SE(3)-equivariant self-attention via invariant features

Nan Chen
School of Computing
National University of Singapore

Soledad Villar
Department of Applied Mathematics and Statistics
Johns Hopkins University

Abstract

In this work, we use classical invariant theory to construct a self-attention module equivariant to 3D rotations and translations. The parameterization is based on the characterization of SE(3)-equivariant functions via the invariants —scalar products of vectors and certain subdeterminants. This parameterization can be seen as a natural extension to a (more straightforward) E(3) equivariant attention based on invariants —scalar products or pairwise distances of vectors. We evaluate our model using a toy N-body particle simulation dataset and a real-world dataset of molecular properties. Our model is easy to implement and it exhibits comparable performance and running time to state-of-the-art methods.

1 Introduction

Symmetry, serving as a kind of inductive bias, has demonstrated a potential for guiding the design of neural networks [6], such as the spatial translation equivariance in convolutional neural networks (CNNs) [27], temporal translation equivariance in recurrent neural networks (RNNs) [14], and permutation equivariance in graph neural networks (GNNs) [7, 9, 22]. By forcing a neural network to respect the exact or approximate symmetries underlying the task in hand, researchers typically restrict the hypothesis class of function which may improve the generalization performance of the resulting model (like [3, 13]). 3D rotation and translation equivariance is one kind of symmetry underlying many problems in physics and chemistry, such as N-body particle simulation [23], 3D molecular structure modeling [2, 21, 36], and point clouds [37]. The group corresponding to this equivariance is called the Special Euclidean group, or SE(3). Under this equivariance constraint, global rotations or translations to the input features result on the same transformations applied to the output. Parameterizing a model respectful of this equivariance is often a desirable choice. Another group of interest is E(3) which includes reflections as well as rotations and translations. Interestingly, parameterizing E(3)-equivariant functions is easier than SE(3)-equivariant functions. Since SE(3) is a subgroup of E(3) the space of E(3)-equivariant functions is a subspace of the space of SE(3)-equivariant functions.

To the best of our knowledge, SE(3) equivariance is implemented by parameterizing neural networks in terms of the irreducible representations of the 3D rotation group, SO(3) on tensor spaces $(\mathbb{R}^3)^{\otimes k}$ [5, 16, 17, 35]. These irreducible representations are computed beforehand (via computing the Clebsch-Gordan coefficients) and kept fixed. The networks learn how to combine these representations for each layer $(\mathbb{R}^3)^{\otimes k_i} \mapsto (\mathbb{R}^3)^{\otimes k_{i+1}}$ by matching the irreducible representations of $(\mathbb{R}^3)^{\otimes k_i}$ with the ones of $(\mathbb{R}^3)^{\otimes k_{i+1}}$ making use of Schur's lemma. Neural networks defined in this form are universal as long as k can be taken arbitrarily large [12]. However, having large order tensors is not an issue because the tensors can be compactly represented using spherical harmonics. Previous works tend to use irreducible representations to construct generalized convolution-like networks [1, 25, 26, 35, 41], and there is a recent trend to use this approach to build equivariant self-attention modules [16].

Classical invariant theory provides another way to parameterize the space of invariant and equivariant functions—using the generators of the invariant ring of polynomials [4, 40, 42]. These ideas provide a simple universal parameterization of the space of equivariant functions that does not require the use

Machine Learning and the Physical Sciences workshop, NeurIPS 2022.

of irreducible representations nor high order tensors, as long as the invariants are explicitly known (which is the case of several simple Lie groups). In this work we show how to use this approach to construct equivariant self-attention modules, and we particularly apply it to SE(3). We note that some E(d)-equivariant self-attention modules in the literature are implicitly defined in terms of the invariants such as [8, 24, 29, 30, 32–34]. One possible reason is that the O(d) invariants are the inner products of the inputs, and therefore it is intuitive to define self-attention mechanisms in those terms, whereas the SO(d) invariants are slightly less intuitive. Here we make the connection with invariants explicit and show how to implement it for SE(3) (to best of our knowledge this hasn't been done before). As a consequence, a straightforward modification of our model can be used to define equivariant self-attention modules with respect to Lorentz, Poincaré, symplectic and unitary groups.

