TopoMole: Topological Message Passing Meets Hyperedge Messages

Pablo Martínez Crespo
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
Gothenburg, SE
pabloma@chalmers.se

Robert S. Jordan
Technology Research
Intel Corporation
Portland, OR, USA
rob.jordan@intel.com

Marisa Gliege
Chief Technology Office
EMD Electronics
Tempe, AZ, USA
marisa.gliege@emdgroup.com

Santiago Miret
Intel Labs
Intel Corporation
Santa Clara, CA, USA
smiret@lila.ai

Vijay Kris Narasimhan
Chief Technology Office
EMD Electronics
San Jose, CA, USA
vijay.narasimhan@emdgroup.com

Rocío Mercado
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
Gothenburg, SE
rocio.mercado@chalmers.se

Abstract

Message-passing neural networks (MPNNs) rely on pairwise edges and local neighborhoods to perform molecular property prediction tasks. As the number of atoms in a system increases, higher-order structures and long-range interactions become increasingly influential. Models for predicting macroscopic material properties and molecular properties of medium-sized molecules would hence benefit from frameworks that can naturally represent the information exchange in these systems. Precisely, topological message passing (TMP) and hypergraph neural networks (HNNs) extend MPNNs to operate on complex data relations by enabling the joint representation of nodes, edges, and higher-dimensional cells. In this workshop paper, we introduce TopoMole, the first open-source JAX package for TMP that supports the generation and aggregation of messages for all adjacency relations within a cell complex together with hyperedge representation. We demonstrate its utility in two molecular property prediction tasks, highlighting its potential in AI-driven materials discovery.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

1 Introduction

Modeling the complex, multiscale interactions that underpin material properties remains a fundamental challenge in AI-driven materials design. Graph neural networks (GNNs) and message-passing neural networks (MPNNs) have shown strong performance in a variety of scientific domains such as chemistry [9, 13], materials science [17, 11], physics [3, 7, 18], and biology [6, 5], to name just a few examples. However, they are fundamentally limited to base their predictions on pairwise relationships and local neighborhoods in the data. In fact, many important phenomena—such as non-additive electronic effects or collective long-range properties—cannot be adequately modeled using standard GNNs [1, 10]. This is a problem in fields like materials science, where not all relevant interactions can be thought of as pairwise interactions between atoms. Although these may offer sufficient predictive power for the automated learning of properties of small molecules or simple periodic structures, long-range electrostatic forces and many-body quantum mechanical effects become more influential as the number of atoms in a system grows. Therefore, property prediction of macroscopic systems and medium-sized molecules would benefit from frameworks that can represent rings, supramolecular structures, and high-order interactions that naturally arise in them.

To address this gap, topological neural networks (TNNs) have emerged as a generalization of GNNs to model higher-dimensional structures and group interactions. We refer the reader to the work by Papillon et al. [14] for an extended survey on this topic. Within TNNs, one of the main branches is topological message passing (TMP) on cellular complexes. Essentially, TMP generalizes the MPNN formulation by incorporating not only nodes (0-cells) and edges (1-cells), but also higher-order elements such as faces (2-cells) and volumes (3-cells), enabling a natural representation of group structures. The models introduced by Bodnar et al. [4] and Giusti et al. [8] have been shown to outperform GNNs in tasks involving molecules from the ZINC, Mol-HIV and TUD datasets. Specifically, Bodnar et al. [4] introduced CW Networks and proved their higher expressivity through several Weisfeiler-Lehman tests. Furthermore, Giusti et al. [8] proved through a series of experiments on the ZINC dataset that long-range interactions can be better captured due to a broader variety of messages allowed by these richer representations, whereas GNNs are limited in their possibilities for information flow. However, existing TMP implementations lack support for coboundary messages, or flexibility in MP operations. Besides, they rely on PyTorch, limiting the possibilities for development of new models by the community.

A related line of work explores hypergraph-based generalizations of GNNs. In particular, the work by Kim et al. [12] introduces the first attempt to a maximally expressive, efficient realization of message passing between hyperedges: the equivariant hypergraph neural network (EHNN). This approach provides a complementary perspective to TMP: whereas TMP leverages cellular complexes to encode higher-dimensional topological structures, EHNNs focus on hypergraphs to capture many-body interactions. Both frameworks highlight the importance of moving beyond simple pairwise edges to model complex relational structures. To the best of our knowledge, this kind of layer has not been tested for molecular property prediction.

In this work, we present TopoMole, the first JAX-based open-source package for topological message passing, addressing the above limitations via the following key contributions:

- **First implementation of topological message passing in JAX at the cellular level**, with support for coboundary adjacencies.
- **First implementation of a hyperedge messaging layer in JAX**, enabling direct modeling of non-pairwise interactions.
- **First integration of the two frameworks**, providing a unified framework for both higher-order and long-range interaction modeling in molecules and materials.
- **Demonstration on materials property prediction**, showcasing robust performance on two common/widespread tasks on a popular benchmark dataset for computational chemistry and materials science.

The package is available open-source on the following repository: `https://github.com/pablomcrespo/JAX_TMP`.

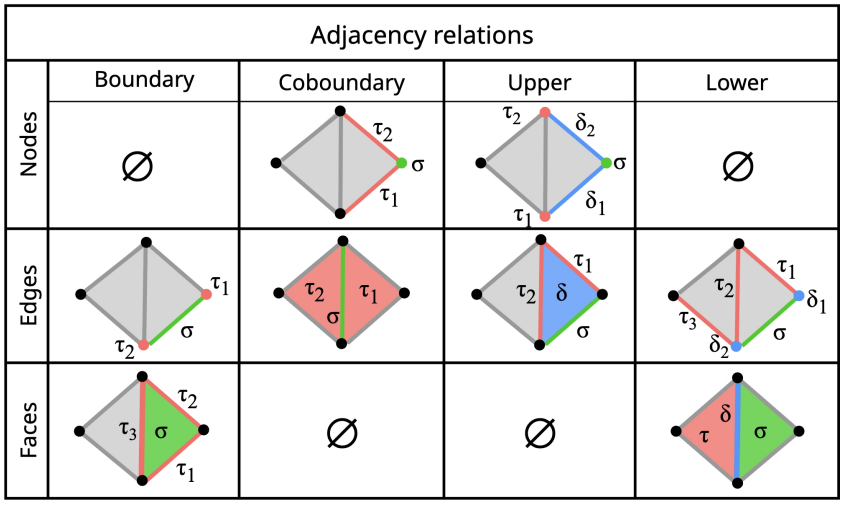

Figure 1: Table of adjacency relations in cell complexes, illustrated for an example complex. Reference cells σ are colored in green, adjacent cells τ are colored in red and intermediary cells δ are colored in blue.

2 Methods

2.1 Definitions

The following definitions are intended as a low-level introduction to the necessary concepts, summarized in Figure 1. For extended definitions, we refer to Bodnar et al. [4] and Kim et al. [12].

Cellular complex and cell: a cellular complex X is a topological space composed of a set of subspaces σ, called cells. Every cell is homeomorphic to $\mathbb{R}^k$ for some k, which defines its dimension, and intersects other cells. For example, nodes in a graph are homeomorphic to $\mathbb{R}^0$, edges are homeomorphic to $\mathbb{R}^1$, and faces are homeomorphic to $\mathbb{R}^2$; an edge intersects its two endpoint nodes as well as the faces it is part of.

