FORCENET: A GRAPH NEURAL NETWORK FOR LARGE-SCALE QUANTUM CHEMISTRY SIMULATION

Weihua Hu[†], **Muhammed Shuaibi**[‡], **Abhishek Das**[♠], **Siddharth Goyal**[♠], **Anuroop Sriram**[♠], **Jure Leskovec**[†], **Devi Parikh**[♠§], **C. Lawrence Zitnick**[♠]
[†] Department of Computer Science, Stanford University
[‡] Department of Chemical Engineering, Carnegie Mellon University
[♠] Facebook AI Research
[§] School of Interactive Computing, Georgia Institute of Technology
weihuahu@cs.stanford.edu, zitnick@fb.com

ABSTRACT

With massive amounts of atomic simulation data available, there is a huge opportunity to develop fast and accurate machine learning models to approximate expensive physics-based calculations. The key quantity to estimate is atomic forces, where the state-of-the-art Graph Neural Networks (GNNs) explicitly enforce basic physical constraints such as rotation-covariance. However, to strictly satisfy the physical constraints, existing models have to make tradeoffs between computational efficiency and model expressiveness. Here we explore an alternative approach. By not imposing explicit physical constraints, we can flexibly design expressive models while maintaining their computational efficiency. Physical constraints are implicitly imposed by training the models using physics-based data augmentation. To evaluate the approach, we carefully design a scalable and expressive GNN model, ForceNet, and apply it to OC20 (Chanussot et al., 2020), an unprecedentedly-large dataset of quantum physics calculations. Our proposed ForceNet is able to predict atomic forces more accurately than state-of-the-art physics-based GNNs while being faster both in training and inference. Overall, our promising results open up an exciting avenue for future research.[1]

1 INTRODUCTION

Recently, massive physics-based data has been generated by ever-increasing scientific compute (Chanussot et al., 2020; Nakata et al., 2019). This provides a huge opportunity for Machine Learning (ML) approaches to efficiently and accurately model complex physical systems (Sanchez-Gonzalez et al., 2020; Bapst et al., 2020; Kipf et al., 2018; Klicpera et al., 2020a; Battaglia et al., 2016; Gilmer et al., 2017; Schütt et al., 2017a; Klicpera et al., 2020b). Of particular practical interest is approximating atomic forces of quantum mechanical systems. This is because the underlying quantum calculations are expensive (several hours per system) (Parr, 1980), and the resulting atomic forces can be used for diverse chemistry applications, such as structure relaxations, molecular dynamics, structural analyses, as well as transition state calculations. Behler (2016); del Río et al. (2019); Frederiksen et al. (2007); Henkelman & Jónsson (2000); Henkelman et al. (2000).

The state-of-the-art approach to predicting atomic forces is physics-based message-passing Graph Neural Networks (GNNs) (Gilmer et al., 2017), with the representative models being SchNet (Schütt et al., 2017a) and DimeNet (Klicpera et al., 2020a;b). These GNNs first predict the energy of the entire system in a rotation-invariant manner, and then predict the per-atom forces by taking the derivative of the energy with respect to the atomic positions. By the architecture's design, these GNNs produce forces that obey the basic physical rules of rotation-covariance and energy-conservation.

However, designing effective GNNs, while satisfying these physical rules is highly non-trivial. For instance, SchNet (Schütt et al., 2017b) is computationally efficient, but the model only uses atomic distances in its message passing in order to ensure rotation-invariance of its energy prediction. Consequently, SchNet fails to capture the 3D structure explicitly, resulting in sub-optimal generalization

[1] The full version available at https://arxiv.org/abs/2103.01436

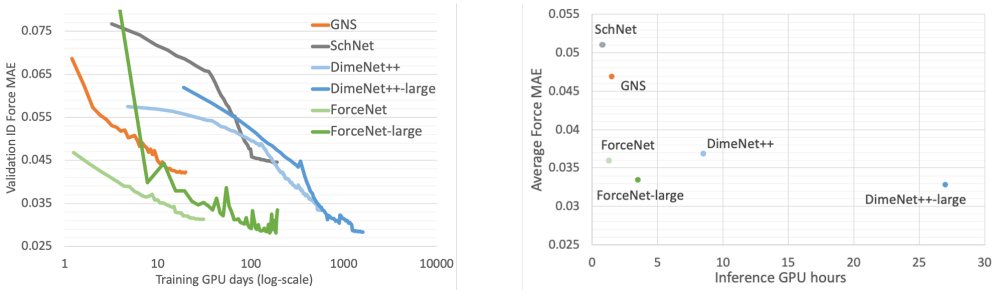

Figure 1: Comparison of S2F (atomic force prediction) performance across different models, while taking computational efficiency into account. **(Left)**: Comparison of validation learning curves, where x-axis is training GPU days in the *log-scale* (lower left is better). **(Right)**: Comparison of validation performance and inference GPU hours (lower left is better; inference time is measured over the in-distribution validation set). GPUs with the same specs are used for fair comparison.

performance. The recent DimeNet and DimeNet++ (Klicpera et al., 2020b;a) additionally capture bond angle information in its message passing, but this comes with the cost of expensive message computations involving atom triplets to ensure the rotation-invariance. As a result, DimeNet necessitates tremendous compute to scale to a massive dataset (Figure 1 (left))—1600 GPU days to train DimeNet++-large.

Here we explore an alternative approach, building on the recent framework of Graph Network-based Simulators (GNS) (Sanchez-Gonzalez et al., 2020; Bapst et al., 2020). Specifically, by not explicitly imposing physical constraints in the model architecture, we can flexibly design expressive GNN models and use the full 3D atomic positions in a scalable manner. In exchange, the predicted forces are translation-invariant but no longer rotation-covariant. As we demonstrate empirically, this issue can be alleviated by training models on a massive dataset with rotation data augmentation. In other words, we impose physical constraints to the model *implicitly through physics-based data* rather than explicitly through architectural constraints. Our model also does not explicitly enforce energy conservation. However, this removes the memory-intensive calculations in physics-based GNNs, *i.e.*, compute forces through energy gradients that require second-order derivatives to optimize.

To realize our approach, we carefully design a GNN model that accurately captures 3D atomic structure in a scalable and flexible manner. The resulting model is ForceNet that uses an expressive message passing architecture with carefully-chosen basis and non-linear activation functions.