Self-attention mechanisms: Given a set of nodes with features, a self-attention module [38] maps each node into a query and a set of key-value pairs to produce the output. The output is a weighted sum of the values, and the weight for each value is the similarity between the query and corresponding key. The implementation of self-attention mechanisms is very simple and it allows for flexible designs [2, 20, 21]. This is why these models have been widely applied: from language modeling [10, 38] to graph-based problems [11, 39]. SE(3) and E(3) equivariant self-attention modules have recently been proposed. [16] proposes an SE(3)-equivariant self-attention module, using irreducible representations to construct SE(3)-equivariant query, key, and value from input. In E(3)-equivariant modules, the L2-norm of the difference of position vectors between nodes is a commonly used invariant feature. Some works compute query, key, and value from scalar features, and integrate the similarity value between query and key with the invariant feature via either addition [29] or multiplication [8, 30, 34]. [24, 32, 33] adopt a more general form of self-attention where attention weights are mapped directly from scalar features together with the invariant features.

Our contributions are summarized as follows:

- We show how to parameterize SE(3)-equivariant self-attention modules based on invariant features, obtaining a model with comparable performance to the state-of-the-art.
- The framework we adopt to build the self-attention modules is flexible, and can be extended to handle input vectors subject to other simple Lie groups without much effort.
- Code is publicly available at `https://github.com/NanChanNN/equi_self_attn`.

2 Method

Assume that we are given a set of n nodes with input features denoted as $\{\vec{x}_i, \{\vec{h}_{i,t}\}, \{s_{i,\ell}\}\}_{i=1}^n$ and (optionally) edge features $\{e_{ji}\}_{j,i=1}^n$. Here, the features are of three types: $\vec{x}_i \in \mathbb{R}^3$ denotes the position vector of node i (subject to rotations and translations), $\{\vec{h}_{i,t}\}_{t=1}^T \subset \mathbb{R}^3$ denotes a set of vectors subject to rotations, like velocity vectors, and $\{s_{i,\ell}\}_{\ell=1}^L \subset \mathbb{R}$ denotes a set of scalar features, like mass or charge. The query q, key k, and value v of the self-attention mechanism are computed as:

$$\{\vec{h}_{i,t'}^{(q)}\}, \{s_{i,\ell'}^{(q)}\} = \Phi^{(q)}(\{\vec{h}_{i,t}\}, \{s_{i,\ell}\}) \tag{1}$$

$$\{\vec{h}_{ji,t'}^{(k)}\}, \{s_{ji,\ell'}^{(k)}\} = \Phi^{(k)}(\{\vec{h}_{j,t}\} \cup \{\vec{x}_i - \vec{x}_j\},\ \{e_{ji}\} \cup \{s_{j,\ell}\}) \tag{2}$$

$$\{\vec{h}_{ji,t'}^{(v)}\}, \{s_{ji,\ell'}^{(v)}\} = \Phi^{(v)}(\{\vec{h}_{j,t}\} \cup \{\vec{x}_i - \vec{x}_j\},\ \{e_{ji}\} \cup \{s_{j,\ell}\}) \tag{3}$$

where $\Phi^{(*)}$ denotes a rotation-equivariant function (or SO(3)-equivariant function) that takes a set of vector and scalar features as input, and generates a new set of vector and scalar features:

$$\{\vec{v'}_{t'}\}_{t'=1}^{T'}, \{s'_{\ell'}\}_{\ell'=1}^{L'} = \Phi^{(*)}(\{\vec{v}_t\}_{t=1}^T, \{s_\ell\}_{\ell=1}^L) \tag{4}$$

where the input and output vector features are constrained by rotation equivariance, namely:

$$\{R(\vec{v'}_{t'})\}, \{s'_{\ell'}\} = \Phi^{(*)}(\{R(\vec{v}_t)\}, \{s_\ell\}) \tag{5}$$

for all $R \in \mathrm{SO}(3)$. (Note we dropped the ranges of the indices to simplify the notation, but they coincide with those of (4)). Previous works [16, 35] use irreducible representations to parameterize the SO(3)-equivariant functions $\Phi^{(*)}$. By contrast, in this work, we apply the formula proposed in