Cochain and cochain complex: a cochain is a map $\psi : X \to F$ from a cellular complex to a feature space. For the purposes of this work, we define a k-cochain $\psi^{(k)} : X^{(k)} \to F^{(k)}$ as the map that assigns features to all k-cells in a complex. A cochain complex of dimension K, $\psi^{(:K)}$, is the set of all cochains that act on a cellular complex. We define $\mathbf{x}_\sigma = \psi(\sigma)$ as the feature vector of σ.

Boundary and coboundary adjacency: two intersecting cells define a boundary relation with respect to each other. The higher dimensional cell is a coboundary of the lower dimensional cell, whereas the lower dimensional cell is a boundary of the higher dimensional one. We denote the set of boundary cells of a given cell σ by $\mathcal{B}(\sigma)$. Similarly, we denote the set of coboundary cells of σ by $\mathcal{C}(\sigma)$. As an example, if σ is an edge, $\mathcal{B}(\sigma)$ contains its endpoint nodes, while $\mathcal{C}(\sigma)$ contains the faces for which the edge is a boundary. Furthermore, given two cells of the same dimension, σ and τ, we refer to the cells that they share in a boundary or coboundary by $\mathcal{B}(\sigma, \tau)$ or $\mathcal{C}(\sigma, \tau)$, respectively. More precisely, $\mathcal{B}(\sigma, \tau) = \mathcal{B}(\sigma) \cap \mathcal{B}(\tau)$ and $\mathcal{C}(\sigma, \tau) = \mathcal{C}(\sigma) \cap \mathcal{C}(\tau)$.

Upper and lower adjacency: two cells of the same dimension, σ and τ, are upper-adjacent if and only if $\mathcal{C}(\sigma, \tau) \neq \emptyset$. Similarly, they are lower-adjacent if and only if $\mathcal{B}(\sigma, \tau) \neq \emptyset$. We denote the set of upper-adjacent cells of σ by $\mathcal{N}_\uparrow(\sigma)$, and the set of lower-adjacent cells by $\mathcal{N}_\downarrow(\sigma)$. We refer to σ as the reference cell of the upper/lower adjacency. We refer to the cells that connect σ with its upper/lower adjacent cells as intermediary cells.

Hypergraph: a hypergraph $\mathcal{H}(\mathcal{V}, \mathcal{E})$ is a set of nodes $\mathcal{V}$ that are connected by sets of varying sizes, called hyperedges $\mathcal{E}$. Each hyperedge can contain an arbitrary number of nodes. Similarly, each node can be contained in an arbitrary number of hyperedges.

Sequence representation of a hypergraph: A hypergraph is k-uniform if all its hyperedges contain k nodes. A hypergraph with max hyperedge order K admits a sequence representation

$\mathcal{H}(\mathcal{V}, \mathcal{E}^{(k)}, \mathbf{X}^{(k)})_{k \leq K}$, where $\mathcal{E}^{(k)}$ is the set of all k-order hyperedges and $\mathbf{X}^{(k)}$ is the stack of their features.

Tensor representation of a k-uniform hypergraph: a k-uniform hypergraph with m nodes can be represented by a tensor $\mathbf{A}^{(k)} \in \mathbb{R}^{m^k \times d}$ defined by

$$\mathbf{A}^{(k)}_{(i_1,...,i_k)} = \begin{cases} \mathbf{x}_e \text{ if } e = \{i_1, ..., i_k\} \in \mathcal{E}^{(k)} \\ 0 \quad \text{else} \end{cases}, \tag{1}$$

where $\mathbf{x}_e$ is the feature vector of hyperedge e. In simple terms, $\mathbf{A}^{(k)}$ corresponds to the feature matrix of k-order relations, with $k = 1$ being the node feature matrix.

2.2 JAX Implementation

Topological message passing. The package introduced in this work is a modular framework for topological message passing. This means that messages can be exchanged between intersecting cells up to arbitrary dimensions. For our package, graph neural networks are a particular case when the maximum dimension of the cells is $k = 1$. To the best of our knowledge, this is the first available JAX implementation of this kind of deep learning framework.

JAX was chosen as the platform for our implementation of TMP, mainly due to its advantages over PyTorch in terms of GPU/TPU acceleration, parallelization and automatic differentiation. The package provides the necessary objects and functions for data containers (`Cochain`, `Complex`), batches of data (`CochainBatch`, `ComplexBatch`) and appropriate padding for compatibility with just-in-time (JIT) compilation. Similarly, a base class with all the necessary sanity checks is included for inheritance when designing custom transformations and models. We use Flax NNX for parameter handling and model training.

Based on the adjacency relations defined above, this type of architecture supports four different message generation channels, with different possibilities for aggregation. These messages can be used to update the cell feature vectors with custom update functions:

$$\mathbf{x}_\sigma^{t+1} = U\left(\mathbf{x}_\sigma^t, m(\sigma, \tau, \delta)\right), \tag{2}$$

where the message aggregation types m are defined below.

Upper messages: they can be generated with any triplet of cells $(\sigma, \tau, \delta) : \delta \in \mathcal{C}(\sigma, \tau)$, and can be aggregated to the cells of either dimension. We distinguish between aggregation at the reference cell

$$m_\uparrow(\sigma \leftarrow \delta) = \underset{\delta \in \mathcal{C}(\sigma,\tau) \,:\, \tau \in \mathcal{N}_\uparrow(\sigma)}{\text{agg}} \left(M_\uparrow\left(\mathbf{x}_\sigma, \mathbf{x}_\tau, \mathbf{x}_\delta\right)\right), \tag{3}$$

and aggregation at the intermediary cell

$$m_\uparrow(\sigma \rightarrow \delta) = \underset{(\sigma,\tau) \,:\, \delta \in \mathcal{C}(\sigma,\tau)}{\text{agg}} \left(M_\uparrow\left(\mathbf{x}_\sigma, \mathbf{x}_\tau, \mathbf{x}_\delta\right)\right). \tag{4}$$

Coboundary messages: they can be generated by any pair of cells $(\sigma \in X^{(k)}, \tau \in X^{(k+1)})$ such that $\tau \in \mathcal{C}(\sigma, \delta)\ \forall \delta \in \mathcal{N}_\uparrow(\sigma)$, i.e., τ connects σ with some other cell in $X^{(k)}$. They can be aggregated to either the reference or the adjacent cell,

$$m_\mathcal{C}(\sigma \leftarrow \tau) = \underset{\tau \in \mathcal{C}(\sigma)}{\text{agg}} \left(M_\mathcal{C}\left(\mathbf{x}_\sigma, \mathbf{x}_\tau\right)\right), \tag{5}$$

$$m_\mathcal{C}(\sigma \rightarrow \tau) = \underset{\sigma \in \mathcal{B}(\tau)}{\text{agg}} \left(M_\mathcal{C}\left(\mathbf{x}_\sigma, \mathbf{x}_\tau\right)\right). \tag{6}$$

