Analyzing the Expressive Power of Geometric Graph Neural Networks

Giorgio Bertone
TU Wien

Abstract

This seminar paper analyzes the expressive power of geometric Graph Neural Networks (GNNs), focusing on their ability to discriminate between non-isomorphic geometric graphs through the Geometric Weisfeiler-Leman (GWL) test framework. This analysis is relevant as existing theoretical frameworks for GNNs, such as the Weisfeiler-Leman (WL) test, fail when applied to geometric graphs due to their inherent physical symmetries. The GWL test, a new approach specifically designed for geometric graphs, addresses these limitations. Moreover, geometric GNNs are able to overcome some limitations of standard GNNs when applied to geometric graphs, which are common in scientific and engineering domains, by incorporating physical symmetries. Examples of such domains include biochemistry (molecular structures), material sciences (crystalline structures), physics (physical simulations of complex systems), and robotics. In particular, the paper addresses the limitations of invariant GNNs, which are unable to distinguish graphs with identical one-hop neighborhoods and cannot capture global geometric properties. It then examines how equivariant GNNs overcome these limitations by propagating geometric information beyond local neighborhoods. Furthermore, the paper explores the impact of depth, body order of scalarization, and tensor order of features on the expressive power of geometric GNNs. Finally, it inspects synthetic experiments to support the theoretical findings.

1 Introduction

Geometric GNNs are very effective for modeling systems with geometric and relational structures. These systems appear across a wide range of fields, including **biochemistry** (e.g., small molecules, proteins, and DNA/RNA) as highlighted by Jamasb et al. [2022], **materials science** (e.g., inorganic crystals, catalysis systems) as demonstrated by Chanussot et al. [2021], and **physical simulations** as shown by Sanchez-Gonzalez et al. [2020]. Beyond these domains, geometric GNNs are also applied in areas such as transportation and logistics, robotic navigation, multiagent robotics [Li et al., 2020] and 3D computer vision. This type of systems can be modelled through geometric graphs embedded in the Euclidean space.

Given a geometric graph, standard GNNs are ill-suited to handle it because they do not take spatial symmetries into account. However, geometric GNNs have emerged as the state of the art for making predictions about their properties. They still follow the message passing paradigm [Gilmer et al., 2017], where neighboring nodes exchange information and influence the embeddings of each other. Specifically, features from local neighbourhoods are aggregated and used to update node features in a permutation equivariant manner. The most important property is that the updated geometric features of the nodes retain the transformation semantics of the initial attributes.

Geometric GNNs have shown promising results as generative models for geometric graphs (e.g., in diffusion models for protein design [Baek et al., 2021]), as wells as molecular dynamics simulators in autoregressive settings [Batzner et al., 2022].

Despite these successes, important theoretical questions remain unanswered. Crucially, a deeper understanding of the **design space of geometric GNNs** is needed to further advance the field, as well as a more rigorous analysis of their expressivity. In particular, comprehending how key factors such as the **body order of the layer** (which defines the number of nodes required for feature computation), **depth** (the number of layers), and **tensor order of features** (how features transform under geometric operations) determine the expressivity and effectiveness of a geometric GNN.

The Weisfeiler-Leman (WL) test [Weisfeiler and Leman, 1968] is widely used to analyze the expressive power of standard GNNs. However, it does not apply to geometric graphs as it fails to account for geometric transformations (e.g. rotations and translations) that affect node positions and geometric features.

Thus, Joshi et al. [2023] study the expressive power of geometric GNNs from the perspective of discriminating non-isomorphic geometric graphs, proposing an extension of the WL test, called GWL, which they prove to be an upper bound on the expressive power of geometric GNNs. Then, they use this framework to understand how design choices influence geometric GNNs expressivity.

1.1 Background

A **standard graph** is a set of nodes connected by edges, where each node can also have scalar features. It can be represented as:

$$\mathcal{G} = (A, S)$$

where A is the $n \times n$ adjacency matrix and $S \in \mathrm{R}^{n \times f}$ are the scalar features.

In contrast, in **geometric graphs** each node is embedded in the Euclidean space and associated with both scalar features and geometric attributes (such as position and orientation). A geometric graph can thus be defined as:

$$\mathcal{G} = (A, S, \vec{X}, \vec{V})$$

where $\vec{X} \in R^{n \times d}$ are the coordinates of the nodes in d-dimensional space, and $\vec{V} \in R^{n \times d}$ are the geometric features of the nodes (e.g., velocities, accelerations).