We evaluate ForceNet on OC20 Chanussot et al. (2020), a recently-introduced large-scale dataset of quantum physics calculations with 200+ million large atomic structures (20–200 atoms) useful for discovering new catalysts for energy applications Seh et al. (2017); Jouny et al. (2018); Zitnick et al. (2020) (Figure 2). Even without any explicit physical constraints, ForceNet is able to achieve higher accuracy than physics-based GNNs when trained with comparable computational resources (Figure 1 (left)). Moreover, ForceNet (resp. ForceNet-large) achieves prediction errors that are comparable to the state-of-the-art DimeNet++ (resp. DimeNet++-large) with 6 (resp. 8) times less inference time (Figure 1 (right)). Finally, compared to DimeNet++, ForceNet-large achieves more accurate force prediction, while being faster both in training and inference (Figure 1 (left) and (right)).

2 FORCENET

ForceNet follows the encoder-decoder architecture of the GNS framework (Sanchez-Gonzalez et al., 2020; Battaglia et al., 2016; Kipf et al., 2018). The input to ForceNet is an atomic structure, *i.e.*, a set of atoms and their 3D spatial positions (Figure 2). Given the input, a radius graph is first constructed, *i.e.*, each atom's neighborhood is defined as atoms that are within the cutoff distance c. We set c to be 6 Angstrom, resulting in the average of 35 neighbors per atom. After the graph is obtained, the encoder uses scalable iterative message passing to compute node embeddings $\boldsymbol{h}_t$ that capture the 3D structure surrounding each atom, and the decoder uses an Multi-Layer Perceptron (MLP) to directly predict per-atom forces from these embeddings.

The critical aspect of ForceNet is its encoder and specifically the scalable message computation that effectively captures the non-linear and complex 3D atomic interactions to predict the atomic forces.

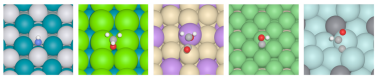

Figure 2: Illustration of sampled systems from the OC20 dataset (Chanussot et al., 2020). Each system consists of adsorbate (the small molecule on the surface) and catalysis (the large grid-like molecule sitting below the adsorbate), and is repeated in the direction of the horizontal axes infinitely. Our ForceNet aims to efficiently predict per-atom forces.

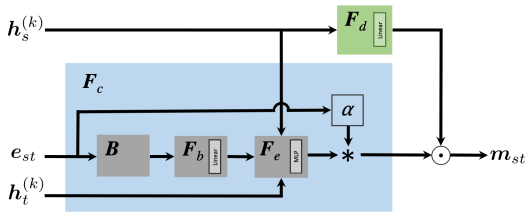

Figure 3: Model diagram for messages $\boldsymbol{m}_{st}$ (from atom s to atom t) used by ForceNet in Eqns. (2). The key components are **(a)** the expressive conditional filter $\boldsymbol{F}_c$ that is dependent on full edge feature $\boldsymbol{e}_{st}$ as well as source and target node embeddings, **(b)** the basis function $\boldsymbol{B}$ over the edge feature that helps the network to accurately capture atomic interactions, and **(c)** the smooth curved non-linearity of the Swish activation.

The encoder updates $\boldsymbol{h}_t$ as:

$$\boldsymbol{h}_t^{(k+1)} = \boldsymbol{F}_n\left(\boldsymbol{m}_t + \sum_{s\in\mathcal{N}_t} \boldsymbol{m}_{st}\right) + \boldsymbol{h}_t^{(k)}, \tag{1}$$

where the messages $\boldsymbol{m}_{st}$ and $\boldsymbol{m}_t$ are summed and passed through the function $\boldsymbol{F}_n : \mathbb{R}^D \to \mathbb{R}^D$ that is a 1-hidden-layer MLP with batch normalization (Ioffe & Szegedy, 2015). The dimensionality of the node and hidden layer features is D. Equation (1) follows standard GNN embedding update formulations (Gilmer et al., 2017) with the addition of a residual connection, $\boldsymbol{h}_t^{(k)}$ (He et al., 2016). There are three key components in our message modeling, which are illustrated in Figure 3 and described below. These designs are then extensively ablated in Appendix F.

(a) Conditional Filter Convolution. To compute the inter-atomic messages $\boldsymbol{m}_{st}$, we start from SchNet's continuous filter convolution (Schütt et al., 2017a), and make two important modifications. First, we use the edge feature $\boldsymbol{e}_{st}$ (described below) that encodes rotation-covariant directional information (not just the rotation-invariant distance). This way, the predicted forces can change as the system is rotated. Second, we condition the message on both the source $\boldsymbol{h}_s^{(k)}$ and target $\boldsymbol{h}_t^{(k)}$ node information. This allows ForceNet to capture complex interactions between different atom types. Including the two modifications, we model the inter-atomic message as:

$$\boldsymbol{m}_{st} = \alpha(\|\boldsymbol{d}_{st}\|) \cdot \boldsymbol{F}_e\left(\boldsymbol{h}_s^{(k)}, \boldsymbol{F}_b(\boldsymbol{B}\left(\boldsymbol{e}_{st}\right)), \boldsymbol{h}_t^{(k)}\right) \odot \boldsymbol{F}_d(\boldsymbol{h}_s^{(k)}), \tag{2}$$

where all $\boldsymbol{F}_*(\cdot)$ functions are learnable and are either linear functions or MLPs (see Figure 3). The edge feature is defined as $\boldsymbol{e}_{st} \equiv \text{Concat}(\boldsymbol{n}_{st}, \boldsymbol{p}_{st}/c) \in \mathbb{R}^7$, where $\boldsymbol{n}_{st} \equiv \boldsymbol{d}_{st}/\|\boldsymbol{d}_{st}\|$ is a normalized directional vector and $\boldsymbol{p}_{st} \in \mathbb{R}^4$ is a list of four atomic distances $\|\boldsymbol{d}_{st}\|, \|\boldsymbol{d}_{st}\| - a_s, \|\boldsymbol{d}_{st}\| - a_t, \|\boldsymbol{d}_{st}\| - a_s - a_t$ that take into account the atomic radii (Slater, 1964) a_s and a_t of atoms s and t, respectively. $\alpha(x) = \cos(\pi x/2c)$ is a scalar that decays to zero as $\|\boldsymbol{d}_{st}\|$ approaches the distance cutoff c.

(b) Basis function. To accurately predict complex atomic forces, it is crucial to capture the subtle change in input 3D structure, which is reflected in subtle change in the edge feature $\boldsymbol{e}_{st}$. However, the raw edge feature is rather low-dimensional (7-dimensional), making it hard for neural networks to capture its subtle change. To resolve this issue, an important aspect of message modeling in Eqn. 2 is the choice of basis function $\boldsymbol{B} : \mathbb{R}^7 \to \mathbb{R}^B$ that transforms the raw distance features $\boldsymbol{e}_{st}$ into ones that are more high-dimensional ($7 \ll B$) and discriminative. Several choices of basis functions have been proposed, such as a Gaussian over 1D distances (Schütt et al., 2017a) and a spherical Bessel function over the joint 2D space of the edge distance and angle (Klicpera et al., 2020b). We extend these ideas to capture the full 3D positional differences between atoms, and systematically study the effectiveness of 5 different choices of basis functions in the context of a force-centric model. Due to space constraint, we provide the detailed descriptions of the basis functions in Appendix A.