Lower messages: similar to the upper messages, but the adjacency between the cells is $(\sigma, \tau, \delta) : \delta \in \mathcal{B}(\sigma, \tau)$. We can also aggregate in either direction,

$$m_\downarrow(\sigma \leftarrow \delta) = \underset{\delta \in \mathcal{B}(\sigma,\tau) \,:\, \tau \in \mathcal{N}_\downarrow(\sigma)}{\text{agg}} \left(M_\downarrow\left(\mathbf{x}_\sigma, \mathbf{x}_\tau, \mathbf{x}_\delta\right)\right), \tag{7}$$

$$m_\downarrow(\sigma \rightarrow \delta) = \underset{(\sigma,\tau) \,:\, \delta \in \mathcal{B}(\sigma,\tau)}{\text{agg}} \left(M_\downarrow\left(\mathbf{x}_\sigma, \mathbf{x}_\tau, \mathbf{x}_\delta\right)\right). \tag{8}$$

Boundary messages: they can be generated by any pair of cells $(\sigma \in X^{(k)}, \tau \in X^{(k+1)})$ such that $\sigma \in \mathcal{B}(\tau, \delta)\ \forall \delta \in \mathcal{N}_{\downarrow}(\tau)$, meaning that σ connects τ with some other cell in $X^{(k+1)}$. They can be aggregated to either the reference or the adjacent cell,

$$m_{\mathcal{B}}(\sigma \leftarrow \tau) = \underset{\tau \in \mathcal{C}(\sigma)}{\text{agg}} \left(M_{\mathcal{B}}\left(\mathbf{x}_{\sigma}, \mathbf{x}_{\tau}\right)\right), \tag{9}$$

$$m_{\mathcal{B}}(\sigma \rightarrow \tau) = \underset{\sigma \in \mathcal{B}(\tau)}{\text{agg}} \left(M_{\mathcal{B}}\left(\mathbf{x}_{\sigma}, \mathbf{x}_{\tau}\right)\right). \tag{10}$$

In the above mentioned message passing schemes, M denotes an arbitrary function to map a set of cell features to a set of message vectors, which are later aggregated by an arbitrary aggregation function agg. These are defined later in Section 3.1 for four models.

Our implementation adapts and expands the original work by Bodnar et al. [4], based on PyTorch. The framework introduced here differs mainly from the original in the nomenclature of the messages. Whereas the original nomenclature dictates the flow of the messages, our nomenclature relates to the set of index lookup arrays used during the message generation. We refer the reader to Appendix A.1.1 for a deeper explanation of the differences.

Hypergraphs. While traditional message passing layers concatenate features of the involved cells in a pairwise manner, it is common to find higher order relations in molecular data, e.g., rings. A possible representation for a ring in a molecule is as a hyperedge with the bonds of the ring as nodes. A full interaction representation of this structure would need a factorial-scaling number permutations of indices to be fully expressive, which for rings of 6 or 7 bonds becomes computationally expensive. To maximize the expressivity for high-order interactions between bonds in a ring at an scalable cost, we embed these into permutation-covariant pairwise interactions for each bond pair within a ring.

Let X be the cellular complex representation of a molecule. Let $f \in X^{(2)}$ be the face representation of a ring in the molecule, and let $\mathcal{B}(f) = \{e_i \in X^{(1)}\}_{i=1}^{K}$ be the set of edges representing the bonds in the ring. Let $\mathbf{x}_f,\ \mathbf{x}_{e_i}$ be the corresponding feature vectors. Specifically, we embed the pairwise interaction between (e_1, e_2) into the ring features with,

$$\mathbf{x}_f(e_1, e_2) = \phi_3 \left(\sum_{\mathcal{I}=\{0,1,2\}} \phi_2 \left(\mathcal{I}, \sum_{\substack{k=\{1,K\} \\ k \geq \mathcal{I}}} \sum_{\mathbf{i}} \phi_1 \left(k, \mathbf{A}_{\mathbf{i}}^{(k)} \right) \right) \right), \tag{11}$$

where

$$\mathbf{A}_{\mathbf{i}}^{(k)} = \begin{cases} \mathbf{x}_{e_i} : e_i \in f, i \neq \{1,2\} & \text{if } \mathcal{I} = 0 \text{ and } k = 1 \\ \mathbf{x}_{e_1} \text{ or } \mathbf{x}_{e_2} & \text{if } \mathcal{I} = 1 \text{ and } k = 1 \\ \mathbf{x}_f & \text{else} \end{cases} \tag{12}$$

restricts the aggregation of features. The multi-index $\mathbf{i}$ follows hypergraph notation, referring to a single bond when $k = 1$ and to a ring when $k = K$ (a bond is a node and a ring is a hyperedge in this case). In simple terms, and following the notation of Kim et al. [12], $\mathcal{I} = 0$ combines features from the rest of the bonds in the ring. $\mathcal{I} = 1$ can be regarded as an interaction between the pair (e_1, e_2) and the ring itself. $\mathcal{I} = 2$ only takes the feature of the ring. ϕ_1, ϕ_2 and ϕ_3 are multi-layer perceptrons (MLPs) that share parameters for different values of $\mathcal{I}$ and k, which are auxiliary integers provided to the MLPs as a one-hot value.

This hyperedge embedding layer is based on the EHNN MLP layer (Eq. 7) from Kim et al. [12]; this type of layer is the first practical attempt to approximate the maximally expressive linear layer for general undirected hypergraphs. Whereas the original implementation is a composition of DeepSet layers that combine features of hyperedges of all orders and with different overlap of nodes, our approach has a different intent. Since we assume no permutation invariance of the cell features in the message generation, we use it to enrich the pairwise edge interactions. To the best of our knowledge, our model is the first to include this kind of layer for molecular property prediction.

2.3 Dataset

Since the capabilities of TMP for long-range interactions and bigger molecules have been shown in other works [4, 8], we chose the QM9 [15, 16] dataset for both showcasing the JAX implementation and benchmarking on a widely used dataset whether there would be advantages in predicting complex tasks for smaller molecules. From the original 133 885 molecules, we removed the 3 054 entries flagged by the dataset authors as failing a geometry consistency check, resulting in a working set of 130 831 molecules.

To create robust train/validation/test splits, we applied a Tanimoto similarity-based clustering procedure. Each molecule was first represented by a Morgan fingerprint (radius 2, 1 024-bit). A random sample of 5 000 molecules was then used to initialize Butina clusters with a minimum similarity threshold of 60%, yielding 4 857 clusters. The remaining molecules were assigned to the nearest cluster centroid. Finally, the clusters were randomly ordered and sequentially allocated to the training, validation, and test sets in a 80/10/10 ratio (104 689, 13 099, 13 043 molecules, respectively). This clustering-based splitting strategy ensures that molecules in different sets are structurally distinct, helping to prevent overfitting and enabling a more reliable assessment of model generalization.

2.4 Hardware and Compute

All models were trained using a single NVIDIA A40 GPU. Training 3 instances of each model for 1000 epochs required a total time under 8 hours (about 1 h 45 min per model) per learning task.