Physical symmetries. The key difference between traditional graphs and geometric graphs lies in the transformation behavior under Euclidean transformations (such as rotations, translations, and reflections). While the scalar features S remain unchanged under these transformations, the geometric attributes (i.e. the vectors $\vec{V}$ and the coordinates $\vec{X}$) transform accordingly. In other words, geometric graphs have additional symmetries, and while geometric GNNs are designed to handle these transformations, ensuring that both scalar and geometric features are updated in a way that preserves the transformation semantics, standard GNNs are not able to provide these guarantees.

Message Passing GNNs. Message Passing Graph Neural Networks (MPGNNs) leverage the structure of graphs to learn meaningful representations by iteratively updating node features using local neighborhood information. The nodes exchange *"messages"* with their neighbors to aggregate information and refine their own features. The key idea is that each node gathers information from its immediate neighbors at every layer, effectively building a **computation tree**. As the network deepens, the receptive field of each node expands, enabling it to incorporate features from larger neighborhoods. This process can be expressed in two main steps:

- **Message Computation** Each edge (i, j) computes a message m_{ij} based on the features of the connected nodes:

  $$m_{ij} = f(s_i, s_j)$$

  Here, s_i and s_j are the features of nodes i and j, and f is a function that encodes their interaction.
- **Message Aggregation and Node Update:** Each node aggregates incoming messages from its neighbors (e.g., using sum, mean, or max pooling) and updates its features:

  $$s_i^{(t+1)} = \text{Update}\left(s_i^{(t)}, \text{Aggregate}\left(\{m_{ij} : j \in \mathcal{N}(i)\}\right)\right)$$

  - $s_i^{(t)}$: Node i's features at layer t.
  - $\mathcal{N}(i)$: The set of neighbors of node i.

- Aggregate: Combines messages from neighbors (e.g., summation).
- Update: Refines the node's features using the aggregated messages.

Graph Isomorphism. Two graphs G, H with attributes are **isomorphic** if there exists an edge-preserving (i.e. two vertices *u* and *v* of *G* are adjacent in *G* iff *b(u)* and *b(v)* are adjacent in *H*) bijection (one-to-one correspondence) between the vertex sets of G and H

$$b : V(G) \to V(H)$$

such that

$$s_i^{(G)} = s_{b(i)}^{(H)}$$

In other words, the problem asks whether two graphs are the same but drawn differently.

Weisfeiler-Leman Test (WL). The WL algorithm checks whether two graphs are isomorphic. At initialization, each node i in the graph is assigned an initial color $c_i^{(0)} \in C$ from a countable space of colors C. If their features are the same, the nodes are coloured the same. Then, at each iteration t, WL iteratively updates the colouring $c_i(t)$, by collecting for each node *i* the multiset of features of *i*'s neighbors, concatenating them *i*'s current label $c_i^{(t-1)}$ with the multiset and then applying an hash function that assigns a unique colour to each input:

$$c_i^{(t)} = HASH(c_i^{(t-1)}, \{\{c_j^{t-1} | j \in N_i\}\}$$

The process stops when the colors no longer change across iterations. After the refinement, the algorithm compares the *multiset of final labels* for the two graphs. If the multisets are different the graph are not isomorphic. If the multisets are identical, the graphs may be isomorphic.

Message passing GNNs can be at most as powerful as WL at distinguishing non-isomorphic graphs. Thus, all the cases where WL will fail, also message passing GNNs will fail.

Group Theory. A group is a collection of elements with a rule for combining them (called an **operation**). This rule satisfies three key properties:

- **Associative**: $(a \cdot b) \cdot c = a \cdot (b \cdot c)$
- **Identity**: There's a special element e that leaves others unchanged, e.g.,$a \cdot e = a$.
- **Inverse**: Every element has a an inverse, e.g., $a \cdot a^{-1} = e$

A group can "act" on a space, meaning it transforms the elements of the space in a structured way (e.g., rotations or shifts). We denote the action of a group $\mathbf{G}$ on a space X by $\mathbf{g} \cdot x$. Given two spaces X and Y on which $\mathbf{G}$ acts, and a function $f : X \to Y$, we say f is:

- $\mathbf{G}$-*invariant* if $f(\mathbf{g} \cdot x) = f(x)$, i.e. the output doesn't change when you apply a transformation to the input (example: imagine X is a square and $\mathbf{G}$ represents rotations. A G-invariant function might just return the area of the square, which stays the same no matter how the square is rotated).
- $\mathbf{G}$-*equivariant* if $f(\mathbf{g} \cdot x) = \mathbf{g} \cdot f(x)$, i.e. transforming the input results in an equivalent transformation of the output (example: imagine X is a set of points in a plane, and Y is the set of their locations and $\mathbf{G}$ represents shifts; then a G-equivariant function might give you the center of mass of the points, and if you shift all points in X, the center of mass in Y shifts the same way).

