ComENet: Towards Complete and Efficient Message Passing for 3D Molecular Graphs

Anonymous Author(s)
Affiliation
Address
email

Abstract

Many real-world data can be modeled as 3D graphs, but learning representations that incorporates 3D information completely and efficiently is challenging. Existing methods either use partial 3D information, or suffer from excessive computational cost. To incorporate 3D information completely and efficiently, we propose a novel message passing scheme that operates within 1-hop neighborhood. Our method guarantees full completeness of 3D information on 3D graphs by achieving global and local completeness. Notably, we propose the important rotation angles to fulfill global completeness. Additionally, we show that our method is orders of magnitude faster than prior methods. We provide rigorous proof of completeness and analysis of time complexity for our methods. As molecules are in essence quantum systems, we build the complete and efficient graph neural network (ComENet) by combing quantum inspired basis functions and the proposed message passing scheme. Experimental results demonstrate the capability and efficiency of ComENet, especially on real-world datasets that are large in both numbers and sizes of graphs.

1 Introduction

In machine learning, structured objects such as molecules [14, 50], proteins [9, 16, 33, 37, 26], materials [51], and quantum systems [31] are usually modeled as graphs. Original modeling shows basic connections between units and the resulted data type is known as 2D graphs. Accordingly, 2D graph neural networks (GNNs) have been intensively studied [14, 21, 19, 20, 52, 56, 32] and the message passing scheme [22, 7, 49] is shown effective to realize 2D GNNs. However, in the modern machine learning era, it is increasingly accepted that modeling real-world data like molecules as 2D graphs leads to inborn defects for succeeding learning models. In practice, 3D information is crucial, based on which some important geometries can be derived, such as chemical bond lengths in molecular modeling. This essentially raises the need of a new data type, known as 3D graphs.

Formally, a 3D graph contains the original 2D graph as well as Cartesian coordinates for all nodes. In this work, we follow invariant 3D GNNs [41, 29, 28, 43, 34, 30] that take relative 3D information, like distances and angles, as inputs to networks. Such relative geometries are naturally SE(3)-invariant. The main challenge comes from what geometries should be computed such that 3D information is incorporated completely. A most recent work SphereNet [34] shows that distance, angle, and torsion information is necessary to incorporate more comprehensive 3D information. However, SphereNet is only complete in local neighborhood, failing to achieve global completeness and distinguish a wide range of molecular structures such as conformers. In addition, existing methods exhibit excessive time complexity of $O(nk^2)$ or even $O(nk^3)$, severely preventing their scalability in real-world applications. Here n and k denote the number of nodes and the average degree in a 3D graph.

In this work, we propose ComENet as a complete and efficient graph neural network for 3D molecular graph learning. We first formally provide the definition of *completeness*. Intuitively, a geometric

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

transformation is considered as complete if it generates distinct representations for any two different 3D graphs. Based on this, we propose a novel message passing scheme by faithfully fulfilling global completeness via the important rotation angles. In addition, we design novel strategies to achieve local completeness, largely reducing the computing complexity to $O(nk)$. To elucidate the merits of the proposed methods, we provide rigorous proof of geometric completeness achieved by our method. Combining the novel message passing scheme and quantum inspired features, ComENet is developed for 3D molecular graphs. We apply ComENet to two large-scale datasets including OC20 and Molecule3D, and a commonly used dataset QM9. Experiments show that ComENet performs similar to existing best methods, but accelerates the training and inference by 6-10 times on various datasets. **We summarize the contributions of ComENet as below**. (i). To our best knowledge, it is the first rigorously complete pipeline for 3D molecular graph learning. Theoretically, it is guaranteed to incorporate 3D information completely without information loss. Practically, it can distinguish all molecular structures in nature. (ii). It is highly efficient. The message passing is shown to be orders of magnitude faster than existing methods in terms of time complexity. (iii). The great capability and efficiency of ComENet allow its scalability to real-world molecule datasets that are large in both numbers and sizes of graphs. (iv). It achieves similar or better performance compared with existing methods, and dramatically accelerates the training and inference by 6-10 times.

Relations with Prior Work. SphereNet [34] is a recent method that achieves local completeness with a complexity of $O(nk^2)$. In summary, SphereNet is not complete and not efficient enough for processing large-scale molecular graphs. However, ComENet is complete and much more efficient with a complexity of $O(nk)$. Technically, a primary difference is that, ComENet proposes to use the important rotation angles to achieve global completeness. As a result, ComENet is provably complete as shown in Sec. 2.2. Practically, ComENet is able to identify a wide range of real-world structures such as conformers, achieving completeness at the conformer level, as introduced in Sec. 2.3. In addition, ComENet and SphereNet both use (d, θ, ϕ) in the spherical coordinate system (SCS) to obtain local completeness. Indeed, the fact that (d, θ, ϕ) can specify the location of a point in SCS is widely known. The key difference lies in that, SphereNet operates within 2-hop neighborhood, while ComENet operates within 1-hop neighborhood. When coupled with rotation angles, ComENet can achieve provable completeness with a reduced complexity of $O(nk)$. This different message passing for local completeness in ComENet entails many differences with SphereNet, including building coordinate systems, defining z-axis, choosing reference nodes, and computing (d, θ, ϕ), as detailed in Sec. 2.4. All the computing procedures for ComeNet are described in detail in Algorithm 1 of Appendix A.1.

2 The Proposed Message Passing Scheme

2.1 Notations & Definitions

We first formally define notations and the concept of *completeness* used in this paper.

Notations. A 3D graph G can be represented as $G = (V, A, P)$. The node feature matrix $V = [\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_n]^T \in \mathbb{R}^{n \times d_v}$ with each $\mathbf{v}_i \in \mathbb{R}^{d_v}$. The adjacency matrix $A \in \mathbb{R}^{n \times n}$, based on which we additionally define there is an edge e_{ij} if $A[i][j] = 1$. The position matrix $P = [\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_n]^T \in \mathbb{R}^{n \times 3}$, where $\mathbf{p}_i = (x_i, y_i, z_i) \in \mathbb{R}^3$ is the position vector for node i given in the Cartesian coordinate system (CSC). The relative position $\mathbf{p}_{ij}$ of node j to node i is defined as $\mathbf{p}_{ij} = \mathbf{p}_j - \mathbf{p}_i$. Particularly, throughout this paper, we define k as the average degree for G.

We then formally define *completeness* given a geometric transformation $\mathcal{T}$. In particular, we aim at incorporating 3D information in 3D molecular graphs. Hence, our definition of *completeness* is set from the geometric view. Generally, $\mathcal{T}$ maps a 3D graph $G = (V, A, P)$ to a geometric representation with size $m \times d$. $\mathcal{T}$ can be different dependent on different methods, resulting in different $m \in \mathbb{N}^+$ or $d \in \mathbb{N}^+$. For example, SchNet generates a distance for each edge, thus, it maps G to a representation with size $m \times d$, where m is the number of edges in G and $d = 1$. However, such geometric transformation is not complete. We provide the definition of *completeness* as below:

Definition 1 (Completeness)**.** *For two 3D graphs $G_1 = (V, A, P_1)$ and $G_2 = (V, A, P_2)$, a geometric transformation $\mathcal{T} : (\mathbb{R}^{n \times d_v}, \mathbb{R}^{n \times n}, \mathbb{R}^{n \times 3}) \mapsto \mathbb{R}^{m \times d}$ is considered as complete when*

$$\mathcal{T}(G_1) = \mathcal{T}(G_2)$$

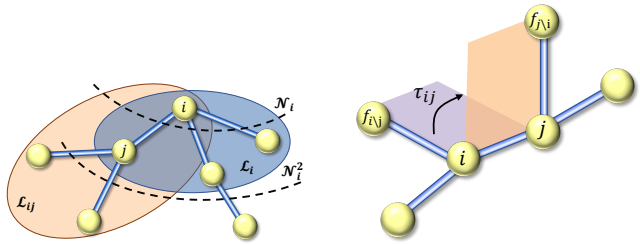
if and only if

$$P_1 \in R(P_2), \forall R \in \mathrm{SE}(3).$$

Intuitively, a complete geometric transformation $\mathcal{T}$ can distinguish any two different 3D graphs. This is to say, as long as two 3D graphs differ in structures, their outputs from $\mathcal{T}$ would be different.