3 Experiments

To assess the effectiveness of (1) the cochain complex representation of molecules and (2) the hypergraph embedding layer for capturing higher-order interactions, particularly in rings, we trained four different models on two regression tasks from QM9. The first task involves predicting the specific heat capacity at constant volume (C_v), a key thermodynamic property linked to thermal stability and energy storage. The second task involves the prediction of the energy difference between the highest occupied molecular orbital (HOMO) and lowest occupied molecular orbital (LUMO)–also known as the HOMO-LUMO gap–which strongly correlates to the optical and electronic behavior of a material. Together, they offer a complementary test of the prediction of thermal/macroscopic and electronic/molecular properties.

3.1 Models

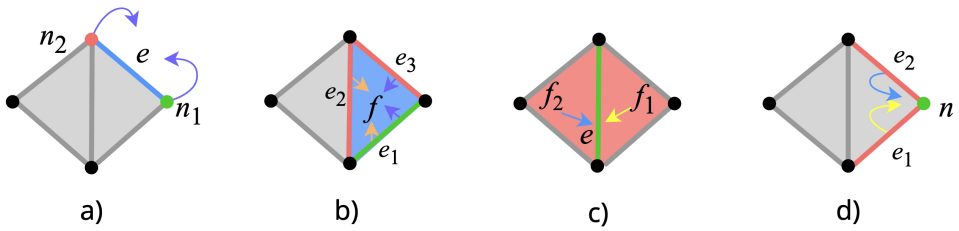

Figure 2: Diagrams illustrating the various message aggregation schemes used by the models. Cell colors follow the scheme of Figure 1. Arrow color represents different concatenated features for message generation. a) Message generated by upper neighborhood of node n_1 is aggregated to update edge e features. b) Messages generated by upper neighborhood of edge e_1 are aggregated to update face f features. c) Messages generated by the coboundary faces $f_1,\ f_2$ of edge e are used to update its features. d) Messages generated by coboundary edges $e_1,\ e_2$ of node n are used to update its features.

In all models, we embed the input representations of atoms, bonds, and rings. Atoms are represented by nodes with feature vectors $\mathbf{x}_n \in \mathbb{R}^6$, a concatenation of their Mulliken charge (given by QM9) with a one-hot encoding of their element (out of C, H, O, N, or F). Bonds are represented by edges with feature vector $\mathbf{x}_e \in \mathbb{R}^7$, their length concatenated to the components of the inertia tensor defined by its endpoint atoms using their Mulliken charges as masses. Similarly, rings are represented by

faces with feature vector $\mathbf{x}_f \in \mathbb{R}^6$, the components of the inertia tensor defined by its atoms, using their Mulliken charges as masses.

The models we trained use messages generated with upper adjacency and coboundary relations (Eqs. 3–6, Fig. 2). In particular, the message functions M are

$$M_{\uparrow}\left(\mathbf{x}_\sigma, \mathbf{x}_\tau, \mathbf{x}_\delta\right) = g_o\left(\mathrm{ReLU}\left(g_h\left(\mathbf{x}_\sigma | \mathbf{x}_\tau | \mathbf{x}_\delta\right)\right)\right), \tag{13}$$

and

$$M_{\mathcal{C}}\left(\mathbf{x}_\sigma, \mathbf{x}_\tau\right) = g_o\left(\mathrm{ReLU}\left(g_h\left(\mathbf{x}_\sigma | \mathbf{x}_\tau\right)\right)\right), \tag{14}$$

where $g(\cdot)_d$ denotes a $\mathrm{LayerNorm}(\mathrm{Linear}_d(\cdot))$ with output feature dimension d (with o and h defined for each model below), and where $|$ denotes concatenation. After the messages are generated, we exploit the flexibility of aggregation of our package depending on the target of the feature update. In all cases, we update the features of the target cell with a residual connection,

$$\mathbf{x}_\sigma^{t+1} = \mathbf{x}_\sigma^t + m(\sigma, \tau, \delta). \tag{15}$$

Finally, the prediction for the molecular property is given by an atomic contribution layer,

$$\hat{y}(X) = \mathrm{Linear}_1\left(\mathrm{ReLU}\left(\underset{n \in X^{(0)}}{\mathrm{agg}}\left(\mathrm{ReLU}\left(g_{16}\left(\mathbf{x}_n^L\right)\right)\right)\right)\right), \tag{16}$$

where $X^{(0)}$ is the set of nodes of the cell complex and L is the total number of message passing steps.

Baseline GNN Our GNN model closely follows Algorithm 1 from the work by Battaglia et al. [2]. We embed atoms and bonds into feature vectors $\mathbf{x}^{t>0}$ of dimension 16, then perform $L = 4$ layers of message passing. In each of these layers, we update edge (e) and node (n) features with

$$\mathbf{x}_e^{t+1} = \mathbf{x}_e^t + \underset{\{n_1, n_2\} \in \mathcal{B}(e)}{\mathrm{agg}}\left(M_{\uparrow}\left(\mathbf{x}_{n_1}^t, \mathbf{x}_{n_2}^t, \mathbf{x}_e^t\right)\right), \tag{17}$$

$$\mathbf{x}_n^{t+1} = \mathbf{x}_n^t + \underset{e \in \mathcal{C}(n)}{\mathrm{agg}}\left(M_{\mathcal{C}}\left(\mathbf{x}_n^t, \mathbf{x}_e^{t+1}\right)\right), \tag{18}$$

where n_i refers to any nodes in the boundary of the edge. For the message functions M, we keep $h = 32$ and $o = 16$ for both node and edge updates.

TNN The TNN model closely follows the structure of the GNN, differing only by a face (f) update and a second edge update before the node update,

$$\mathbf{x}_e^{t+1/2} = \mathbf{x}_e^t + \underset{\{n_1, n_2\} \in \mathcal{B}(e)}{\mathrm{agg}}\left(M_{\uparrow}\left(\mathbf{x}_{n_1}^t, \mathbf{x}_{n_2}^t, \mathbf{x}_e^t\right)\right), \tag{19}$$

$$\mathbf{x}_f^{t+1} = \mathbf{x}_f^t + \underset{\{e_1, e_2\} \in \mathcal{B}(f)}{\mathrm{agg}}\left(M_{\uparrow}\left(\mathbf{x}_{e_1}^{t+1/2}, \mathbf{x}_{e_2}^{t+1/2}, \mathbf{x}_f^t\right)\right), \tag{20}$$

$$\mathbf{x}_e^{t+1} = \mathbf{x}_e^{t+1/2} + \underset{f \in \mathcal{C}(e)}{\mathrm{agg}}\left(M_{\mathcal{C}}\left(\mathbf{x}_e^{t+1/2}, \mathbf{x}_f^{t+1}\right)\right), \tag{21}$$

$$\mathbf{x}_n^{t+1} = \mathbf{x}_n^t + \underset{e \in \mathcal{C}(n)}{\mathrm{agg}}\left(M_{\mathcal{C}}\left(\mathbf{x}_n^t, \mathbf{x}_e^{t+1}\right)\right), \tag{22}$$

where e_i refers to any edges in the boundary of the face. For this model, we use $L = 2$ layers, $h = 32$ and $o = 16$ for all messages. Equations 19 to 22 are represented in Figure 2 by message diagrams a) to d), respectively.

