approach can achieve an overall better performance against single-graph baselines.

As a proof of concept, we create multiple graph views using a fixed partitioning strategy depending on the edge feature values, which may be sub-optimal, as observed in the cosmological parameter prediction experiments. Future work could explore adaptive mechanisms, such as a learnable threshold or flexible relationships between scales, to allow the multi-view graph-tuple framework to tailor its structure to the specific task and data. Another interesting direction is to apply our framework for other dense-graph applications, such as brain connectomes and combinatorial optimization problems.

References

[1] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34 (4):18–42, 2017.

[2] TN Kipf. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[3] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

[4] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 639–648, 2020.

[5] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. Pmlr, 2017.

[6] Tian Xie and Jeffrey C Grossman. Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties. *Physical review letters*, 120(14):145301, 2018.

[7] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

[8] 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.

[9] Ningyuan Huang, Richard Stiskalek, Jun-Young Lee, Adrian E Bayer, Charles C Margossian, Christian Kragh Jespersen, Lucia A Perez, Lawrence K Saul, and Francisco Villaescusa-Navarro. Cosmobench: A multiscale, multiview, multitask cosmology benchmark for geometric deep learning. *arXiv preprint arXiv:2507.03707*, 2025.

[10] Ben Blum-Smith, Ningyuan Huang, Marco Cuturi, and Soledad Villar. Learning functions on symmetric matrices and point clouds via lightweight invariant features. *CoRR*, 2024.

[11] Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. Graph transformer networks. *Advances in neural information processing systems*, 32, 2019.

[12] Kaveh Hassani and Amir Hosein Khasahmadi. Contrastive multi-view representation learning on graphs. In *International conference on machine learning*, pages 4116–4126. PMLR, 2020.

[13] Chanyoung Park, Donghyun Kim, Jiawei Han, and Hwanjo Yu. Unsupervised attributed multiplex network embedding. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 5371–5378, 2020.

[14] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer, 2018.

[15] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. Heterogeneous graph attention network. In *The world wide web conference*, pages 2022–2032, 2019.

[16] Mauricio Velasco, Kaiying O'Hare, Bernardo Rychtenberg, and Soledad Villar. Graph neural networks and non-commuting operators. *Advances in neural information processing systems*, 37:95662–95691, 2024.

[17] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. *arXiv preprint arXiv:1905.12265*, 2019.

[18] 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.

[19] Yilin He, Chaojie Wang, Hao Zhang, Bo Chen, and Mingyuan Zhou. A variational edge partition model for supervised graph representation learning. *Advances in Neural Information Processing Systems*, 35:12339–12351, 2022.

[20] Ziqiao Zhang, Jihong Guan, and Shuigeng Zhou. Fragat: a fragment-oriented multi-scale graph attention model for molecular property prediction. *Bioinformatics*, 37(18):2981–2987, 2021.

[21] Meire Fortunato, Tobias Pfaff, Peter Wirnsberger, Alexander Pritzel, and Peter Battaglia. Multiscale meshgraphnets. *arXiv preprint arXiv:2210.00612*, 2022.

[22] Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018.

[23] Lorenz C Blum and Jean-Louis Reymond. 970 million druglike small molecules for virtual screening in the chemical universe database gdb-13. *Journal of the American Chemical Society*, 131(25):8732–8733, 2009.

[24] Grégoire Montavon, Matthias Rupp, Vivekanand Gobre, Alvaro Vazquez-Mayagoitia, Katja Hansen, Alexandre Tkatchenko, Klaus-Robert Müller, and O Anatole Von Lilienfeld. Machine learning of molecular electronic properties in chemical compound space. *New Journal of Physics*, 15(9):095003, 2013.

[25] Kristof T Schütt, Farhad Arbabzadah, Stefan Chmiela, Klaus R Müller, and Alexandre Tkatchenko. Quantum-chemical insights from deep tensor neural networks. *Nature communications*, 8(1):13890, 2017.

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

A Full Details for Theoretical Analysis

This section provides the complete definitions, lemmas, and proofs for the theoretical results presented in Section 5.

A.1 Setup and Model Classes

Let $S_1, S_2 \in \mathbb{R}^{n\times n}$ be two graph shift operators on a common node set, representing two different connection strenghts (e.g., strong and weak). We also define a dense-graph operator $S_{\text{dense}} = S_1 + S_2$ which includes both the strong and weak connections. For a fixed polynomial degree bound $m \in \mathbb{N}$, we consider three classes of linear graph filters.