(c) Expressive Non-Linearity in MLPs. Our final key component is simple but crucial: the choice of non-linear activation function plays a central role in modeling complex non-linearities of atomic

intereactions. The ReLU activation (Glorot et al., 2011) is widely used in many deep learning models, including the existing GNS model (Sanchez-Gonzalez et al., 2020; Bapst et al., 2020). However, ReLU may not be ideal in modeling atomic forces, since it results in outputs being modeled as piece-wise linear hyper-planes with sharp boundaries. Ideally, we desire a smooth and expressive non-linear activation function. Instead of ReLU, we propose to use the Swish activation (Ramachandran et al., 2017). Swish provides a smoother output landscape and has non-zero activation for negative inputs. The replacement of ReLU with Swish consistently and significantly improves the predictive accuracy while maintaining scalability across all choices of basis functions.

Rotation data augmentation. The prediction of ForceNet is not necessarily rotation-covariant. To encourage the rotation-covariance, we apply random rotation data augmentation along the axis that is vertical to the material surface (see (Figure 2)). Specifically, we randomly rotate the entire system and per-atom forces by the same degree, and let ForceNet predict the rotated forces based on the rotated system.

3 EXPERIMENTS

In this section, we evaluate ForceNet's performance in predicting atomic forces. We do so by applying the model to OC20 (Chanussot et al., 2020), a massive dataset on quantum physics calculations on non-equilibrium atomic structures relevant to catalysis discovery. Following Chanussot et al. (2020), the prediction error is measured by Mean Absolute Error (MAE). We normalize for computational time when comparing models' predictive performance, *i.e.*, we compare models with similar computational cost in training and inference. This is crucial because simply using more computational resources to train larger models is shown to lead better results in OC20 tasks (Chanussot et al., 2020). However, training time of most existing models is already more than 100 GPU days (defined as the number of GPUs times the number of days the GPUs are used). and even goes up to 1600 GPU days, making it harder to further scale up without improving the models' computational efficiency. Moreover, fast model inference is crucial for the application of catalyst material discovery, where an ML model needs to make predictions over an enormous number of potential candidates (Zitnick et al., 2020).

As baseline models, we use the force-centric model proposed in the GNS work (Sanchez-Gonzalez et al., 2020), as well as the energy-centric SchNet (Schütt et al., 2017a) and DimeNet++ (Klicpera et al., 2020a). $*$-large denotes that the model size is larger. The detailed model settings can be found in Appendix C.

Table 1: Analysis of how training data and rotation augmentation affect the stability of ForceNet's prediction against rotation. 1000 validation ID structures are sampled and randomly rotated 100 times along the vertical axis. For each rotated structure, ForceNet predicts per-atom forces that are then rotated back to compare with the originally-predicted forces. Instability is measured by the average standard deviation of the errors across the 100 rotations for each (free) atom. Smaller instability values indicate the model is closer to being rotation-covariant, where a fully rotation-covariant model would always give a value of 0.

Dataset	Rotation aug.	Average instability of per-atom force pred.	Val Force MAE ID	Val Force MAE Average
All (130M)	✔	**0.0037**	**0.0313**	**0.0360**
All (130M)		0.0069	0.0314	0.0366
2M	✔	0.0041	0.0332	0.0382
2M		0.0093	0.0346	0.0400

Results Our main results are plotted in Figure 1. In Table 2 of Appendix F, we provide complete numerical results. From Figure 1 (left), we see that ForceNet gives superior force prediction performance given limited training GPU budgets. In Figure 1 (right), we see that ForceNet achieves prediction performance comparable to DimeNet++, while enabling much faster inference speed. In Table 1, We also analyze whether ForceNet is able to learn rotation-covariance when predicting atomic forces. We do so by measuring the prediction instability of ForceNet when validation systems are rotated. We see a clear trend that both large data and rotation augmentation help to reduce the instability of ForceNet's prediction against rotation. Moreover, the instability of ForceNet trained on all data and rotation augmentation is relatively small compared to its force MAE, suggesting that ForceNet's prediction is close to be rotation covariant in the practical sense. Furthermore, in Appendix F, we perform extensive ablation studies on ForceNet's key design choices presented in Section 2. We demonstrate that each design choice is crucial in ForceNet's final performance.

4 CONCLUSIONS

In physical modeling, the conventional approach has been to explicitly enforce physical constraints into model architecture, but this presents a trade-off between model's expressiveness and computational efficiency. In this work, we explore an alternative approach: *not* enforcing physical constraints explicitly, thereby allowing expressive and scalable models to be built in a flexible way. We carefully design the ForceNet architecture and train it on a huge amount of data with physics-based data augmentation. We demonstrate that ForceNet, without any explicit physical constraints, are able to predict atomic forces more accurately than state-of-the-art energy-centric GNN models, while being faster both in training and inference. We hope our unconventional approach and the promising empirical evidence open up an exciting avenue for further research.

ACKNOWLEDGEMENTS

This work was done while Weihua Hu and Muhammed Shuaibi were at Facebook AI Research. We acknowledge Pytorch (Paszke et al., 2019) and Pytorch Geometric (Fey & Lenssen, 2019). We thank Ryotatsu Yanagimoto and Zack Ulissi for insightful discussions on physics and chemistry.

We also gratefully acknowledge the support of DARPA under Nos. N660011924033 (MCS); ARO under Nos. W911NF-16-1-0342 (MURI), W911NF-16-1-0171 (DURIP); NSF under Nos. OAC-1835598 (CINES), OAC-1934578 (HDR), CCF-1918940 (Expeditions), IIS-2030477 (RAPID); Stanford Data Science Initiative, Wu Tsai Neurosciences Institute, Chan Zuckerberg Biohub, Amazon, JPMorgan Chase, Docomo, Hitachi, JD.com, KDDI, NVIDIA, Dell, Toshiba, and UnitedHealth Group. Weihua Hu is supported by Funai Overseas Scholarship and Masason Foundation Fellowship. Jure Leskovec is a Chan Zuckerberg Biohub investigator.

REFERENCES

The atomic simulation environment—a python library for working with atoms. *Journal of Physics Condensed Matter*, 2017. ISSN 1361648X. doi: 10.1088/1361-648X/aa680e.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Victor Bapst, Thomas Keck, A Grabska-Barwińska, Craig Donner, Ekin Dogus Cubuk, SS Schoenholz, Annette Obika, AWR Nelson, Trevor Back, Demis Hassabis, et al. Unveiling the predictive power of static structure in glassy systems. *Nature Physics*, 16(4):448–454, 2020.

Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 4502–4510, 2016.

Jörg Behler. Perspective: Machine learning potentials for atomistic simulations. *The Journal of chemical physics*, 145(17):170901, 2016.

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. The Open Catalyst 2020 (oc20) dataset and community challenges. *arXiv preprint arXiv:2010.09990*, 2020.

Estefanía Garijo del Río, Jens Jørgen Mortensen, and Karsten Wedel Jacobsen. Local bayesian optimizer for atomic structures. *Physical Review B*, 100(10):104103, 2019.

Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with pytorch geometric. *arXiv preprint arXiv:1903.02428*, 2019.

Thomas Frederiksen, Magnus Paulsson, Mads Brandbyge, and Antti-Pekka Jauho. Inelastic transport theory from first principles: Methodology and application to nanoscale devices. *Physical Review B*, 75(20):205413, 2007.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International Conference on Machine Learning (ICML)*, pp. 1273–1272, 2017.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 249–256, 2010.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 315–323, 2011.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

Graeme Henkelman and Hannes Jónsson. Improved tangent estimate in the nudged elastic band method for finding minimum energy paths and saddle points. *The Journal of chemical physics*, 113(22):9978–9985, 2000.

Graeme Henkelman, Blas P Uberuaga, and Hannes Jónsson. A climbing image nudged elastic band method for finding saddle points and minimum energy paths. *The Journal of chemical physics*, 113(22):9901–9904, 2000.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, pp. 448–456, 2015.

Matthew Jouny, Wesley Luc, and Feng Jiao. High-rate electroreduction of carbon monoxide to multi-carbon products. *Nature Catalysis*, 2018. ISSN 25201158. doi: 10.1038/s41929-018-0133-2.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. Neural relational inference for interacting systems. In *International Conference on Machine Learning (ICML)*, 2018.

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

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

Georg Kresse and Jürgen Furthmüller. Efficiency of ab-initio total energy calculations for metals and semiconductors using a plane-wave basis set. *Computational Materials Science*, 6(1):15–50, 1996a. ISSN 09270256. doi: 10.1016/0927-0256(96)00008-0.

Georg Kresse and Jürgen Furthmüller. Efficient iterative schemes for ab initio total-energy calculations using a plane-wave basis set. *Physical Review B*, 54(16):11169–11186, 1996b. ISSN 0163-1829. doi: 10.1103/PhysRevB.54.11169.

Georg Kresse and Jürgen Hafner. Ab initio molecular-dynamics simulation of the liquid-metal–amorphous-semiconductor transition in germanium. *Physical Review B*, 49(20):14251–14269, 1994. ISSN 0163-1829. doi: 10.1103/PhysRevB.49.14251.

Thomas Murray MacRobert. Spherical harmonics: an elementary treatise on harmonic functions with applications. 1947.

Maho Nakata, Tomomi Shimazaki, Masatomo Hashimoto, and Toshiyuki Maeda. Pubchemqc pm6: Data sets of 221 million molecules with optimized molecular geometries and electronic properties. *Journal of Chemical Information and Modeling*, 2019.

Robert G Parr. Density functional theory of atoms and molecules. In *Horizons of quantum chemistry*, pp. 5–15. Springer, 1980.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 8024–8035, 2019.

Prajit Ramachandran, Barret Zoph, and Quoc V Le. Swish: a self-gated activation function. *arXiv preprint arXiv:1710.05941*, 7, 2017.

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

Kristof Schütt, Pieter-Jan Kindermans, Huziel Enoc Sauceda Felix, Stefan Chmiela, Alexandre Tkatchenko, and Klaus-Robert Müller. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 991–1001, 2017a.

Kristof T Schütt, Farhad Arbabzadah, Stefan Chmiela, Klaus R Müller, and Alexandre Tkatchenko. Quantum-chemical insights from deep tensor neural networks. *Nature communications*, 8:13890, 2017b.

Zhi Wei Seh, Jakob Kibsgaard, Colin F Dickens, Ib Chorkendorff, Jens K Nørskov, and Thomas F Jaramillo. Combining theory and experiment in electrocatalysis: Insights into materials design. *Science*, 355(6321), 2017. ISSN 0036-8075. doi: 10.1126/science.aad4998. URL `https://science.sciencemag.org/content/355/6321/eaad4998`.

John C Slater. Atomic radii in crystals. *The Journal of Chemical Physics*, 41(10):3199–3204, 1964.

C. L. Zitnick, L. Chanussot, A. Das, S. Goyal, J. Heras-Domingo, C. Ho, W. Hu, T. Lavril, A. Palizhati, M. Riviere, M. Shuaibi, A. Sriram, K. Tran, B. Wood, J. Yoon, D. Parikh, and Z. Ulissi. An introduction to electrocatalyst design using machine learning for renewable energy storage. *arXiv preprint arXiv:2010.09435*, 2020.

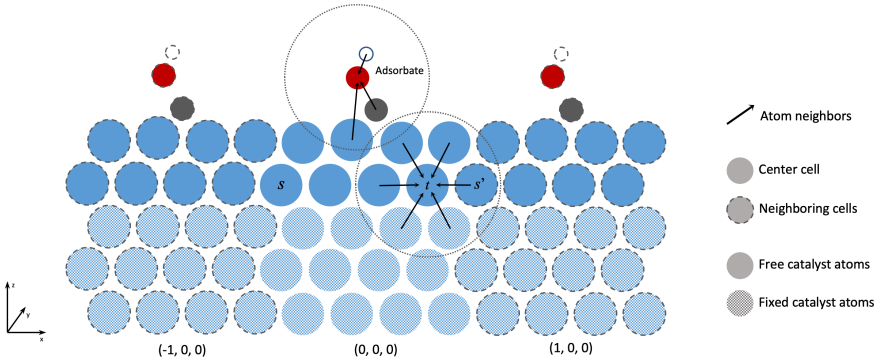

Figure 4: 2D Illustration of a slab that represents a catalyst's surface and an adsorbate. The slab is tiled in the x and y directions to create the surface (neighboring cells shown as atoms with dashed outlines). Only the cells to the left ($[-1, 0, 0]$) and right ($[1, 0, 0]$) are shown. The adsorbate is also assumed to be tiled with the slab (white, red, and grey atoms). Only the top 2 layers of the slab are allowed to move during a relaxation (dark blue), and the others are fixed (light blue). Neighboring atoms (black arrows) can be from the same cell or neighboring cells (t and s'). All atoms within a radius (dotted circle) are assumed to be neighbors.

A Basis Function Design

Here we present 5 different choices of basis functions to encode the raw edge features.