H-TNN The H-TNN model differs from the TNN in the dimensions of the face up-messages ($h = 16$, $d = 8$, face embedding is also 8) and by a hypergraph embedding layer (Eq. 11) before the message passing layers. Essentially, this replaces the face features with a representation of the high-order interactions within the face. Necessarily, this modifies the update of the face features,

$$\mathbf{x}_f^{t+1}(e_1, e_2) = \mathbf{x}_f^t(e_1, e_2) + M_{\uparrow}\left(\mathbf{x}_{e_1}^{t+1/2}, \mathbf{x}_{e_2}^{t+1/2}, \mathbf{x}_f^t(e_1, e_2)\right). \tag{23}$$

H(agg)-TNN The H(agg)-TNN model aggregates the pairwise interactions into the faces to which each pair belongs,

$$\mathbf{x}_f^0 = \mathrm{MLP}\left(\underset{\{e_1,e_2\}\in\mathcal{B}(f)}{\mathrm{agg}}\left(\mathbf{x}_f(e_1, e_2)\right)\right), \tag{24}$$

where MLP indicates two linear layers sandwiching a ReLU activation function, the first of dimension 16 and the second of dimension 8. In this case, the layer equations are the same as for the TNN. However, the dimensions of the face upper messages are the same as for the H-TNN.

A comparison of the four models and their parameters is presented in Table 1. To make the comparison of the four architectures as similar as possible, we aimed to keep the number of parameters similar between the four models, taking care to adjust the dimensions of the different layers in each model to accomplish this.

Table 1: Overview of four models compared in these experiments. Number of parameters per model and dimensions of embedding vectors.

Model	# Parameters	# Node features	# Edge features	# Face features
GNN	16 049	16	16	N/A
TNN	16 161	16	16	16
H-TNN	15 281	16	16	8
H(agg)-TNN	15 953	16	16	8

We regard the GNN model as baseline for its simpler and more standard architecture. For the other three models, edges and faces were obtained directly from the bonds and rings of the SMILES. In Appendix A.2, we present additional results (parity plots, loss curves) for both tasks examined herein. In all regression results, error bars represent the standard deviation.

3.2 Task 1: Predicting C_{v}

Using the four models described above, we first evaluated the models in predicting C_{v}. In Figure 3, we compare the performance of the four models. All topological models show a noticeable increase in predictive power. As mentioned in the discussion, this could be due to a greater relevance of many-body interactions.

3.3 Task 2: Predicting HOMO-LUMO Gap

For the second task, we compared the performance of the four models in predicting the HOMO-LUMO gap (Figure 3). In this case, the topological models perform worse than the GNN. This property is arguably harder to predict than the C_{v}, since it depends on many-electron quantum interactions in the molecule. Because of this, the deeper latent representation achieved by the GNN (due to its higher number of layers) could be advantageous.

4 Discussion

We observe a greater benefit from higher-order message passing for C_{v} prediction than for the prediction of HOMO-LUMO gap compared to standard GNN baselines. This may be explained because C_{v} is a bulk thermodynamic property that emerges from collective/global vibrational modes that involve the coordinated motion of many atoms at once. Capturing these correlations may thus

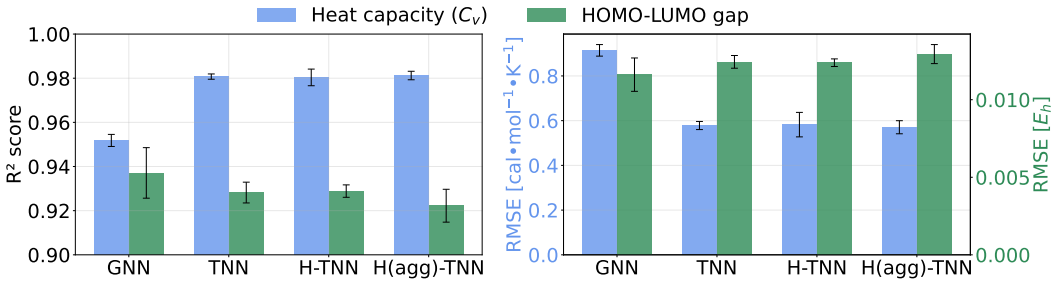

Figure 3: Model performance comparison, showing the R^2 score (left) and RMSE (right) on the held out test set from QM9 for both tasks. Error bars represent the standard deviation from three distinct training runs per model. See Appendix A.2.1 for exact values.

benefit more from integrating information over extended multi-atom structures, precisely the type of representations that higher-order cells and hypergraph connections in TMP can help model. In contrast, the HOMO-LUMO gap prediction does not show the same trend. These energy levels depend on the quantum many-body interaction of the electrons in the molecule, making it a complex property to predict. As a result, the higher number of layers of the GNN could be providing a deeper latent representation that results in better predictions. Besides, it is also worth mentioning that the influence of the EHNN layer in the results of both tasks is practically negligible. This suggests that the TNN is already capable of learning the group interactions in faces.

However, these findings must be interpreted within the context of the inherent limitations of the QM9 data set. QM9 consists exclusively of small organic molecules (≤ 9 CNOF atoms), fundamentally constraining the extent to which higher-order and long-range interactions can affect a property. While our experiments demonstrate that TMP can match or exceed standard GNN performance on certain tasks, the small molecular size in QM9 limits the potential for truly global or macroscale phenomena that would most clearly benefit from higher-order topological representations. In these compact molecular systems, even "global" properties like C_{v} may not require the full expressivity of higher-order message passing, as the limited spatial extent means that few atoms are ever truly distant from one another in either real space or through-bond connectivity.

This suggests that TMPs may only yield significant gains when applied to systems where global or macroscale properties dominate, such as bulk materials properties, extended supramolecular assemblies, or larger biomolecular complexes, rather than the small-molecule properties represented in QM9. For the molecular systems in this dataset, the added expressivity may be underutilized, especially given that the dataset size may not be sufficient to leverage the richer topological structure of the data. Future work should therefore focus on identifying appropriate tasks that may benefit from the representation power of TMPs, possibly those involving larger molecular systems or materials where higher-order interactions are expected to play a more decisive role, possibly involving larger models and tasks with inherently global dependencies to demonstrate the full learning potential and expressivity of TMPs.

5 Conclusion

We introduce TopoMole, the first JAX-based framework for topological message passing, and present the first results involving a hyperedge embedding layer for molecular property prediction. Our results demonstrate that, at similar parameter counts, different model architectures can elucidate whether specific property prediction tasks benefit more from pairwise or higher-order relations in the data. However, these findings are constrained by the limitation of the QM9 dataset to small molecules (≤ 9 CNOF atoms), where the potential for long-range, higher-order interactions is inherently restricted. The chemical world extends far beyond small organic molecules to encompass complex biological macromolecules, crystalline materials, and supramolecular assemblies. Long-range interactions and many-body quantum mechanical effects, which cannot be adequately captured by pairwise interactions alone, play a crucial role in emergent collective properties in these systems. The modest improvements observed for certain properties like C_{v} suggest that the true advantages of TMP may

show in learning tasks involving larger systems. As computational chemistry increasingly tackles problems involving protein folding, reaction pathways, and materials design, the ability to model higher-order correlations becomes not just advantageous but essential. Despite the limited gains on small molecules, TMP represents a crucial step toward more expressive molecular representations capable of capturing the high-order interactions that govern real-world chemical complexity. Further experiments on appropriately scaled benchmarks are necessary to fully demonstrate the potential of the framework and identify the scenarios where TMP provides decisive advantages over conventional GNNs.

