arXiv:2306.12059v1 [cs.LG] 21 Jun 2023

EquiformerV2: Improved Equivariant Transformer for Scaling to Higher-Degree Representations

Yi-Lun Liao[1] **Brandon Wood**[2] **Abhishek Das**[2]* **Tess Smidt**[1]*

[1]Massachusetts Institute of Technology [2]FAIR, Meta AI

{ylliao,tsmidt}@mit.edu {bmwood,abhshkdz}@meta.com

https://github.com/atomicarchitects/equiformer_v2

Abstract

Equivariant Transformers such as Equiformer have demonstrated the efficacy of applying Transformers to the domain of 3D atomistic systems. However, they are still limited to small degrees of equivariant representations due to their computational complexity. In this paper, we investigate whether these architectures can scale well to higher degrees. Starting from Equiformer, we first replace $SO(3)$ convolutions with eSCN convolutions to efficiently incorporate higher-degree tensors. Then, to better leverage the power of higher degrees, we propose three architectural improvements – attention re-normalization, separable S^2 activation and separable layer normalization. Putting this all together, we propose EquiformerV2, which outperforms previous state-of-the-art methods on the large-scale OC20 dataset by up to 12% on forces, 4% on energies, offers better speed-accuracy trade-offs, and $2\times$ reduction in DFT calculations needed for computing adsorption energies.

1 Introduction

In recent years, machine learning (ML) models have shown promising results in accelerating and scaling high-accuracy but compute-intensive quantum mechanical calculations by effectively accounting for key features of atomic systems, such as the discrete nature of atoms, and Euclidean and permutation symmetries [1–10]. By bringing down computational costs from hours or days to fractions of seconds, these methods enable new insights in many applications such as molecular simulations, material design and drug discovery. A promising class of ML models that have enabled this progress is equivariant graph neural networks (GNNs) [5, 11–18].

Equivariant GNNs treat 3D atomistic systems as graphs, and incorporate inductive biases such that their internal representations and predictions are equivariant to 3D translations, rotations and optionally inversions. Specifically, they build up equivariant features of each node as vector spaces of irreducible representations (or irreps) and have interactions or message passing between nodes based on equivariant operations such as tensor products. Recent works on equivariant Transformers, specifically Equiformer [17], have shown the efficacy of applying Transformers [19, 20], which have previously enjoyed widespread success in computer vision [21–23], language [24, 25], and graphs [26–29], to this domain of 3D atomistic systems.

A bottleneck in scaling Equiformer as well as other equivariant GNNs is the computational complexity of tensor products, especially when we increase the maximum degree of irreps L_{max}. This limits these models to use small values of L_{max} (e.g., $L_{max} \leqslant 3$), which consequently limits their performance. Higher degrees can better capture angular resolution and directional information, which is critical to accurate prediction of atomic energies and forces. To this end, eSCN [18] recently proposes efficient convolutions to reduce $SO(3)$ tensor products to $SO(2)$ linear operations, bringing down the computational cost from $\mathcal{O}(L_{max}^6)$ to $\mathcal{O}(L_{max}^3)$ and enabling scaling to larger values of L_{max} (e.g., L_{max} up to 8). However, except using efficient convolutions for higher L_{max}, eSCN still follows

*denotes equal contribution.

Preprint. Under review.

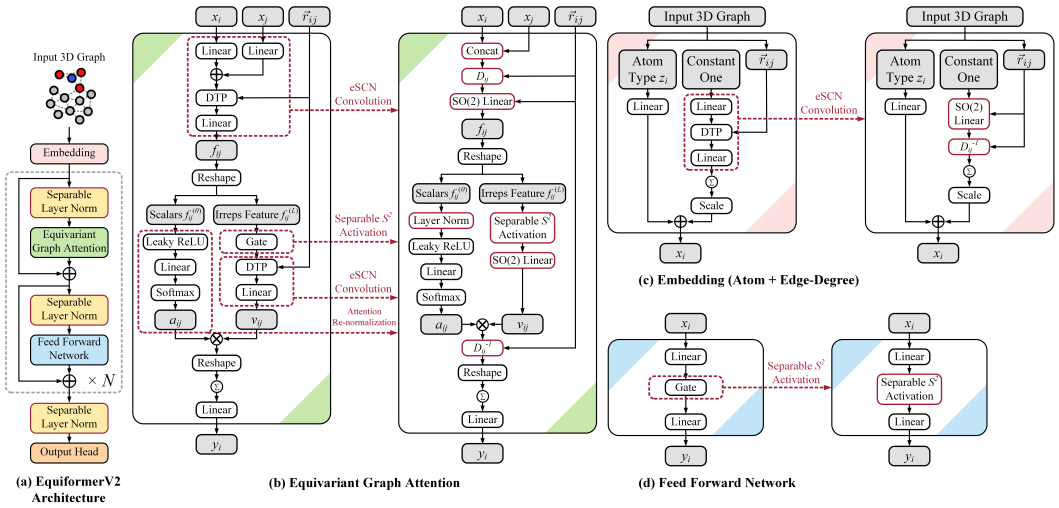


Figure 1: Overview of EquiformerV2. We highlight the differences from Equiformer [17] in red. For (b), (c), and (d), the left figure is the original module in Equiformer, and the right figure is the revised module in EquiformerV2. Input 3D graphs are embedded with atom and edge-degree embeddings and processed with Transformer blocks, which consist of equivariant graph attention and feed forward networks. "$\otimes$" denotes multiplication, "$\oplus$" denotes addition, and $\sum$ within a circle denotes summation over all neighbors. "DTP" denotes depth-wise tensor products used in Equiformer. Gray cells indicate intermediate irreps features.

SEGNN [15]-like message passing network design, and Equiformer has been shown to improve upon SEGNN. Additionally, this ability to use higher L_{max} challenges whether the previous design of equivariant Transformers can scale well to higher-degree representations.

In this paper, we are interested in adapting eSCN convolutions for higher-degree representations to equivariant Transformers. We start with Equiformer [17] and replace $SO(3)$ convolutions with eSCN convolutions. We find that naively incorporating eSCN convolutions does not result in better performance than the original eSCN model. Therefore, to better leverage the power of higher degrees, we propose three architectural improvements – attention re-normalization, separable S^2 activation and separable layer normalization. Putting this all together, we propose EquiformerV2, which is developed on the large and diverse OC20 dataset [30]. Experiments on OC20 show that EquiformerV2 outperforms previous state-of-the-art methods with improvements of up to 12% on forces and 4% on energies, and offers better speed-accuracy trade-offs compared to existing invariant and equivariant GNNs. Additionally, when used in the AdsorbML algorithm [10] for performing adsorption energy calculations, EquiformerV2 achieves the highest success rate and $2\times$ reduction in DFT calculations to achieve comparable adsorption energy accuracies as previous methods.

2 Related Works

***SE(3)/E(3)*-Equivariant GNNs.** Equivariant neural networks [5, 7, 11–18, 31–38] use equivariant irreps features built from vector spaces of irreducible representations (irreps) to achieve equivariance to 3D rotation [11–13]. They operate on irreps features with equivariant operations like tensor products. Previous works differ in equivariant operations used in their networks and how they combine those operations. TFN [11] and NequIP [5] use equivariant graph convolution with linear messages built from tensor products, with the latter utilizing extra equivariant gate activation [12]. SEGNN [15] introduces non-linearity to messages passing [1, 39] with equivariant gate activation, and the non-linear messages improve upon linear messages. SE(3)-Transformer [14] adopts equivariant dot product attention [19] with linear messages. Equiformer [17] improves upon previously mentioned equivariant GNNs by combining MLP attention and non-linear messages. Equiformer additionally introduces equivariant layer normalization and regularizations like dropout [40] and stochastic depth [41]. However, the networks mentioned above rely on compute-intensive $SO(3)$ tensor products to mix the information of vectors of different degrees during message passing, and therefore they are limited to small values for maximum degrees L_{max} of equivariant representations. SCN [42] proposes rotating irreps features based on relative position vectors and identifies a subset of spherical harmonics coefficients, on which they can apply unconstrained functions. They further propose

relaxing the requirement for strict equivariance and apply typical functions to rotated features during message passing, which trades strict equivariance for computational efficiency and enables using higher values of L_{max}. eSCN [18] further improves upon SCN by replacing typical functions with $SO(2)$ linear layers for rotated features and imposing strict equivariance during message passing. However, except using more efficient operations for higher L_{max}, SCN and eSCN mainly adopt the same network design as SEGNN, which is less performant than Equiformer. In this work, we propose EquiformerV2, which includes all the benefits of the above networks by incorporating eSCN convolutions into Equiformer and adopts three additional architectural improvements.

Invariant GNNs. Prior works [4, 8, 43–51] extract invariant information from 3D atomistic graphs and operate on the resulting graphs augmented with invariant features. Their differences lie in leveraging different geometric features such as distances, bond angles (3 atom features) or dihedral angles (4 atom features). SchNet [43] models interaction between atoms with only relative distances. DimeNet series [4, 46] use triplet representations of atoms to incorporate bond angles. SphereNet [48] and GemNet [50, 51] further include dihedral angles by considering quadruplet representations. However, the memory complexity of triplet and quadruplet representations of atoms do not scale well with the number of atoms, and this requires additional modifications like interaction hierarchy used by GemNet-OC [51] for large datasets like OC20 [30]. Additionally, for the task of predicting DFT calculations of energies and forces on the large-scale OC20 dataset, invariant GNNs have been surpassed by equivariant GNNs recently.

3 Background

3.1 $SE(3)/E(3)$-Equivariant Neural Networks

We discuss the relevant background of $SE(3)/E(3)$-equivariant neural networks here. Please refer to Sec. A in appendix for more details of equivariance and group theory.

Including equivariance in neural networks can serve as a strong prior knowledge, which can therefore improve data efficiency and generalization. Equivariant neural networks use equivariant irreps features built from vector spaces of irreducible representations (irreps) to achieve equivariance to 3D rotation. Specifically, the vector spaces are $(2L+1)$-dimensional, where degree L is a non-negative integer. L can be intuitively interpreted as the angular frequency of the vectors, i.e., how fast the vectors rotate with respect to a rotation of the coordinate system. Higher L is critical to tasks sensitive to angular information like predicting forces [5, 18, 42]. Vectors of degree L are referred to as type-L vectors, and they are rotated with Wigner-D matrices $D^{(L)}$ when rotating coordinate systems. Euclidean vectors $\vec{r}$ in $\mathbb{R}^3$ can be projected into type-L vectors by using spherical harmonics $Y^{(L)}(\frac{\vec{r}}{||\vec{r}||})$. We use order m to index the elements of type-L vectors, where $-L \leqslant m \leqslant L$. We concatenate multiple type-L vectors to form an equivariant irreps feature f. Concretely, f has C_L type-L vectors, where $0 \leqslant L \leqslant L_{max}$ and C_L is the number of channels for type-L vectors. In this work, we mainly consider $C_L = C$, and the size of f is $(L_{max}+1)^2 \times C$. We index f by channel i, degree L, and order m and denote as $f^{(L)}_{m,i}$.