2.2 Global Completeness via Rotation Angles

Existing studies focus on the complete representation learning of local neighborhood. Earlier methods like SchNet and DimeNet cannot achieves local completeness. In a more recent method SphereNet, completeness is guaranteed within local neighborhood, but fails to hold in the whole 3D graph. In this section, we move a step forward to formally fulfill global completeness for a given 3D molecular graph. Particularly, for the purpose of clear illustration, we safely assume local completeness is already obtained by exiting methods like SphereNet.



(a) Illustration of a 2-hop local structure. (b) Illustration of computing the rotation angle for an edge e_{ij}.

Figure 1: Illustrations of how to achieve global completeness in our proposed methods.

Without loss of generality, we start by describing our method to attain full completeness within 2-hop neighborhood, as illustrated in Fig. 1(a). Formally, for a center node i, we let $\mathcal{N}_i$ and $\mathcal{N}_i^2$ denote two sets of indices of i's 1-hop and 2-hop neighboring nodes, respectively. We also define any node i and its 1-hop neighborhood as a local structure. Then the whole 2-hop neighborhood of node i can be viewed as $1 + |\mathcal{N}_i|$ local structures centered in i and $\mathcal{N}_i$, defined as $\mathcal{L}_i$ and $\mathcal{L}_{ij,j\in\mathcal{N}_i}$, respectively. Apparently, each local structure $\mathcal{L}_{ij,j\in\mathcal{N}_i}$ shares the common edge e_{ij} with the local structure $\mathcal{L}_i$. Given the complete representation for each local structure, for the structure $\mathcal{L}_i \cup \mathcal{L}_{ij}$, the only remaining degree of freedom is the rotation angle of edge e_{ij}, denoted as τ_{ij}. With the rotation angles for all the $|\mathcal{N}_i|$ common edges specified, we can obtain a complete representation for 2-hop neighborhood of node i. Achieving completeness beyond 2-hop neighborhood is similar. Overall, after considering rotation angles, the global completeness can be easily guaranteed when it gradually generalizes from n- to $(n+1)$-hop neighborhood.

Essentially, each edge in an input graph can be treated as a common edge between different local structures. We reveal how to compute the rotation angle τ_{ij} for each edge e_{ij} in Fig. 1(b). Specifically, we choose two reference nodes whose indices are $f_{i\backslash j}$ and $f_{j\backslash i}$ for node i and j, respectively. $f_{i\backslash j}$ denotes the index of i's nearest neighboring node except j, and $f_{j\backslash i}$ denotes j's nearest neighboring node except i. Then τ_{ij} for edge e_{ij} is the angle from the plane formed by $f_{i\backslash j}, i, j$ to the plane formed by $i, j, f_{j\backslash i}$. As analyzed previously, the global conformation of the input graph can be identified based on all local structures and rotation angles. As a result, given fixed local structures, the global completeness the input 3D molecular graph is fulfilled by additionally considering the rotation angle of each edge, as introduced in this section.

2.3 Rotation Angles for Conformer Identification

The proposed rotation angles in Sec. 2.2 play a crucial role in identifying some important molecular structures, such as conformers. In nature, a real molecule exits as an ensemble of interconventing 3D structures, known as conformers [5, 4, 18]. Different conformers posses the same 2D molecular graph, but differ in 3D structures. Generally, a conformer for a molecule exits with a certain probability and may exhibit distinct properties [5, 4]. As shown in Fig. 2(a), different conformers for the molecule butane (C_4H_{10}) show varying conformation energy. To this end, it is important to design complete 3D GNNs for identifying molecules at the conformer level. From the geometry perspective of view, a conformer distinguishes itself from others mainly through varying rotation angles of chemical bonds [18]. As shown in Fig. 2(b), given the fixed ethyl (-C_2H_5) of both sides, the only degree of freedom is the rotation angle of the C-C bond. In literature, the ethyl is formulated as a local 3D graph. Existing studies focus on the complete representation of such local structures, failing to identify the whole 3D graph globally. Essentially, they can only distinguish different molecules, trying to achieve

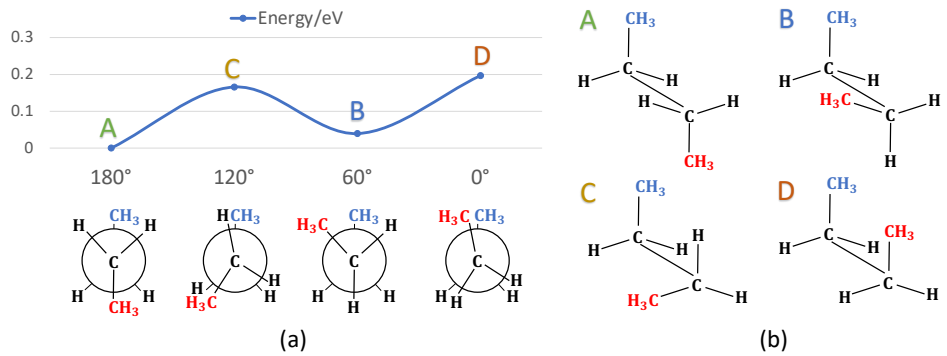


Figure 2: (a). Illustration that the relative conformation energy of butane is a function of the rotation angle of the C-C bond. (b). A 3D view of the four conformers in (a).

completeness at the molecule level rather that the finer comformer level. By integrating the rotation angles as in Sec. 2.2 into the message passing scheme, our methods can fulfill rigorous completeness at the conformer level and can distinguish all conformers in nature.

2.4 Local Completeness with Improved Efficiency

Formally, each node and its local neighborhood can be viewed as a local structure. Existing studies focus on the learning of local structures and SphereNet achieves local completeness. However, SphereNet induces the complexity of $O(nk^2)$, restricting its scalability on large molecules in practice. Here, we design a novel strategy to guarantee local completeness with the computing cost of $O(nk)$.

Specifically, we follow SphereNet and perform on the spherical coordinate system. It is commonly known that the location of each node can be completely determined using the tuple (d, θ, ϕ) in SCS. SphereNet employs the directional message passing (DMP) fashion that operates within 2-hop neighborhood. It first updates messages over edges thus the center edge is z-axis in SCS. For node i, the computing of the tuple (d, θ, ϕ) involves 2-hop information. However, we view 1-hop neighborhood as a local structure, which requires all strategies in our local completeness to be different from SphereNet, including building local coordinate systems, defining z-axis, picking reference nodes, and computing (d, θ, ϕ). First, we build a light local coordinate system for any node i's corresponding local structure. Similarly, the center node i serves as the origin. Then z-axis is defined as the direction from i to its nearest neighbor f_i, and xz-plane is further formed by z-axis and i's second nearest neighbor s_i. Finally, the tuple (d, θ, ϕ) is computed within 1-hop neighborhood with a complexity of $O(nk)$. Particularly, we analyze efficiency versus model expressiveness in Sec. 3.2. We show that compared with the DMP fashion used by SphereNet, our method operating within 1-hop neighborhood hurts the model expressiveness a bit by largely improves the efficiency.

2.5 Message Passing Scheme

Based upon global completeness achieved in Sec. 2.2 and improved local completeness introduced in Sec. 2.4, the complete geometric transformation $\mathcal{T}$ required by Def. 1 should be formulated based on a 4-tuple as (d, θ, ϕ, τ). Specifically, we build such transformation within 1-hop neighborhood, and a 4-tuple is computed for each edge. Hence, for a 3D graph $G = (V, E, P)$, the full expression for $\mathcal{T}$ is $\mathcal{T}(G) = [(d_{ij}, \theta_{ij}, \phi_{ij}, \tau_{ij})]_{i=1,\ldots,n; j\in\mathcal{N}_i} \in \mathbb{R}^{m\times 4}$, where m is the number of edges in G. Especially, as $\mathcal{T}$ converts absolute Cartesian coordinates in P to relative information, it is naturally SE(3)-invariant as required in Def. 1. Thus, we formally build our message passing scheme as

$$\mathbf{v}_i' = g\left(\mathbf{v}_i, \sum_{j\in\mathcal{N}_i} f\left(\mathbf{v}_j, d_{ij}, \theta_{ij}, \phi_{ij}, \tau_{ij}\right)\right), \tag{1}$$

where g and f can be implemented by neural networks or mathematical operations. Intuitively, our message passing is established in 1-hop local neighborhood and all edges connecting regions beyond. Essentially, d_{ij}, θ_{ij}, and ϕ_{ij} specify the 1-hop local neighborhood, and τ_{ij} determines the orientation of the local neighborhood. By doing this, the complete representation for a whole 3D molecular graph is eventually achieved. The formulas of computing of d_{ij}, θ_{ij}, and ϕ_{ij}, and τ_{ij} are shown in Algorithm 1 in Appendix A.1, along with detailed description of the complexity of $O(nk)$.