When a group G acts on a space X, each element $x \in X$ can be transformed by all elements of G. The $\mathbf{G}$-*orbit* of x, denoted $O_{\mathbf{G}}(x)$, is the set of all the results of applying the group's transformations to x:

$$O_{\mathbf{G}}(x) = \{g \cdot x \mid g \in G\}.$$

We write $x \simeq x'$ if x and x' are in the same orbit (i.e. x' is just a transformed version of x). We say a function $f : X \to Y$ is $\mathbf{G}$-*orbit injective* if:

$$f(x_1) = f(x_2) \quad \text{if and only if} \quad x_1 \simeq x_2 \quad \forall x_1, x_2 \in X.$$

This means that f assigns the **same output** to all elements in the same orbit, but **different outputs** to elements from *different orbits*.If f is G-orbit injective, it is automatically G-invariant. This is because transforming an input x by any group element g keeps x within its orbit.

Example: Let X be a set of points on a plane, and G represent rotations around a fixed point. The orbit $O_G(x)$ of a point x is the circle it traces under all rotations. A G-orbit injective function f could assign the same value (e.g., the radius) to all points in the same orbit (i.e., the same circle). Points on different circles would receive different values.

Actions of groups on geometric graphs.

- **Symmetric Group S_n.** The symmetric group S_n consists of all possible permutations of n elements. In the context of geometric graphs, S_n acts on the graph by permuting its nodes. Specifically, it applies a permutation σ that transforms the adjacency matrix A, the node features S, the node positions $\vec{V}$, and the node attributes $\vec{X}$ as follows:

  $$P_\sigma G := (P_\sigma A P_\sigma^T, P_\sigma S, P_\sigma \vec{V}, P_\sigma \vec{X}),$$

  where P_σ is the permutation matrix corresponding to σ. This action reshuffles the nodes of the graph, preserving the structure and relationships but changing their labeling.
- **Orthogonal Group $O(d)$.** The orthogonal group $O(d)$ consists of all $d \times d$ orthogonal matrices, i.e., matrices that preserve the Euclidean distance between points. These transformations represent rotations and reflections in d-dimensional space. In geometric graphs, the group $O(d)$ acts on the node positions $\vec{V}$ and node attributes $\vec{X}$, transforming them as:

  $$\vec{V}Q_g \quad \text{and} \quad \vec{X}Q_g,$$

  where $Q_g \in O(d)$ is an orthogonal transformation. This action alters the spatial configuration of the nodes, but preserves the geometric structure (e.g., angles and distances between nodes).
- **Translation Group $T(d)$.** The translation group $T(d)$ represents translations in d-dimensional space. A translation is defined by a vector $\vec{t} \in T(d)$ that shifts all points in the space by the same amount. In the context of geometric graphs, translations act on the node coordinates $\vec{X}$, shifting the positions of all nodes uniformly by adding $\vec{t}$ to each node's position:

  $$\vec{x}_i + \vec{t} \quad \text{for each node} \quad i.$$

  This transformation does not alter the relative distances or angles between nodes, but it changes their absolute positions in the space.

2 Related work

In this section, we review the key results in the literature of Geometric Neural Networks, highlighting the two main classes of geometric GNNs — *invariant* and *equivariant* models—and discuss the open questions and challenges that remain in the field .

2.1 Design Space of Geometric GNNs

Geometric GNNs are typically categorized into two main classes based on how they treat spatial transformations: *invariant* and *equivariant* models.

1. **Invariant GNNs**: The first class of models, *G-invariant GNNs*, only update scalar features by scalarization of local geometric information, i.e. converting geometric information within local neighbourhoods into an invariant quantity. Thus, they transform local geometric information into invariant quantities that do not change under rotations or translations.

   $$s_i^{(t+1)} := \text{UPD}\Big(s_i^{(t)}, \text{AGG}\Big(\{\{(s_i^{(t)}, \vec{v}_i^{(t)}), \Big(s_j^{(t)}, \vec{v}_j^{(t)}\Big), \vec{x}_{ij} \mid j \in N_i\}\}\Big)\Big)$$

   For instance, **SchNet** [Schütt et al., 2018] concatenates the pairwise Euclidean distance between nodes to the message passed between them. This approach is invariant becuase

relative distances are unchanged under global rotations and translations. SchNet is an example of a GNN with a body order of two, meaning that it requires two nodes to compute a pairwise distance. The body order of a model refers to the number of nodes involved in computing a local invariant scalar. **DimeNet** [Gasteiger et al., 2020], on the other hand, uses both distances and angles among triplets of nodes, making it an invariant model with a body order of three. The inclusion of angles requires the use of three nodes, thus it increases the body order, but it also enables to capture more complex geometric relationships.

