Using Multiple Vector Channels Improves $E(n)$-Equivariant Graph Neural Networks

Daniel Levy [* 1 2] **Sékou-Oumar Kaba** [* 1 2] **Carmelo Gonzales** [3] **Santiago Miret** [3] **Siamak Ravanbakhsh** [1 2]

Abstract

We present a natural extension to $E(n)$-equivariant graph neural networks that uses multiple equivariant vectors per node. We formulate the extension and show that it improves performance across different physical systems benchmark tasks, with minimal differences in runtime or number of parameters. The proposed multichannel EGNN outperforms the standard single-channel EGNN on N-body charged particle dynamics, molecular property predictions, and predicting the trajectories of solar system bodies. Given the additional benefits and minimal additional cost of multi-channel EGNN, we suggest that this extension may be of practical use to researchers working in machine learning for astrophysics and cosmology.

1. Introduction

Designing neural network architectures that correctly account for the symmetry of physical laws is an important requirement for applications of artificial intelligence in science. In particular, for dynamical systems in astrophysics, the relevant properties transform invariantly or equivariantly under Euclidean transformations. This is also the case when modelling particles or atomistic systems, for which machine learning simulators are already widely used.

There now exist a wide variety of equivariant neural network architectures leveraging a diverse set of mathematical formulations, among which we highlight two important classes. First, some architectures apply spherical harmonics mappings to incorporate directional information in an equivariant way (Thomas et al., 2018; Fuchs et al.,

2020). These architectures have the advantage of being highly expressive but are also computationally expensive and challenging to implement. Second, we highlight models that fall under the equivariant multilayer perceptron (E-MLP) paradigm (Finzi et al., 2021). The idea behind this paradigm is to simply generalize standard multilayer perceptrons by composing equivariant linear layers with appropriate non-linear functions. These architectures are much simpler to work with and more computationally efficient than those using spherical harmonics, but in principle, require lifting input quantities to high-order tensors to achieve high expressivity (Finkelshtein et al., 2022). Prior works, however, show that one can achieve satisfactory modelling performance without requiring higher-order representations. One such example is the Vector Neurons (Deng et al., 2021) model, which can be seen as an equivariant multilayer perceptron with order-1 vector features. This architecture also leverages the fact that the number of vector channels (neurons) in each layer can be arbitrary to increase expressivity.

The $E(n)$-equivariant Graph Neural Network (EGNN) model by Satorras et al. (2021) is an example of a model that does not clearly fit into one of the categories above. Nevertheless, EGNN has become widely applied mainly due to efficiency and simple model design. EGNN uses the message-passing framework, which captures the inductive bias that sparse interactions between entities should lead to better generalization. EGNN also has the advantage of separating equivariant features into a separate channel that only follows equivariant operations. The work of (Brandstetter et al., 2022) extends EGNN by using ideas inspired by spherical-harmonics-type architectures. Their Steerable $E(n)$-Equivariant Graph Neural Network (SEGNN) achieves better performance across some benchmarks but suffers from similar conceptual shortcomings in addition to increased computational complexity.

In this paper, we explore the direction of generalizing EGNN by drawing from E-MLP-type architectures. EGNN only updates a single vector for each node in the graph over each layer. A natural way to increase the expressivity of this model is to make the number of vector channels arbitrary. In our experiments, we show that this change alone

---

[*]Equal contribution [1]School of Computer Science, McGill University, Montreal, Canada [2]Mila, Quebec AI Institute [3]Intel Labs. Correspondence to: Daniel Levy <daniel.levy@mila.quebec>, Sékou-Oumar Kaba <kabaseko@mila.quebec>.