6 Code and Data Availability

For implementation details and source code, please refer to ourGitHub repository `https://github.com/pablomcrespo/JAX_TMP`. Additionally, the QM9 dataset used for experiments in this study is available at doi.org/10.6084/m9.figshare.c.978904.v5.

Acknowledgments and Disclosure of Funding

PMC acknowledges funding from the Intel-Merck joint university research center for AI-Aware Pathways to Sustainable Semiconductor Process and Manufacturing Technologies (AWASES). RM acknowledges funding provided by the Wallenberg AI, Autonomous Systems, and Software Program (WASP), supported by the Knut and Alice Wallenberg Foundation. The computations and data storage were enabled by resources provided by Chalmers e-Commons at Chalmers. The computations and data storage were enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS), partially funded by the Swedish Research Council through grant agreement no. 2022-06725.

References

[1] Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications, 2021. URL https://arxiv.org/abs/2006.05205.

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

[3] Suresh Bishnoi, Ravinder Bhattoo, Jayadeva Jayadeva, Sayan Ranu, and N M Anoop Krishnan. Learning the dynamics of physical systems with hamiltonian graph neural networks. In *ICLR 2023 Workshop on Physics for Machine Learning*, 2023. URL https://openreview.net/forum?id=Ugl-B_at5n.

[4] Cristian Bodnar, Fabrizio Frasca, Nina Otter, Yuguang Wang, Pietro Lio, Guido F Montufar, and Michael Bronstein. Weisfeiler and Lehman go cellular: CW networks. *Advances in Neural Information Processing Systems*, 34:2625–2640, 2021.

[5] Nicholas E Charron, Klara Bonneau, Aldo S Pasos-Trejo, Andrea Guljas, Yaoyi Chen, Félix Musil, Jacopo Venturin, Daria Gusew, Iryna Zaporozhets, Andreas Krämer, et al. Navigating protein landscapes with a machine-learned transferable coarse-grained model. *Nature Chemistry*, pages 1–9, 2025.

[6] Justas Dauparas, Ivan Anishchenko, Nathaniel Bennett, Hua Bai, Robert J Ragotte, Lukas F Milles, Basile IM Wicky, Alexis Courbet, Rob J de Haas, Neville Bethel, et al. Robust deep learning–based protein sequence design using proteinmpnn. *Science*, 378(6615):49–56, 2022.

[7] Adam Foster, Zeno Schätzle, P. Bernát Szabó, Lixue Cheng, Jonas Köhler, Gino Cassella, Nicholas Gao, Jiawei Li, Frank Noé, and Jan Hermann. An ab initio foundation model of wavefunctions that accurately describes chemical bond breaking, 2025. URL https://arxiv.org/abs/2506.19960.

[8] Lorenzo Giusti, Teodora Reu, Francesco Ceccarelli, Cristian Bodnar, and Pietro Liò. Topological message passing for higher-order and long-range interactions. In *2024 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2024.

[9] Esther Heid, Kevin P Greenman, Yunsie Chung, Shih-Cheng Li, David E Graff, Florence H Vermeire, Haoyang Wu, William H Green, and Charles J McGill. Chemprop: a machine learning package for chemical property prediction. *Journal of Chemical Information and Modeling*, 64(1):9–17, 2023.

[10] Yinan Huang, Xingang Peng, Jianzhu Ma, and Muhan Zhang. Boosting the cycle counting power of graph neural networks with i^2-gnns, 2023. URL https://arxiv.org/abs/2210.13978.

[11] Shengli Jiang and Michael A Webb. Physics-guided neural networks for transferable property prediction in architecturally diverse copolymers. *Macromolecules*, 58(10):4971–4984, 2025.

[12] Jinwoo Kim, Saeyoon Oh, Sungjun Cho, and Seunghoon Hong. Equivariant hypergraph neural networks. In *European Conference on Computer Vision*, pages 86–103. Springer, 2022.

[13] Rıza Ozcelik, Helena Brinkmann, Emanuele Criscuolo, and Francesca Grisoni. Generative deep learning for de novo drug designa chemical space odyssey. *Journal of Chemical Information and Modeling*, 2025.

[14] Mathilde Papillon, Sophia Sanborn, Mustafa Hajij, and Nina Miolane. Architectures of topological deep learning: A survey of message-passing topological neural networks, 2024. URL https://arxiv.org/abs/2304.10031.

[15] Raghunathan Ramakrishnan, Pavlo O Dral, Matthias Rupp, and O Anatole Von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. https://doi.org/10.6084/m9.figshare.c.978904.v5, 2014. Accessed: 2025-08-11.

[16] Raghunathan Ramakrishnan, Pavlo O Dral, Matthias Rupp, and O Anatole Von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data*, 1(1):1–7, 2014.

[17] Ali Ramlaoui, Théo Saulus, Basile Terver, Victor Schmidt, David Rolnick, Fragkiskos D. Malliaros, and Alexandre Duval. Improving molecular modeling with geometric gnns: an empirical study, 2024. URL https://arxiv.org/abs/2407.08313.

[18] Maksim Zhdanov, Nabil Iqbal, Erik Bekkers, and Patrick Forré. AdS-GNN – a conformally equivariant graph neural network, 2025. URL https://arxiv.org/abs/2505.12880.

A Technical Appendices and Supplementary Material

A.1 Notation

- X: cell complex
- $X^{(k)}$: subset of cells homeomorphic to $\mathbb{R}^k$
- ψ: cochain map
- F: feature space
- $\mathbf{x}_\sigma$: feature vector of $\sigma \in X$
- $\mathcal{B}(\sigma)$: boundary set of the reference cell σ
- $\mathcal{C}(\sigma)$: coboundary set of the reference cell σ
- $\mathcal{N}_\downarrow(\sigma)$: lower-adjacent set of the reference cell σ
- $\mathcal{N}_\uparrow(\sigma)$: upper-adjacent set of the reference cell σ
- $\mathcal{H}(\mathcal{V}, \mathcal{E})$: hypergraph defined by node set $\mathcal{V}$ and hyperedge set $\mathcal{E}$
- $\mathbf{X}^{(k)}$: feature matrix of k-order hyperedges
- $\mathbf{A}^{(k)}_{(i_1,\ldots,i_k)}$: tensor representation of a k-uniform hypergraph.

A.1.1 Difference in Notation from Bodnar et al. [4]

As mentioned in the main text, the implementation in Bodnar et al. [4] names the messaging schemes based on the flow of the messages. Upper messages flow between upper adjacent cells (e.g. nodes that share an edge), lower messages between lower adjacent cells (e.g. edges that share a node), boundary messages come from boundary cells (e.g. a face receives messages from its edges), and coboundary messages come from coboundary cells (e.g. edges receive messages from the faces they are part of). In order to handle the index lookup for the cells of each dimension involved in the messages, each cochain has upper neighbors and shared coboundaries (for upper messages), lower neighbors and shared boundaries (for lower messages), and specific boundary adjacency arrays (for boundary messages). In the date of our publication, the original work codebase does not support coboundary messages or coboundary adjacency arrays.