2. **Equivariant GNNs**: The second class of models, *G-equivariant GNNs*, not only update scalar features, but also propagate geometric vector features.

$$s_i^{(t+1)}, \vec{v}_i^{(t+1)} := \text{UPD}\Big((s_i^{(t)}, \vec{v}_i^{(t)}), \text{AGG}\Big(\{\{(s_i^{(t)}, \vec{v}_i^{(t)}), \Big(s_j^{(t)}, \vec{v}_j^{(t)}\Big), \vec{x}_{ij} \mid j \in N_i\}\}\Big)\Big)$$

They aim to preserve the physical symmetries of the system by ensuring that node features transform in a predictable way under Euclidean transformations. The first class of equivariant GNNs operates in Cartesian coordinates (x, y, z). For example, **PaiNN** [Schütt et al., 2021] updates both scalar and vector features by propagating geometric messages. In order to ensure equivariance and respect physical symmetries, however, certain constraints must be forced to the network's operations, such as limiting the types of non-linearities used (e.g., no ReLU on vector features). The second class of equivariant models employs higher-order spherical tensors as node features. These models use a spherical basis, which involves features based on radius and angle, to describe the geometry. Thus, every node has an associated higher order spherical tensor as feature, and these features are updated via tensor products of neighbourhood features with spherical harmonic expansion of the displacement vector. An example of this type of models are **Tensor Field Networks (TFN)** [Thomas et al., 2018].

While invariant models have proven effective in many tasks, equivariant models offer a more flexible framework for capturing complex geometric transformations. However, the trade-offs between these models in terms of expressivity and computational complexity is not sufficiently explored in the literature. Moreover, it is unclear how key design choices influence geometric GNN expressivity.

Furthermore, it has recently been shown (Dym and Maron [2020], Villar et al. [2021]) that architectures such as TFN, GemNet and GVP-GNN can be universal approximators of continuous, G-equivariant or G-invariant multiset functions over fully connected graphs. However, universality is a binary property (a model is either universal or not), often not vey useful in practice. However, there may be varying degrees of discrimination, depending on which classes of geometric graphs it can or cannot distinguish, and this has not yet been studied.

3 Expressivity of Geometric GNNs

3.1 How powerful are Geometric GNNs ?

GNNs struggle to distinguish all local neighbourhoods when they only use unordered sets of distances and angles. Some pairs of geometric graphs cannot be differentiated by scalar features alone, so to address this issue, geometric information, such as the relative orientation of local neighbourhoods, is essential. However, theoretical frameworks for traditional GNNs, such as the WL test, do not apply to geometric GNNs due to physical symmetries inherent in geometric graphs, thus it is necessary to develop a framework to evaluate the expressivity of these models.

Geometric Graph Isomorphism. Two geometric graphs G and H are *geometrically isomorphic* if there exists an attributed graph isomorphism $b : V(G) \to V(H)$, such that the geometric attributes are equivalent for all nodes up to rotations Q_G and translations $\vec{t} \in T(d)$.

$$\Big(s_{b(i)}^{(H)}, Q_g \vec{v}_{b(i)}^{(H)}, Q_g(\vec{x}_{b(i)}^{(H)} + \vec{t})\Big) = \Big(s_i^{(G)}, \vec{v}_i^{(G)}, \vec{x}_i^{(G)}\Big) \quad \text{for all } i \in V(G)$$

3.2 Geometric Weisfeiler-Leman (GWL) Test

We now need a test to solve the problem of determining whether two graphs are identical up to geometric transformations. We want to retain the node-centric procedure and injective aggregation

from local neighbours of the WL test. However, we also need to include in the neighbourhood the set of invariant and equivariant geometric features, and assign a unique color for each neighbourhood type (i.e., pattern) making use of the geometric information that captures how the neighbourhood type is oriented or rotated in space. Thus, the test developed must have the following properties:

1. **Orbit injectivity of colors:** If two neighbours are identical up to an action of $\mathbf{G}$, the colours assigne to the central nodes must be the same. This means the colouring function must be orbit injective.
2. **Preservation of local geometry:** Satisfying the first property removes orientation information, making the aggregation no longer injective. To preserve spatial properties, we introduce auxiliary geometric variables g_i that must be updated in a way that is both injective and equivariant.