(a) Full solar system (b) Close-up of the inner solar system (c) Close-up on 4 moons of Saturn
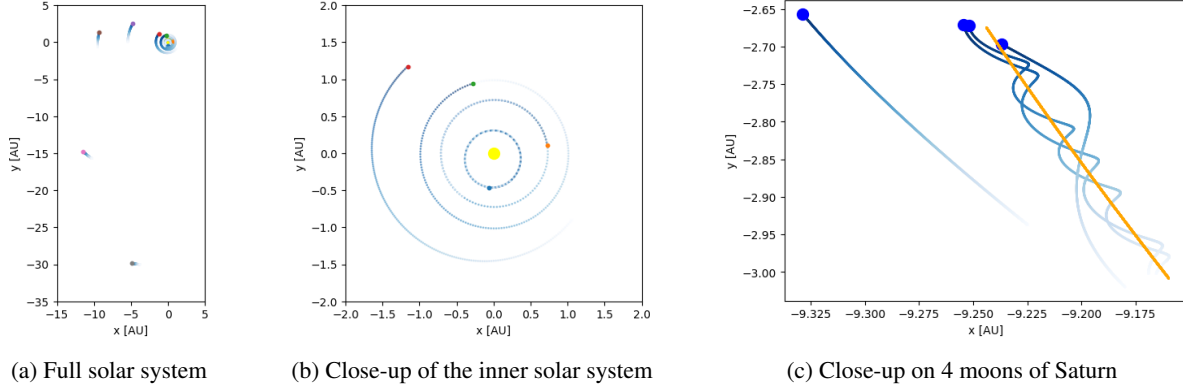
Figure 1: Visualization of the Solar Systems dynamics dataset. Solid circles represent initial positions of celestial bodies and traces their training trajectories.

leads to an important increase in performance for some physical modelling tasks. This multi-channel extension also retains the simplicity and computational efficiency of the original architecture and makes intuitive physical sense: the network may use the different channels to store additional physical quantities relevant to the prediction task.

We note that GMN (Huang et al., 2022) proposes to use multiple channels as part of a generalized EGNN-like model. However, since this is one contribution amongst several others in GMN, the advantage of using multiple channels is not clear. Here we show that we can obtain significant benefits only with the additional channels. In this short paper, we highlight that simply adding multiple channels to EGNN can lead to a significant performance increase compared to much more expensive methods such as SEGNN. We believe this result should be of use to practitioners looking to preserve the advantages of EGNN.

2. Method

Following the formulation of EGNN, we assume that the model operates on graphs embedded in n-dimensional Euclidean space (typically $n = 3$). To each node is associated coordinates $x_i \in \mathbb{R}^n$ and node features $h_i \in \mathbb{R}^{d_h}$. Edge features $a_{i,j} \in \mathbb{R}^{d_e}$ can also be considered. The original EGNN layer is defined by the following equations:

$$x_{ij} = x_i - x_j \tag{1}$$
$$m_{ij} = \phi_e\left(h_i, h_j, ||x_{ij}||^2, a_{ij}\right) \tag{2}$$
$$x_i^{t+1} = x_i^t + C \sum_{j \in \mathcal{N}(i)} x_{ij}^t \phi_x(m_{ij}) \tag{3}$$
$$h_i^{t+1} = \phi_h\left(h_i^t, \sum_{j \in \mathcal{N}(i)} m_{ij}\right) \tag{4}$$

where $\phi_e : \mathbb{R}^{d_h + d_h + 1 + d_e} \to \mathbb{R}^d$, $\phi_x : \mathbb{R}^d \to \mathbb{R}$ and $\phi_h : \mathbb{R}^{d_h + d} \to \mathbb{R}^{d_h}$ are multilayer perceptrons (MLPs)

and $\mathcal{N}(i)$ is the neighborhood of node i.