[40] to the construction of the functions:

$$\vec{v'}_{t'} = \sum_{t=1}^{T} f_{t't}(\vec{v}_1, \ldots, \vec{v}_T, s_1, \ldots, s_L) \cdot \vec{v}_t + \sum_{S \in \binom{[T]}{2}} f_{t'S}(\vec{v}_1, \ldots, \vec{v}_T, s_1, \ldots, s_L) \cdot \vec{v}_S \tag{6}$$

$$s_{\ell'} = f_{\ell'}(\vec{v}_1, \ldots, \vec{v}_T, s_1, \ldots, s_L) \tag{7}$$

where f_* are SO(3)-invariant scalar functions (assuming SO(3) acts by multiplication on the vector inputs, and trivially on the scalar inputs), and $\vec{v}_S$ is the cross product of the two vectors $\vec{v}_r$ $\vec{v}_{r'}$ with $r, r' \in S$. The SO(3)-invariant functions can be written as a function of the scalar products of the v_i and the 3×3 subdeterminants of the $3 \times T$ matrix of vectors

$$f_*(\vec{v}_1, \ldots, \vec{v}_T, s_1, \ldots, s_L) = g(\langle \vec{v}_i, \vec{v}_j \rangle_{i,j=1}^{T}, \det[\vec{v}_1, \ldots, \vec{v}_T]_{3\times 3}, s_1, \ldots, s_L) \tag{8}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product, and g is implemented as a multilayer perceptron (MLP) with a uniform architecture in our model: Input → { LinearLayer() → LayerNorm() → ReLU() → LinearLayer() } → Output. The query vector q_i and key vector k_{ji} are constructed by concatenating vector and scalar features together:

$$q_i = \vec{h}_{i,1}^{(q)} \oplus \vec{h}_{i,2}^{(q)} \cdots \oplus \vec{h}_{i,T'}^{(q)} \oplus s_{i,1}^{(q)} \oplus s_{i,2}^{(q)} \cdots \oplus s_{i,L'}^{(q)} \in \mathbb{R}^{3\times T' + L'} \tag{9}$$

$$k_{ji} = \vec{h}_{ji,1}^{(k)} \oplus \vec{h}_{ji,2}^{(k)} \cdots \oplus \vec{h}_{ji,T'}^{(k)} \oplus s_{ji,1}^{(k)} \oplus s_{ji,2}^{(k)} \cdots \oplus s_{ji,L'}^{(k)} \in \mathbb{R}^{3\times T' + L'} \tag{10}$$

After that, the self-attention mechanism is applied as usual. Here we use the dot-product attention:

$$\alpha_{ji} = \frac{\exp(\langle q_i, k_{ji} \rangle)}{\sum_{j'} \exp(\langle q_i, k_{j'i} \rangle)} \tag{11}$$

where attention weights α_{ji} are applied to each channel of the value features to derive the final result:

$$\vec{h'}_{i,t'} = \sum_j \alpha_{ji} \cdot \vec{h}_{ji,t'}^{(v)}, \qquad s'_{i,\ell'} = \sum_j \alpha_{ji} \cdot s_{ji,\ell'}^{(v)} \tag{12}$$

In this way, each node i gets new features $\{\vec{h'}_{i,t'}\}$ and $\{s'_{i,\ell'}\}$, which will be fed into the next layer.

Multi-head attentions Multi-head attention mechanisms can be formulated by evenly partitioning the set of vector and scalar features into multiple parts, applying the self-attention mechanism described above to each part, and finally merging results of different parts via concatenation or summation.

Extensibility This model can be extended to express equivariant functions with respect to other simple Lie groups by redefining the function $\Phi^{(*)}$ (see [40]).

Scalability One limitation of this model is that the number of pairwise scalar products in the definition of the invariant functions (8) scales quadratically with the number of vectors, and the number of subdeterminants scales cubically. A discussion in [40] claims that by using techniques from matrix completion and rigidity theory one could potentially replace the scalar products by a subset of size linear in n. For reducing the number of subdeterminants, one possible approach is to randomly sample the subdeterminants that go into (8).

3 Experiments

N-Body Simulations In this part, we conduct an N-Body simulation experiment to evaluate the performance of our method. We follow the same experiment procedure as that in [16], and we use an extension of the dataset from [23] to the 3D space. The input to the model is an N-Body system consisting of five particles, each of which carries a negative or positive charge, and has position and velocity vectors in the 3D space. The goal of the experiment is to predict future positions and velocities of particles in the system after 500 time steps of simulation.