Definition 1 (Filter Classes)**.** *The strong-connection (S_1-based), dense-graph, and multi-view graph tuple filter classes are defined as:*

$$H_1(m) = \Big\{\, p(S_1) = \sum_{k=0}^{m} a_k S_1^k \,\Big\},$$
$$H_0(m) = \Big\{\, q(S_{\text{dense}}) = \sum_{k=0}^{m} b_k (S_1 + S_2)^k \,\Big\},$$
$$H_{\text{Gt}}(m) = \Big\{\, g(S_1, S_2) = \sum_{w\in\mathcal{W}_{\le m}} c_w\, w(S_1, S_2) \,\Big\},$$

where $\mathcal{W}_{\le m}$ is the set of all words of length up to m formed from the symbols $\{1, 2\}$, and $w(S_1, S_2) = \prod_{t=1}^{|w|} S_{w_t}$.

Remark 1 (Degrees of Freedom)**.** *The classes $H_1(m)$ and $H_0(m)$ are defined by $m + 1$ free parameters each. In contrast, the multi-view graph-tuple class $H_{\text{Gt}}(m)$ is defined by $2^{m+1} - 1$ parameters, reflecting its ability to assign an independent coefficient to every possible path of length up to m.*

A.2 Expressivity and Risk Dominance

Our analysis is grounded in a standard linear data model and the corresponding prediction risk.

Assumption 1 (Data model and risk)**.** *The input $x \in \mathbb{R}^n$ is zero-mean with covariance $\Sigma \succ 0$. For a linear predictor $M \in \mathbb{R}^{n\times n}$ and target $y^\star$, we define the risk as*

$$R(M) = \mathbb{E}\big[\|Mx - y^\star\|_2^2\big].$$

We assume $y^\star = M^\star x + \varepsilon$ with $\mathbb{E}[\varepsilon \mid x] = 0$ for some oracle predictor $M^\star \in \overline{H_{\text{Gt}}(m)}$, where $\overline{H_{\text{Gt}}(m)}$ is the closure of $H_{\text{Gt}}(m)$.

For matrices $A, B \in \mathbb{R}^{n\times n}$, we define the weighted inner product, Frobenius norm, and the corresponding distance:

$$\langle A, B\rangle_\Sigma = \operatorname{tr}(B^\top A\,\Sigma), \qquad \|A\|_{\Sigma,F}^2 = \langle A, A\rangle_\Sigma, \qquad \operatorname{dist}_\Sigma(A, B) = \|A - B\|_{\Sigma,F}.$$

We use $\operatorname{dist}_\Sigma(M, \mathcal{C}) = \inf_{M'\in\mathcal{C}} \operatorname{dist}_\Sigma(M, M')$ to denote the distance from a matrix M to a set $\mathcal{C}$.

Our analysis relies on two lemmas. The first recasts the prediction risk as a best approximation problem in a matrix space, and the second provides a combinatorial expansion.

Lemma 1 (Risk Decomposition)**.** *Under Assumption 1, for any predictor M, the risk can be decomposed as $R(M) = R(M^\star) + \|M - M^\star\|_{\Sigma,F}^2$.*

Proof. By definition, $R(M) = \mathbb{E}[\|(M - M^\star)x - \varepsilon\|_2^2]$. Expanding this and taking the expectation, the cross-term $\mathbb{E}[\langle (M - M^\star)x, \varepsilon\rangle]$ vanishes due to the condition $\mathbb{E}[\varepsilon|x] = 0$. The remaining terms are $\mathbb{E}[\|(M - M^\star)x\|_2^2] = \|M - M^\star\|_{\Sigma,F}^2$ and $\mathbb{E}[\|\varepsilon\|_2^2] = R(M^\star)$, which yields the result. □

Lemma 2 (Noncommutative Binomial Expansion)**.** *For any integer $k \ge 0$, we have $(S_1 + S_2)^k = \sum_{w\in\mathcal{W}_k} w(S_1, S_2)$, where $\mathcal{W}_k$ is the set of words of length k.*

These lemmas allow us to establish our main theoretical results concerning the expressivity and risk of the multi-view graph-tuple class.

Proposition 1 (Expressivity)**.** *For any $m \geq 0$, the multi-view graph-tuple class contains the strong-connection and dense-graph classes: $H_1(m) \subseteq H_{\mathrm{Gt}}(m)$ and $H_0(m) \subseteq H_{\mathrm{Gt}}(m)$. If the operators do not commute, i.e., $[S_1, S_2] = S1S2 - S2S1 \neq 0$, and $m \geq 2$, this latter inclusion is strict: $H_0(m) \subsetneq H_{\mathrm{Gt}}(m)$.*