We define the Multi-Channel $E(n)$-Equivariant Graph Neural Network (MC-EGNN) by replacing x_i with the matrix $X_i \in \mathbb{R}^{3 \times m}$, where m is the number of vector channels. Using this, we modify the above equations as follows:

$$X_{ij} = X_i - X_j \tag{5}$$
$$m_{ij} = \phi_e\left(h_i, h_j, ||X_{ij}||_c^2, a_{ij}\right) \tag{6}$$
$$X_i^{t+1} = X_i^t + C \sum_{j \in \mathcal{N}(i)} X_{ij}^t \Phi_x(m_{ij}) \tag{7}$$

for MLPs $\phi_e : \mathbb{R}^{d_h + d_h + m + d_e} \to \mathbb{R}^d$ and $\Phi_x : \mathbb{R}^d \to \mathbb{R}^{m \times m'}$, where m' is the output channel dimension. $||X_{ij}||_c$ denotes the channel-wise Euclidean norm. Equation 4 stays the same. We set $m = 1$ for the first and last layer to use a single vector for each node's inputs and outputs. The modification to EGNN does not affect the equivariance properties of the architecture since the Euclidean and permutation groups do not act over the channels dimension.

3. Experiments

3.1. Solar System Predictions

3.1.1. DATASET

To investigate the necessity of different number of vector channels, we perform experiments on prediction of dynamics on an N-body system based on the solar system. We look at a dataset of 30 years of real observations of the 31 highest-mass bodies in the solar system (including the sun, 8 planets, and 22 moons) curated by Lemos et al. (2022) and sourced originally from NASA Horizons [1]. We set up the task of simultaneously predicting the positions of each of the bodies 60 Earth-days into the future, given each of

---

[1]https://ssd.jpl.nasa.gov/horizons/

their positions, velocities, and log-masses.

We constructed a train/validation/test split using three subsequent years of data, with one year of data for each partition. We provide a visualization of sample training set trajectories at Figure 1. We trained our models on a fully connected graph of the solar system bodies using a mean squared error loss normalized by the true distance between the initial and final positions of each bodies. This is done to account for the broad ranges in velocities in the system.
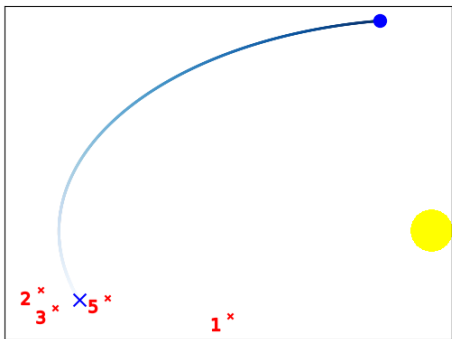

Figure 2: Predictions of the position of Venus for different numbers of vectors channels. The cross indicates the ground truth position.

3.1.2. RESULTS

Most of the predictions made in this task involve moons orbiting planets while also orbiting the sun, and so we hypothesized that 3 vectors would be needed to approximate their dynamics efficiently: to keep track of the coordinates, the angular momentum around the sun, and the angular momentum around the planet. Note that velocity is already considered since we use a variant of the multichannel EGNN described in Appendix A.

We first conducted a hyperparameter search using a one vector-channel EGNN to maximize its performance on the validation set, and then used those hyperparameters when testing the EGNN models with 2, 3, and 5 vector channels.

Our results, shown in Table 1, validate our hypothesis. While using 2 vector channels improves over using 1, it takes 3 vector channels for the model to achieve its highest performance. Note that the difference in performance between using 3 and 5 channels is not statistically significant, it is therefore not crucial to tune this parameter to an exact value. Figure 2, shows clearly that original single channel EGNN model is not able to provide an accurate estimate of the future and is widely off the trajectory.

Table 1: Performance on the solar system prediction task using differing number of vector channels. Performance is shown averaged across all 31 solar system bodies, and normalized by true distance.

# of Vectors	Normalized MSE
1	0.109 ± 0.051
2	0.082 ± 0.047
3	**0.024 ± 0.007**
5	0.030 ± 0.008

3.2. Charged Particles System

We also compare againt other models using a widely used benchmark. In the Charged Particles N-body experiment (Kipf et al., 2018), the task is to predict the positions of charged particles several timesteps into the future, given their charges, positions, and velocities.