Identity: $\boldsymbol{B}_{\text{id}}(\boldsymbol{x}) = \boldsymbol{x}$. The baseline is to use the edge features $\boldsymbol{e}_{st}$ directly.

Linear + Act: $\boldsymbol{B}_{\text{linact}}(\boldsymbol{x}) = \text{g}(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b})$, where $\text{g}(\cdot)$ is the non-linear activation function, and $\boldsymbol{W}$ and $\boldsymbol{b}$ are the learnable parameters. When followed by the linear layer $\boldsymbol{F}_b$, this is equivalent to applying an 1-hidden-layer MLP over the edge features $\boldsymbol{e}_{st}$.

Gaussian: $\boldsymbol{B}_{\text{gauss}}(\boldsymbol{x}) = [b_1, \dots, b_J]$, where b_j is the output of the j-th basis function $b_j(x) = \exp^{(x-\mu_j)^2/(2\cdot\sigma^2)}$. The Gaussian means are evenly distributed on the interval between 0 and 1, *i.e.*, $\mu_j = j/(J-1)$ and the standard deviation $\sigma = 1/(J-1)$. All values of x are normalized to lie between 0 and 1, where normalization is performed globally, *e.g.*, dividing the atomic distance $\|\boldsymbol{d}_{st}\|$ by c. $\boldsymbol{B}_{\text{gauss}}$ is applied to each dimension of $\boldsymbol{e}_{st}$, resulting in a $B = J \times E$ vector.

Sine: $\boldsymbol{B}_{\text{sin}}(\boldsymbol{x}) = [b_1, \dots, b_J]$, where b_j is the output of the j-th basis function $b_j(x) = \sin(1.1^j x)$. The design is based on function approximation using the Fourier series. In our experiments, we find that using only the sinusoidal component of the Fourier series is sufficient. $\boldsymbol{B}_{\text{sin}}$ is applied to each dimension of $\boldsymbol{e}_{st}$, resulting in an $B = J \times E$ vector.

Spherical harmonics: $\boldsymbol{B}_{\text{sph}}(\boldsymbol{e}_{st}) = \boldsymbol{Y}_L(\theta, \phi)\boldsymbol{R}(\boldsymbol{p}_{st})^\top$ where $\boldsymbol{Y}_L$ is the list of Laplace's spherical harmonics (MacRobert, 1947) used to encode the angular information and $\boldsymbol{R}(\boldsymbol{p}_{st})$ encodes the distance. We use spherical harmonic functions up to degree L, which gives us L^2 orthogonal basis in total. The angles θ and ϕ can be directly computed from $\boldsymbol{n}_{st} \in \mathbb{R}^3$ in $\boldsymbol{e}_{st}$. $\boldsymbol{R}$ uses a linear combination of the above sine basis functions computed from $\boldsymbol{p}_{st} \in \mathbb{R}^4$ in $\boldsymbol{e}_{st}$ (thus, $4J$ basis functions in total) to encode distance information. Specifically, $\boldsymbol{R}(\boldsymbol{p}_{st}) = \boldsymbol{W}_{\text{rad}}\boldsymbol{B}_{\text{sin}}(\boldsymbol{p}_{st}) + \boldsymbol{b}_{\text{rad}} \in \mathbb{R}^S$, where $\boldsymbol{W}_{\text{rad}}$ and $\boldsymbol{b}_{\text{rad}}$ are learnable parameters. $\boldsymbol{B}_{\text{sph}}$ is flattened into a vector before being passed into $\boldsymbol{F}_b$. The dimensionality of $\boldsymbol{B}_{\text{sph}}$ is $B = SL^2$, where we set S to be the dimensionality of $\boldsymbol{p}_{st}$.

B Description of OC20 dataset

The OC20 dataset (Chanussot et al., 2020) contains over 130M non-equilibrium structures for training for the S2F task (*i.e.*, atomic force prediction). The structures come from over 650K relaxation trajectories—the movement of the atoms from the initial structure to relaxed structure (equilibrium 3D structures with all-zero atomic forces).

Each structure contains the 3D positions of atoms in an adsorbate and catalyst slab, Figure 4. The adsorbate is a molecule involved in the chemical reaction that interacts with the catalyst's surface. The adsorbate contains 1 to 11 atoms. The catalyst is represented as a "slab" that repeats infinitely in the x and y directions. The slab structure is repeated in a grid pattern where each repetition is referred to as a "cell". The center cell has coordinate $(0, 0, 0)$ with the cell to the left right being $(-1, 0, 0)$ and $(1, 0, 0)$ respectively. The slab is not repeated in the z direction. Instead, the atoms at the bottom of the slab are assumed to be fixed and not move during a relaxation, which approximates how they would be held in place by the catalyst's atoms below the slab. Typically, only the top two layers of the catalyst's surface are assumed to be free and are moved according to their forces during a relaxation (see Figure 4). Therefore, forces are only evaluated on free catalyst atoms and the adsorbate.

The forces on the same atom in different cells are identical, since their atom neighbors are identical, resulting in their GNN's node embeddings to be also identical, *e.g.*, the node embedding of atoms marked s and s' in Figure 4 are the same. When computing the neighborhood of an atom, atoms in neighboring cells need to be also taken into consideration (atom t in Figure 4). Notice that the message from s to t and the message from s' to t are different since the the relative placements of the two atoms are different, resulting in different edge features $\boldsymbol{e}_{st}$ and $\boldsymbol{e}_{s't}$. Computing the edge features between atoms from different cells can be done using the supplied information in the OC20 dataset for periodic boundary conditions.

C Details of Model Settings

Below, we describe hyper-parameter settings of ForceNet and baseline models, along with the computational time (in GPU days) to train these models. Table 2 shows the summary of training time, inference time, and sizes of different models. All the training is run under Tesla V100 Volta. The inference time is measured on the validation ID set under GeForce RTX 2080, where the largest possible batch size is used for each model.

ForceNet. We use the spherical function and Swish activation as the default basis and activation functions, since this combination consistently provides the best results (Figure 5). The default model size has 5-layers of message passing and 512-dimensional hidden channels, and the training batch size is set to 256. We also consider a larger variant, ForceNet-large, that uses 7-layer message passing, 768-dimensional hidden channels, and the batch size of 512. Training ForceNet and ForceNet-large takes 31 and 194 GPU days, respectively.

During training, we apply the data-efficient rotation augmentation strategy. We also find it useful to train on both free and fixed atoms, even though the evaluation is only on the free atoms. Specifically, we give a small relative weight of 0.05 to the loss of fixed atoms during training, which is ablated in Table 7 of Appendix G. Further implementation details and hyper-parameters are provided in Appendix D.

Baseline models. We compare ForceNet against the following three strong baseline GNN models.