3 Merits of Our Methods

3.1 Geometric Completeness

Proposition 1. *For a strongly connected 3D graph $G = (V, E, P)$, its geometric transformation $\mathcal{T}(G) = [(d_{ij}, \theta_{ij}, \phi_{ij}, \tau_{ij})]_{i=1,\dots,n;j\in\mathcal{N}_i}$ is complete.*

Proof. We employ mathematical induction and assume the number of nodes in a 3D graph is n.

Base case: It is obvious that the 3D structure of G can be identified when $n = 1, 2$. Hence, we let $n = 3$ be the base case, where the completeness can be achieved by only considering d and θ in $\mathcal{T}$.

Inductive hypothesis: The claim that $\mathcal{T}$ is complete holds for the node numbers of n up to $k \geq 3$.

Inductive step: Let $n = k + 1$. Without loss of generality, among the existing k nodes, we safely assume i and its neighboring nodes $c_{k=1,2,\dots}$ form the local region of interest. Then j is the index of the newly $(k + 1)$-th node connected to the center node i. To show global completeness of the whole graph, based on Def. 1, we only need to prove that the relative position of the new node j is uniquely determined given $\mathcal{T}$. We propose the following lemma for this as

Lemma 1. *Assume a strongly connected 3D graph $G = (V, E, P)$ with more than 2 nodes is fully identified. If a new node j is connected to a node i of G following the geometric transformation $\mathcal{T}(G) = [(d_{ij}, \theta_{ij}, \phi_{ij}, \tau_{ij})]_{i=1,\dots,n;j\in\mathcal{N}_i}$, then $\boldsymbol{p}_{ij}$ is uniquely determined.*

The proof of Lemma 1 is provided in Appendix A.2. With Lemma 1 successfully proved, we show that such geometric transformation $\mathcal{T}$ can determine a unique 3D graph. Hence, the *if* condition in Def. 1 holds. In addition, as $\mathcal{T}$ renders purely relative 3D information like distance and angle, it's naturally SE(3) invariant. Hence, the *only if* condition in Def. 1 holds. Overall, based on Def. 1, we complete the proof of Prop. 1. □

Intuitively, since a molecular graph is strongly connected, two arbitrary nodes are connected by at least one path. Hence, starting from the existing structure, we can restore the relative position of any new node step by step along a path with finite length. As a result, geometric completeness for the whole 3D graph with any number of nodes can be guaranteed in our message passing scheme. Theoretically, 3D information of the input 3D molecular graph is fully captured without information loss. In practice, our method can distinguish all structures in nature.

3.2 Efficiency

Efficiency versus Model Expressiveness: Our method induces the complexity of $O(nk)$ by operating within 1-hop neighborhood. Existing methods, such as like DimeNet and SphereNet, employ the DMP fashion that update edges within 2-hop neighborhood, inducing the complexity of $O(nk^2)$. Notably, DMP [46, 54, 29, 34] incorporates 2-hop information in one single layer but $(n + 1)$-hop when stacking n layers. For each center edge In DMP, stacking n DMP layers incorporates information from its n-hop neighboring edges. To further aggregate information from edges to nodes, studies like Stokes et al. [46], Yang et al. [54] use an aggregation layer after the n-th DMP layer while Klicpera et al. [29], Liu et al. [34] add such aggregation layer after each DMP layer. Then each node is updated based on the nodes within $(n + 1)$-hop edges. Thus, stacking n DMP layers integrates $(n + 1)$-hop information. However, stacking n proposed message passing layers is already able to incorporate information from n-hop neighborhood. Obviously, compared with a network containing several DMP layers, a network with the same number of our proposed message passing layers merely hurts the model expressiveness a bit, but significantly improves the model efficiency.

Efficiency via Less Torsion Angles: In addition, our method achieves completeness by computing $O(n)$ torsion angles, which is efficient especially compared with methods including Klicpera et al. [30], Adams et al. [1], Ganea et al. [18]. Given a scenario where a global region is the union of two local regions with n_1 and n_2 nodes. As introduced in Sec. 2.2 and Sec. 2.4, our method computes the same number of torsion angles as nodes in a local region, then employs a rotation angle (torsion angle essentially). Hence, the number of torsion angles that our method computes is $n_1 + n_2 + 1$. Ganea et al. [18] computes a torsion angle based on one pair of nodes, each of which is from a separate local region. Hence, the number of torsion angles needed is $n_1 \times n_2$. Apparently, our method reduces the number of torsion angles from $O(n^2)$ to $O(n)$, which is significant considering the computing of torsion is excessively expensive.

4 ComENet

Based upon the message passing scheme introduced in Sec. 2.5, we propose the complete and efficient graph neural network (ComENet) as shown in Fig. 3. Generally, ComENet consists of an embedding layer, multiple interaction layers, a self-atom layer, and a pooling layer. To be in line with the message passing scheme in Sec. 2.5, we take the updating process for node i's feature vector $\mathbf{v}_i$ as an example to describe the network. In particular, we omit all indices in Fig. 3 and below for clear presentation.

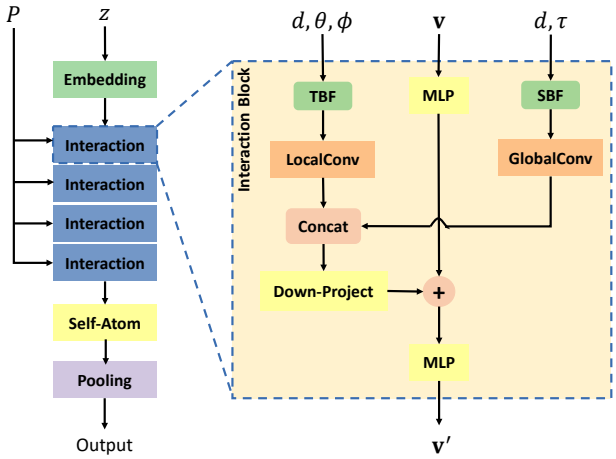


Figure 3: Illustration of ComENet with an overview (left) and the interaction layer (right). TBF and SBF denote the basis functions for tuples (d, θ, ϕ) and (d, τ). LocalConv and GlobalConv denote the proposed local and global convolution layers. Concat is the concatenation operation and Down-Project is a linear layer to reduce feature dimensions. + denotes the element-wise sum operation.

Embedding Layer converts each atom type z to an initial node feature vector via learnable atom type embeddings [41, 29].

Interaction Layer updates each node feature vector $\mathbf{v}$ based on features of the neighboring nodes and the corresponding 3D information in P. Firstly, it converts 3D information in P to a set of geometries based on the proposed complete geometric transformation $\mathcal{T}$ and message passing scheme. Since distance is the most important geometry, we also consider d in global representation and split the output of $\mathcal{T}$ into two tuples (d, θ, ϕ) and (d, τ), for local and global representations, respectively.

Importantly, the tuples (d, θ, ϕ) and (d, τ) cannot serve as immediate inputs to the network. They need to be transformed into physically meaningful vectors based on quantum-based basis functions. As in previous studies [25, 29, 34, 30], we test different basis functions including MLP, Gaussian and sine functions, spherical Bessel functions, and spherical harmonics. We found spherical Bessel and spherical harmonics perform best. Formally, the basis function for tuple (d, θ, ϕ) is TBF $j_\ell\left(\frac{\beta_{\ell n}}{c} d\right) Y_\ell^m(\theta, \phi)$, where $j_\ell(\cdot)$ is a spherical Bessel function of order ℓ, Y_ℓ^m is a spherical harmonic function of degree m and order ℓ, c is the cutoff, and $\beta_{\ell n}$ is the n-th root of the Bessel function of order ℓ. The basis function for tuple (d, τ) is SBF $j_\ell\left(\frac{\beta_{\ell n}}{c} d\right) Y_\ell^0(\tau)$. These two basis functions are also used in SphereNet [34] and GemNet [30].