Equivariant GNNs update irreps features by passing messages of transformed irreps features between nodes. To interact different type-L vectors during message passing, we use tensor products, which generalize multiplication to equivariant irreps features. Denoted as $\otimes^{L_3}_{L_1,L_2}$, the tensor product uses Clebsch-Gordan coefficients to combine type-L_1 vector $f^{(L_1)}$ and type-L_2 vector $g^{(L_2)}$ and produces type-L_3 vector $h^{(L_3)}$:

$$h^{(L_3)}_{m_3} = (f^{(L_1)} \otimes^{L_3}_{L_1,L_2} g^{(L_2)})_{m_3} = \sum_{m_1=-L_1}^{L_1} \sum_{m_2=-L_2}^{L_2} C^{(L_3,m_3)}_{(L_1,m_1)(L_2,m_2)} f^{(L_1)}_{m_1} g^{(L_2)}_{m_2} \tag{1}$$

where m_1 denotes order and refers to the m_1-th element of $f^{(L_1)}$. Clebsch-Gordan coefficients $C^{(L_3,m_3)}_{(L_1,m_1)(L_2,m_2)}$ are non-zero only when $|L_1 - L_2| \leqslant L_3 \leqslant |L_1 + L_2|$ and thus restrict output vectors to be of certain degrees. We typically discard vectors with $L > L_{max}$, where L_{max} is a hyper-parameter, to prevent vectors of increasingly higher dimensions. In many works, message passing is implemented as equivariant convolutions, which perform tensor products between input irreps features $x^{(L_1)}$ and spherical harmonics of relative position vectors $Y^{(L_2)}(\frac{\vec{r}}{||\vec{r}||})$.

3.2 Equiformer

Equiformer [17] is an $SE(3)/E(3)$-equivariant GNN that combines the inductive biases of equivariance with the strength of Transformers [19, 22]. First, Equiformer replaces scalar node features with equivariant irreps features to incorporate equivariance. Next, it performs equivariant operations on these irreps features and equivariant graph attention for message passing. These operations include tensor products and equivariant linear operations, equivariant layer normalization [52] and gate activation [12, 34]. For stronger expressivity in the attention compared to typical Transformers, Equiformer uses non-linear functions for both attention weights and message passing. Additionally, Equiformer incorporates regularization techniques common in Transformers applied to other domains, e.g., dropout [40] to attention weights [53] and stochastic depth [54] to the outputs of equivariant graph attention and feed forward networks. Please refer to the Equiformer paper [17] for more details.

3.3 eSCN Convolution

While tensor products are necessary to interact vectors of different degrees, they are compute-intensive. To reduce the complexity, eSCN convolutions [18] are proposed to use $SO(2)$ linear operations for efficient tensor products. We provide an outline and intuition for their method here, please refer to Sec. A.3 and their work [18] for mathematical details.

A traditional $SO(3)$ convolution interacts input irreps features $x_{m_i}^{(L_i)}$ and spherical harmonic projections of relative positions $Y_{m_f}^{(L_f)}(\vec{r_{ij}})$ with an $SO(3)$ tensor product with Clebsch-Gordan coefficients $C_{(L_i,m_i),(L_f,m_f)}^{(L_o,m_o)}$. The projection $Y_{m_f}^{(L_f)}(\vec{r_{ij}})$ becomes sparse if we rotate the relative position vector $\vec{r}_{ij}$ with a rotation matrix D_{ij} to align with the direction of $L = 0$ and $m = 0$, which corresponds to the z axis traditionally but the y axis in the conventions of `e3nn` [55]. Concretely, given $D_{ij}\vec{r}_{ij}$ aligned with the y axis, $Y_{m_f}^{(L_f)}(D_{ij}\vec{r}_{ij}) \neq 0$ only for $m_f = 0$. If we consider only $m_f = 0$, $C_{(L_i,m_i),(L_f,m_f)}^{(L_o,m_o)}$ can be simplified, and $C_{(L_i,m_i),(L_f,0)}^{(L_o,m_o)} \neq 0$ only when $m_i = \pm m_o$. Therefore, the original expression depending on m_i, m_f, and m_o is now reduced to only depend on m_o. This means we are no longer mixing all integer values of m_i and m_f, and outputs of order m_o are linear combinations of inputs of order $\pm m_o$. eSCN convolutions go one step further and replace the remaining non-trivial paths of the $SO(3)$ tensor product with an $SO(2)$ linear operation to allow for additional parameters of interaction between $\pm m_o$ without breaking equivariance. To summarize, eSCN convolutions achieve efficient equivariant convolutions by first rotating irreps features based on relative position vectors and then performing $SO(2)$ linear operations on the rotated features. The key idea is that the rotation sparsifies tensor products and simplifies the computation.

4 EquiformerV2

Starting from Equiformer [17], we first use eSCN convolutions to scale to higher-degree representations (Sec. 4.1). Then, we propose three architectural improvements, which yield further performance gain when using higher degrees: attention re-normalization (Sec. 4.2), separable S^2 activation (Sec. 4.3) and separable layer normalization (Sec. 4.4). Figure 1 illustrates the overall architecture of EquiformerV2 and the differences from Equiformer.

4.1 Incorporating eSCN Convolutions for Efficient Tensor Products and Higher Degrees

The computational complexity of $SO(3)$ tensor products used in traditional $SO(3)$ convolutions during equivariant message passing scale unfavorably with L_{max}. Because of this, it is impractical for Equiformer to use beyond $L_{max} = 1$ for large-scale datasets like OC20 [30] and beyond $L_{max} = 3$ for small-scale datasets like MD17 [56–58]. Since higher L_{max} can better capture angular information and are correlated with model expressivity [5], low values of L_{max} can lead to limited performance on certain tasks such as predicting forces. Therefore, we replace original tensor products with eSCN convolutions [18] for efficient tensor products, enabling Equiformer to scale up L_{max} to 6 or 8 on the large-scale OC20 dataset.

Equiformer uses equivariant graph attention for message passing. The attention consists of depth-wise tensor products, which mix information across different degrees, and linear layers, which mix information between channels of the same degree. Since eSCN convolutions mix information across both degrees and channels, we replace the $SO(3)$ convolution, which involves one depth-wise tensor product layer and one linear layer, with a single eSCN convolutional layer, which consists of a rotation matrix D_{ij} and an $SO(2)$ linear layer as shown in Figure 1b.

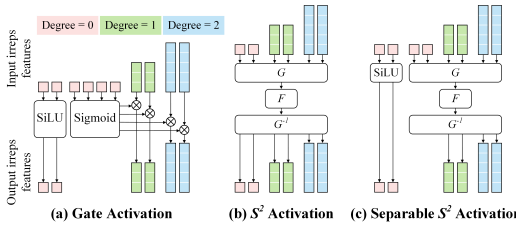


Figure 2: Illustration of different activation functions. G denotes conversion from vectors to point samples on a sphere, F can typically be a SiLU activation or MLPs, and G^{-1} is the inverse of G.

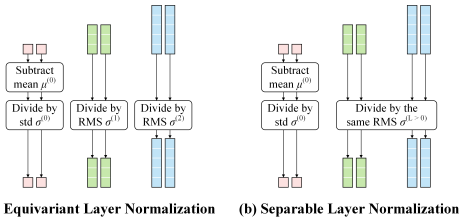


Figure 3: Illustration of how statistics are calculated in different normalizations. "std" denotes standard deviation, and "RMS" denotes root mean square.

4.2 Attention Re-normalization

Equivariant graph attention in Equiformer uses tensor products to project node embeddings x_i and x_j, which contain vectors of different degrees, to scalar features $f_{ij}^{(0)}$ and applies non-linear functions to $f_{ij}^{(0)}$ for attention weights a_{ij}. The node embeddings x_i and x_j are obtained by applying equivariant layer normalization [17] to previous outputs. We note that vectors of different degrees in x_i and x_j are normalized independently, and therefore when they are projected to the same degree, the resulting $f_{ij}^{(0)}$ can be less well-normalized. To address the issue, we propose attention re-normalization and introduce one additional layer normalization (LN) [52] before non-linear functions. Specifically, given $f_{ij}^{(0)}$, we first apply LN and then use one leaky ReLU layer and one linear layer to calculate $z_{ij} = a^{\top}\text{LeakyReLU}(\text{LN}(f_{ij}^{(0)}))$ and $a_{ij} = \text{softmax}_j(z_{ij}) = \frac{\exp(z_{ij})}{\sum_{k \in \mathcal{N}(i)} \exp(z_{ik})}$, where a is a learnable vector of the same dimension as $f_{ij}^{(0)}$.

4.3 Separable S^2 Activation

The gate activation [12] used by Equiformer applies sigmoid activation to scalar features to obtain non-linear weights and then multiply irreps features of degree > 0 with non-linear weights to add non-linearity to equivariant features. The activation, however, only accounts for the interaction from vectors of degree 0 to those of degree > 0 and could be sub-optimal when we scale up L_{max}.

To better mix the information across degrees, SCN [42] and eSCN [18] propose to use S^2 activation [59]. The activation first converts vectors of all degrees to point samples on a sphere for each channel, applies unconstrained functions F to those samples, and finally convert them back to vectors. Specifically, given an input irreps feature $x \in \mathbb{R}^{(L_{max}+1)^2 \times C}$, the output is $y = G^{-1}(F(G(x)))$, where G denotes the conversion from vectors to point samples on a sphere, F can be typical SiLU activation [60, 61] or typical MLPs, and G^{-1} is the inverse of G.

While S^2 activation can better mix vectors of different degrees, we find that directly replacing the gate activation with S^2 activation results in training instability (row 3 in Table 1a). To address the issue, we propose separable S^2 activation, which separates activation for vectors of degree 0 and those of degree > 0. Similar to gate activation, we have more channels for vectors of degree 0. As shown in Figure 2c, we apply a SiLU activation to the first part of vectors of degree 0, and the second part of vectors of degree 0 are used for S^2 activation along with vectors of higher degrees. After S^2 activation, we concatenate the first part of vectors of degree 0 with vectors of degrees > 0 as the final output and ignore the second part of vectors of degree 0. Additionally, we also use separable S^2 activation in point-wise feed forward networks (FFNs). Figure 2 illustrates the differences between gate activation, S^2 activation and separable S^2 activation.