Geometric WL Test - Steps The previous properties lead to the following algorithm for the test.

1. **Initialization**: assign to each node $i \in V$ a scalar node colour $c_i \in C'$ and an auxiliary object g_i containing the geometric information associated to the sub-graph around it:

$$c_i^{(0)} := HASH(s_i), \quad g_i^{(0)} = (c_i^{(0)}, \vec{v_i})$$

   where $HASH$ is an injective map over the scalar attributes s_i of node i.
2. **Aggregate local information**: at each iteration aggregate the geometric information around node i into a new nested object $g_i^{(t)}$:

$$g_i^{(t)} = \Big((c_i^{(t-1)}, g_i^{(t-1)}), \{\{(c_j^{(t-1)}, g_j^{(t-1)}, \vec{x}_{ij}) | j \in N_i\}\}\Big)$$

   Importantly, the group $\mathbf{G}$ can act on the geometric objects by applying its action to the geometric information contained within them. This nested aggregation is injective and equivariant. Each iteration can expand $g_i^{(t)}$ to larger t-hop subgraphs $N_i^{(t)}$
3. **Update node colouring**: node colouring $c_i^{(t)}$ aggregates the information in $g_i^{(t)}$ by using an orbit injective and invariant colouring function *I-HASH*:

$$c_i^{(t)} = \textit{I-HASH}^{(t)}(g_i^{(t)})$$

   For any geometric objects g and g', *I-HASH(g)* $=$ *I-HASH*(g') if and only if there exists $\mathbf{g} \in \mathbf{G}$ such that $g = \mathbf{g} \cdot g'$
4. **Termination**: upon reaching stable coloring or set number of iterations

3.3 Upper Bounding the Expressivity of Geometric GNNs

Equivariant GNNs can be at most as powerful as GWL in distinguishing non-isomorphic geometric graphs.

Theorem 1. Any pair of geometric graphs distinguishable by a G-equivariant GNN is also distinguishable by GWL.

3.4 Invariant GWL (IGWL)

To better understand the role of equivariance, a restricted version of GWL that only updates node colours using orbit injective I-Hash function and does not propagate geometric information is introduced. The color update works according to this formula:

$$c_i^{(t)} := \textit{I-HASH}\Big((c_i^{(t-1)}, \vec{v}_i), \{\{(c_j^{(t-1)}, \vec{v}_j, \vec{x}_{ij}) \mid j \in \mathcal{N}_i\}\}\Big)$$

Furthermore, the authors also consider *IGWL(k)* a version of *IGWL* that is constrained to extract information only from all the possible k-sized tuples of nodes in a neighbourhood:

$$c_i^{(t)} := \text{I-HASH}^{(k)}\big((c_i^{(t-1)}, \vec{v}_i), \{\{(c_j^{(t-1)}, \vec{v}_j, \vec{x}_{ij}) \mid j \in \mathcal{N}_i\}\}\big)$$

where *I-HASH*$^{(k+1)}$ is defined as:

$$\text{HASH}\Big(\big\{\big\{\text{I-HASH}\Big(\big(c_i^{(t-1)}, \vec{v}_i\big), \\ \big\{\big\{(c_{j_1}^{(t-1)}, \vec{v}_{j_1}, \vec{x}_{ij_1}), \ldots, (c_{j_k}^{(t-1)}, \vec{v}_{j_k}, \vec{x}_{ij_k})\big\}\big\}\Big) \mid j \in (\mathcal{N}_i)^{(k)}\big\}\big\}\Big)$$

where $j = [j_1, ..., j_k]$ are all possible k-tuples formed of elements of $\mathcal{N}_i$.

In the same way that *GWL* is an upper bound for the expressivity of G-equivariant GNNs, *IGWL* constitutes an **upper bound** on the expressivity of *G-invariant GNNs*.

3.5 Understanding the Design Space

The GWL framework can now be used to understand how design choices influence GNN expressivity, to discover what are the limitations of current models and how they could be improved.

3.5.1 Role of depth

Consider two geometric graphs $G_1 = (A_1, S_1, \vec{V}_1, \vec{X}_1)$ and $G_2 = (A_2, S_2, \vec{V}_2, \vec{X}_2)$ such that the underlying attributed graphs (A_1, S_1) and (A_2, S_2) are isomorphic (for example, even though two molecules might have the same atoms connected by the same types of bonds, their spatial arrangements could differ due to rotational or conformational differences). We define G_1 and G_2 as **k-hop distinct** if, for every graph isomorphism b, there exists a node $i \in V_1$ such that $b(i) \in V_2$, and the corresponding k-hop subgraphs $N_i^{(k)}$ and $N_{b(i)}^{(k)}$ are distinct. Conversely, the graphs are **k-hop identical** if, for every graph isomorphism b, all corresponding k-hop subgraphs are identical up to group actions.