- **SchNet** (Schütt et al., 2017a) is an energy-centric GNN that uses scalable atom-pair-based message passing; hence, a relatively large model size (5-layer message passing with 1024-dimensional hidden channels) can be trained with 194 GPU hours, which is comparable to ForceNet-large.
- **DimeNet++** (Klicpera et al., 2020a) is a recent improvement of DimeNet (Klicpera et al., 2020b) and is also an energy-centric GNN. It uses atom-triplet-based message passing to capture angular information, which makes it computationally expensive. Even training DimeNet++ of a relatively-small model size (3-layer message passing with 192-dimensional hidden channels) requires 587 GPU days—18.9 and 3.0 times more expensive than ForceNet and ForceNet-large, respectively. Training DimeNet++-large (3-layer message-passing with 512-dimensional hidden channels) takes a significant 1600 GPU days of compute, being 51.6 and 8.2 times more expensive than ForceNet and ForceNet-large, respectively.
- **GNS model** (Sanchez-Gonzalez et al., 2020) is a scalable force-centric model that directly predicts atomic forces. We make its model size (in terms of the number of parameters) comparable to ForceNet. The training takes 20 GPU days, which is $1.6\times$ faster than ForceNet. However, as we will see, the performance of ForceNet is better even if ForceNet's training is truncated at 20 GPU days.

All the results of SchNet and DimeNet++ are directly adopted from the OC20 paper (Chanussot et al., 2020). These energy-centric models are trained only on atomic forces, although in principle, they can be aso trained on per-system energy. Chanussot et al. (2020) report that training on forces and energy seperately achieves better performance on each task compared to joint training. For the GNS model, we reproduce the original model architecture ourselves based on the feedback from the original author of GNS (Sanchez-Gonzalez et al., 2020). Refer to Appendix E for implementation details. On the GNS model, we apply the same training strategies as ForceNet.

D HYPER-PARAMETERS

For training, we use the Adam optimizer (Kingma & Ba, 2015), with an initial learning rate of 0.0005. We train ForceNet and the GNS model for 500K iterations with the batch size of 256, which is equivalent to 1 epoch for the entire dataset[2]. For ForceNet-large, we use the batch size of 512. All the parameters of the force-centric models are initialized with Xavier uniform initialization (Glorot & Bengio, 2010). The learning rate is kept constant for the first 250K iterations, after which it is halved every 50K iterations. We use the checkpoint with the best validation ID performance, and evaluate the saved model over all four validation sets. MAE over forces is used as the training loss. We will evaluate our models on the hidden test sets once the test server is ready.

For Gaussian and sine basis functions, we use $J = 50$, which gives an output dimensionality of $B = 350$. For Linear+Act, we set $B = 350$. For spherical basis, we use $L = 3$ and $S = 4$, which results in $B = 36 (= 3^2 \cdot 4)$, and we set $J = 50$ for the internally-used sine basis function. For encoding the input atomic node features, we first normalize each dimension to lie between 0 and 1, and adopt the same basis function as used for encoding the edge features. The exception is spherical basis that is specialized for 3D spaces, in which case, the sine basis is used to encode the input atomic node features. We find that increasing J and L beyond the above values does not improve the performance, while significantly decreasing them worsens the performance.

E DETAILS OF GNS MODEL

For the GNS results in this paper, we reimplemented the original GNS model (Sanchez-Gonzalez et al., 2020). Since the public code for the original GNS model (Sanchez-Gonzalez et al., 2020) was not available at the time of our experiments, we communicated with one of the authors to confirm the implementation details.

The message in the GNS model is defined as

$$m(\boldsymbol{h}_t^{(l)}, \boldsymbol{e}_{st}, \boldsymbol{h}_s^{(l)}) = \mathbf{MLP}\left(\mathrm{Concat}\left(\boldsymbol{h}_t^{(l)}, \boldsymbol{e}_{st}, \boldsymbol{h}_s^{(l)}\right)\right),$$

where $\mathbf{MLP}(\cdot)$ is a 1-hidden-layer MLP with ReLU activation and layer normalization (Ba et al., 2016) applied before the activation. For aggregating the message, the GNS model used either mean or sum, so we tried both in our experiments. We found sum aggregation to perform better, and report results of sum aggregation in this paper. After the messages are aggregated, GNS uses a learnable linear function to transform the node embeddings. Similar to ForceNet, we additionally apply a batch normalization on the node embeddings, which alleviates training instability and significantly improves performance. The GNS model uses a residual connection, where the computed node embeddings are added into the node embeddings from the previous layer. For the decoder, the GNS model uses a 1-hidden-layer MLP with ReLU activation. All the node embeddings and hidden units in the MLPs have the same dimensionality.

F EXPERIMENTAL RESULTS

Here we provide our experimental results for predicting atomic forces in the OC20 dataset, including the comparison to the baseline model (Appendix F.1) and extensive ablation studies of ForceNet (Appendix F.2).

[2]We do not observe much gain by training models longer than 1 epoch. This is probably because of the redundancy in data, *i.e.*, out of 130M data points, there are 650k unique atom configurations (ignoring the positional differences).

Table 2: Comparison of ForceNet to existing GNN models. We mark as bold the best performance and close ones, *i.e.*, within 0.0005 MAE, which according to our preliminary experiments, is a good threshold to meaningfully distinguish model performance. Training time is in GPU days, and inference time is in GPU hours. Median represents the trivial baseline of always predicting the median training force across all the validation atoms.

Model	**Hidden dim**	**#Msg layers**	**#Params**	**Train time**	**Inference time**	**Validation Force MAE (eV/Å)** ID	OOD Ads.	OOD Cat.	OOD Both	**Average**
Median	–	–	–			0.0810	0.0799	0.0799	0.0943	0.0838
GNS	768	5	12.5M	20d	1.5h	0.0421	0.0466	0.0430	0.0559	0.0469
SchNet	1024	5	9.1M	194d	0.8h	0.0443	0.0514	0.0465	0.0618	0.0510
DimeNet++	192	3	1.8M	587d	8.5h	0.0332	0.0366	0.0344	0.0436	0.0369
DimeNet++-large	512	3	10.7M	1600d	27.0h	**0.0281**	**0.0318**	**0.0315**	**0.0396**	**0.0328**
ForceNet	512	5	11.3M	31d	1.3h	0.0313	0.0355	0.0334	0.0439	0.0360
ForceNet-large	768	7	34.8M	194d	3.5h	**0.0281**	**0.0320**	0.0327	0.0412	0.0335

We evaluate models on four validation datasets that test different levels of model generalization: In Domain (ID), Out of Domain Adsorbate (OOD Adsorbate), OOD Catalyst, and OOD Both (both the adsorbate and catalyst's material are not seen in training). Each split contains 1M examples. All the models and their settings are explained in Appendix C.