4.4 Separable Layer Normalization

As mentioned in Sec. 4.2, equivariant layer normalization used by Equiformer normalizes vectors of different degrees independently, and when those vectors are projected to the same degree, the projected vectors can be less well-normalized. Therefore, instead of performing normalization to each degree independently, we propose separable layer normalization (SLN), which separates normalization for vectors of degree 0 and those of degrees > 0. Mathematically, let $x \in \mathbb{R}^{(L_{max}+1)^2 \times C}$ denote an input irreps feature of maximum degree L_{max} and C channels, and $x_{m,i}^{(L)}$ denote the L-th degree, m-th order

and i-th channel of x. SLN calculates the output y as follows. For $L = 0$, $y^{(0)} = \gamma^{(0)} \circ \left(\frac{x^{(0)} - \mu^{(0)}}{\sigma^{(0)}} \right) + \beta^{(0)}$, where $\mu^{(0)} = \frac{1}{C} \sum_{i=1}^{C} x_{0,i}^{(0)}$ and $\sigma^{(0)} = \sqrt{\frac{1}{C} \sum_{i=1}^{C} (x_{0,i}^{(0)} - \mu^{(0)})^2}$. For $L > 0$, $y^{(L)} = \gamma^{(L)} \circ \left(\frac{x^{(L)}}{\sigma^{(L>0)}} \right)$, where $\sigma^{(L>0)} = \sqrt{\frac{1}{L_{max}} \sum_{L=1}^{L_{max}} \left(\sigma^{(L)} \right)^2}$ and $\sigma^{(L)} = \sqrt{\frac{1}{C} \sum_{i=1}^{C} \frac{1}{2L+1} \sum_{m=-L}^{L} \left(x_{m,i}^{(L)} \right)^2}$. $\gamma^{(0)}, \gamma^{(L)}, \beta^{(0)} \in \mathbb{R}^C$ are learnable parameters, $\mu^{(0)}$ and $\sigma^{(0)}$ are mean and standard deviation of vectors of degree 0, $\sigma^{(L)}$ and $\sigma^{(L>0)}$ are root mean square values (RMS), and $\circ$ denotes element-wise product. The computation of $y^{(0)}$corresponds to typical layer normalization. We note that the difference between equivariant layer normalization and SLN lies only in $y^{(L)}$ with $L > 0$ and that equivariant layer normalization divides $x^{(L)}$ by $\sigma^{(L)}$, which is calculated independently for each degree L, instead of $\sigma^{(L>0)}$, which considers all degrees $L > 0$. Figure 3 compares how $\mu^{(0)}$, $\sigma^{(0)}$, $\sigma^{(L)}$ and $\sigma^{(L>0)}$ are calculated in equivariant layer normalization and SLN.

4.5 Overall Architecture

Here, we discuss all the other modules in EquiformerV2 and focus on the differences from Equiformer.

Equivariant Graph Attention. Figure 1b illustrates equivariant graph attention after the above modifications. As described in Sec. 4.1, given node embeddings x_i and x_j, we first concatenate them along the channel dimension and then rotate them with rotation matrices D_{ij} based on their relative positions or edge directions $\vec{r}_{ij}$. The rotation enables reducing $SO(3)$ tensor products to $SO(2)$ linear operations, and we replace depth-wise tensor products and linear layers between x_i, x_j and f_{ij} with a single $SO(2)$ linear layer. To consider the information of relative distances $||\vec{r}_{ij}||$, in the same way as eSCN [18], we transform $||\vec{r}_{ij}||$ with a radial function to obtain edge distance embeddings and then multiply the edge distance embeddings with concatenated node embeddings before the first $SO(2)$ linear layer. We split the outputs f_{ij} of the first $SO(2)$ linear layer into two parts. The first part is scalar features $f_{ij}^{(0)}$, which only contains vectors of degree 0, and the second part is irreps features $f_{ij}^{(L)}$ and includes vectors of all degrees up to L_{max}. As mentioned in Sec. 4.2, we first apply an additional LN to $f_{ij}^{(0)}$ and then follow the design of Equiformer by applying one leaky ReLU layer, one linear layer and a final softmax layer to obtain attention weights a_{ij}. As for value v_{ij}, we replace the gate activation with separable S^2 activation with F being a single SiLU activation and then apply the second $SO(2)$ linear layer. While in Equiformer, the message m_{ij} sent from node j to node i is $m_{ij} = a_{ij} \times v_{ij}$, here we need to rotate $a_{ij} \times v_{ij}$ back to original coordinate frames and the message m_{ij} becomes $D_{ij}^{-1}(a_{ij} \times v_{ij})$. Finally, we can perform h parallel equivariant graph attention functions given f_{ij}. The h different outputs are concatenated and projected with a linear layer to become the final output y_i. Parallelizing attention functions and concatenating can be implemented with "Reshape".

Feed Forward Network. As illustrated in Figure 1d, we replace the gate activation with separable S^2 activation. The function F consists of a two-layer MLP, with each linear layer followed by SiLU, and a final linear layer.

Embedding. This module consists of atom embedding and edge-degree embedding. The former is the same as that in Equiformer. For the latter, as depicted in the right branch in Figure 1c, we replace original linear layers and depth-wise tensor products with a single $SO(2)$ linear layer followed by a rotation matrix D_{ij}^{-1}. Similar to equivariant graph attention, we consider the information of relative distances by multiplying the outputs of the $SO(2)$ linear layer with edge distance embeddings.

Radial Basis and Radial Function. We represent relative distances $||\vec{r}_{ij}||$ with a finite radial basis like Gaussian radial basis functions [43] to capture their subtle changes. We transform radial basis with a learnable radial function to generate edge distance embeddings. The function consists of a two-layer MLP, with each linear layer followed by LN and SiLU, and a final linear layer.

Output Head. To predict scalar quantities like energy, we use one feed forward network to transform irreps features on each node into a scalar and then perform sum aggregation over all nodes. As for predicting forces acting on each node, we use a block of equivariant graph attention and treat the output of degree 1 as our predictions.

5 OC20 Experiments

Our experiments focus on the large and diverse OC20 dataset [30] (Creative Commons Attribution 4.0 License), which consists of 1.2M DFT relaxations for training and evaluation, computed with the

	Attention re-normalization	Activation	Normalization	Epochs	forces	energy
1	✗	Gate	LN	12	21.85	286
2	✓	Gate	LN	12	21.86	279
3	✓	S^2	LN	12	didn't converge	
4	✓	Sep. S^2	LN	12	20.77	285
5	✓	Sep. S^2	SLN	12	20.46	285
6	✓	Sep. S^2	LN	20	20.02	276
7	✓	Sep. S^2	SLN	20	19.72	278
8	eSCN baseline			12	21.3	294

(a) Architectural improvements. Attention re-normalization improves energies, and separable S^2 activation ("Sep. S^2") and separable layer normalization ("SLN") improve forces.

		eSCN		EquiformerV2	
L_{max}	Epochs	forces	energy	forces	energy
6	12	21.3	294	20.46	285
6	20	20.6	290	19.78	280
6	30	20.1	285	19.42	278
8	12	21.3	296	20.46	279
8	20	-	-	19.95	273

(b) Training epochs. Training for more epochs consistently leads to better results.

	eSCN		EquiformerV2	
L_{max}	forces	energy	forces	energy
4	22.2	291	21.37	284
6	21.3	294	20.46	285
8	21.3	296	20.46	279

(c) Degrees L_{max}. Higher degrees are consistently helpful.

	eSCN		EquiformerV2	
M_{max}	forces	energy	forces	energy
2	21.3	294	20.46	285
3	21.2	295	20.24	284
4	21.2	298	20.24	282
6	-	-	20.26	278

(d) Orders M_{max}. Higher orders mainly improve energy predictions.

	eSCN		EquiformerV2	
Layers	forces	energy	forces	energy
8	22.4	306	21.18	293
12	21.3	294	20.46	285
16	20.5	283	20.11	282

(e) Number of Transformer blocks. Adding more blocks can help both force and energy predictions.

Table 1: Ablation results with EquiformerV2. We report mean absolute errors for forces in meV/Å and energy in meV, and lower is better. All models are trained on the 2M subset of OC20 [30], and errors are averaged over the four validation splits of OC20. The base model setting is marked in gray.

revised Perdew-Burke-Ernzerhof (RPBE) functional [62]. Each structure in OC20 has an adsorbate molecule placed on a catalyst surface, and the core task is Structure-to-Energy-Forces (S2EF), which is to predict the energy of the structure and per-atom forces. Models trained for the S2EF task are evaluated on energy and force mean absolute error (MAE). These models can in turn be used for performing structure relaxations by using the model's force predictions to iteratively update the atomic positions until a relaxed structure corresponding to a local energy minimum is found. These relaxed structure and energy predictions are evaluated on the Initial Structure to Relaxed Structure (IS2RS) and Initial Structure to Relaxed Energy (IS2RE) tasks. The "All" split of OC20 contains 134M training structures spanning 56 elements, and the "MD" split consists of 38M structures. We first conduct ablation studies on EquiformerV2 trained on the smaller S2EF-2M subset (Sec. 5.1). Then, we report the results of training on S2EF-All and S2EF-All+MD splits (Sec. 5.2). Additionally, we investigate the performance of EquiformerV2 when used in the AdsorbML algorithm [10] (Sec. 5.3). Please refer to Sec. B and C for details of models and training.

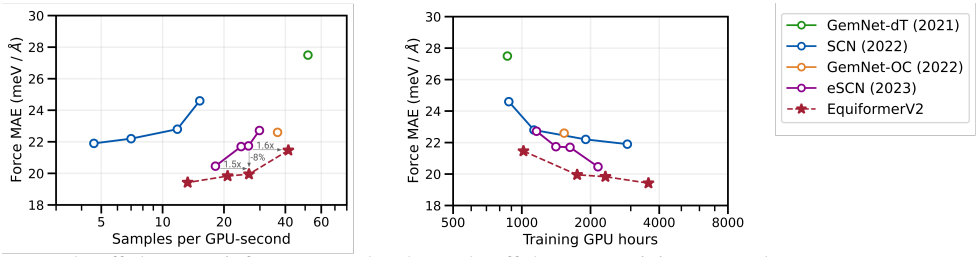
5.1 Ablation Studies

Architectural Improvements. In Table 1a, we ablate the three proposed architectural changes – attention re-normalization, separable S^2 activation and separable layer normalization. First, with attention re-normalization (row 1 and 2), energy errors improve by 2.4%, while force errors are about the same. Next, we replace the gate activation with S^2 activation used in SCN [42] and eSCN [18], but that does not converge (row 3). Instead, using the proposed separable S^2 activation (row 4), where we have separate paths for invariant and equivariant features, converges to 5% better forces albeit hurting energies. Similarly, replacing equivariant layer normalization with separable layer normalization (row 5) further improves forces by 1.5%. Finally, these modifications enable training for longer without overfitting (row 7), further improving forces by 3.6% and recovering energies to similar accuracies as row 2. Overall, our modifications improve forces by 10% and energies by 3%. Note that simply incorporating eSCN convolutions into Equiformer (row 1) and using higher degrees does not result in improving over the original eSCN baseline (row 8), and that the proposed architectural changes are necessary.