In [Joshi et al., 2023] the authors show that:

- Within k iterations, the GWL algorithm can distinguish any k-hop distinct geometric graphs G_1 and G_2, provided their underlying attributed graphs are isomorphic
- Within k iterations, GWL cannot distinguish any k-hop identical geometric graphs G_1 and G_2 where the underlying attributed graphs are isomorphic.

Moreover, they also explore the infleunce of depth for *Invariant GNN*s using the *IGWL* test. and they conclude that no number of iterations of *IGWL* can distinguish any 1-hop identical geometric graphs G_1 and G_2 where the underlying attributed graphs are isomorphic as it cannot propagate geometric orientation information beyond 1-hop.

3.5.2 Limitations of invariant message passing

Given the previous observations, we can conclude that ***GWL* is strictly more powerful than *IGWL*** as it can distinguish a broader class of geometric graphs. Equivariant message passing propagates geometric information across the computational tree, while invariant message passing does not propagate local geometry, and this restricts their expressive power. *IGWL* and *G-invariant* GNNs fail to understand how various 1-hop neighborhoods in a graph are oriented with respect to each other. As a consequence they fail to capture global geometry and thus cannot decide numerous geometric graph properties like: area, volume, and perimeter of the bounding box, distance of the center of mass and dihedral angles. These results have important **practical implications**, as they suggest that *G-equivariant GNNs* are preferable for large geometric graphs (e.g. macromolecules).

3.5.3 Role of scalarization body order

To analyze the role of body order, the authors investigate the IGWL(k) hierarchy. They find that *IGWL(k)* is at least as powerful as *IGWL(k-1)*, and for $k \leq 5$, *IGWL(k)* is strictly more powerful than *IGWL(k-1)*. Thus, increasing the order (k), enables to capture more complex geometric interactions. In other words, higher body scalarization allows to capture richer geometric relationships among multiple nodes, distinguishing more complex spatial patterns. However, the challenge is that this comes with higher computational costs as k increases.

4 Synthetic Experiments

4.1 Depth and Oversquashing

Theory: The Geometric Weisfeiler-Leman (GWL) framework guarantees perfect propagation of geometric information with each iteration.

Practice: In geometric GNNs, stacking multiple layers may distort distant geometric information.

- **k-chains geometric graphs:**$(\lfloor \frac{k}{2} \rfloor + 1)$**-hop distinguishable graphs**
  - **Invariant GNNs** are unable to distinguish k-chains, highlighting the limits of their expressive power.
  - **Equivariant GNNs**, on the other hand, may require more iterations than the GWL framework suggests. This discrepancy points to the **oversquashing** of geometric information across multiple layers, where relevant spatial relationships are progressively compressed and lost.

As theorized, **Invariant GNNs** are inherently limited in their ability to differentiate geometric graphs, while **Equivariant GNNs** may need additional iterations, highlighting the potential issue of **oversquashing** of geometric information as it propagates through layers.

4.2 Higher-Order Tensors and Rotational Symmetry

Theory: The GWL framework aggregates equivariant geometric information perfectly, capturing complex geometric interactions across layers.

Practice: Geometric GNNs face trade-offs when choosing between Cartesian and spherical coordinate systems, and there are considerations regarding the rank of the tensors used.

- **Fold-symmetric structures** remain invariant when rotated by specific angles. However, there are two distinct rotated versions of each structure, and this rotational symmetry presents challenges for certain models.
- **Low-order tensor models** fail to capture the full extent of these symmetries, especially when the structure undergoes rotations beyond the tensor's capacity to represent them.
- **Spherical frameworks** and **higher-order tensors** become necessary for accurate representation and identification of orientations in structures with **rotational symmetry** greater than the order L. Specifically, layers using order-L tensors cannot identify orientations in structures with rotational symmetry greater than L-fold.

This issue is particularly prevalent in models using **first-order tensors** (e.g., Cartesian vectors), which lack the flexibility to handle the rotational symmetries inherent in complex geometric structures.

4.3 Results and Discussion

We summarize the experimental/theoretical results obtained and discuss their implications.