The two physically meaningful vectors from TBF and SBF are then imported into a local convolution layer and a global convolution layer, respectively. For both convolution layers, we use the GraphConv [38] implemented in the PyTorch Geometric library [15]. The vectors from the basis functions are used as edge weights in the convolution layers. The outputs of local and global convolution layers are concatenated to generate a new node feature vector. Then the concatenated vector is forwarded into several linear layers to generate the updated feature vector $\mathbf{v}'$.

Self-Atom and Pooling layers. The self-atom layer is used to update each node feature and project the feature dimension into 1, similar to the atom-wise layer used in SchNet. The pooling layer is a sum-pooling performing on all node features to obtain final predictions.

5 Related Work

We consider how to represent 3D information in 3D graphs for facilitating molecular learning tasks [3, 10]. One category of methods are equivariant 3DGNNs that directly use coordinates in the CSC as inputs to networks [47, 2, 17, 42, 8]. These methods are efficient and do not need to consider nodes beyond 1-hop neighborhood. However, they suffer from several setbacks. Firstly, each network component needs to be carefully designed to be rotation equivariant of input graphs. For example, NequIP [8] employs equivariant convolutions based on spherical harmonics and Clebsch-Gordon

transforms. Non-linearities also need careful design and common techniques like ReLu cannot be used. Secondly, the reason why this category of methods are efficient lies in that they only use the type-1 basis in Spherical harmonics, which is an approximation essentially. It is proved in tensor field networks [47] that theoretically, l is infinite in terms of type-l basis. In practice, l should be at least 2 for achieving satisfactory performance. Type-1 basis essentially coerces the conv kernel to be in a narrow learning space, which imposes a hard constraint on the network capability. However, when using type-2 basis, the conv kernel can be more expressive while the efficiency issue would emerge as a new bottleneck. Thirdly, the performance of such equivariant GNNs is shown to be worse than the other category that is invariant 3D GNNs [34].

In this work, we follow invariant 3D GNNs to inherit the merit of SE(3)-invariance by using relative 3D information, such as bond distances, angles between bonds, and angles between planes. Such relative information is used in both representation learning tasks [41, 29, 28, 43, 34, 30] and coordinates generation tasks [18, 44, 45, 53, 26, 6]. Since the relative information is SE(3)-invariant of input 3D graphs, the employed networks favorably achieve the invariance merit. We focus on the 3D graph learning problem and existing methods either capture partial 3D information or suffer from high computational cost. For example, SchNet [41] only considers distance information and DimeNet [29] further incorporates angles between bonds. They both integrate incomplete 3D information that the network capacity is limited in practice. SphereNet [34] generates approximate complete 3D representations by using distance, angle, and torsion information but the complexity is $O(nk^2)$. GemNet [30] is based on quadruplets of nodes, which is more expensive. Our objective is to build a fully complete 3D graph net with a much lower computational budget. There also exist some methods in literature using both absolute and relative 3D information [23, 25, 55]. In this work, we only use relative information as input to avoid inherent limitations of equivariant 3D GNNs. Moreover, we design a novel message passing such that the computational cost is comparable with that of equivariant 3D GNNs.

Table 1: Statistics of the datasets.

Dataset	OC20	Molecule3D	QM9
# Graphs	659,110	3,899,647	130,831
Split Type	Pre-defined	Random/Scaffold	Random
Split Ratio	70:15:15	6:2:2	84:8:8
# Nodes/Graph	77.75	29.11	18.02

6 Experiments

We examine the power and efficiency of ComENet on two large-sacle datasets including Open Catalyst 2020 (OC20) [12] and Molecule3D [53], and the mostly commonly used datastet QM9 [40]. The statistics of three datasets are provided in Table 1. Detailed descriptions of the datasets are provided in Appendix A.3. In particular, the Molecule3D contains about 4 million 3D molecular graphs, and the OC20 has the average graph size of 77.75. Baseline methods include GIN-Virtual [24], CGCNN [51], SchNet [41], PhysNet [48], MGCN [35], DimeNet [29], DimeNet++ [28], SphereNet [34], PaiNN [42], GemNet [30]. Unless otherwise specified, the values for baseline methods are taken from the referred papers. For the ComENet, we use data loader in the PyTorch Geometric library [15] to load the datasets. All the models are trained using the Adam optimizer [27] and the optimal hyperparameters are obtained on validation sets using grid search. Experimental setup and search space for all models are provided in Appendix A.4. Code will be released after the anonymous review period.

Table 2: Results on IS2RE including computing cost in training&inference and performance in terms of energy MAE and the percentage of EwT of the ground truth energy. Training time is the average time per epoch during training using 1 GPU. Performance is reported for models trained on the All training dataset. The best performance is shown in bold and the second best is shown with underlines.

	Time		Energy MAE [eV] ↓					EwT ↑				
Model	Train	Infer.	ID	OOD Ads	OOD Cat	OOD Both	Average	ID	OOD Ads	OOD Cat	OOD Both	Average
CGCNN	18min	1min	0.6203	0.7426	0.6001	0.6708	0.6585	3.36%	2.11%	3.53%	2.29%	2.82%
SchNet	10min	1min	0.6465	0.7074	0.6475	0.6626	0.6660	2.96%	2.22%	3.03%	2.38%	2.65%
DimeNet++	230min	4min	0.5636	0.7127	0.5612	0.6492	0.6217	4.25%	2.48%	4.40%	2.56%	3.42%
GemNet-T	200min	4min	0.5561	0.7342	0.5659	0.6964	0.6382	4.51%	2.24%	4.37%	2.38%	3.38%
SphereNet	290min	5min	0.5632	0.6682	0.5590	0.6190	0.6023	**4.56%**	2.70%	**4.59%**	2.70%	**3.64%**
ComENet	20min	1min	**0.5558**	**0.6602**	**0.5491**	**0.5901**	**0.5888**	4.17%	**2.71%**	4.53%	**2.83%**	3.56%

6.1 OC20

The Open Catalyst 2020 (OC20) dataset is a newly released large-scale dataset with millions of DFT relaxations to model and discover catalysts. In this work, we focus on Initial Structure to Relaxed Energy (IS2RE) task, which is the most common task in catalyst discovery. Descriptions of the data and tasks are provided in Appendix A.3. The ground truth of the test set is not publicly available, therefore, we compare the results of different methods on the validation set. The evaluation metrics include the energy MAE and the percentage of Energies within a Threshold (EwT) of the ground truth energy. The values for the baseline methods are taken from Chanussot et al. [11], Liu et al. [34]. Notably, we aim to predict relaxed energy directly from initial structure and do not compare with some methods using relaxation [12], trajectory information, or relaxed structures. Using relaxation [12, 30, 43] is computationally expensive during prediction while the relaxation trajectory and relaxed structures [23, 55] are hard to obtain in practice.

Table 2 shows that ComENet outperforms all the baseline methods in terms of energy MAE, which is also used as the main evaluation metric in the Open Catalyst Challenge [13]. ComENet achieves best performance on two splits and the second best on the other two splits in terms of EwT. Specifically, ComENet reduces the average energy MAE by 0.0135, which is 2.2% of the second best model. Note that ComENet achieves the best results on the OOD Both split in terms of both energy MAE and EwT. In practice, it is common that test data is in the different domain with the training data. Hence, OOD Both can test the generalization capability of learning models. More importantly, ComENet is much more efficient than methods like DimeNet++ and SphereNet. For example, SphereNet needs 5 hours per epoch while ComENet only requires 20 minutes using the same computing infrastructure (NVIDIA RTX A6000 48GB). Overall, compared with existing best methods, ComENet achieves better performance and largely reduces training by at least 10 times.

6.2 Molecule3D

The Molecule3D dataset [53] is a newly proposed large-scale dataset, including around 4 million molecules with precise ground-state 3D information derived from DFT and molecular properties. We focus on the prediction of the HOMO-LUMO gap as it is one of the most practically-relevant quantum chemical properties of molecules. A detailed description of the Molecule3D dataset is provided in Appendix A.3. As this is a newly proposed dataset, we run baseline methods including GIN-Virtual [24], SchNet [41], DimeNet++[28] and SphereNet [34], among

Table 3: Comparisons between ComENet and other models in terms of computing cost and HOMO-LUMO gap MAE on Molecule3D for both random and scaffold splits. Train time is the average training time per epoch.