We mainly compare our model to three other models, namely, the SE(3)-Transformer [16], tensor field networks (TFN) [35], and E(n) equivariant graph neural networks (EGNNs) [32]. We include two

Table 1: Quantitative results in the N-Body simulation experiment. SIZE denotes the amount of trainable parameters. TIME denotes the time to train the model for one epoch in terms of seconds. MSE x and MSA v denote the mean squared error of predicted positions and velocities, respectively. Top: Non-equivariant models whose results are taken from [16]. Bottom: Equivariant models.

MODEL	SIZE	TIME (s)	MSE x ($\cdot\ 10^{-4}$)	MSE v ($\cdot\ 10^{-3}$)
Linear	-	-	691.0	261.0
DeepSet [43]	-	-	639.0 ± 86.0	246.0 ± 17.0
Set Transformer [28]	-	-	139.0 ± 4.0	101.0 ± 4.0
SE(3)-Transformer [16]	163K	33.3 ± 0.06	35.8 ± 6.1	60.1 ± 1.5
TFN [35]	86K	23.8 ± 4.2	100.6 ± 5.4	96.5 ± 1.2
EGNNs [32]	134K	$\mathbf{5.3 \pm 0.4}$	$\mathbf{31.6 \pm 1.0}$	-
Ours-SE(3)	184K	9.9 ± 0.2	34.9 ± 1.8	58.2 ± 3.0
Ours-E(3)	115K	5.9 ± 0.2	$\mathbf{31.2 \pm 1.2}$	$\mathbf{54.4 \pm 1.0}$

variants of our model: one is equivariant to SE(3) and the other is equivariant to E(3). The E(3) variant is implemented by simply removing the cross product term in (6) and the subdeterminant term in (8). Both variants are constructed to have 4 self-attention layers. For the SE(3)-Transformer and TFN, we adopt the implementation provided by [16]. For the EGNNs, we use the official implementation provided by its authors [32]. As the official implementation of the EGNNs does not update the input velocity features and only generates future coordinates as output, we do not test it on predicting future velocities. We use the same hyperparameter settings as those in [16], except that, for our models and the EGNNs, we sweep over different learning rates and select the one with the best performance for each model. Experiments are conducted on a NVIDIA A5000 GPU. Each model is run 5 times using different seeds with 500 epochs per run. Quantitative results are summarized in Table 1 with training time per epoch and model size included as well. We also borrow results of non-equivariant models from [16] to examine the effect of equivariance. These non-equivariant models include a linear baseline, the DeepSet model [43], and the Set Transformer model [28], whose results are organized on the top half of Table 1. From the quantitative results, we find that our SE(3) variant performs similarly to the SE(3)-Transformer, its counterpart using irreducible representations, in terms of prediction accuracy on future positions and velocities. Our E(3) variant has a comparable performance in position prediction to the state-of-the-art method, the EGNNs. As the NBody system is equivariant to E(3), we see advantages of E(3) models over SE(3) ones. Generally, equivariant models outperform non-equivariant ones by a considerable margin.

QM9 In this part, we evaluate our model on a molecular property regression dataset, the QM9 dataset [31]. This dataset contains 134k small molecules with up to 29 atoms. Each atom is encoded as a node with a five-dimension one-hot embedding indicating the atom type and a three-dimension position vector. Chemical bonds between atoms are represented as edges with one-hot embedding indicating bond types. The task is to predict different chemical properties for each molecule given. For a detailed description of these chemical properties, readers can refer to [31].

We use the scripts provided by [16] to conduct the experiment. Following common practice, we tune our architecture on $\varepsilon_{\mathsf{HOMO}}$ and reuse the same architecture on the other tasks. Similar to the preceding experiment, this experiment includes an SE(3) and E(3) variant of our model. Each variant has 7 self-attention layers, followed by a sum pooling to aggregate node features. Each self-attention layer adopts a multi-head attention mechanism with 8 attention heads. We use a batch size of 128 and an epoch size of 200 to make sure that all models get the same number of updates for fair comparison. The other hyperparameter settings are kept the same as those in [16]. We measure the performance on the test set of [1] in 6 regression tasks. Quantitative results are summarized in Table 2. Note that results of other models are taken from [16], and the top half of Table 2 includes a non-equivariant model, WaveScatt [19]. The quantitative results are similar to those in the N-body experiment. Our SE(3) variant delivers a similar performance to that of the SE(3)-Transformer, and our E(3) variant shows a comparable result to the state-of-the-art. Overall, the non-equivariant model performs worse.