Following Chanussot et al. (2020), the Mean Absolute Error (MAE) of forces on free atoms is evaluated for each validation set. Here the free atoms represent atoms that are close to the material surface and are free to move during atomic relaxation simulation (Figure 4 in Appendix B). We use the "average force MAE" to represent the MAE of forces averaged over the four validation sets.

F.1 Comparison to Baseline Models

In Table 2, we provide the complete results for our comparison of ForceNet and the baseline models.

F.2 Ablation on ForceNet's model designs

Here we perform extensive ablation studies on ForceNet's key design choices presented in Section 2.

F.2.1 Basis and activation functions

We systematically study how choices of different basis and non-linear activation functions affect the model performance. The baseline functions are explained in Appendix A. We also compare with the "None" baseline, which does not use a basis function and directly concatenates the input raw edge feature into the node embeddings. In Figure 5, we see that the combination of spherical basis and Swish activation performs the best. For comparison, the GNS model uses no basis function ("None") and ReLU, see Appendix E for full GNS model details.

F.2.2 Architecture design

Next, we study the architectural building blocks of our conditional-filter-based message passing with the fixed basis and activation functions. We consider seven cases: **(1) Only-dist**: we remove $\boldsymbol{n}_{st}$ from the input edge feature, *i.e.*, $\boldsymbol{e}_{st} \equiv \boldsymbol{p}_{st}$, resulting in the edge features being rotation invariant. **(2) No-atomic-radii**: we set the input edge features to $\boldsymbol{e}_{st} \equiv \mathrm{Concat}\left(\boldsymbol{n}_{st}, \|\boldsymbol{d}_{st}\|\right)$ (atomic radii information is dropped), **(3) No-node-emb**: filter $\boldsymbol{F}_c$ is a function of only $\boldsymbol{e}_{st}$ (conditioning on source and target node embeddings $\boldsymbol{h}_s^{(k)}, \boldsymbol{h}_t^{(k)}$ is dropped), **(4) Only-$\boldsymbol{F}_c$**: Filter is directly aggregated, *i.e.*, $\boldsymbol{m}_{st} = \boldsymbol{F}_c$, and self-message m_t is omitted. **(5) Edge-linear-BN**: MLP $\boldsymbol{F}_{\mathrm{e}}$ is replaced with a linear function followed by batch normalization, **(6) Node-linear-BN**: MLP $\boldsymbol{F}_{\mathrm{n}}$ is replaced with a linear function followed by batch normalization. **(7) No-$\boldsymbol{m}_t$**: self-message $\boldsymbol{m}_t$ is removed. Note that in **(5)** and especially **(6)**, we find it critical to add the batch normalization to facilitate training.

Table 3 shows the results of the seven ablation studies. Most notably, **(1)** is significantly worse than the rest, because rotation-invariant node embeddings are insufficient for predicting rotation-covariant forces. We also see from **(2)** and **(3)** that making the filter less expressive, especially by dropping the dependency on node embeddings, significantly hurts performance. The improvement from element-wise product parameterization $\boldsymbol{F}_c \odot \boldsymbol{F}_d$ is demonstrated in **(4)**. From **(5)**, we see that it is critical to

Table 3: Ablations on the architecture of ForceNet.

Ablation	Average Force MAE (eV/Å)
ForceNet	**0.0360**
(1) Only-dist	0.0699
(2) No-atomic-radii	0.0368
(3) No-node-emb	0.0410
(4) Only-$\boldsymbol{F}_c$	0.0378
(5) Edge-linear-BN	0.0427
(6) Node-linear-BN	**0.0364**
(7) No-$\boldsymbol{m}_t$	**0.0364**

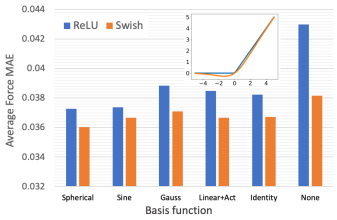

Figure 5: Ablations on basis and activation functions in ForceNet.

Table 4: Ablation of model scaling in terms of : (a) hidden dimensionality, (b) number of message passing layers, and (c) training batch size. Training time is roughly proportional to (b) and (c) and quadratic in (a).

Hidden dim	#Msg layers	Batch size	Average Force MAE (eV/Å)
512	5	256	0.0360
768	5	256	0.0352
512	7	256	0.0355
768	7	256	0.0352
768	**7**	**512**	**0.0345**

Table 5: Ablations on basis and activation functions in the ForceNet architecture.

Basis	**Act.**	**Validation Force MAE (eV/Å)** ID	OOD Ads.	OOD Cat.	OOD Both	**Average**
Spherical	ReLU	0.0324	0.0367	0.0346	0.0454	0.0373
Spherical	**Swish**	**0.0313**	**0.0355**	**0.0334**	**0.0439**	**0.0360**
Sine	ReLU	0.0324	0.0367	0.0346	0.0456	0.0374
Sine	Swish	**0.0317**	**0.0360**	0.0342	0.0448	0.0367
Gauss	ReLU	0.0335	0.0384	0.0359	0.0476	0.0389
Gauss	Swish	**0.0318**	0.0364	0.0346	0.0456	0.0371
Linear+Act	ReLU	0.0340	0.0379	0.0356	0.0464	0.0385
Linear+Act	Swish	0.0321	**0.0359**	0.0342	0.0445	0.0367
Identity	ReLU	0.0335	0.0377	0.0353	0.0464	0.0382
Identity	Swish	0.0322	0.0364	**0.0338**	0.0445	0.0368
None	ReLU	0.0379	0.0430	0.0391	0.0519	0.0430
None	Swish	0.0330	0.0383	0.0347	0.0466	0.0382

utilize non-linear models for edge features, as atomic forces are highly dependent on their subtle changes, but non-linearities are not essential for node embeddings **(6)**. Finally, from **(7)**, we see that the self-message $\boldsymbol{m}_t$ is not essential in performance.

Overall, our analysis suggests that ForceNet benefits most from its expressive *edge-level* computation via the conditional filter, which is directly responsible for accurately encoding the 3D neighborhood structure on which the atomic forces depend.

F.2.3 Model Scaling

Comparing ForceNet and ForceNet-large in Table 2, we see that a larger model provides significant performance gain, at the cost of 6.3 times more training time and 2.4 times more inference time. Extrapolating through Figure 1, we expect ForceNet to significantly outperform DimeNet++, once comparable computational resources are used. We leave this investigation to future work. More fine-grained ablations on model scaling are shown in Table 4. We see that all the three scaling components help in the current regime of ForceNet.

G Full Ablation Results

Here we provide full S2F results of our ablation studies, reporting the force MAE on each of the four validation sets.

Table 6: Ablations on the message passing architecture of ForceNet.