We use the variant created by (Satorras et al., 2021), which consists of 3,000 training samples, each consisting of a system of 5 particles with their charges and 3d coordinates and velocities given, and we train our network on a loss of the mean squared error of the particle's position after 1000 timesteps. We also use the velocity version of the multichannel EGNN in this experiment. For this experiment, we use the implementation and hyperparameters of the EGNN used by (Satorras et al., 2021). The only modification we make is to incorporate multiple vector channels. Specific architecture details and hyperparameters are listed in Appendix C.1.

Table 2: Test set MSE for the N-body experiment. Our results are averaged over 5 random seeds.

Method	MSE
SE(3)-Tr (Fuchs et al., 2020)	0.0244
TFN (Thomas et al., 2018)	0.0155
NMP (Gilmer et al., 2017)	0.107
Radial Field (Köhler et al., 2019)	0.0104
CN-GNN (Kaba et al., 2022)	0.0043 ± 0.0001
FA-GNN (Puny et al., 2022)	0.0057 ± 0.0002
SEGNN (Brandstetter et al., 2022)	0.0043 ± 0.0002
EGNN (Satorras et al., 2021)	0.0070 ± 0.0005
MC-EGNN-2	**0.0041 ± 0.0006**
MC-EGNN-5	0.0043 ± 0.0003
MC-EGNN-10	0.0044 ± 0.0005
MC-EGNN-25	0.0048 ± 0.0005

The results, shown in Table 2, demonstrate that using just 1 more vector channel (MC-EGNN-2) yields greatly improves performance over the single-vector EGNN and matches the performance of more sophisticated models such as SEGNN (Brandstetter et al., 2022). This is at a

negligible added cost: Table 4 (Appendix B.1) shows that for a small number of vector channels, the forward time and the model's number of parameters are largely unaffected.

3.3. QM9 – Molecular Property Prediction

Lastly, we applied the EGNN the task of predicting chemical properties of small molecules. The QM9 dataset consists of 100,000 training samples of molecules, each described by the atom type and positions of their constituent atoms (Ramakrishnan et al., 2014). We use the same training setup as EGNN which facilitates comparison. The hyperparameters are detailed in Appendix C.2.

We predict 12 chemical properties using the multi-channel EGNN. In theory, the properties are entirely determined by the atom types and positions. Unlike in the N-body experiment or the solar system experiment, the predicted properties are coordinate-invariant. The output is therefore obtained by pooling the invariant node embeddings. The vector channels are only used in the intermediate layers, but as we report hereafter, they still contribute to increased performance on all targets compared to the standard EGNN. The results, shown in Table 3, demonstrate that performance is also comparable to SEGNN when using 8 vector channels.

4. Conclusion

We show here that adding multiple channels to the EGNN model leads to performance improvements in prediction tasks on physical systems, sometimes matching more complicated architectures. This is achieved without a significant increase in the forward runtime of the model because only a small number of vector channels are needed to obtain improvements.

This generalization could also be useful for tasks where the number of input vectors attached to each node or to predict is arbitrary (for example, when quantities such as angular velocity, spin, or polarization are included). Translationally invariant predictions can also be produced simply by removing the residual connection in the positions update equation.

We plan to investigate further whether there is a particular semantic meaning to the different vectors computed by the multi-channel model that makes it helpful. One possible downside we noticed with the multi-channel model was that training could be less stable when more vector channels were used. In practice, we found that gradient clipping could be used to help address this issue, but this was not used in our experiments. Analyzing the learned vectors could lead to better architecture design and hyperparameter selection.

We further plan to apply this model to larger datasets, such

as predictions on dark matter halos, where we may have many more interactions than in any of the experiments tested in this paper. We believe that the relative computational efficiency of the method proposed here may allow it to prove useful in applications that were previously unattainable for $E(n)$ equivariant neural networks.

References

Anderson, B., Hy, T.-S., and Kondor, R. Cormorant: Covariant molecular neural networks. *arXiv preprint arXiv:1906.04015*, 2019.