Table 2: Quantitative results in the QM9 experiment. Models are evaluated by mean absolute error. Top: Non-equivariant models. Bottom: Equivariant models.

TASK	α	$\Delta\varepsilon$	$\varepsilon_{\mathsf{HOMO}}$	$\varepsilon_{\mathsf{LUMO}}$	μ	C_v
UNITS	bohr^3	meV	meV	meV	D	cal/mol K
WaveScatt [19]	.160	118	85	76	.340	.049
NMP [18]	.092	69	43	38	.030	.040
SchNet [33]	.235	63	41	34	.033	.033
Cormorant [1]	.085	61	34	38	.038	**.026**
LieConv(T3) [15]	.084	49	30	**25**	.032	.038
TFN [35]	.223	58	40	38	.064	.101
SE(3)-Transformer [16]	.142	53	35	33	.051	.054
Ours-SE(3)	.107	57	38	33	.035	.043
Ours-E(3)	**.083**	**48**	**29**	27	**.029**	.037

4 Conclusion

In this work, we present a self-attention module equivariant to global rotations and translations, which is constructed based on invariant features. Our experimental results demonstrate that our model has a comparative performance to other methods that use irreducible representations. In the future, we would like to explore several directions based on the current work. Current experiments are on datasets with small inputs, one future direction is to evaluate the model on larger systems. The current work only empirically compares the difference between our model and those using irreducible representations, another future direction would be to theoretically analyze such difference.

Broader impact

We believe this work can have impacts on several domains. In physics, our work can be applied to the simulation of particle behaviors in dynamical systems. Our model can be easily extended to other simple Lie groups in classical physics including Lorentz, unitary, and symplectic groups, independent of the dimension. This allows applications to higher-dimensional spaces and more complicated physical problems. In biology and medicine, our work can be applied to learn representations of macro- or micro- molecules effectively. The learned representations can then be used in many downstream tasks, such as drug chemical property prediction, molecule generation, and protein folding. The simple implementation of our model may also allow for its integration into existing state-of-the-art algorithms in biology.

References

[1] B. M. Anderson, T. Hy, and R. Kondor. Cormorant: Covariant molecular neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[2] M. Baek, F. DiMaio, I. Anishchenko, J. Dauparas, S. Ovchinnikov, G. R. Lee, J. Wang, Q. Cong, L. N. Kinch, R. D. Schaeffer, C. Millán, H. Park, C. Adams, C. R. Glassman, A. DeGiovanni, J. H. Pereira, A. V. Rodrigues, A. A. van Dijk, A. C. Ebrecht, D. J. Opperman, T. Sagmeister, C. Buhlheller, T. Pavkov-Keller, M. K. Rathinaswamy, U. Dalwadi, C. K. Yip, J. E. Burke, K. C. Garcia, N. V. Grishin, P. D. Adams, R. J. Read, and D. Baker. Accurate prediction of protein structures and interactions using a three-track neural network. *Science*, 2021.

[3] A. Bietti, L. Venturi, and J. Bruna. On the sample complexity of learning under geometric stability. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[4] B. Blum-Smith and S. Villar. Equivariant maps from invariant functions. *arXiv preprint*, 2022.

[5] A. Bogatskiy, B. M. Anderson, J. T. Offermann, M. Roussi, D. W. Miller, and R. Kondor. Lorentz group equivariant neural network for particle physics. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2020.

[6] M. M. Bronstein, J. Bruna, T. Cohen, and P. Velickovic. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *CoRR*, 2021.

[7] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun. Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations (ICLR)*, 2014.

[8] Y. Choukroun and L. Wolf. Geometric transformer for end-to-end molecule properties prediction. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2022.

[9] M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[10] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, 2018.