Scaling of Parameters. In Tables 1c, 1d, 1e, we systematically vary the maximum degree L_{max}, the maximum order M_{max}, and the number of Transformer blocks and compare with equivalent eSCN variants. There are several key takeaways. First, across all experiments, EquiformerV2 performs better than its eSCN counterparts. Second, while one might intuitively expect higher resolution features and larger models to perform better, this is only true for EquiformerV2, not eSCN. For example, increasing L_{max} from 6 to 8 or M_{max} from 3 to 4 degrades the performance of eSCN on energy predictions but helps that of EquiformerV2. In Table 1b, we show that longer training regimes

Training set	Model	Throughput Samples / GPU sec. ↑	S2EF validation Energy MAE (meV) ↓	S2EF validation Force MAE (meV/Å) ↓	S2EF test Energy MAE (meV) ↓	S2EF test Force MAE (meV/Å) ↓	IS2RS test AFbT (%) ↑	IS2RS test ADwT (%) ↑	IS2RE test Energy MAE (meV) ↓
OC20 All	CGCNN [44]	-	590	74.0	608	73.3	-	-	-
	SchNet [43]	-	549	56.8	540	54.7	-	14.4	764
	ForceNet-large [63]	15.3	-	33.5	-	32.0	12.7	49.6	-
	DimeNet++-L-F+E [4]	4.6	515	32.8	480	31.3	21.7	51.7	559
	SpinConv [49]	6.0	371	41.2	336	29.7	16.7	53.6	437
	GemNet-dT [50]	25.8	315	27.2	292	24.2	27.6	58.7	400
	GemNet-XL [8]	1.5	-	-	270	20.5	30.8	62.7	371
	GemNet-OC [51]	18.3	244	21.7	233	20.7	35.3	60.3	355
	SCN L=8 K=20 [42]	-	-	-	244	17.7	40.3	67.1	330
	eSCN L=6 K=20 [18]	2.9	-	-	242	17.1	48.5	65.7	341
	EquiformerV2 ($\lambda_E = 2$, 153M)	1.8	**236**	**15.7**	**229**	**14.8**	**53.0**	**69.0**	**316**
OC20 All+MD	GemNet-OC-L-E [51]	7.5	239	22.1	230	21.0	-	-	-
	GemNet-OC-L-F [51]	3.2	252	20.0	241	19.0	40.6	60.4	-
	GemNet-OC-L-F+E [51]	-	-	-	-	-	-	-	348
	SCN L=6 K=16 (4-tap 2-band) [42]	-	-	-	228	17.8	43.3	64.9	328
	SCN L=8 K=20 [42]	-	-	-	237	17.2	43.6	67.5	321
	eSCN L=6 K=20 [18]	2.9	243	17.1	236	16.2	50.3	66.7	327
	EquiformerV2 ($\lambda_E = 4$, 31M)	7.1	232	16.3	228	15.5	47.6	68.3	315
	EquiformerV2 ($\lambda_E = 2$, 153M)	1.8	**230**	**14.6**	227	**13.8**	**55.4**	**69.8**	**311**
	EquiformerV2 ($\lambda_E = 4$, 153M)	1.8	**227**	15.0	**219**	14.2	54.4	69.4	**309**

Table 2: OC20 results on S2EF validation and test splits, and IS2RS and IS2RE test splits when trained on OC20 S2EF-All or S2EF-All+MD splits. Throughput is reported as the number of structures processed per GPU-second during training and measured on V100 GPUs. λ_E is the coefficient of the energy loss.



(a) Trade-offs between inference speed and validation force MAE.

(b) Trade-offs between training cost and validation force MAE.

Figure 4: EquiformerV2 offers better accuracy trade-offs both in terms of inference speed as well as training cost compared to prior works. All models in this analysis are trained on the S2EF-2M split.

are crucial. Increasing the training epochs from 12 to 30 with $L_{max} = 6$ improves force and energy predictions by 5% and 2.5%, respectively.

Comparison of Speed-Accuracy Trade-offs. To be practically useful for atomistic simulations and material screening, models should offer flexibility in speed-accuracy tradeoffs. We compare these trade-offs for EquiformerV2 with prior works in Figure 4a. Here, the speed is reported as the number of structures processed per GPU-second during inference and measured on V100 GPUs. For the same force MAE as eSCN, EquiformerV2 is up to $1.6\times$ faster, and for the same speed as eSCN, EquiformerV2 is up to 8% more accurate. Compared to GemNet-OC [51] at the same speed, EquiformerV2 is 5% more accurate. Comparing to the closest available EquifomerV2 point, GemNet-dT [50] is $1.25\times$ faster but 30% worse. Overall, EquiformerV2 clearly offers a better trade-off between speed and accuracy. In similar spirit, we also study the training cost of EquiformerV2 compared to prior works in Figure 4b, and find that it is substantially more training efficient.

5.2 Main Results

Table 2 reports results on the test splits for all the three tasks of OC20, averaged across the in-distribution, out-of-distribution adsorbates, out-of-distribution catalysts, and out-of-distribution both subsplits. Models are trained on either OC20 S2EF-All or S2EF-All+MD splits. All test results are computed via the EvalAI evaluation server[2]. We train EquiformerV2 of two sizes, one with 153M parameters and the other with 31M parameters. As shown in Table 4, the smaller one is obtained by reducing the maximum degree L_{max} from 6 to 4, the maximum order M_{max} from 3 to 2 and the number of Transformer blocks from 20 to 8. EquiformerV2 outperforms all previous models across all tasks. When trained on the S2EF-All+MD split, EquiformerV2 ($\lambda_E = 4$, 153M) improves previous state-of-the-art S2EF energy MAE by 4%, S2EF force MAE by 12%, IS2RS Average Forces below Threshold (AFbT) by absolute 4% and IS2RE energy MAE by 4%. In particular, the improvements in

[2] eval.ai/web/challenges/challenge-page/712

	$k=1$		$k=2$		$k=3$		$k=4$		$k=5$	
Model	Success	Speedup	Success	Speedup	Success	Speedup	Success	Speedup	Success	Speedup
SchNet [43]	2.77%	4266.13	3.91%	2155.36	4.32%	1458.77	4.73%	1104.88	5.04%	892.79
DimeNet++ [4]	5.34%	4271.23	7.61%	2149.78	8.84%	1435.21	10.07%	1081.96	10.79%	865.20
PaiNN [34]	27.44%	4089.77	33.61%	2077.65	36.69%	1395.55	38.64%	1048.63	39.57%	840.44
GemNet-OC [51]	68.76%	4185.18	77.29%	2087.11	80.78%	1392.51	81.50%	1046.85	82.94%	840.25
GemNet-OC-MD [51]	68.76%	4182.04	78.21%	2092.27	81.81%	1404.11	83.25%	1053.36	84.38%	841.64
GemNet-OC-MD-Large [51]	73.18%	4078.76	79.65%	2065.15	83.25%	1381.39	85.41%	1041.50	86.02%	834.46
SCN-MD-Large [42]	77.80%	3974.21	84.28%	1989.32	86.33%	1331.43	87.36%	1004.40	87.77%	807.00
EquiformerV2 ($\lambda_E = 4$, 31M)	84.17%	3983.41	87.15%	1992.64	87.87%	1331.35	88.69%	1000.62	89.31%	802.95
EquiformerV2 ($\lambda_E = 4$, 153M)	**85.41%**	4001.71	**88.90%**	2012.47	**90.54%**	1352.08	**91.06%**	1016.31	**91.57%**	815.87

Table 3: AdsorbML results with EquiformerV2 ($\lambda_E = 4$, 31M) and ($\lambda_E = 4$, 153M) trained on S2EF-All+MD from Table 2.

force predictions are significant. Going from SCN [42] to eSCN [18], S2EF test force MAE improves from 17.2 meV/Å to 16.2 meV/Å , largely due to replacing approximate equivariance in SCN with strict equivariance in eSCN during message passing and scaling to higher degrees. Similarly, by scaling up the degrees of representations in Equiformer [17], EquiformerV2 ($\lambda_E = 4$, 153M) further improves force MAE to 14.2 meV/Å, doubling the gain of going from SCN to eSCN. These better force predictions also translate to higher IS2RS test AFbT, which is computed via DFT single-point calculations to check if the DFT forces on the predicted relaxed structures are close to zero. A 4% improvement on AFbT is a strong step towards replacing DFT with ML. Additionally, the smaller EquiformerV2 ($\lambda_E = 4$, 31M) also improves upon previously best results for all metrics except IS2RS AFbT and achieves comparable training efficiency to the fastest GemNet-OC-L-E [51] model.

5.3 AdsorbML Results

Lan et al. [10] recently proposes the AdsorbML algorithm, wherein they show that recent state-of-the-art GNNs (e.g., SCN [42]) can achieve more than 1000× speedup over DFT relaxations at computing adsorption energies within a 0.1eV margin of DFT results with an 87% success rate. This is done by using OC20-trained models to perform structure relaxations for an average 90 configurations of an adsorbate placed on a catalyst surface, followed by DFT single-point calculations for the top-k structures with lowest predicted relaxed energies, as a proxy for calculating the global energy minimum or adsorption energy. We refer the reader to Sec. C.3 and the AdsorbML paper [10] for more details. We benchmark AdsorbML with EquiformerV2, and Table 3 shows that EquiformerV2 ($\lambda_E = 4$, 153M) improves over SCN by a significant margin, with 8% and 5% absolute improvements at $k = 1$ and $k = 2$, respectively. Moreover, EquiformerV2 ($\lambda_E = 4$, 153M) at $k = 2$ is more accurate at adsorption energy calculations than all the other models even at $k = 5$, thus requiring at least 2× fewer DFT calculations. Since the speedup is with respect to using DFT for structure relaxations and that ML models are much faster than DFT, the speedup is dominated by the final DFT single-point calculations and ML models with the same value of k have roughly the same speedup. To better understand the speed-accuracy trade-offs of different models, we compare the AdsorbML success rate averaged over k from 1 to 5 and average GPU-seconds of running one structure relaxation in Figure 5. EquiformerV2 ($\lambda_E = 4$, 31M) improves upon previous methods while being 3.7× to 9.8× faster than GemNet-OC-MD-Large and SCN, respectively. Besides, the larger EquiformerV2 is 2.1% more accurate but 3.7× slower than the smaller one.

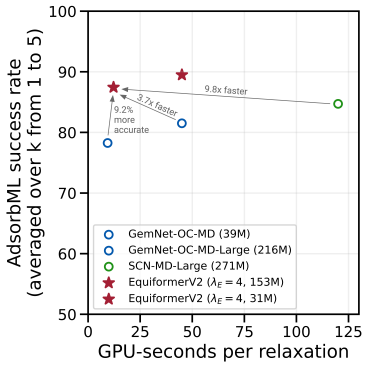


Figure 5: Speed-accuracy trade-offs of different models when used in the AdsorbML algorithm.

6 Conclusion

In this work, we investigate how equivariant Transformers can be scaled up to higher degrees of equivariant representations. We start by replacing $SO(3)$ convolutions in Equiformer with eSCN convolutions, and then we propose three architectural improvements to better leverage the power of higher degrees – attention re-normalization, separable S^2 activation and separable layer normalization. With these modifications, we propose EquiformerV2, which outperforms state-of-the-art methods on the S2EF, IS2RS, and IS2RE tasks on the OC20 dataset, improves speed-accuracy trade-offs, and achieves the best success rate when used in AdsorbML.

Broader Impacts. EquiformerV2 achieves more accurate approximation of quantum mechanical calculations and demonstrates one further step toward replacing DFT force fields with machine learned ones. By demonstrating its promising results, we hope to encourage the community to make further progress in applications like material design and drug discovery than to use it for adversarial purposes. Additionally, the method only facilitates identification of molecules or materials of specific properties, and there are substantial hurdles from their large-scale deployment. Finally, we note that the proposed method is general and can be applied to different problems like protein structure prediction [64] as long as inputs can be modeled as 3D graphs.

Limitations. Although EquiformerV2 improves upon state-of-the-art methods on the large and diverse OC20 dataset, we acknolwdge that the performance gains brought by scaling to higher degrees and the proposed architectural improvements can depend on tasks and datasets. For example, the increased expressivity may lead to overfitting on smaller datasets like QM9 [65, 66] and MD17 [56–58]. However, the issue can be mitigated by pre-training on large datasets like OC20 [30] and PCQM4Mv2 [67] optionally via denoising [68] and then finetuning on smaller datasets.