	Time		MAE	
Model	Train	Inference	Random	Scaffold
GIN-Virtual	15min	2min	0.1036	0.2371
SchNet	14min	3min	0.0428	0.1511
DimeNet++	133min	16min	0.0306	0.1214
SphereNet	182min	28min	0.0301	0.1182
ComENet	22min	3min	0.0326	0.1273

Table 4: Comparisons between ComENet and other models in terms of MAE and the overall mean std. MAE on QM9.

Property	Unit	SchNet	PhysNet	MGCN	DimeNet	DimeNet++	PaiNN	SphereNet	ComENet
μ	D	0.033	0.0529	0.0560	0.0286	0.0297	0.012	0.0245	0.0245
α	${a_0}^3$	0.235	0.0615	0.0300	0.0469	0.0435	0.045	0.0449	0.0452
ϵ_{HOMO}	meV	41	32.9	42.1	27.8	24.6	27.6	22.8	23.1
ϵ_{LUMO}	meV	34	24.7	57.4	19.7	19.5	20.4	18.9	19.8
$\Delta\epsilon$	meV	63	42.5	64.2	34.8	32.6	45.7	31.1	32.4
$\langle R^2 \rangle$	${a_0}^2$	0.073	0.765	0.110	0.331	0.331	0.066	0.268	0.259
ZPVE	meV	1.7	1.39	1.12	1.29	1.21	1.28	1.12	1.20
U_0	meV	14	8.15	12.9	8.02	6.32	5.85	6.26	6.59
U	meV	19	8.34	14.4	7.89	6.28	5.83	6.36	6.82
H	meV	14	8.42	14.6	8.11	6.53	5.98	6.33	6.86
G	meV	14	9.4	16.2	8.98	7.56	7.35	7.78	7.98
c_{v}	$\frac{\text{cal}}{\text{mol K}}$	0.033	0.028	0.038	0.025	0.023	0.024	0.022	0.024
std. MAE	%	1.76	1.37	1.86	1.05	0.98	1.01	0.91	0.93

which GIN-Virtual is a powerful baseline for 2D graphs while the others are for 3D graphs. All the models are trained using the same computing infrastructure (Nvidia GeForce RTX 2080 Ti 11GB).

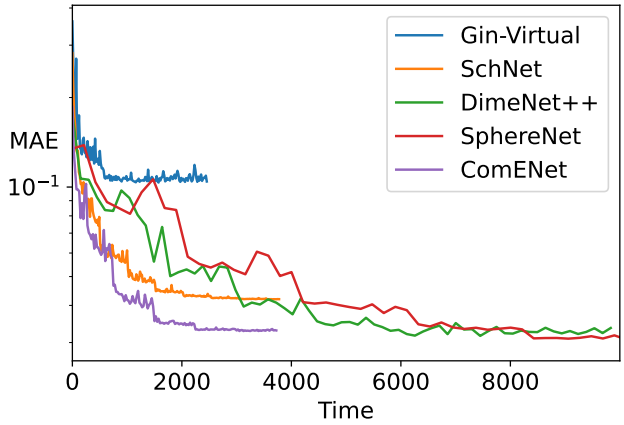


Figure 4: Total training time for different methods on Molecule3D.

Table 3 shows that ComENet dramatically reduces training time by 6-9 times compared with DimeNet++ and SphereNet, and costs similar time as GIN-virtual and SchNet that only considers distance information. In terms of performance, our ComENet performs much better than SchNet for both random and scaffold splits with similar time and computing costs, but a little worse than DimeNet++ and SphereNet. This may be due to the molecules are relatively small in Molecule3D compared with OC20, the structures that our complete strategy can distinguish may not exist in the dataset. However, considering the comparable performance and high efficiency, our ComENet is more practically useful than other methods on such large datasets. In addition, Fig. 4 also shows ComENet either converges much faster in terms of total training time or performs much better compared with other baselines.

6.3 QM9

The QM9 dataset is a widely used dataset for predicting various properties of molecules. The evaluation metrics include the MAE for each property and the overall mean standardized MAE (std. MAE) for all the 12 properties. Notably, we do not list results for PPGN [36] and Cormorant [2] since they use different training/val/test sizes. Table 4 shows that ComENet is much better than the methods operating in 1-hop neighborhood like SchNet, PhyNet, and MGCN. Compared with DimeNet, DimeNet++, PaiNN, and SphereNet, our ComENet achieves similar results on all properties and the overall mean std. MAE. Consistently, compared with those methods operating within 2-hop neighborhood including DimeNet and SphereNet, ComENet is much more efficient.

6.4 Ablation Study

The proposed ComENet considers both local and global representations with two tuples (d, θ, ϕ) and (d, τ) to achieve full completeness. We investigate contributions of different representations to demonstrate the advances of our complete ComENet. We remove local and global convolutional layers from ComENet and denote as "w/o (d, θ, ϕ)" and "w/o (d, τ)", respectively. The other parts of the network are remaining the same. Table 5 shows that the ComENet performs the best and the other two models achieve similar results. These results demonstrate the effectiveness of both local and global representations in ComENet.

Table 5: Comparisons between ComENet and models with partial 3D information on 4 properties of QM9 in terms of MAE.

Property	w/o (d, θ, ϕ)	w/o (d, τ)	ComENet
ϵ_{HOMO}	24.3	24.1	23.1
ϵ_{LUMO}	20.5	20.0	19.8
U_0	6.89	6.73	6.59
μ	0.0273	0.0278	0.0245

7 Conclusion

3D information is crucial for 3D molecular graph learning. Existing methods either learn partial 3D information or induce high time complexity. We propose ComENet that is both complete in incorporating 3D information and efficient with time complexity of $O(nk)$. Particularly, we propose the novel rotation angles to fulfill global completeness. ComENet can generalize to large-scale datasets, accelerating training and inference by 6-10 times with superior or comparable performance.

References

[1] Keir Adams, Lagnajit Pattanaik, and Connor W Coley. Learning 3d representations of molecular chirality with invariance to bond rotations. *arXiv preprint arXiv:2110.04383*, 2021.

[2] Brandon Anderson, Truong-Son Hy, and Risi Kondor. Cormorant: Covariant molecular neural networks. In *Proceedings of the 33st International Conference on Neural Information Processing Systems*, pages 14537–14546, 2019.

[3] Kenneth Atz, Francesca Grisoni, and Gisbert Schneider. Geometric deep learning on molecular representations. *Nature Machine Intelligence*, pages 1–10, 2021.

[4] Simon Axelrod and Rafael Gomez-Bombarelli. Molecular machine learning with conformer ensembles. *arXiv preprint arXiv:2012.08452*, 2020.

[5] Simon Axelrod and Rafael Gomez-Bombarelli. Geom, energy-annotated molecular conformations for property prediction and molecular generation. *Scientific Data*, 9(1):1–14, 2022.

[6] Minkyung Baek, Frank DiMaio, Ivan Anishchenko, Justas Dauparas, Sergey Ovchinnikov, Gyu Rie Lee, Jue Wang, Qian Cong, Lisa N Kinch, R Dustin Schaeffer, et al. Accurate prediction of protein structures and interactions using a 3-track network. *bioRxiv*, 2021.

[7] 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.

[8] Simon Batzner, Tess E Smidt, Lixin Sun, Jonathan P Mailoa, Mordechai Kornbluth, Nicola Molinari, and Boris Kozinsky. Se (3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *arXiv preprint arXiv:2101.03164*, 2021.

[9] Karsten M Borgwardt, Cheng Soon Ong, Stefan Schönauer, SVN Vishwanathan, Alex J Smola, and Hans-Peter Kriegel. Protein function prediction via graph kernels. *Bioinformatics*, 21 (suppl_1):i47–i56, 2005.

[10] Michael M Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021.

[11] Lowik Chanussot, Abhishek Das, Siddharth Goyal, Thibaut Lavril, Muhammed Shuaibi, Morgane Riviere, Kevin Tran, Javier Heras-Domingo, Caleb Ho, Weihua Hu, et al. Correction to "the open catalyst 2020 (oc20) dataset and community challenges". *ACS Catalysis*, 11(21): 13062–13065, 2021.