[11] V. P. Dwivedi and X. Bresson. A generalization of transformer networks to graphs. *CoRR*, 2020.

[12] N. Dym and H. Maron. On the universality of rotation equivariant point cloud networks. In *International Conference on Learning Representations (ICLR)*, 2021.

[13] B. Elesedy and S. Zaidi. Provably strict generalisation benefit for equivariant models. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2021.

[14] J. L. Elman. Finding structure in time. *Cognitive Science*, 1990.

[15] M. Finzi, S. Stanton, P. Izmailov, and A. G. Wilson. Generalizing convolutional neural networks for equivariance to lie groups on arbitrary continuous data. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2020.

[16] 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.

[17] M. Geiger and T. Smidt. e3nn: Euclidean neural networks. *CoRR*, 2022.

[18] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. Neural message passing for quantum chemistry. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2017.

[19] M. J. Hirn, S. Mallat, and N. Poilvert. Wavelet scattering regression of quantum chemical energies. *Multiscale Model. Simul.*, 2017.

[20] J. Ho, N. Kalchbrenner, D. Weissenborn, and T. Salimans. Axial attention in multidimensional transformers. *CoRR*, 2019.

[21] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, A. Bridgland, C. Meyer, S. A. A. Kohl, A. J. Ballard, A. Cowie, B. Romera-Paredes, S. Nikolov, R. Jain, J. Adler, T. Back, S. Petersen, D. Reiman, E. Clancy, M. Zielinski, M. Steinegger, M. Pacholska, T. Berghammer, S. Bodenstein, D. Silver, O. Vinyals, A. W. Senior, K. Kavukcuoglu, P. Kohli, and D. Hassabis. Highly accurate protein structure prediction with alphafold. *Nature*, 2021.

[22] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[23] T. N. Kipf, E. Fetaya, K. Wang, M. Welling, and R. S. Zemel. Neural relational inference for interacting systems. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.

[24] J. Köhler, L. Klein, and F. Noé. Equivariant flows: Exact likelihood generative learning for symmetric densities. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2020.

[25] R. Kondor. N-body networks: a covariant hierarchical neural network architecture for learning atomic potentials. *CoRR*, 2018.

[26] 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 (NeurIPS)*, 2018.

[27] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 2015.

[28] J. Lee, Y. Lee, J. Kim, A. R. Kosiorek, S. Choi, and Y. W. Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019.

[29] L. Maziarka, T. Danel, S. Mucha, K. Rataj, J. Tabor, and S. Jastrzebski. Molecule attention transformer. *CoRR*, 2020.

[30] A. Morehead, C. Chen, and J. Cheng. Geometric transformers for protein interface contact prediction. In *International Conference on Learning Representations (ICLR)*, 2022.

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

[32] V. G. Satorras, E. Hoogeboom, and M. Welling. E (n) equivariant graph neural networks. In *International Conference on Learning Representations (ICLR)*, 2021.

[33] K. Schütt, P. Kindermans, H. E. S. Felix, S. Chmiela, A. Tkatchenko, and K. Müller. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

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

[35] 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. *CoRR*, 2018.

[36] R. J. L. Townshend, S. Eismann, A. M. Watkins, R. Rangan, M. Karelina, R. Das, and R. O. Dror. Geometric deep learning of rna structure. *Science*, 2021.

[37] M. A. Uy, Q. Pham, B. Hua, D. T. Nguyen, and S. Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *2019 IEEE/CVF International Conference on Computer Vision, (ICCV)*, 2019.

[38] 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.

[39] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph attention networks. In *International Conference on Learning Representations (ICLR)*, 2018.

[40] S. Villar, D. W. Hogg, K. Storey-Fisher, W. Yao, and B. Blum-Smith. Scalars are universal: Equivariant machine learning, structured like classical physics. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[41] 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 (NeurIPS)*, 2018.

[42] H. Weyl. *The classical groups*. Princeton University Press, 1946.

[43] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola. Deep sets. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

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? [Yes]
   - (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? [N/A]
   - (b) Did you include complete proofs of all theoretical results? [N/A]
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)? [Yes]
   - (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   - (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   - (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]
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? [Yes] Mentioned in the anonymous repository.
   - (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   - (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] Mentioned in the anonymous repository.
   - (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]