Acknowledgement

We thank Larry Zitnick and Saro Passaro for helpful discussions. We also thank Muhammed Shuaibi for helping with the DFT evaluations for AdsorbML [10]. We acknowledge the MIT SuperCloud and Lincoln Laboratory Supercomputing Center [69] for providing high performance computing and consultation resources that have contributed to the research results reported within this paper.

Yi-Lun Liao and Tess Smidt were supported by DOE ICDI grant DE-SC0022215.

References

[1] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *International Conference on Machine Learning (ICML)*, 2017. 1, 2

[2] L. Zhang, J. Han, H. Wang, R. Car, and W. E, "Deep potential molecular dynamics: A scalable model with the accuracy of quantum mechanics," *Phys. Rev. Lett.*, vol. 120, p. 143001, Apr 2018. 1

[3] W. Jia, H. Wang, M. Chen, D. Lu, L. Lin, R. Car, W. E, and L. Zhang, "Pushing the limit of molecular dynamics with ab initio accuracy to 100 million atoms with machine learning," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '20, IEEE Press, 2020. 1

[4] J. Gasteiger, S. Giri, J. T. Margraf, and S. Günnemann, "Fast and uncertainty-aware directional message passing for non-equilibrium molecules," in *Machine Learning for Molecules Workshop, NeurIPS*, 2020. 1, 3, 8, 9

[5] S. Batzner, A. Musaelian, L. Sun, M. Geiger, J. P. Mailoa, M. Kornbluth, N. Molinari, T. E. Smidt, and B. Kozinsky, "E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials," *Nature Communications*, vol. 13, May 2022. 1, 2, 3, 4, 16

[6] D. Lu, H. Wang, M. Chen, L. Lin, R. Car, W. E, W. Jia, and L. Zhang, "86 pflops deep potential molecular dynamics simulation of 100 million atoms with ab initio accuracy," *Computer Physics Communications*, vol. 259, p. 107624, 2021. 1

[7] O. T. Unke, M. Bogojeski, M. Gastegger, M. Geiger, T. Smidt, and K. R. Muller, "SE(3)-equivariant prediction of molecular wavefunctions and electronic densities," in *Advances in Neural Information Processing Systems (NeurIPS)* (A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, eds.), 2021. 1, 2

[8] A. Sriram, A. Das, B. M. Wood, and C. L. Zitnick, "Towards training billion parameter graph neural networks for atomic simulations," in *International Conference on Learning Representations (ICLR)*, 2022. 1, 3, 8

[9] J. A. Rackers, L. Tecot, M. Geiger, and T. E. Smidt, "A recipe for cracking the quantum scaling limit with machine learned electron densities," *Machine Learning: Science and Technology*, vol. 4, p. 015027, feb 2023. 1, 16

[10] J. Lan, A. Palizhati, M. Shuaibi, B. M. Wood, B. Wander, A. Das, M. Uyttendaele, C. L. Zitnick, and Z. W. Ulissi, "AdsorbML: Accelerating adsorption energy calculations with machine learning," *arXiv preprint arXiv:2211.16486*, 2022. 1, 2, 7, 9, 10, 20, 21

[11] N. Thomas, T. E. 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. 1, 2

[12] M. Weiler, M. Geiger, M. Welling, W. Boomsma, and T. Cohen, "3D Steerable CNNs: Learning Rotationally Equivariant Features in Volumetric Data," in *Advances in Neural Information Processing Systems 32*, pp. 10402–10413, 2018. 1, 2, 4, 5

[13] R. Kondor, Z. Lin, and S. Trivedi, "Clebsch–gordan nets: a fully fourier space spherical convolutional neural network," in *Advances in Neural Information Processing Systems 32*, pp. 10117–10126, 2018. 1, 2

[14] F. Fuchs, D. E. Worrall, V. Fischer, and M. Welling, "Se(3)-transformers: 3d roto-translation equivariant attention networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 1, 2

[15] J. Brandstetter, R. Hesselink, E. van der Pol, E. J. Bekkers, and M. Welling, "Geometric and physical quantities improve e(3) equivariant message passing," in *International Conference on Learning Representations (ICLR)*, 2022. 1, 2

[16] A. Musaelian, S. Batzner, A. Johansson, L. Sun, C. J. Owen, M. Kornbluth, and B. Kozinsky, "Learning local equivariant representations for large-scale atomistic dynamics," *arxiv preprint arxiv:2204.05249*, 2022. 1, 2

[17] Y.-L. Liao and T. Smidt, "Equiformer: Equivariant graph attention transformer for 3d atomistic graphs," in *International Conference on Learning Representations (ICLR)*, 2023. 1, 2, 4, 5, 9, 15, 18

[18] S. Passaro and C. L. Zitnick, "Reducing SO(3) Convolutions to SO(2) for Efficient Equivariant GNNs," in *International Conference on Machine Learning (ICML)*, 2023. 1, 2, 3, 4, 5, 6, 7, 8, 9, 15, 16, 17, 18, 20, 21

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 1, 2, 4

[20] S. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, and M. Shah, "Transformers in vision: A survey," *arXiv preprint arxiv:2101.01169*, 2021. 1

[21] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European Conference on Computer Vision (ECCV)*, 2020. 1

[22] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations (ICLR)*, 2021. 1, 4

[23] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," *arXiv preprint arXiv:2012.12877*, 2020. 1

[24] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *arxiv preprint arxiv:1810.04805*, 2019. 1

[25] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 1

[26] V. P. Dwivedi and X. Bresson, "A generalization of transformer networks to graphs," *arxiv preprint arxiv:2012.09699*, 2020. 1

[27] D. Kreuzer, D. Beaini, W. L. Hamilton, V. Létourneau, and P. Tossou, "Rethinking graph transformers with spectral attention," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 1

[28] C. Ying, T. Cai, S. Luo, S. Zheng, G. Ke, D. He, Y. Shen, and T.-Y. Liu, "Do transformers really perform badly for graph representation?," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 1

[29] Y. Shi, S. Zheng, G. Ke, Y. Shen, J. You, J. He, S. Luo, C. Liu, D. He, and T.-Y. Liu, "Benchmarking graphormer on large-scale molecular modeling datasets," *arxiv preprint arxiv:2203.04810*, 2022. 1

[30] L. Chanussot*, A. Das*, S. Goyal*, T. Lavril*, M. Shuaibi*, M. Riviere, K. Tran, J. Heras-Domingo, C. Ho, W. Hu, A. Palizhati, A. Sriram, B. Wood, J. Yoon, D. Parikh, C. L. Zitnick, and Z. Ulissi, "Open catalyst 2020 (oc20) dataset and community challenges," *ACS Catalysis*, 2021. 2, 3, 4, 6, 7, 10, 19

[31] B. K. Miller, M. Geiger, T. E. Smidt, and F. Noé, "Relevance of rotationally equivariant convolutions for predicting molecular properties," *arxiv preprint arxiv:2008.08461*, 2020. 2

[32] R. J. L. Townshend, B. Townshend, S. Eismann, and R. O. Dror, "Geometric prediction: Moving beyond scalars," *arXiv preprint arXiv:2006.14163*, 2020. 2

[33] B. Jing, S. Eismann, P. Suriana, R. J. L. Townshend, and R. Dror, "Learning from protein structure with geometric vector perceptrons," in *International Conference on Learning Representations (ICLR)*, 2021. 2

[34] K. T. Schütt, O. T. Unke, and M. Gastegger, "Equivariant message passing for the prediction of tensorial properties and molecular spectra," in *International Conference on Machine Learning (ICML)*, 2021. 2, 4, 9

[35] V. G. Satorras, E. Hoogeboom, and M. Welling, "E(n) equivariant graph neural networks," in *International Conference on Machine Learning (ICML)*, 2021. 2

[36] P. Thölke and G. D. Fabritiis, "Equivariant transformers for neural network based molecular potentials," in *International Conference on Learning Representations (ICLR)*, 2022. 2

[37] T. Le, F. Noé, and D.-A. Clevert, "Equivariant graph attention networks for molecular property prediction," *arXiv preprint arXiv:2202.09891*, 2022. 2

[38] I. Batatia, D. P. Kovacs, G. N. C. Simm, C. Ortner, and G. Csanyi, "MACE: Higher order equivariant message passing neural networks for fast and accurate force fields," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 2, 18

[39] A. Sanchez-Gonzalez, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, and P. W. Battaglia, "Learning to simulate complex physics with graph networks," in *International Conference on Machine Learning (ICML)*, 2020. 2

[40] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014. 2, 4, 18

[41] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, "Deep networks with stochastic depth," in *European Conference on Computer Vision (ECCV)*, 2016. 2, 18

[42] L. Zitnick, A. Das, A. Kolluru, J. Lan, M. Shuaibi, A. Sriram, Z. Ulissi, and B. Wood, "Spherical channels for modeling atomic interactions," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 2, 3, 5, 7, 8, 9

[43] K. T. Schütt, P.-J. Kindermans, H. E. Sauceda, S. Chmiela, A. Tkatchenko, and K.-R. Müller, "Schnet: A continuous-filter convolutional neural network for modeling quantum interactions," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 3, 6, 8, 9

[44] T. Xie and J. C. Grossman, "Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties," *Physical Review Letters*, 2018. 3, 8

[45] O. T. Unke and M. Meuwly, "PhysNet: A neural network for predicting energies, forces, dipole moments, and partial charges," *Journal of Chemical Theory and Computation*, vol. 15, pp. 3678–3693, may 2019. 3

[46] J. Gasteiger, J. Groß, and S. Günnemann, "Directional message passing for molecular graphs," in *International Conference on Learning Representations (ICLR)*, 2020. 3

[47] Z. Qiao, M. Welborn, A. Anandkumar, F. R. Manby, and T. F. Miller, "OrbNet: Deep learning for quantum chemistry using symmetry-adapted atomic-orbital features," *The Journal of Chemical Physics*, 2020. 3

[48] Y. Liu, L. Wang, M. Liu, Y. Lin, X. Zhang, B. Oztekin, and S. Ji, "Spherical message passing for 3d molecular graphs," in *International Conference on Learning Representations (ICLR)*, 2022. 3

[49] M. Shuaibi, A. Kolluru, A. Das, A. Grover, A. Sriram, Z. Ulissi, and C. L. Zitnick, "Rotation invariant graph neural networks using spin convolutions," *arxiv preprint arxiv:2106.09575*, 2021. 3, 8

[50] J. Klicpera, F. Becker, and S. Günnemann, "Gemnet: Universal directional graph neural networks for molecules," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 3, 8

[51] J. Gasteiger, M. Shuaibi, A. Sriram, S. Günnemann, Z. Ulissi, C. L. Zitnick, and A. Das, "GemNet-OC: Developing Graph Neural Networks for Large and Diverse Molecular Simulation Datasets," *Transactions on Machine Learning Research (TMLR)*, 2022. 3, 8, 9

[52] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arxiv preprint arxiv:1607.06450*, 2016. 4, 5

[53] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *International Conference on Learning Representations (ICLR)*, 2018. 4