Proof. The inclusion $H_1(m) \subseteq H_{\mathrm{Gt}}(m)$ is trivial by construction. The inclusion $H_0(m) \subseteq H_{\mathrm{Gt}}(m)$ follows from Lemma 2, which shows that any polynomial in S_{dense} is a sum over words with coefficients tied according to their length ($c_w = b_{|w|}$). The inclusion is strict under non-commutativity because an element like the commutator $[S_1, S_2]$ is in $H_{\mathrm{Gt}}(m)$ but not in $H_0(m)$, as the latter requires the coefficients of S_1S_2 and S_2S_1 to be equal. □

Remark 2 (The Commuting Case)**.** *Even if $[S_1, S_2] = 0$, $H_0(m)$ generally remains a proper subset of $H_{\mathrm{Gt}}(m)$ unless S_1 and S_2 are algebraically dependent (e.g., $S_2 = cS_1$), because $H_0(m)$ still enforces coefficient tying across all same-degree terms.*

The greater expressivity of the multi-view graph-tuple class can translate into improved generalization performance.

Proposition 2 (Oracle risk dominance)**.** *Let $\mathcal{U}(m) = \overline{H_{\mathrm{Gt}}(m)}$ and $\mathcal{V}(m) = \overline{H_0(m)}$ be the closures of the multi-view graph-tuple and dense-graph classes. If $m \geq 2$ and the oracle predictor $M^\star$ has a non-zero component in the orthogonal complement of $\mathcal{V}(m)$ within $\mathcal{U}(m)$ (i.e., $\Pi_{\mathcal{V}(m)^\perp}(M^\star) \neq 0$), then the multi-view graph-tuple class achieves a strictly lower oracle risk. The performance gap is given precisely by:*

$$\inf_{q \in H_0(m)} R(q) - \inf_{g \in H_{\mathrm{Gt}}(m)} R(g) = \|\Pi_{\mathcal{V}(m)^\perp}(M^\star)\|^2_{\Sigma,F} > 0.$$

Proof. By Lemma 1, the minimum risk for a closed class $\mathcal{C}$ is $\inf_{M \in \mathcal{C}} R(M) = R(M^\star) + \mathrm{dist}_\Sigma(M^\star, \mathcal{C})^2$. Subtracting the expressions for $\mathcal{C} = \mathcal{U}(m)$ and $\mathcal{C} = \mathcal{V}(m)$ yields the risk gap:

$$\inf_{q \in H_0(m)} R(q) - \inf_{g \in H_{\mathrm{Gt}}(m)} R(g) = \mathrm{dist}_\Sigma(M^\star, \mathcal{V}(m))^2 - \mathrm{dist}_\Sigma(M^\star, \mathcal{U}(m))^2.$$

Since $M^\star \in \mathcal{U}(m)$ by assumption, $\mathrm{dist}_\Sigma(M^\star, \mathcal{U}(m))$ is zero. Since the shortest distance from $M^\star$ to the subspace $\mathcal{V}(m)$ is the norm of its component in the orthogonal complement, $\mathrm{dist}_\Sigma(M^\star, \mathcal{V}(m))^2 = \|\Pi_{\mathcal{V}(m)^\perp}(M^\star)\|^2_{\Sigma,F}$. Substituting this gives the claimed result. As the proposition's premise is that this projection is non-zero, the squared norm is strictly positive. □

Corollary 1 (Sufficient Condition for Strict Improvement)**.** *The condition for strict risk dominance in Proposition 2 is satisfied if $m \geq 2$, the operators do not commute, $[S_1, S_2] \neq 0$, and the degree-2 component of $M^\star$ contains a non-zero multiple of the commutator $[S_1, S_2]$.*

Proof. This follows because, as established in the proof of Proposition 1, the commutator $[S_1, S_2]$ is an element of the orthogonal complement $\mathcal{V}(m)^\perp$. If $M^\star$ contains a non-zero multiple of this element, its projection onto this subspace, $\Pi_{\mathcal{V}(m)^\perp}(M^\star)$, must be non-zero. □

In summary, the ability of our multi-view graph-tuple framework to assign distinct weights to distinct interaction paths makes it more expressive than models constrained to polynomials of a single operator. This greater expressivity guarantees a lower or equal modeling risk for target functions that satisfies our modeling assumptions: the oracle predictor $M^\star$ that is expressible within our framework (i.e., $M^\star \in \overline{H_{\mathrm{Gt}}(m)}$ in Assumption 1).

These theoretical results directly apply to the linear backbone of the GNNs used in our experiments. Specifically, our analysis focuses on these linear operators and does not consider the nonlinear activation functions applied to their outputs. While a full characterization of the nonlinearities is more complex, our analysis provides a clean conceptual baseline: any nonlinear architecture built upon this backbone inherits the fundamental expressivity gap between the underlying operator classes. Extending such guarantees to fully nonlinear settings is an interesting but technically nontrivial direction that we leave for future work.

B Implementation Details

B.1 Molecular Property Prediction

B.1.1 Feature Construction.