While working on our JAX implementation, we found it simplest to keep only the upper/lower neighbors and shared (co)boundary arrays (`up_senders, up_receivers, shared_coboundaries, down_senders, down_receivers, shared_boundaries`). Besides, this greatly simplifies the padding and masking mechanisms needed for JIT-compatibility, needing only an upper mask and a lower mask and avoiding (co)boundary masks. The key for this simplification is that the boundary adjacency array from the original implementation can be formed by concatenating our `up_receivers` and `shared_coboundaries` arrays. Therefore, our nomenclature for each of the messaging channels is motivated by the specific arrays used for index lookup, rather than by the flow of the messages. The flexibility of the aggregation and update options of our package is precisely motivated by the way adjacency relations are represented. In any case, our framework contains all the messaging options of the work presented in [4].

However, this simplification comes with its own particularities. In high-order relations, the entries in both arrays have several repeated values (one per pair of interactions), generating one message per pair even though only the features of the receiver and the coboundary are being looked up. On the other hand, our boundary messages remain limited to cells that are joined by a lower neighboring relation (e.g. an edge and a node only exchange boundary messages if the node intersects other edges, a face and an edge only exchange boundary messages if the edge is part of other rings). We are already working on extending the framework to account each boundary/coboundary relation only once.

A.2 Additional Results

A.2.1 Tabular Results for Figure 3

In Tables 2 and 3 we provide the exact values for the various evaluation metrics on the hold-out test set for C_v and HOMO-LUMO gap prediction.

Table 2: R^2 score and RMSE for each model in C_v prediction on the hold-out test set.

Model	$\mathbf{R^2}$	**RMSE** [cal·mol^{-1}·K^{-1}]
GNN	0.9518 ± 0.0027	0.9144 ± 0.0259
TNN	0.9807 ± 0.0012	0.5783 ± 0.0182
H-TNN	0.9804 ± 0.0038	0.5822 ± 0.0547
H(agg)-TNN	0.9812 ± 0.0019	0.5706 ± 0.0293

Table 3: R^2 score and RMSE for each model in HOMO-LUMO gap prediction on the hold-out test set.

Model	$\mathbf{R^2}$	**RMSE** [E_h]
GNN	0.9371 ± 0.0114	0.0116 ± 0.0011
TNN	0.9282 ± 0.0047	0.0124 ± 0.0004
H-TNN	0.9289 ± 0.0028	0.0124 ± 0.0002
H(agg)-TNN	0.9222 ± 0.0074	0.0129 ± 0.0006

A.2.2 Parity Plots

In Figures 4 and 5, the regression plots for each task are presented.

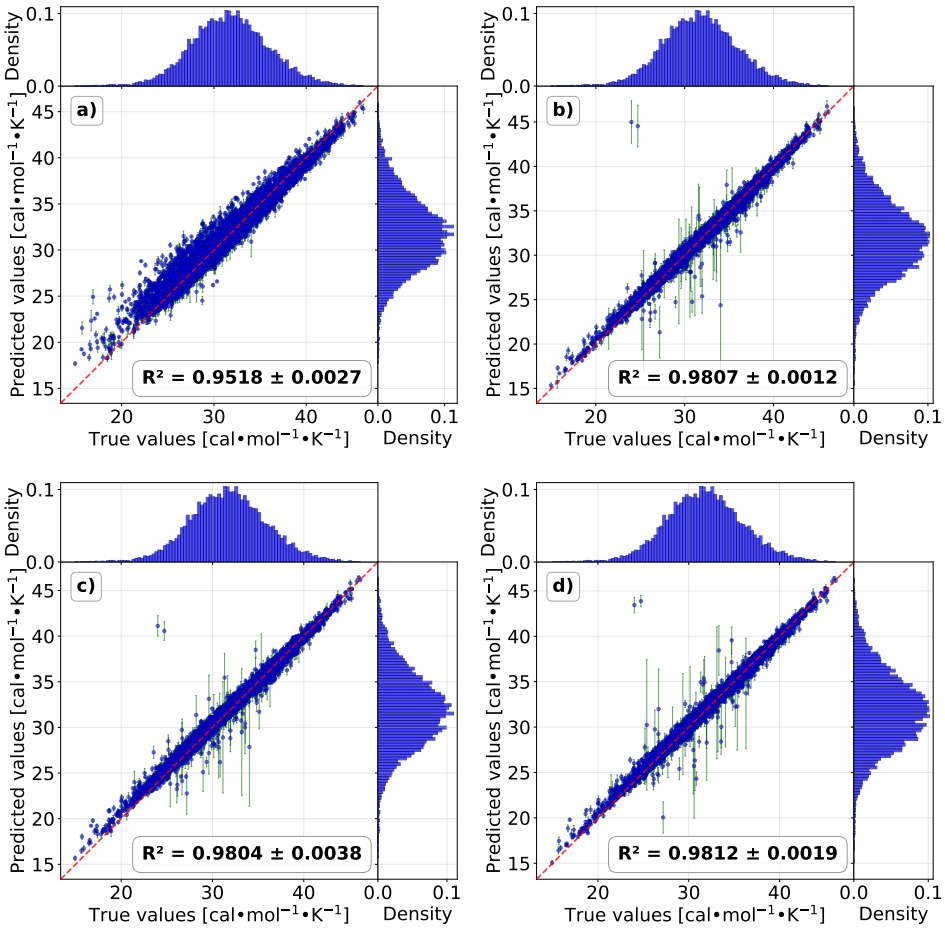

Figure 4: Parity plots showing C_v predictions versus true values on the test set. a) GNN, b) TNN, c) H-TNN, and d) H(agg)-TNN.