[12] Lowik Chanussot, Abhishek Das, Siddharth Goyal, Thibaut Lavril, Muhammed Shuaibi, Morgane Riviere, Kevin Tran, Javier Heras-Domingo, Caleb Ho, Weihua Hu, et al. Open catalyst 2020 (oc20) dataset and community challenges. *ACS Catalysis*, 11(10):6059–6072, 2021.

[13] Lowik Chanussot, Abhishek Das, Siddharth Goyal, Thibaut Lavril, Muhammed Shuaibi, Morgane Riviere, Kevin Tran, et al. Open Catalyst Challenge. https://opencatalystproject.org/challenge.html, 2021.

[14] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. *Advances in neural information processing systems*, 28:2224–2232, 2015.

[15] Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[16] Alex Fout, Jonathon Byrd, Basir Shariat, and Asa Ben-Hur. Protein interface prediction using graph convolutional networks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6533–6542, 2017.

[17] Fabian Fuchs, Daniel Worrall, Volker Fischer, and Max Welling. Se (3)-transformers: 3d roto-translation equivariant attention networks. *Advances in Neural Information Processing Systems*, 33, 2020.

[18] Octavian-Eugen Ganea, Lagnajit Pattanaik, Connor W Coley, Regina Barzilay, Klavs F Jensen, William H Green, and Tommi S Jaakkola. Geomol: Torsional geometric generation of molecular 3d conformer ensembles. *Advances in Neural Information Processing Systems*, 2021.

[19] Hongyang Gao and Shuiwang Ji. Graph u-net. In *International conference on machine learning*, 2019.

[20] Hongyang Gao, Zhengyang Wang, and Shuiwang Ji. Large-scale learnable graph convolutional networks. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1416–1424. ACM, 2018.

[21] Hongyang Gao, Yi Liu, and Shuiwang Ji. Topology-aware graph pooling networks. *arXiv preprint arXiv:2010.09834*, 2020.

[22] 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-Volume 70*, pages 1263–1272. JMLR. org, 2017.

[23] Jonathan Godwin, Michael Schaarschmidt, Alexander Gaunt, Alvaro Sanchez-Gonzalez, Yulia Rubanova, Petar Veličković, James Kirkpatrick, and Peter Battaglia. Very deep graph neural networks via noise regularisation. *arXiv preprint arXiv:2106.07971*, 2021.

[24] Weihua Hu, Matthias Fey, Hongyu Ren, Maho Nakata, Yuxiao Dong, and Jure Leskovec. Ogb-lsc: A large-scale challenge for machine learning on graphs. *arXiv preprint arXiv:2103.09430*, 2021.

[25] Weihua Hu, Muhammed Shuaibi, Abhishek Das, Siddharth Goyal, Anuroop Sriram, Jure Leskovec, Devi Parikh, and C Lawrence Zitnick. Forcenet: A graph neural network for large-scale quantum calculations. *arXiv preprint arXiv:2103.01436*, 2021.

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

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

[28] Johannes Klicpera, Shankari Giri, Johannes T. Margraf, and Stephan Günnemann. Fast and uncertainty-aware directional message passing for non-equilibrium molecules. In *NeurIPS-W*, 2020.

[29] Johannes Klicpera, Janek Groß, and Stephan Günnemann. Directional message passing for molecular graphs. In *International Conference on Learning Representations (ICLR)*, 2020.

[30] Johannes Klicpera, Florian Becker, and Stephan Günnemann. Gemnet: Universal directional graph neural networks for molecules. *arXiv preprint arXiv:2106.08903*, 2021.

[31] Dmitrii Kochkov, Tobias Pfaff, Alvaro Sanchez-Gonzalez, Peter Battaglia, and Bryan K Clark. Learning ground states of quantum hamiltonians with graph networks. *arXiv preprint arXiv:2110.06390*, 2021.

[32] Meng Liu, Youzhi Luo, Limei Wang, Yaochen Xie, Hao Yuan, Shurui Gui, Haiyang Yu, Zhao Xu, Jingtun Zhang, Yi Liu, et al. Dig: a turnkey library for diving into graph deep learning research. *Journal of Machine Learning Research*, 22(240):1–9, 2021.

[33] Yi Liu, Hao Yuan, Lei Cai, and Shuiwang Ji. Deep learning of high-order interactions for protein interface prediction. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 679–687, 2020.

[34] Yi Liu, Limei Wang, Meng Liu, Yuchao Lin, Xuan Zhang, Bora Oztekin, and Shuiwang Ji. Spherical message passing for 3d molecular graphs. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=givsRXsOt9r.

[35] Chengqiang Lu, Qi Liu, Chao Wang, Zhenya Huang, Peize Lin, and Lixin He. Molecular property prediction: A multilevel quantum interactions modeling perspective. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 1052–1060, 2019.

[36] Haggai Maron, Heli Ben-Hamu, Hadar Serviansky, and Yaron Lipman. Provably powerful graph networks. *Advances in Neural Information Processing Systems*, 32:2153–2164, 2019.

[37] Alex Morehead, Chen Chen, and Jianlin Cheng. Geometric transformers for protein interface contact prediction. *arXiv preprint arXiv:2110.02423*, 2021.

[38] Christopher Morris, Martin Ritzert, Matthias Fey, William L Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4602–4609, 2019.

[39] Maho Nakata and Tomomi Shimazaki. Pubchemqc project: a large-scale first-principles electronic structure database for data-driven chemistry. *Journal of chemical information and modeling*, 57(6):1300–1308, 2017.

[40] 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.

[41] Kristof Schütt, Pieter-Jan Kindermans, Huziel Enoc Sauceda Felix, Stefan Chmiela, Alexandre Tkatchenko, and Klaus-Robert Müller. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. In *Advances in neural information processing systems*, pages 991–1001, 2017.

[42] Kristof T Schütt, Oliver T Unke, and Michael Gastegger. Equivariant message passing for the prediction of tensorial properties and molecular spectra. *arXiv preprint arXiv:2102.03150*, 2021.

[43] Muhammed Shuaibi, Adeesh Kolluru, Abhishek Das, Aditya Grover, Anuroop Sriram, Zachary Ulissi, and C Lawrence Zitnick. Rotation invariant graph neural networks using spin convolutions. *arXiv preprint arXiv:2106.09575*, 2021.

[44] Gregor N. C. Simm, Robert Pinsler, and José Miguel Hernández-Lobato. Reinforcement learning for molecular design guided by quantum mechanics. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 8959–8969. PMLR, 2020.

[45] Gregor N. C. Simm, Robert Pinsler, Gábor Csányi, and José Miguel Hernández-Lobato. Symmetry-aware actor-critic for 3d molecular design. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=jEYKjPE1xYN.

[46] Jonathan M Stokes, Kevin Yang, Kyle Swanson, Wengong Jin, Andres Cubillos-Ruiz, Nina M Donghia, Craig R MacNair, Shawn French, Lindsey A Carfrae, Zohar Bloom-Ackermann, et al. A deep learning approach to antibiotic discovery. *Cell*, 180(4):688–702, 2020.

[47] Nathaniel Thomas, Tess Smidt, Steven Kearnes, Lusann Yang, Li Li, Kai Kohlhoff, and Patrick Riley. Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds. *arXiv preprint arXiv:1802.08219*, 2018.

[48] Oliver T Unke and Markus Meuwly. Physnet: A neural network for predicting energies, forces, dipole moments, and partial charges. *Journal of chemical theory and computation*, 15(6): 3678–3693, 2019.

[49] Clement Vignac, Andreas Loukas, and Pascal Frossard. Building powerful and equivariant graph neural networks with message-passing. *arXiv preprint arXiv:2006.15107*, 2020.

[50] 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.

[51] 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.

[52] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.

[53] Zhao Xu, Youzhi Luo, Xuan Zhang, Xinyi Xu, Yaochen Xie, Meng Liu, Kaleb Dickerson, Cheng Deng, Maho Nakata, and Shuiwang Ji. Molecule3d: A benchmark for predicting 3d geometries from molecular graphs. *arXiv preprint arXiv:2110.01717*, 2021.