[54] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, "Deep networks with stochastic depth," in *European Conference on Computer Vision (ECCV)*, 2016. 4

[55] M. Geiger, T. Smidt, A. M., B. K. Miller, W. Boomsma, B. Dice, K. Lapchevskyi, M. Weiler, M. Tyszkiewicz, S. Batzner, D. Madisetti, M. Uhrin, J. Frellsen, N. Jung, S. Sanborn, M. Wen, J. Rackers, M. Rød, and M. Bailey, "e3nn/e3nn: 2022-04-13," Apr. 2022. 4, 15, 16

[56] S. Chmiela, A. Tkatchenko, H. E. Sauceda, I. Poltavsky, K. T. Schütt, and K.-R. Müller, "Machine learning of accurate energy-conserving molecular force fields," *Science Advances*, vol. 3, no. 5, p. e1603015, 2017. 4, 10

[57] K. T. Schütt, F. Arbabzadah, S. Chmiela, K. R. Müller, and A. Tkatchenko, "Quantum-chemical insights from deep tensor neural networks," *Nature Communications*, vol. 8, jan 2017. 4, 10

[58] S. Chmiela, H. E. Sauceda, K.-R. Müller, and A. Tkatchenko, "Towards exact molecular dynamics simulations with machine-learned force fields," *Nature Communications*, vol. 9, sep 2018. 4, 10

[59] T. S. Cohen, M. Geiger, J. Köhler, and M. Welling, "Spherical CNNs," in *International Conference on Learning Representations (ICLR)*, 2018. 5

[60] S. Elfwing, E. Uchibe, and K. Doya, "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning," *arXiv preprint arXiv:1702.03118*, 2017. 5

[61] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for activation functions," *arXiv preprint arXiv:1710.05941*, 2017. 5

[62] B. Hammer, L. B. Hansen, and J. K. Nørskov, "Improved adsorption energetics within density-functional theory using revised perdew-burke-ernzerhof functionals," *Phys. Rev. B*, 1999. 7

[63] W. Hu, M. Shuaibi, A. Das, S. Goyal, A. Sriram, J. Leskovec, D. Parikh, and C. L. Zitnick, "Forcenet: A graph neural network for large-scale quantum calculations," *arxiv preprint arxiv:2103.01436*, 2021. 8

[64] J. H. Lee, P. Yadollahpour, A. Watkins, N. C. Frey, A. Leaver-Fay, S. Ra, K. Cho, V. Gligorijevic, A. Regev, and R. Bonneau, "Equifold: Protein structure prediction with a novel coarse-grained structure representation," *bioRxiv*, 2022. 10

[65] R. Ramakrishnan, P. O. Dral, M. Rupp, and O. A. von Lilienfeld, "Quantum chemistry structures and properties of 134 kilo molecules," *Scientific Data*, vol. 1, 2014. 10

[66] L. Ruddigkeit, R. van Deursen, L. C. Blum, and J.-L. Reymond, "Enumeration of 166 billion organic small molecules in the chemical universe database gdb-17," *Journal of Chemical Information and Modeling*, vol. 52, no. 11, pp. 2864–2875, 2012. PMID: 23088335. 10

[67] M. Nakata and T. Shimazaki, "Pubchemqc project: A large-scale first-principles electronic structure database for data-driven chemistry," *Journal of chemical information and modeling*, vol. 57 6, pp. 1300–1308, 2017. 10

[68] S. Zaidi, M. Schaarschmidt, J. Martens, H. Kim, Y. W. Teh, A. Sanchez-Gonzalez, P. Battaglia, R. Pascanu, and J. Godwin, "Pre-training via denoising for molecular property prediction," in *International Conference on Learning Representations (ICLR)*, 2023. 10

[69] A. Reuther, J. Kepner, C. Byun, S. Samsi, W. Arcand, D. Bestor, B. Bergeron, V. Gadepally, M. Houle, M. Hubbell, M. Jones, A. Klein, L. Milechin, J. Mullen, A. Prout, A. Rosa, C. Yee, and P. Michaleas, "Interactive supercomputing on 40,000 cores for machine learning and data analysis," in *2018 IEEE High Performance extreme Computing Conference (HPEC)*, pp. 1–6, IEEE, 2018. 10

[70] A. Zee, *Group Theory in a Nutshell for Physicists*. USA: Princeton University Press, 2016. 15

[71] M. S. Dresselhaus, G. Dresselhaus, and A. Jorio, *Group theory*. Berlin, Germany: Springer, 2008 ed., Mar. 2007. 15

[72] N. Frey, R. Soklaski, S. Axelrod, S. Samsi, R. Gomez-Bombarelli, C. Coley, and V. Gadepally, "Neural scaling of deep chemical models," *ChemRxiv*, 2022. 16

[73] D. E. Worrall, S. J. Garbin, D. Turmukhambetov, and G. J. Brostow, "Harmonic networks: Deep translation and rotation equivariance," *arxiv preprint arxiv:1612.04642*, 2016. 17

Appendix

A Additional Background

We first provide relevant mathematical background on group theory and equivariance. We note that most of the content is adapted from Equiformer [17] and that these works [70, 71] have more in-depth and pedagogical discussions. Then, we provide mathematical details of eSCN convolutions.

A.1 Group Theory

Definition of Groups. A group is an algebraic structure that consists of a set G and a binary operator $\circ : G \times G \rightarrow G$. Typically denoted as G, groups satisfy the following four axioms:

1. Closure: $g \circ h \in G$ for all $g, h \in G$.
2. Identity: There exists an identity element $e \in G$ such that $g \circ e = e \circ g = g$ for all $g \in G$.
3. Inverse: For each $g \in G$, there exists an inverse element $g^{-1} \in G$ such that $g \circ g^{-1} = g^{-1} \circ g = e$.
4. Associativity: $g \circ h \circ i = (g \circ h) \circ i = g \circ (h \circ i)$ for all $g, h, i \in G$.

In this work, we consider 3D Euclidean symmetry, and relevant groups are:

1. The Euclidean group in three dimensions $E(3)$: 3D rotation, translation and inversion.
2. The special Euclidean group in three dimensions $SE(3)$: 3D rotation and translation.
3. The orthogonal group in three dimensions $O(3)$: 3D rotation and inversion.
4. The special orthogonal group in three dimensions $SO(3)$: 3D rotation.

Since eSCN [18] and this work only consider equivariance to 3D rotation and invariance to 3D translation but not inversion, we mainly discuss $SE(3)$-equivariance in the main text and in appendix and note that more details of $E(3)$-equivariance can be found in the work of Equiformer [17].

Group Representations. Given a vector space X, the way a group G acts on X is given by the group representation D_X. D_X is parameterized by $g \in G$, with $D_X(g) : X \rightarrow X$. Group representations D_X are invertible matrices, and group transformations, or group actions, take the form of matrix multiplications. This definition of group representations satisfies the requirements of groups, including associativity, $D(g)D(h) = D(g \circ h)$ for all $g, h \in G$. We say that the two group representations $D(g)$ and $D'(g)$ are equivalent if there exists a change-of-basis $N \times N$ matrix P such that $P^{-1}D(g)P = D'(g)$ for all $g \in G$. $D(g)$ is reducible if $D'(g)$ is block diagonal for all $g \in G$, meaning that $D'(g)$ acts on multiple independent subspaces of the vector space. Otherwise, the representation $D(g)$ is said to be irreducible. Irreducible representations, or irreps, are a class of representations that are convenient for composing different group representations. Specifically, for the case of $SO(3)$, Wigner-D matrices are irreducible representations, and we can express any group representation of $SO(3)$ as a direct sum (concatentation) of Wigner-D matrices [55, 70, 71]:

$$D(g) = P^{-1} \left(\bigoplus_i D^{(L_i)}(g) \right) P = P^{-1} \begin{pmatrix} D^{(L_0)}(g) & & & \\ & D^{(L_1)}(g) & & \\ & & \ddots & \\ & & & \end{pmatrix} P \tag{2}$$

where $D^{(L_i)}(g)$ are Wigner-D matrices of degree L_i.

A.2 Equivariance

A function f mapping between vector spaces X and Y is equivariant to a group of transformations G if for any input $x \in X$, output $y \in Y$ and group element $g \in G$, we have $f(D_X(g)x) = D_Y(g)f(x) = D_Y(g)y$, where $D_X(g)$ and $D_Y(g)$ are transformation matrices or group representations parametrized by g in X and Y. Additionally, f is invariant when $D_Y(g)$ is an identity matrix for any $g \in G$.

As neural networks comprise many composable operations, equivariant neural networks comprise many equivariant operations to maintain the equivariance of input, intermediate, and output features. Incorporating equivariance as a strong prior knowledge can help improve data efficiency and generalization of neural networks [5, 9, 72]. In this work, we achieve equivariance to 3D rotation by operating on vector spaces of $SO(3)$ irreps, incorporate invariance to 3D translation by acting on relative positions, but do not consider inversion.

A.3 eSCN Convolution

Message passing is used to update equivariant irreps features and is typically implemented as $SO(3)$ convolutions. A traditional $SO(3)$ convolution interacts input irrep features $x_{m_i}^{(L_i)}$ and spherical harmonic projections of relative positions $Y_{m_f}^{(L_f)}(\vec{r}_{ts})$ with an $SO(3)$ tensor product with Clebsch-Gordan coefficients $C_{(L_i,m_i),(L_f,m_f)}^{(L_o,m_o)}$. Since tensor products are compute-intensive, eSCN convolutions [18] are proposed to reduce the complexity of tensor products when they are used in $SO(3)$ convolutions. Rotating the input irreps features $x_{m_i}^{(L_i)}$ based on the relative position vectors $\vec{r}_{ts}$ simplifies the tensor products and enables reducing $SO(3)$ convolutions to $SO(2)$ linear operations. Below we provide the mathematical details of $SO(3)$ convolutions built from tensor products and how rotation can reduce their computational complexity.

Tensor products interact type-L_i vector $x^{(L_i)}$ and type-L_f vector $f^{(L_f)}$ to produce type-L_o vector $y^{(L_o)}$ with Clebsch-Gordan coefficients $C_{(L_i,m_i),(L_f,m_f)}^{(L_o,m_o)}$. Clebsch-Gordan coefficients $C_{(L_i,m_i),(L_f,m_f)}^{(L_o,m_o)}$ are non-zero only when $|L_i - L_o| \leqslant L_f \leqslant |L_i + L_o|$. Each non-trivial combination of $L_i \otimes L_f \rightarrow L_o$ is called a path, and each path is independently equivariant and can be assigned a learnable weight w_{L_i,L_f,L_o}.