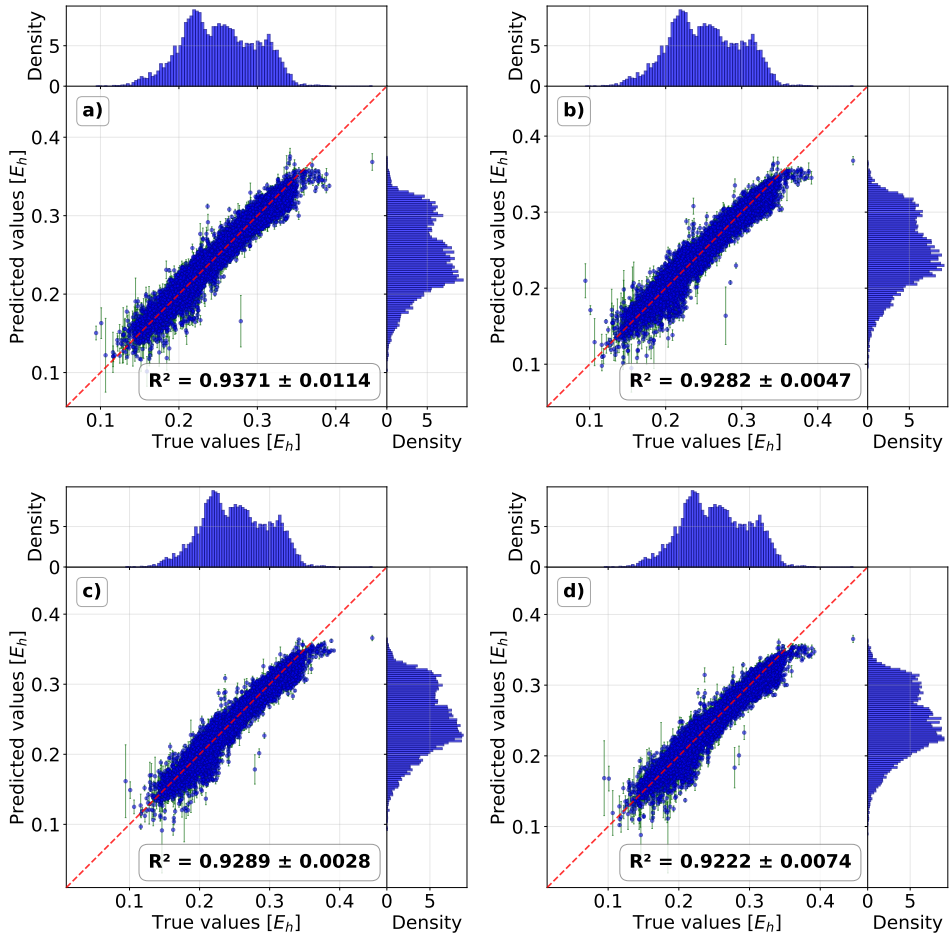

Figure 5: Parity plots showing HOMO-LUMO gap predictions versus true values on the test set. a) GNN, b) TNN, c) H-TNN, and d) H(agg)-TNN.

We noticed that the three TNN models displayed the same outliers in the prediction of C_{v}, all containing complex ring systems. These two molecules have SMILES `C1N2C3C4C2C11CN4C31` and `C1C2C3C4C2C11CN4C31` and are illustrated in Figure 6 a) and b), respectively.

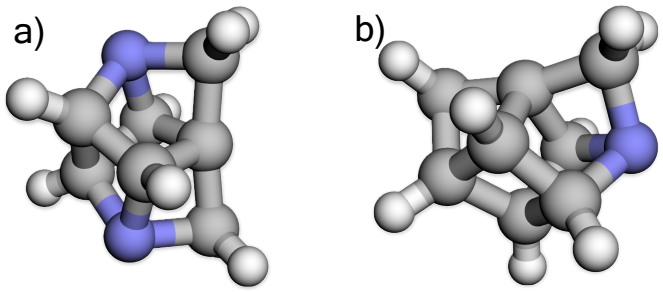

Figure 6: Outliers in the C_{v} prediction of the topological models.

A.2.3 Training Curves

In Figures 7 and 8 we present the training curves for each of the four models (GNN, TNN, H-TNN, and H(agg)-TNN), demonstrating the rate of model convergence on both tasks.

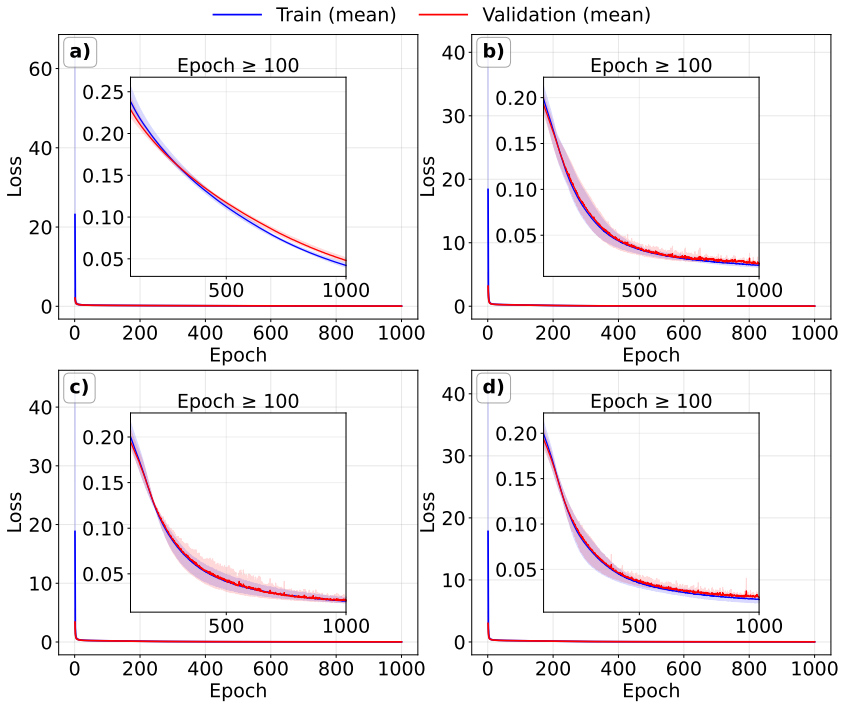

Figure 7: Loss curves for C_{v} training. a) GNN, b) TNN, c) H-TNN, d) H(agg)-TNN.

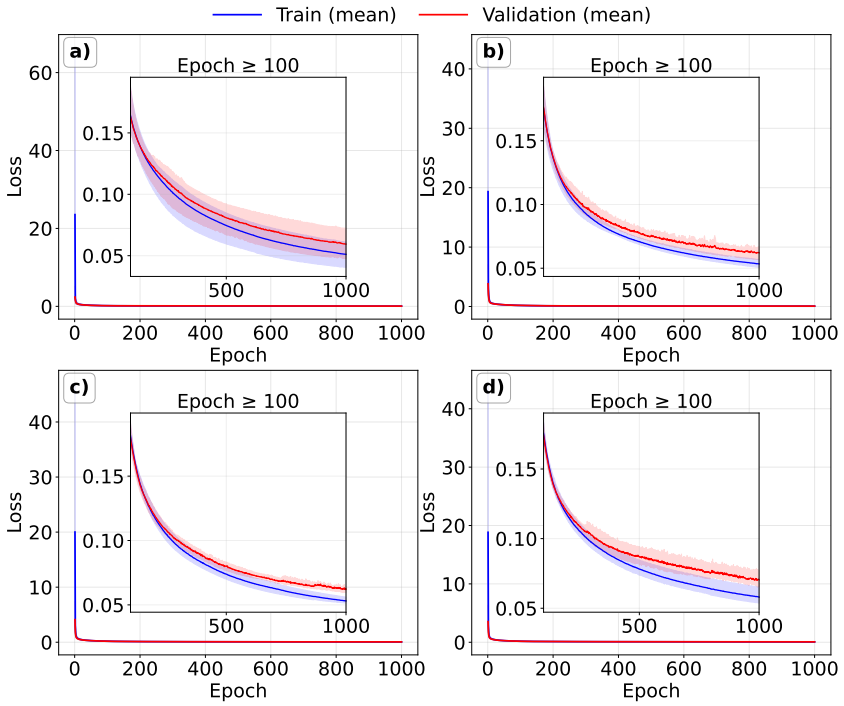

Figure 8: Loss curves for HOMO-LUMO gap training. a) GNN, b) TNN, c) H-TNN, d) H(agg)-TNN.