Since the QM7b dataset is feature-scarce, we first construct node and edge features from the molecule's Coulomb matrix, X. Following prior work [10], we derive initial features directly from the Coulomb matrix entries. Specifically, we apply a "binary expansion" technique to expand the scalar diagonal entries (X_{ii}) and off-diagonal entries (X_{ij}) into 100-dimensional vectors, which serve as the initial node and edge features, respectively. These raw scalar values are then projected into a 100-dimensional hidden space by learnable encoders.

Table 2: Percentage of remaining strong edges under different Coulomb thresholds (c).

Threshold (c)	0	0.5	1	2	2.5
Remaining Edges (%)	100.00	68.02	50.10	25.89	24.90

B.1.2 Model Architecture and Training.

All GNN models are constructed with two GNN layers and a hidden dimension of 100. The MLPs within each GINEConv layer consist of two linear layers separated by a ReLU activation. The edge encoders within each path of the GINE-Gt model are implemented as single linear layers. For graph-level prediction, we apply a global mean pooling to the node features of the final GNN layer, and the resulting graph vector is passed through a 3-layer MLP with a ReLU activation to produce the final output.

For training, all models use a batch size of 128. We train the models by minimizing the L1 Loss (Mean Absolute Error) using the Adam optimizer [26] with an initial learning rate of 5×10^{-3} and weight decay of 10^{-5}. A cosine–plateau scheduler reduces the learning rate by a factor of 0.8 after five epochs without validation improvement (minimum 10^{-5}). Early stopping is triggered after 20 idle epochs or when the run reaches a maximum of 1000 epochs.

B.1.3 Evaluation Protocol and Environment.

To ensure a robust evaluation, we employ a stratified ten-fold cross-validation scheme. For each fold, we reserve 10% of the data for testing, while the remainder is split into a 9:1 train/validation ratio. Then we report the mean and standard error of the Mean Absolute Error (MAE) across the ten test set folds. These experiments were performed on a MacBook Air (15-inch, 2023) featuring an Apple M2 processor and 16 GB of unified memory, running macOS Ventura (13.4). All models were implemented in PyTorch.

B.2 Cosmological Parameter Inference

B.2.1 Feature Construction.

For all constructed graphs, since the dark matter halos are treated as identical particles, the initial feature for each node is set to a 1-dimensional unit vector ($h_i^{(0)} = [1]$). This vector is then projected into the model's hidden dimension by an embedding layer. Edge attributes are dynamically generated by expanding the Euclidean distance between halos into a 32-dimensional feature vector using a Radial Basis Function (RBF) encoding.

B.2.2 Model Architecture and Training.

Our EGNN-Gt models are constructed with 3 layers and a hidden dimension of 96. The MLPs within each EGCL operator use the SiLU activation function. For the primary task of cosmological parameter prediction, a global mean pooling is applied to the final node features, and the resulting graph-level representation is passed through a 2-layer MLP to produce the output. We train all models for a maximum of 300 epochs by minimizing the Mean Squared Error (MSE) loss, using a batch size

Table 3: Search space for the strong-connection cutoff radius (c_1) for different datasets and target parameters. The search for c_1 is performed with a step of 0.5. The weak-connection radius (c_2) is always set to $2c_1$, so its corresponding search is performed with a step of 1.0.

Dataset	**Target Parameter**	$\mathbf{c_1}$ **Values (Mpc/h)**	$\mathbf{c_2}$ **Values (Mpc/h)**
CAMELS (TNG)	Ω_m	0.5, 1.0, ..., 3.5	1.0, 2.0, ..., 7.0
CAMELS (TNG)	σ_8	1.5, 2.0, ..., 5.0	3.0, 4.0, ..., 10.0
CAMELS-SAM	Ω_m	0.5, 1.0, ..., 3.5	1.0, 2.0, ..., 7.0
CAMELS-SAM	σ_8	1.0, 1.5, ..., 4.0	2.0, 3.0, ..., 8.0

of 8. The AdamW optimizer is used with an initial learning rate of 5×10^{-4} and a weight decay of 1×10^{-5}. The learning rate is dynamically adjusted using a ReduceLROnPlateau scheduler, which reduces it by a factor of 0.7 if the validation loss does not improve for 5 consecutive epochs, down to a minimum of 1×10^{-5}.

B.2.3 Evaluation Protocol and Environment.

The datasets are randomly partitioned into training (60%), validation (20%), and test (20%) sets. To ensure the robustness of our findings, each experiment is repeated 10 times with different random seeds, and we report the mean and standard error of the performance metrics on the test set. All experiments were conducted on a single NVIDIA RTX 6000 Ada Generation GPU, equipped with 48 GB of VRAM.