We consider the message m_{ts} sent from source node s to target node t in an $SO(3)$ convolution. The L_o-th degree of m_{ts} can be expressed as:

$$m_{ts}^{(L_o)} = \sum_{L_i,L_f} w_{L_i,L_f,L_o} \left(x_s^{(L_i)} \otimes_{L_i,L_f}^{L_o} Y^{(L_f)}(\hat{r}_{ts}) \right) \tag{3}$$

where x_s is the irreps feature at source node s, $x_s^{(L_i)}$ denotes the L_i-th degree of x_s, and $\hat{r}_{ts} = \frac{\vec{r}_{ts}}{|\vec{r}_{ts}|}$. The spherical harmonic projection of relative positions $Y^{(L_f)}(\hat{r}_{ts})$ becomes sparse if we rotate $\hat{r}_{ts}$ with a rotation matrix R_{ts} to align with the direction of $L = 0$ and $m = 0$, which corresponds to the z axis traditionally but the y axis in the conventions of `e3nn` [55]. Concretely, given $R_{ts}\hat{r}_{ts}$ aligned with the y axis, $Y_{m_f}^{(L_f)}(R_{ts}\hat{r}_{ts}) \neq 0$ only for $m_f = 0$. Without loss of equivariance, we re-scale $Y_0^{(L_f)}(R_{ts}\vec{r}_{ts})$ to be one. Therefore, by rotating $x_s^{(L_i)}$ and $Y^{(L_f)}$ based on $\hat{r}_{ts}$, we can simplify Eq. 3

as follows:

$$
\begin{aligned}
m_{ts}^{(L_o)} &= \left(D^{(L_o)}(R_{ts})\right)^{-1} \sum_{L_i,L_f} w_{L_i,L_f,L_o} \left(D^{(L_i)}(R_{ts}) x_s^{(L_i)} \otimes_{L_i,L_f}^{L_o} Y^{(L_f)}(R_{ts}\hat{r}_{ts})\right) \\
&= \left(D^{(L_o)}\right)^{-1} \sum_{L_i,L_f} w_{L_i,L_f,L_o} \bigoplus_{m_o} \left(\sum_{m_i,m_f} \left(D^{(L_i)} x_s^{(L_i)}\right)_{m_i} C^{(L_o,m_o)}_{(L_i,m_i),(L_f,m_f)} \left(Y^{(L_f)}(R_{ts}\hat{r_{ts}})\right)_{m_f} \right) \\
&= \left(D^{(L_o)}\right)^{-1} \sum_{L_i,L_f} w_{L_i,L_f,L_o} \bigoplus_{m_o} \left(\sum_{m_i} \left(D^{(L_i)} x_s^{(L_i)}\right)_{m_i} C^{(L_o,m_o)}_{(L_i,m_i),(L_f,0)} \right) \\
&= \left(D^{(L_o)}\right)^{-1} \sum_{L_i,L_f} w_{L_i,L_f,L_o} \bigoplus_{m_o} \left(\sum_{m_i} \left(\tilde{x}_s^{(L_i)}\right)_{m_i} C^{(L_o,m_o)}_{(L_i,m_i),(L_f,0)} \right)
\end{aligned}
\tag{4}
$$

where $D^{(L_i)}(R_{ts}) = D^{(L_i)}$ and $D^{(L_o)}(R_{ts}) = D^{(L_o)}$ denote Wigner-D matrices of degrees L_i and L_o based on rotation matrix R_{ts}, respectively, $\bigoplus$ denotes concatenation, and $D^{(L_i)} x_s^{(L_i)} = \tilde{x}_s^{(L_i)}$. Additionally, given $m_f = 0$, Clebsch-Gordan coefficients $C^{(L_o,m_o)}_{(L_i,m_i),(L_f,0)}$ are sparse and are non-zero only when $m_i = \pm m_o$, which further simplifies Eq. 4:

$$
m_{ts}^{(L_o)} = \left(D^{(L_o)}\right)^{-1} \sum_{L_i,L_f} w_{L_i,L_f,L_o} \bigoplus_{m_o} \left(\left(\tilde{x}_s^{(L_i)}\right)_{m_o} C^{(L_o,m_o)}_{(L_i,m_o),(L_f,0)} + \left(\tilde{x}_s^{(L_i)}\right)_{-m_o} C^{(L_o,m_o)}_{(L_i,-m_o),(L_f,0)} \right)
\tag{5}
$$

By re-ordering the summations and concatenation in Eq. 5, we have:

$$
\left(D^{(L_o)}\right)^{-1} \sum_{L_i} \bigoplus_{m_o} \left(\left(\tilde{x}_s^{(L_i)}\right)_{m_o} \sum_{L_f} \left(w_{L_i,L_f,L_o} C^{(L_o,m_o)}_{(L_i,m_o),(L_f,0)} \right) + \left(\tilde{x}_s^{(L_i)}\right)_{-m_o} \sum_{L_f} \left(w_{L_i,L_f,L_o} C^{(L_o,m_o)}_{(L_i,-m_o),(L_f,0)} \right) \right)
\tag{6}
$$

Instead of using learnable parameters for w_{L_i,L_f,L_o}, eSCN proposes to parametrize $\tilde{w}^{(L_i,L_o)}_{m_o}$ and $\tilde{w}^{(L_i,L_o)}_{-m_o}$ as below:

$$
\begin{aligned}
\tilde{w}^{(L_i,L_o)}_{m_o} &= \sum_{L_f} w_{L_i,L_f,L_o} C^{(L_o,m_o)}_{(L_i,m_o),(L_f,0)} = \sum_{L_f} w_{L_i,L_f,L_o} C^{(L_o,-m_o)}_{(L_i,-m_o),(L_f,0)} \quad \text{for } m >= 0 \\
\tilde{w}^{(L_i,L_o)}_{-m_o} &= \sum_{L_f} w_{L_i,L_f,L_o} C^{(L_o,-m_o)}_{(L_i,m_o),(L_f,0)} = -\sum_{L_f} w_{L_i,L_f,L_o} C^{(L_o,m_o)}_{(L_i,-m_o),(L_f,0)} \quad \text{for } m > 0
\end{aligned}
\tag{7}
$$

The parametrization of $\tilde{w}^{(L_i,L_o)}_{m_o}$ and $\tilde{w}^{(L_i,L_o)}_{-m_o}$ enables removing the summation over L_f and further simplifies the computation. By combining Eq. 6 and Eq. 7, we have:

$$
\begin{aligned}
m_{ts}^{(L_o)} &= \left(D^{(L_o)}\right)^{-1} \sum_{L_i} \bigoplus_{m_o} \left(y_{ts}^{(L_i,L_o)}\right)_{m_o} \\
\left(y_{ts}^{(L_i,L_o)}\right)_{m_o} &= \tilde{w}^{(L_i,L_o)}_{m_o} \left(\tilde{x}_s^{(L_i)}\right)_{m_o} - \tilde{w}^{(L_i,L_o)}_{-m_o} \left(\tilde{x}_s^{(L_i)}\right)_{-m_o} \quad \text{for } m_o > 0 \\
\left(y_{ts}^{(L_i,L_o)}\right)_{-m_o} &= \tilde{w}^{(L_i,L_o)}_{-m_o} \left(\tilde{x}_s^{(L_i)}\right)_{m_o} + \tilde{w}^{(L_i,L_o)}_{m_o} \left(\tilde{x}_s^{(L_i)}\right)_{-m_o} \quad \text{for } m_o > 0 \\
\left(y_{ts}^{(L_i,L_o)}\right)_{m_o} &= \tilde{w}^{(L_i,L_o)}_{m_o} \left(\tilde{x}_s^{(L_i)}\right)_{m_o} \quad \text{for } m_o = 0
\end{aligned}
\tag{8}
$$

The formulation of $y_{ts}^{(L_i,L_o)}$ coincides with performing $SO(2)$ linear operations [18, 73]. Additionally, eSCN convolutions can further simplify the computation by considering only a subset of m_o components in Eq. 8, i.e., $|m_o| \leqslant M_{max}$.

In summary, efficient $SO(3)$ convolutions can be achieved by first rotating irreps features $x_s^{(L_i)}$ based on relative position vectors $\vec{r}_{ts}$ and then performing $SO(2)$ linear operations on rotated features. The key idea is that rotation simplifies the computation as in Eq. 4, 5, 7, and 8. Please refer to their

work [18] for more details. We note that eSCN convolutions consider only simplifying the case of taking tensor products between input irreps features and spherical harmonic projections of relative position vectors. eSCN convolutions do not simplify general cases such as taking tensor products between input irreps features and themselves [38] since the relative position vectors used to rotate irreps features are not clearly defined.

B Details of Architecture

In this section, we define architectural hyper-parameters like maximum degrees and numbers of channels in certain layers in EquiformerV2, which are used to specify the detailed architectures in Sec. C.1. Besides, we note that eSCN [18] and this work mainly consider $SE(3)$-equivariance.

We denote embedding dimensions as d_{embed}, which defines the dimensions of most irreps features. Specifically, the output irreps features of all modules except the output head in Figure 1a have dimension d_{embed}. For separable S^2 activation as illustrated in Figure 2c, we denote the resolution of point samples on a sphere as R, which can depend on maximum degree L_{max}, and denote the unconstrained functions after projecting to point samples as F.

For equivariant graph attention in Figure 1b, the input irreps features x_i and x_j have dimension d_{embed}. The dimension of the irreps feature $f_{ij}^{(L)}$ is denoted as d_{attn_hidden}. Equivariant graph attention can have h parallel attention functions. For each attention function, we denote the dimension of the scalar feature $f_{ij}^{(0)}$ as d_{attn_alpha} and denote the dimension of the value vector, which is in the form of irreps features, as d_{attn_value}. For the separable S^2 activation used in equivariant graph attention, the resolution of point samples is R, and we use a single SiLU activation for F. We share the layer normalization in attention re-normalization across all h attention functions but have different h linear layers after that. The last linear layer projects the dimension back to d_{embed}. The two intermediate $SO(2)$ linear layers operate with maximum degree L_{max} and maximum order M_{max}.

For feed forward networks (FFNs) in Figure 1d, we denote the dimension of the output irreps features of the first linear layer as d_{ffn}. For the separable S^2 activation used in FFNs, the resolution of point samples is R, and F consists of a two-layer MLP, with each linear layer followed by SiLU, and a final linear layer. The linear layers have the same number of channels as d_{ffn}.

For radial functions, we denote the dimension of hidden scalar features as d_{edge}. For experiments on OC20, same as eSCN [18], we use Gaussian radial basis to represent relative distances and additionally embed the atomic numbers at source nodes and target nodes with two scalar features of dimension d_{edge}. The radial basis and the two embeddings of atomic numbers are fed to the radial function to generate edge distance embeddings.

The maximum degree of irreps features is denoted as L_{max}. All irreps features have degrees from 0 to L_{max} and have C channels for each degree. We denote the dimension as (L_{max}, C). For example, irreps feature x_{irreps} of dimension $(6, 128)$ has maximum degree 6 and 128 channels for each degree. The dimension of scalar feature x_{scalar} can be expressed as $(0, C_{scalar})$.

Following Equiformer [17], we apply dropout [40] to attention weights and stochastic depth [41] to outputs of equivariant graph attention and feed forward networks. However, we do not apply dropout or stochastic depth to the output head.

C Details of Experiments on OC20

C.1 Training Details

Hyper-Parameters. We summarize the hyper-parameters for the base model setting on OC20 S2EF-2M dataset and the main results on OC20 S2EF-All and S2EF-All+MD datasets in Table 4. For the ablation studies on OC20 S2EF-2M dataset, when trained for 20 or 30 epochs as in Table 1b, we increase the learning rate from 2×10^{-4} to 4×10^{-4}. When using $L_{max} = 8$ as in Table 1c, we increase the resolution of point samples R from 18 to 20. We vary L_{max} and the widths for speed-accuracy trade-offs in Figure 4. Specifically, we first decrease L_{max} from 6 to 4. Then, we multiply h and the number of channels of $(d_{embed}, d_{attn\_hidden}, d_{ffn})$ by 0.75 and 0.5. We train all models for 30 epochs. The same strategy to scale down eSCN models is adopted for fair comparisons.