- **Results of the evaluation/experiments:** The experiments evaluated the expressive power of geometric GNNs using the Geometric Weisfeiler-Leman (GWL) framework. Specifically, the performance of invariant and equivariant GNNs was observed in distinguishing various geometric graph classes, focusing on the challenges of oversquashing and rotational symmetry. The experiments showed that equivariant GNNs performed better than invariant ones, especially in graphs with more complex geometric properties.
- **What the results say about the problem considered:** The results suggest that geometric GNNs, particularly equivariant ones, are more effective in preserving geometric information over multiple layers. However, they also highlight the limitations of current methods when dealing with complex symmetries and distant relationships within graphs.
- **Were hypotheses confirmed or disproved? Were results expected or not?** The hypotheses that equivariant GNNs would outperform invariant GNNs in capturing geometric

relationships across layers were confirmed. However, the extent of oversquashing observed in some equivariant models was more pronounced than initially expected, indicating that additional considerations such as higher-order tensors may be necessary for complex graph structures.

- **What can be learned from the results?** The experiments reinforce the importance of understanding the structural limitations of GNNs in geometric domains. It becomes clear that while equivariant GNNs have higher expressive power, the challenge of oversquashing remains. Higher-order tensors and spherical frameworks may provide a solution to better capture rotational symmetries and more complex relationships.
- **Assumptions/limitations of the results:** The primary limitation of the study lies in the reliance on synthetic graphs, which may not fully capture the complexity of real-world geometric data.
- **Advantages/disadvantages of the presented techniques:** The advantages of the techniques presented, particularly GWL and equivariant GNNs, lie in their ability to effectively capture geometric properties such as distances and symmetries across graph layers. However, the trade-off is the need for more computational resources and the risk of oversquashing in deeper networks.

5 Conclusion

The Geometric Weisfeiler-Leman (GWL) framework was introduced to assess the expressive power of geometric GNNs, filling a crucial gap as traditional GNNs and their theoretical tools are inadequate for geometric graphs. Apart from this new framework, the contributions of Joshi et al. [2023] are twofold:

1. They highlight how key design choices influence the expressivity of geometric GNNs using the GWL framework.
2. They present synthetic experiments that uncover challenges in developing expressive geometric models.

A key takeaway from this work is a deeper understanding of the **geometric GNN design space**. Specifically:

- **Invariant layers** exhibit limited expressivity, as they cannot distinguish 1-hop identical geometric graphs.
- **Equivariant layers**, on the other hand, can distinguish larger classes of graphs and propagate geometric information beyond local neighborhoods, thus enhancing model performance.
- The utility of **higher-order tensors** and **scalarization** is emphasized as crucial for building more powerful GNNs capable of capturing complex geometric interactions.

Future Directions. Given this theoretical framework, it is now necessary to develop models that meet the conditions of Proposition 2, which remains challenging in practice. Future work should aim to develop practical, maximally powerful geometric GNNs based on these insights. Some steps have already been made in this direction, with several papers (Amir et al. [2024], Du et al. [2024] , Wang et al. [2024], Li et al. [2024]) leveraging these insights and results to present new geometric deep learning models.

References

Tal Amir et al. Neural injective functions for multisets, measures and graphs via a finite witness theorem. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, 2024.

Minjoon Baek, Frank DiMaio, Igor Anishchenko, Jurgis Dauparas, et al. Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, 2021.

Sebastian Batzner, Armen Musaelian, Lijun Sun, Michael Geiger, Jason P. Mailoa, Michael Kornbluth, Nicholas Molinari, T. Edward Smidt, and Boris E. Kozinsky. (3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *Nature Communications*, 2022.

Lowik Chanussot, Abhishek Das, Siddharth Goyal, Thibaut Lavril, Muhammed Shuaibi, et al. Open catalyst 2020 (oc20) dataset and community challenges. *ACS Catalysis*, 2021.

Yuanqi Du et al. A new perspective on building efficient and expressive 3d equivariant graph neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, 2024.

N. Dym and H. Maron. On the universality of rotation equivariant point cloud networks. In *International Conference on Learning Representations (ICLR)*, 2020.

J. Gasteiger, J. Groß, and S. Günnemann. Directional message passing for molecular graphs. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020. URL https://openreview.net/forum?id=Hylw8aHtPr.

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017.