Brandstetter, J., Hesselink, R., van der Pol, E., Bekkers, E. J., and Welling, M. Geometric and physical quantities improve e(3) equivariant message passing. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=_xwr8gOBeV1.

Chanussot, L., Das, A., Goyal, S., Lavril, T., Shuaibi, M., Riviere, M., Tran, K., Heras-Domingo, J., Ho, C., Hu, W., et al. Open catalyst 2020 (oc20) dataset and community challenges. *Acs Catalysis*, 11(10):6059–6072, 2021.

Chen, C., Ye, W., Zuo, Y., Zheng, C., and Ong, S. P. Graph networks as a universal machine learning framework for molecules and crystals. *Chemistry of Materials*, 31(9):3564–3572, 2019. doi: 10.1021/acs. chemmater.9b01294. URL https://doi.org/10. 1021/acs.chemmater.9b01294.

Deng, C., Litany, O., Duan, Y., Poulenard, A., Tagliasacchi, A., and Guibas, L. J. Vector neurons: A general framework for so (3)-equivariant networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12200–12209, 2021.

Finkelshtein, B., Baskin, C., Maron, H., and Dym, N. A simple and universal rotation equivariant point-cloud network. In *Topological, Algebraic and Geometric Learning Workshops 2022*, pp. 107–115. PMLR, 2022.

Finzi, M., Stanton, S., Izmailov, P., and Wilson, A. G. Generalizing convolutional neural networks for equivariance to lie groups on arbitrary continuous data. In *International Conference on Machine Learning*, pp. 3165–3176. PMLR, 2020.

Finzi, M., Welling, M., and Wilson, A. G. A practical method for constructing equivariant multilayer perceptrons for arbitrary matrix groups. *arXiv preprint arXiv:2104.09459*, 2021.

Fuchs, F. B., Worrall, D. E., Fischer, V., and Welling, M. Se(3)-transformers: 3d roto-translation equivariant at-

Table 3: Mean absolute error for property prediction on QM9.

Task	α	$\Delta\varepsilon$	$\varepsilon_{\text{HOMO}}$	$\varepsilon_{\text{LUMO}}$	μ	C_ν	G	H	R^2	U	U_0	ZPVE
Units	bohr^3	meV	meV	meV	D	cal/mol K	meV	meV	bohr^3	meV	meV	meV
NMP (Gilmer et al., 2017)	.092	69	43	38	.030	.040	19	17	.180	20	20	1.500
SchNet (Schütt et al., 2017) *	.235	63	41	34	.033	.033	14	14	.073	19	14	1.700
Cormorant (Anderson et al., 2019)	.085	61	34	38	.038	.026	20	21	.961	21	22	2.027
L1Net (Miller et al., 2020)	.088	68	46	35	.043	.031	14	14	.354	14	13	1.561
LieConv (Finzi et al., 2020)	.084	49	30	25	.032	.038	22	24	.800	19	19	2.280
TFN (Thomas et al., 2018)	.223	58	40	38	.064	.101	-	-	-	-	-	-
SE(3)-Tr. (Fuchs et al., 2020)	.142	53	35	33	.051	.054	-	-	-	-	-	-
DimeNet++ (Klicpera et al., 2020) *	.0435	32.6	24.6	19.5	.0297	.0230	7.56	6.53	.331	6.28	6.32	1.21
SphereNet (Liu et al., 2022) *	.0465	**32.3**	**23.6**	**18.9**	.0269	**.0215**	8.0	6.40	.292	7.33	6.26	**1.12**
PaiNN (Schütt et al., 2021) *	**.045**	45.7	27.6	20.4	**.012**	.024	**7.35**	**5.98**	**.066**	**5.83**	**5.85**	1.28
SEGNN (Brandstetter et al., 2022)	.060	42	24	21	.023	.031	15	16	.66	13	15	1.62
EGNN (Satorras et al., 2021)	.071	48	29	25	.029	.031	12	12	.106	12	12	1.55
MC-EGNN-8	.063	42	24	23	.016	.028	10	11	.101	11	10	1.45

* these methods use different train/val/test partitions.

tention networks. In *Advances in Neural Information Processing Systems 34 (NeurIPS)*, 2020.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1263–1272, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. URL `http://proceedings.mlr.press/v70/gilmer17a.html`.