Hyper-parameters	Base model setting on S2EF-2M	EquiformerV2 (31M) on S2EF-All+MD	EquiformerV2 (153M) on S2EF-All/S2EF-All+MD
Optimizer	AdamW	AdamW	AdamW
Learning rate scheduling	Cosine learning rate with linear warmup	Cosine learning rate with linear warmup	Cosine learning rate with linear warmup
Warmup epochs	0.1	0.01	0.01
Maximum learning rate	2×10^{-4}	4×10^{-4}	4×10^{-4}
Batch size	64	512	256 for S2EF-All, 512 for S2EF-All+MD
Number of epochs	12	3	1
Weight decay	1×10^{-3}	1×10^{-3}	1×10^{-3}
Dropout rate	0.1	0.1	0.1
Stochastic depth	0.05	0.1	0.1
Energy coefficient λ_E	2	4	2 for S2EF-All, 2, 4 for S2EF-All+MD
Force coefficient λ_F	100	100	100
Gradient clipping norm threshold	100	100	100
Model EMA decay	0.999	0.999	0.999
Cutoff radius (Å)	12	12	12
Maximum number of neighbors	20	20	20
Number of radial basis	600	600	600
Dimension of hidden scalar features in radial functions d_{edge}	$(0, 128)$	$(0, 128)$	$(0, 128)$
Maximum degree L_{max}	6	4	6
Maximum order M_{max}	2	2	3
Number of Transformer blocks	12	8	20
Embedding dimension d_{embed}	$(6, 128)$	$(4, 128)$	$(6, 128)$
$f_{ij}^{(L)}$ dimension d_{attn_hidden}	$(6, 64)$	$(4, 64)$	$(6, 64)$
Number of attention heads h	8	8	8
$f_{ij}^{(0)}$ dimension d_{attn_alpha}	$(0, 64)$	$(0, 64)$	$(0, 64)$
Value dimension d_{attn_value}	$(6, 16)$	$(4, 16)$	$(6, 16)$
Hidden dimension in feed forward networks d_{ffn}	$(6, 128)$	$(4, 128)$	$(6, 128)$
Resolution of point samples R	18	18	18

Table 4: Hyper-parameters for the base model setting on OC20 S2EF-2M dataset and the main results on OC20 S2EF-All and S2EF-All+MD datasets.

Training set	Attention re-normalization	Activation	Normalization	L_{max}	M_{max}	Number of Transformer blocks	Training time (GPU-hours)	Inference speed (Samples / GPU sec.)	Number of parameters
S2EF-2M	✗	Gate	LN	6	2	12	965	19.06	91.06M
	✓	Gate	LN	6	2	12	998	19.07	91.06M
	✓	S^2	LN	6	2	12	1476	12.80	81.46M
	✓	Sep. S^2	LN	6	2	12	1505	12.51	83.16M
	✓	Sep. S^2	SLN	6	2	12	1412	13.22	83.16M
	✓	Sep. S^2	SLN	4	2	12	965	19.86	44.83M
	✓	Sep. S^2	SLN	8	2	12	2709	7.86	134.28M
	✓	Sep. S^2	SLN	6	3	12	1623	11.92	95.11M
	✓	Sep. S^2	SLN	6	4	12	2706	7.98	102.14M
	✓	Sep. S^2	SLN	6	6	12	3052	7.13	106.63M
S2EF-All	✓	Sep. S^2	SLN	6	3	20	20499	6.08	153.60M
S2EF-All+MD ($\lambda_E = 2$)	✓	Sep. S^2	SLN	6	3	20	32834	6.08	153.60M
S2EF-All+MD ($\lambda_E = 4$)	✓	Sep. S^2	SLN	4	2	8	16931	29.21	31.06M
S2EF-All+MD ($\lambda_E = 4$)	✓	Sep. S^2	SLN	6	3	20	37692	6.08	153.60M

Table 5: Training time, inference speed and numbers of parameters of different models trained on OC20 S2EF-2M, S2EF-All and S2EF-All+MD datasets. All numbers are measured on V100 GPUs with 32GB.

Training Time, Inference Speed and Numbers of Parameters. Table 5 summarizes the training time, inference speed and numbers of parameters of models in Tables 1a (row 1, 2, 3, 4, 5), 1c, 1d and 2. V100 GPUs with 32GB are used to train all models. We use 16 GPUs to train each individual model on S2EF-2M dataset, 64 GPUs for S2EF-All, 64 GPUs for EquiformerV2 (31M) on S2EF-All+MD, and 128 GPUs for EquiformerV2 (153M) on S2EF-All+MD.

C.2 Details of Running Relaxations

A structural relaxation is a local optimization where atom positions are iteratively updated based on forces to minimize the energy of the structure. We perform ML relaxations using the LBFGS optimizer (quasi-Newton) implemented in the Open Catalyst Github repository [30]. The structural relaxations for OC20 IS2RE and IS2RS tasks are allowed to run for 200 steps or until the maximum predicted force per atom $F_{max} \leqslant 0.02$ eV/Å , and the relaxations for AdsorbML are allowed to run for 300 steps or until $F_{max} \leqslant 0.02$ eV/Å . These settings are chosen to be consistent with prior works. We run relaxations on V100 GPUs with 32GB. The computational cost of running relaxations with EquiformerV2 (153M) for OC20 IS2RE and IS2RS tasks is 1011 GPU-hours, and that of running ML relaxations for AdsorbML is 1075 GPU-hours. The time for running relaxations with EquiformerV2 (31M) is 240 GPU-hours for OC20 IS2RE and IS2RS and 298 GPU-hours for AdsorbML.

C.3 Details of AdsorbML

We run the AdsorbML algorithm on the OC20-Dense dataset in accordance with the procedure laid out in the paper [10], which is summarized here:

1. Run ML relaxations on all initial structures in the OC20-Dense dataset. There are around 1000 different adsorbate-surface combinations with about 90 adsorbate placements per combination, and therefore we have roughly 90k structures in total.
2. Remove invalid ML relaxed structures based on physical constraints and rank the other ML relaxed structures in order of lowest to highest ML predicted energy.
3. Take the top k ML relaxed structures with the lowest ML predicted energies for each adsorbate-surface combination and run DFT single-point calculations. The single-point calculations are performed on the ML relaxed structures to improve the energy predictions without running a full DFT relaxation and are run with VASP using the same setting as the original AdsorbML experiments. As shown in Table 3, we vary k from 1 to 5.
4. Compute success and speedup metrics based on our lowest DFT single-point energy per adsorbate-surface combination and the DFT labels provided in the OC20-Dense dataset.

We visualize some examples of relaxed structures from eSCN [18], EquiformerV2 and DFT in Figure 6.

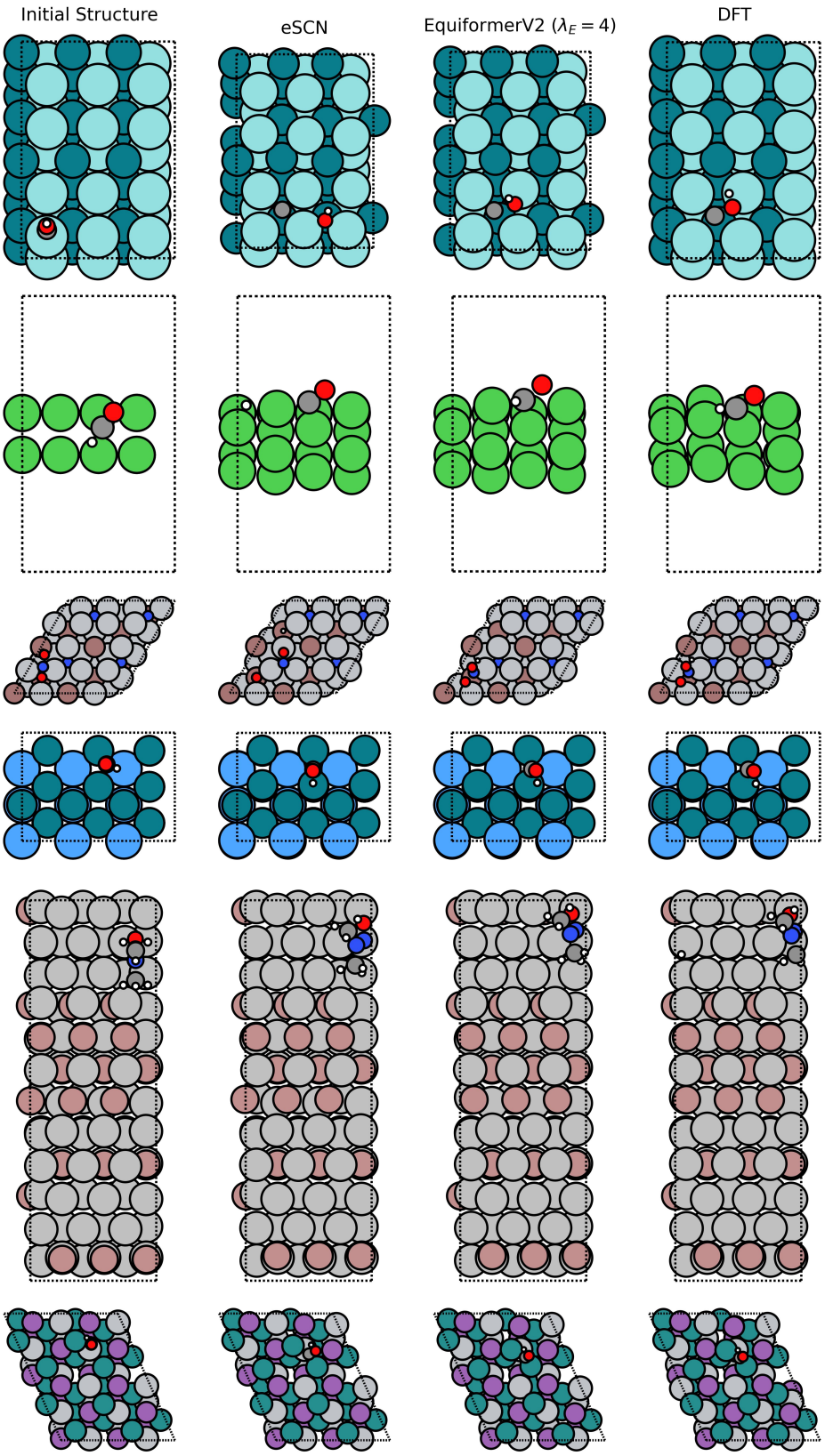


Figure 6: Qualitative examples of the initial configuration of an adsorbate on a catalyst surface (column 1), and corresponding relaxed configurations obtained from eSCN [18] (column 2), EquiformerV2 (column 3), and DFT (column 4). All examples are selected from the OC20-Dense dataset [10]. We show top-down views of each structure, with dashed lines showing the boundary of the unit cell repeating in the x and y directions.