Amir R. Jamasb, Roberto Vicente Torné, Ethan J. Ma, Yang Du, Christopher Harris, Kangning Huang, David Hall, Pietro Liò, and Tom L. Blundell. Graphein: A python library for geometric deep learning and network analysis on biomolecular structures and interaction networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Chaitanya K. Joshi, Cristian Bodnar, Simon V. Mathis, Taco Cohen, and Pietro Liò. On the expressive power of geometric graph neural networks. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

Qi Li, Fabio Gama, Alberto Ribeiro, and Achim Prorok. Graph neural networks for decentralized multi-robot path planning. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

Zian Li et al. Is distance matrix enough for geometric deep learning? In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, 2024.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, 2020.

K. Schütt, O. Unke, and M. Gastegger. Equivariant message passing for the prediction of tensorial properties and molecular spectra. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2021. URL https://proceedings.mlr.press/v139/schutt21a.html.

K. T. Schütt, H. E. Sauceda, P.-J. Kindermans, A. Tkatchenko, and K.-R. Müller. Schnet–a deep learning architecture for molecules and materials. *The Journal of Chemical Physics*, 148(24): 241722, 2018. doi: 10.1063/1.5019779.

N. Thomas, T. Smidt, S. Kearnes, L. Yang, L. Li, K. Kohlhoff, and P. Riley. Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds. *arXiv preprint arXiv:1802.08219*, 2018. URL https://arxiv.org/abs/1802.08219.

S. Villar, D. W. Hogg, K. Storey-Fisher, W. Yao, and B. Blum-Smith. Scalars are universal: Equivariant machine learning, structured like classical physics. In *NeurIPS*, 2021.

Yusong Wang et al. Enhancing geometric representations for molecules with equivariant vector-scalar interactive message passing. *Nature*, 2024.

Boris Weisfeiler and Andrei Leman. The reduction of a graph to canonical form and the algebra which appears therein. *NTI, Series*, 1968.

A Appendix

A.1 Proof of Theorem 1

Proof by contradiction. Consider two geometric graphs G and H. Then, **Theorem 1** implies that if a GNN output is $f(G) \neq f(H)$, then also the GWL test will always determine that G and H are non-isomorphic, i.e. $G \neq H$. Assume now the opposite and suppose that after T iterations the GWL cannot distinguish G and H (which means they must have the same colours for all the iterations from zero until T), yet the GNN output is $f(G) \neq f(H)$. The proof works by now showing that on the same graph for nodes i and k, if $(c_i^{(t)}, g_i^{(t)}) = (c_k^{(t)}, \mathbf{g} \cdot g_k^{(t)})$, we will always have GNN features such that $(s_i^{(t)}, \vec{v}_i^{(t)}) = (s_k^{(t)}, Q_g \vec{v}_k^{(t)})$ for any iteration t. This holds for $t = 0$ because both GWL and the GNN start with the same initial node features and geometric embeddings.

Inductive Step

Suppose now this holds also for iteration t:

$$\big(c_i^{(t)}, g_i^{(t)}\big) = \big(c_k^{(t)}, g \cdot g_k^{(t)}\big) \implies \big(s_i^{(t)}, \vec{v}_i^{(t)}\big) = \big(s_k^{(t)}, Q_g \vec{v}_k^{(t)}\big).$$

Then at iteration $t+1$ we will have:

$$\big(c_i^{(t+1)}, g_i^{(t+1)}\big) = \big(c_k^{(t+1)}, g \cdot g_k^{(t+1)}\big),$$

then:

$$\Big\{\big(c_i^{(t)}, g_i^{(t)}\big), \big\{\big\{\big(c_j^{(t)}, g_j^{(t)}\big) \mid j \in \mathcal{N}_i\big\}\big\}\Big\} = \Big\{\big(c_k^{(t)}, g \cdot g_k^{(t)}\big), \big\{\big\{\big(c_j^{(t)}, g \cdot g_j^{(t)}\big) \mid j \in \mathcal{N}_k\big\}\big\}\Big\}$$

Since the GNN applies the same aggregate and update operations at every node, the same inputs are mapped to the same outputs. Thus, by our assumption on iteration t:

$$\big(s_i^{(t+1)}, \vec{v}_i^{(t+1)}\big) = \big(s_k^{(t+1)}, Q_g \vec{v}_k^{(t+1)}\big).$$

By induction, this holds for all t.

If G and H have the same collection of node colors and geometric multisets for all iterations t, then the GNN produces the same collection of node features:

$$\Big\{\big(s_i^{(t)}, \vec{v}_i^{(t)}\big), \big\{\big\{\big(s_j^{(t)}, \vec{v}_j^{(t)}\big) \mid j \in \mathcal{N}_i\big\}\big\}\Big\}$$

Thus, the graph-level readout of the GNN outputs $f(G) = f(H)$, contradicting the assumption.