[54] Kevin Yang, Kyle Swanson, Wengong Jin, Connor Coley, Philipp Eiden, Hua Gao, Angel Guzman-Perez, Timothy Hopper, Brian Kelley, Miriam Mathea, et al. Analyzing learned molecular representations for property prediction. *Journal of chemical information and modeling*, 59 (8):3370–3388, 2019.

[55] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=OeWooOxFwDa.

[56] Muhan Zhang, Zhicheng Cui, Marion Neumann, and Yixin Chen. An end-to-end deep learning architecture for graph classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [No]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix A.2.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] We will include these details upon acceptance.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Appendix A.4.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A] Error bars are not necessary for baseline methods and our method.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] All of these are reported in Appendix A.4.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]

(c) Did you include any new assets either in the supplemental material or as a URL? [N/A]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Appendix

A.1 Algorithm for the Proposed Message Passing

Algorithm 1 The Proposed Message Passing

for $i = 1, ..., n$ **do**
  for $j = 1, ..., k$ **do**
    Compute $d_{ij} = ||\mathbf{p}_i - \mathbf{p}_j||_2$
  end for
  Get reference nodes via

$$f_i = \text{argmin}_{k \in \mathcal{N}_i}(d_{ik}), s_i = \text{argmin}_{k \in \mathcal{N}_i \setminus \{f_i\}}(d_{ik})$$

end for
for $i = 1, ..., n$ **do**
  for $j = 1, ..., k$ **do**
    Get reference nodes via

$$f_{i\setminus j} = \begin{cases} f_i, & \text{if } f_i \neq j \\ s_i, & \text{otherwise} \end{cases}, f_{j\setminus i} = \begin{cases} f_j, & \text{if } f_j \neq i \\ s_j, & \text{otherwise} \end{cases}$$

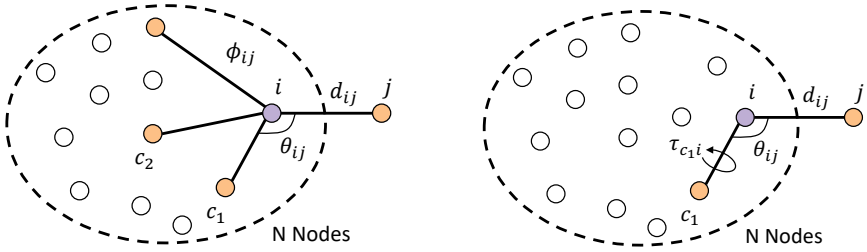
    Compute angles via

$$\theta_{ij} = \text{angle}(f_i, i, j),$$
$$\phi_{ij} = \text{angle}(\text{plane}_{f_i,i,s_i}, \text{plane}_{f_i,i,j}),$$
$$\tau_{ij} = \text{angle}(\text{plane}_{f_{i\setminus j},i,j}, \text{plane}_{i,j,f_{j\setminus i}}),$$

  end for
  Update node features via Eq. 1
end for

As rigorously shown in Algorithm 1 , there are two nested loops in the message passing. For each nested loop, the complexity for the outer loop is $O(n)$, and the complexity for the inner loop is $O(k)$. Particularly, within the inner loop, the operations are picking proper reference nodes, thus the complexity is simply $O(1)$. Overall, the total complexity of our message passing is $O(nk)$. Importantly, the efficiency of our method is also demonstrated in Sec. 6 that ComeNet is 6-10 times faster than SphereNet. Actually, the training time of ComENet is similar to SchNet whose complexity is also $O(nk)$.

A.2 Proofs of Lemma 1



Case (1): i has more than one neighboring nodes $c_{1,2,...}$

Case (2): i only has one neighboring node c_1

Figure 5: Two cases for proving Lemma 1 when generalizing the size of a 3D graph from k to $k+1$.

Proof. We employ contradiction to prove it. Basically, there exist two cases when adding this new node j, as illustrated in Fig. 5 and described as following. *Case (1): i has more than one neighboring nodes $c_{1,2,...}$; Case (2): i only has one neighboring node c_1.*

Generally, in case (1), we need to prove $\mathbf{p}_{ij}$ is uniquely defined as $(d_{ij}, \theta_{ij}, \phi_{ij})$; in case (2), we need to prove $\mathbf{p}_{ij}$ is uniquely defined as $(d_{ij}, \theta_{ij}, \tau_{c_1 i})$. We assume there exists another location for j such

that there is a different relative location vector $\widetilde{\mathbf{p}}_{ij}$, and this assumption leads to contradiction. The proofs of both cases are provided below. □

Proof of case (1): In case (1), we need to prove $\mathbf{p}_{ij}$ is uniquely defined as $(d_{ij}, \theta_{ij}, \phi_{ij})$. Based on the notations defined in Sec. 2.4, we have

$$\begin{aligned} \left\langle \mathbf{p}_{ij}, \mathbf{p}_{ij} \right\rangle &= d_{ij}^2, \\ \left\langle \mathbf{p}_{if_i}, \mathbf{p}_{ij} \right\rangle &= d_{if_i} d_{ij} cos\theta_{ij}, \\ \left\langle T(\mathbf{p}_{is_i}), T(\mathbf{p}_{ij}) \right\rangle &= \|T(\mathbf{p}_{is_i})\| \|T(\mathbf{p}_{ij})\| cos\phi_{ij}, \\ \left\langle T(\mathbf{p}_{is_i}) \times T(\mathbf{p}_{ij}), \mathbf{p}_{if_i} \right\rangle &= d_{if_i} \|T(\mathbf{p}_{is_i})\| \|T(\mathbf{p}_{ij})\| sin\phi_{ij}, \end{aligned} \tag{2}$$

where T denotes an operator of projection to the plane perpendicular to $\mathbf{p}_{if_i}$. Apparently, all the quantities on the right side are known based on Sec. 2.4 and Fig. 1**??**. Assume the solution set contains at least two different solutions $\mathbf{p}_{ij}$ and $\widetilde{\mathbf{p}}_{ij}$. Eq. 2 can simply imply that

$$\begin{aligned} \left\langle \mathbf{p}_{if_i}, \mathbf{p}_{ij} - \widetilde{\mathbf{p}}_{ij} \right\rangle &= 0, \\ \left\langle T(\mathbf{p}_{is_i}), \mathbf{p}_{ij} - \widetilde{\mathbf{p}}_{ij} \right\rangle &= 0, \\ \left\langle T(\mathbf{p}_{is_i}) \times (\mathbf{p}_{ij} - \widetilde{\mathbf{p}}_{ij}), \mathbf{p}_{if_i} \right\rangle &= 0. \end{aligned} \tag{3}$$

Apparently, both $T(\mathbf{p}_{is_i})$ and $\mathbf{p}_{ij} - \widetilde{\mathbf{p}}_{ij}$ are in the plane perpendicular to $\mathbf{p}_{if_i}$. Hence, $T(\mathbf{p}_{is_i}) \times (\mathbf{p}_{ij} - \widetilde{\mathbf{p}}_{ij}) = \alpha \mathbf{p}_{if_i}$ holds for some $\alpha \neq 0$. However, from Eq. 3, we have

$$\left\langle T(\mathbf{p}_{is_i}) \times (\mathbf{p}_{ij} - \widetilde{\mathbf{p}}_{ij}), \mathbf{p}_{if_i} \right\rangle = \alpha \left\langle \mathbf{p}_{if_i}, \mathbf{p}_{if_i} \right\rangle = 0. \tag{4}$$

Since $\alpha \neq 0$ and $\mathbf{p}_{if_i} \neq \mathbf{0}$, Eq. 4 causes a contradiction. Thus, the solution set contains a unique solution.