Huang, W., Han, J., Rong, Y., Xu, T., Sun, F., and Huang, J. Equivariant graph mechanics networks with constraints. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=SHbhHHfePhP`.

Kaba, S.-O., Mondal, A. K., Zhang, Y., Bengio, Y., and Ravanbakhsh, S. Equivariance with learned canonicalization functions. *arXiv preprint arXiv:2211.06489*, 2022.

Kipf, T., Fetaya, E., Wang, K.-C., Welling, M., and Zemel, R. Neural relational inference for interacting systems. In *International Conference on Machine Learning*, pp. 2688–2697. PMLR, 2018.

Klicpera, J., Giri, S., Margraf, J. T., and Günnemann, S. Fast and uncertainty-aware directional message passing for non-equilibrium molecules. *arXiv preprint arXiv:2011.14115*, 2020.

Köhler, J., Klein, L., and Noé, F. Equivariant flows: sampling configurations for multi-body systems with symmetric energies. *arXiv preprint arXiv:1910.00753*, 2019.

Lemos, P., Jeffrey, N., Cranmer, M., Ho, S., and Battaglia, P. Rediscovering orbital mechanics with machine learning. *arXiv preprint arXiv:2202.02306*, 2022.

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

Miller, B. K., Geiger, M., Smidt, T. E., and Noé, F. Relevance of rotationally equivariant convolutions for predicting molecular properties. *arXiv preprint arXiv:2008.08461*, 2020.

Miret, S., Lee, K. L. K., Gonzales, C., Nassar, M., and Spellings, M. The open matsci ml toolkit: A flexible framework for machine learning in materials science. *arXiv preprint arXiv:2210.17484*, 2022.

Puny, O., Atzmon, M., Smith, E. J., Misra, I., Grover, A., Ben-Hamu, H., and Lipman, Y. Frame averaging for invariant and equivariant network design. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=zIUyj55nXR`.

Ramakrishnan, R., Dral, P. O., Rupp, M., and Von Lilienfeld, O. A. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific data*, 1(1):1–7, 2014.

Satorras, V. G., Hoogeboom, E., and Welling, M. E (n) equivariant graph neural networks. *arXiv preprint arXiv:2102.09844*, 2021.

Schütt, K., Kindermans, P.-J., Felix, H. E. S., Chmiela, S., Tkatchenko, A., and Müller, K.-R. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. In *Advances in Neural Information Processing Systems*, pp. 991–1001, 2017.

Schütt, K., Unke, O., and Gastegger, M. Equivariant message passing for the prediction of tensorial properties and molecular spectra. In *International Conference on Machine Learning*, pp. 9377–9388. PMLR, 2021.

Spellings, M. Geometric algebra attention networks for small point clouds. *arXiv preprint arXiv:2110.02393*, 2021.

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

A. Velocity Variant

For problems in which we are provided with node velocities, a similar reformulation of the model to (Satorras et al., 2021) is used. This is done simply by replacing Equation 7 with the following:

$$V_{(i)}^{t+1} = v_{(i)}^0 \phi_v(h_{(i)}^t) + C \sum_{j \neq i} X_{(ij)} \Phi_x(m_{(ij)}) \quad (8)$$

$$X_{(i)}^{t+1} = X_{(i)}^t + V_{(i)}^{t+1} \quad (9)$$

Where $\phi_v \in \mathbb{R}^d \to \mathbb{R}^{m'}$ is an MLP that weighs the impact of the initial velocity vector on the position update.

B. Additional Results

B.1. Comparison of forward time and parameters

We show hereafter a comparison of the forward time of the model as well as the number of parameters for different values of number of vectors.