Proof of case (2): In case (2), we need to prove $\mathbf{p}_{ij}$ is uniquely defined by $(d_{ij}, \theta_{ij}, \tau_{c_1 i})$. Based on the notations defined in Sec. 2.4 and Sec. 2.2, we have

$$\begin{aligned} \left\langle \mathbf{p}_{ij}, \mathbf{p}_{ij} \right\rangle &= d_{ij}^2, \\ \left\langle \mathbf{p}_{if_i}, \mathbf{p}_{ij} \right\rangle &= d_{if_i} d_{ij} cos\theta_{ij}, \\ \left\langle T(\mathbf{p}_{f_i f_{f_i \setminus i}}), T(\mathbf{p}_{ij}) \right\rangle &= \|T(\mathbf{p}_{f_i f_{f_i \setminus i}})\| \|T(\mathbf{p}_{ij})\| cos\tau_{c_1 i}, \\ \left\langle T(\mathbf{p}_{f_i f_{f_i \setminus i}}) \times T(\mathbf{p}_{ij}), \mathbf{p}_{if_i} \right\rangle &= d_{if_i} \|T(\mathbf{p}_{f_i f_{f_i \setminus i}})\| \|T(\mathbf{p}_{ij})\| sin\tau_{c_1 i}, \end{aligned} \tag{5}$$

where T still denotes an operator of projection to the plane perpendicular to $\mathbf{p}_{if_i}$. Similarly, all the quantities on the right side are known based on Sec. 2.4 and Sec. 2.2. Assume the solution set contains at least two different solutions $\mathbf{p}_{ij}$ and $\widetilde{\mathbf{p}}_{ij}$. Eq. 5 can simply imply that

$$\begin{aligned} \left\langle \mathbf{p}_{if_i}, \mathbf{p}_{ij} - \widetilde{\mathbf{p}}_{ij} \right\rangle &= 0, \\ \left\langle T(\mathbf{p}_{f_i f_{f_i \setminus i}}), \mathbf{p}_{ij} - \widetilde{\mathbf{p}}_{ij} \right\rangle &= 0, \\ \left\langle T(\mathbf{p}_{f_i f_{f_i \setminus i}}) \times (\mathbf{p}_{ij} - \widetilde{\mathbf{p}}_{ij}), \mathbf{p}_{if_i} \right\rangle &= 0. \end{aligned} \tag{6}$$

Both $T(\mathbf{p}_{f_i f_{f_i \setminus i}})$ and $\mathbf{p}_{ij} - \widetilde{\mathbf{p}}_{ij}$ are in the plane perpendicular to $\mathbf{p}_{if_i}$. Hence, $T(\mathbf{p}_{f_i f_{f_i \setminus i}}) \times (\mathbf{p}_{ij} - \widetilde{\mathbf{p}}_{ij}) = \alpha \mathbf{p}_{if_i}$ holds for some $\alpha \neq 0$. However, from Eq. 6, we have

$$\left\langle T(\mathbf{p}_{f_i f_{f_i \setminus i}}) \times (\mathbf{p}_{ij} - \widetilde{\mathbf{p}}_{ij}), \mathbf{p}_{if_i} \right\rangle = \alpha \left\langle \mathbf{p}_{if_i}, \mathbf{p}_{if_i} \right\rangle = 0. \tag{7}$$

Since $\alpha \neq 0$ and $\mathbf{p}_{if_i} \neq \mathbf{0}$, Eq. 7 causes a contradiction. Thus, the solution set contains a unique solution.

A.3 Data Description

OC20. The Open Catalyst 2020 (OC20) dataset [12] is a newly released dataset to model and discover catalysts. Specifically, the goal is efficient DFT approximation of structure relaxation, which is a fundamental calculation in catalysis to determine a structure's activity and selectivity. All the structures in the dataset contain a surface and an adsorbate, and the surface is defined by a unit cell that is periodic in all directions. There are three tasks including Structure to Energy and Forces (S2EF), Initial Structure to Relaxed Structure (IS2RS), and Initial Structure to Relaxed Energy (IS2RE).

In this work, we focus on Initial Structure to Relaxed Energy (IS2RE) task, which is the most common task in catalysis as the relaxed energies are often correlated with catalyst activity and selectivity. The dataset for IS2RE is originally split into training, validation, and test sets. The training set contains 460,328 structures and the validation set has four splits including in-domain (ID), out-of-domain adsorbate (OOD Ads), out-of-domain catalyst (OOD Cat), and out-of-domain adsorbate and catalyst (OOD Both), with 24,733, 24,961, 24,738, 24,971 structures respectively.

Molecule3D. The Molecule3D dataset [53] is a newly proposed large-scale dataset, including around 4 million molecules with precise ground-state geometries derived from DFT. The dataset is collected from PubChemQC [39] and designed for predicting 3D geometries from molecular graphs [53] while we aim to learn representations and predict properties for molecules based on their geometries. The dataset contains 3,899,647 molecules and is split into training, validation, and test sets via random and scaffold split with ratio 6:2:2. Random split ensures the training, validation, and test data are sampled from the same distribution while scaffold split leads to a distribution shift between training and test data. The evaluation metric is the MAE between the predictions and the ground truth.

A.4 Experimental Setup

ComENet. The values/search space of model and training hyperparameters for ComENet on OC20, Molecule3D, and QM9 are provided in Table 6 and Table 7. For Molecule3D and QM9, the optimal hyperparameters are chosen by the performance on the validation sets. For OC20, since the final comparison results are on validation set, we firstly use 10% of the training data to choose optimal hyperparameters, then train our model on whole training data. Specifically, we use a larger cutoff value for OC20 to generate graphs and larger hidden dimensions for OC20 and Molecule3D. All models are trained on NVIDIA GeForce RTX 2080 Ti 11GB GPU for Molecule3D and QM9. For the OC20 dataset, we use NVIDIA RTX A6000 48GB GPU. Note that one experiment is only conducted on one GPU.

Baselines for Molecule3D. As Molecule3D is a newly proposed dataset, we run experiments and provide results for baseline methods including GIN-Virtual [24], SchNet [41], DimeNet++[28] and SphereNet [34]. All the models are trained on one GPU (Nvidia GeForce RTX 2080 Ti 11GB). The model hyperparameters and training hyperparameters are listed in Table 8 and Table 9. The optimal hyperparameters are chosen by the performance on validation set. Note that for DimeNet++[28] and SphereNet [34], the maximum batch size is 32 due to the GPU memory limitation.

Table 6: Model hyperparameters for ComENet.

Model Hyperparameters	Values/ search space		
	OC20	Molecule3D	QM9
Number of layers	4, 5, 6	4, 5, 6, 8	4, 5, 6
Cutoff	6.0, 8.0	5.0	5.0
Hidden dim	128, 256, 512	128, 256, 512	128, 256
Hidden dim in Self-Atom layer	128, 256, 512	128, 256, 512	128, 256
Number of layers in Self-Atom layer	2, 3, 4	2, 3, 4	2, 3, 4
Number of layers of the MLP in Interaction layer	2, 3, 4	2, 3, 4	2, 3, 4
Distance embedding dim	6, 12	6, 12	6, 12
Angle embedding dim	3, 6	3, 6	3, 6

Table 7: Training hyperparameters for ComENet.

Training hyperparameters	Values/ search space		
	OC20	Molecule3D	QM9
Epochs	20	300	1000
Batch size	64, 128	128, 256	32, 64, 128
Learning rate 1e-3, 5e-4, 2e-4	1e-3, 5e-4, 2e-4	1e-3, 5e-4, 2e-4	1e-3, 5e-4, 2e-4
Learning rate decay factor	0.4, 0.5, 0.6	0.4, 0.5, 0.6	0.4, 0.5, 0.6
Learning rate decay epochs	Milestone [4,7,10,12]	20, 30, 50	100, 200
Warmup epochs	2	–	–
Warmup factor	0.2	–	–

Table 8: Model hyperparameters for baseline methods on Molecule3D.

Model Hyperparameters	Values/ search space			
	GIN-Virtual	SchNet	DimeNet++	SphereNet
Number of layers	4, 6, 8	4, 6, 8	3, 4, 5, 6	3, 4, 5, 6
Cutoff	-	10.0	6.0	6.0
Hidden dim	600	256	128	128

Table 9: Training hyperparameters for baseline methods on Molecule3D.

Training hyperparameters	Values/ search space			
	GIN-Virtual	SchNet	DimeNet++	SphereNet
Epochs	300	300	300	300
Batch size	64, 128, 256	64, 128, 256	16, 32	16, 32
Learning rate	1e-3, 5e-4, 2e-4	1e-3, 5e-4, 2e-4	1e-3, 5e-4, 2e-4	1e-3, 5e-4, 2e-4
Learning rate decay factor	0.4, 0.5, 0.6	0.4, 0.5, 0.6	0.4, 0.5, 0.6	0.4, 0.5, 0.6
Learning rate decay epochs	20, 30, 50	20, 30, 50	20, 30, 50	20, 30, 50