Table 4: Comparison of the forward time (for a batch of 100 samples) and number of parameters of EGNN models with different numbers of vector channels used for the N-body experiment.

# of Vectors	Forward Time (s)	# of Parameters
1 (EGNN)	0.003118	134020
2	0.003229	135244
5	0.003135	142012
10	0.003405	163612
25	0.004188	305812

B.2. Open Catalyst Project – IS2RE

The Open Catalyst (OC20) Dataset is a large-scale dataset designed for applying machine learning to the task of catalyst discovery, of particular interest to the materials science community (Chanussot et al., 2021). The dataset contains over 1.2 million simulated molecular relaxations involving different adsorbates and catalytic surfaces. One challenge associated with the dataset is the Initial Structure to Relaxed Energy (IS2RE) task, wherein a neural network is trained to predict the relaxation energy of a system described by the initial positions and atom types of a molecule, a catalytic surface, and a bulk crystal substrate below the surface.

We conducted experiments using the Open MatSci ML Toolkit (Miret et al., 2022), which provides an interface into OC20 tasks while abstracting away implementation complexities. The toolkit also contains an implementation of EGNN, which we modified to include our multi-channel

extension. We used a training set of ∼500K training samples and ∼25K validation samples and trained on the mean absolute error (MAE) of the relaxed energy for a maximum of 100 epochs, with early stopping.

Table 5: In-Distribution Validation set MAE for the IS2RE task. EGNN variants was run across 5 random seeds, but due to instabilities in training, seeds where training failed were discarded. MegNet and Gala results are taken from (Miret et al., 2022)

Model	Validation MAE (eV)
MegNet (Chen et al., 2019)	0.233
Gala (Spellings, 2021)	0.240
EGNN	0.239 ± 0.003
MC-EGNN-2	0.262 ± 0.017
MC-EGNN-5	0.239 ± 0.006
MC-EGNN-10	0.253 ± 0.019

We report a negative result on the IS2RE task as shown in Table 5. We do not observe any clear benefit with using multiple vectors, with different resulting performances being mostly determined by the random seed used.

C. Experimental details

C.1. N-Body System

We used the same architecture and hyperparameters as EGNN (Satorras et al., 2021) in their N-Body system experiment. We used 4 EGNN layers. Each layer used 64 channel, 2 layer MLPs for the node, edge, and coordinate update functions, with a Swish function nonlinearity. We trained on 3000 training samples for 10,000 epochs with a batch size of 100, using the Adam optimizer with a learning rate of 5×10^{-4}.

C.2. QM9

We used the same architecture and hyperparameters as EGNN (Satorras et al., 2021) in their QM9 experiments. Training was performed using the Adam optimizer with learning rate 5×10^{-4} for all targets except the gap, homo and lumo. We used a cosine learning rate scheduler. Batch size was set at 96 and number of epochs to 1000 with early stopping. We used 8 vector channels and 128 channels for node features and messages. The number of layers was set at 7.

C.3. Solar System

We again used the same architecture as EGNN. We conducted a hyperparameter search using a baseline 1-vector EGNN, looking at layers between 3 to 6, and number of

features between $[64, 128, 256, 512]$. We then used a 5-layer EGNN with 128-channels for node features and messages, the Adam optimizer with a learning rate of 1×10^{-4} for all experiments with different numbers of vectors. We trained for 1000 epochs with early stopping.

C.4. Open Catalyst Project

We used the same EGNN implementation, hyperparameters, and optimization procedure as (Miret et al., 2022). We used a 3-layer EGNN with 2-layer MLPs for the node, edge, and coordinate update functions, that used 48, 16, and 64 channels respectively. Node features were first embedded using a 3-layer, 64-channel MLP, and a 3-layer, 128-channel MLP was used to predict relaxed energy using the EGNN's pooled node embeddings. Training was performed over a maximum of 100 epochs with a with early stopping, with a batch size of 8. The Adam optimizer was used with a learning rate of 1e-5, a gamma of 0.5, and a cosine annealing learning rate scheduler.