Ablation	Validation Force MAE (eV/Å)				
	ID	OOD Ads.	OOD Cat.	OOD Both	**Average**
ForceNet	**0.0313**	**0.0355**	**0.0334**	**0.0439**	**0.0360**
(1) Only-dist	0.0658	0.0673	0.0660	0.0805	0.0699
(2) No-atomic-radii	0.0321	0.0362	0.0342	0.0447	0.0368
(3) No-node-emb	0.0361	0.0409	0.0374	0.0495	0.0410
(4) Only-$\boldsymbol{F}_c$	0.0333	0.0372	0.0350	0.0455	0.0378
(5) Edge-linear-BN	0.0371	0.0430	0.0388	0.0520	0.0427
(6) Node-linear-BN	**0.0317**	**0.0356**	**0.0339**	**0.0442**	**0.0364**
(7) No-$\boldsymbol{m}_t$	**0.0314**	**0.0360**	**0.0336**	**0.0444**	**0.0364**

Table 7: Ablations on training strategies for ForceNet.

Model	Rotation aug.	Weight on fixed atoms	Validation Force MAE (eV/Å)				
			ID	OOD Ads.	OOS Cat.	OOD Both	**Average**
ForceNet	✔	0.05	**0.0313**	**0.0355**	**0.0334**	**0.0439**	**0.0360**
ForceNet		0.05	**0.0314**	**0.0359**	0.0341	0.0448	0.0366
ForceNet	✔	1	0.0369	0.0411	0.0390	0.0506	0.0419
ForceNet	✔	0	0.0333	0.0385	0.0348	0.0465	0.0383

Full Results on ForceNet Designs First, we provide the full ablation results on ForceNet designs. Table 5 shows the ablations on basis and activation functions, while Table 6 shows the ablations on the message passing architectures.

Overall, we see trends that are consistent with the averaged results in Figure 5 and Table 3. Specifically, from Table 5, we see that the combination of spherical basis functions and the Swish activation results in the best performance across the four validation sets. From Table 6, we see that the conditional filter convolution design gives superior performance compared to the more simplified architectures, except for (6) and (7), in which the performance is comparable.

Full Results on Training Strategies Next, we provide the full ablation results of our training strategies, fixing the model architecture to the default ForceNet.

The results are shown in Table 7. First, we see that rotation augmentation helps, especially for the three out-of-distribution validation sets. Second, we see that providing small reweighted supervision on fixed atoms is also helpful, significantly improving the validation performance (evaluated on *free atoms*) compared to the two baseline strategies: (1) uniform loss weighting (equally weighting the losses for fixed and free atoms) and (2) zero-loss weighting (ignoring losses on fixed atoms during training).

H Structure Relaxation Simulation Results

Here we apply ForceNet to the IS2RS (**I**nitial **S**tructure to **R**elaxed **S**tructure) task. The goal is to predict the relaxed structure, *i.e.*, 3D structure with zero-forces on all free atoms, from the initial structure. This can be achieved by simulating a relaxation trajectory: iteratively updating the atomic positions of free atoms according to their predicted forces until convergence, *i.e.*, the predicted forces are below a pre-specified threshold.

Specifically, the structure relaxations are performed using a PyTorch implementation of the Atomic Simulation Environment's (*ASE*) (Hjo, 2017) L-BFGS optimizer. Relaxations were terminated when a max-absolute per-atom force of 0.01 eV/Å or 200 simulation steps, whichever comes first. All DFT calculations were performed in the *Vienna Ab Initio Simulation Package* (VASP) (Kresse &

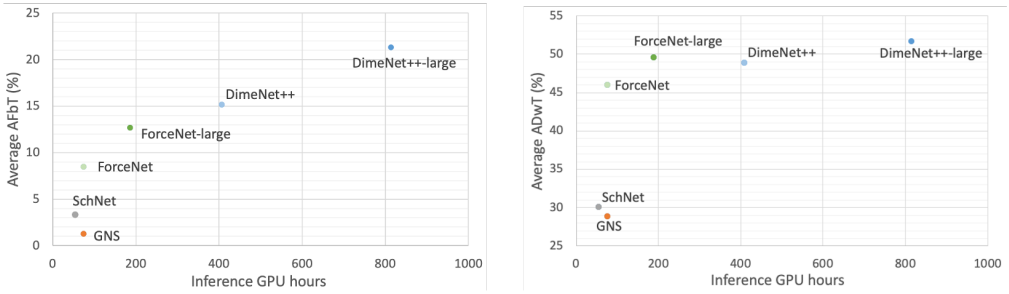

Figure 6: Comparison of IS2RS performance in terms of AFbT and ADwT averaged over the four validation sets. The x-axis is the IS2RS inference time in GPU hours measured over 100K relaxations.

Hafner, 1994; Kresse & Furthmüller, 1996a;b). Both ASE and VASP are popular packages within the computational chemistry and catalysis communities.

The performance on the IS2RS task is evaluated by the two standard metrics (Chanussot et al., 2020): (1) Average Force below Threshold (AFbT), measuring whether the predicted relaxed structure actually has small forces calculated by ground-truth DFT, and (2) Average Distance within Threshold (ADwT), measuring the geometrical closeness between the predicted relaxed structure and ground-truth relaxed structure. For both metrics, the higher, the better.

Figure 6 compares the performance of different models, while taking the inference efficiency into account. The full results for all the validation sets are provided in Table 8. Here the inference time is measured on a Tesla V100 Volta GPU, where we use the largest possible batch size for each model and perform 100K relaxations. All the models are the same as Figure 1 and Table 2, originally trained for the S2F task.

We see from Figure 6 (left) that in terms of AFbT, both ForceNet models outperform GNS and SchNet, while the inference time of ForceNet, GNS and SchNet is comparable to each other. Compared to DimeNet++, both ForceNet and ForceNet-large have lower AFbT. However, the inference of both ForceNet models is much faster than that of DimeNet++ (5.4 times faster for ForceNet, and 2.2 times faster for ForceNet-large). Moreover, we see that there is still a room for ForceNet-large to be further scaled up to give AFbT comparable to DimeNet++. Regarding ADwT, from Figure 6 (right), we see that ForceNet-large outperforms DimeNet++, while being 2.2 times faster in inference. ForceNet-large is slightly worse than DimeNet++-large, but is 5.4 times faster in inference.

Overall, the above results are encouraging given the faster inference time of ForceNet compared to DimeNet++. However, the results also suggest a potential limitation of ForceNet's force-centric approach: the superior performance of ForceNet-large over DimeNet++ in the S2F task (*i.e.*, estimate the forces of 3D structures along the simulation trajectory) does not directly translate into its superior performance on the IS2RS simulation task. We deduce this is due to the compounding error problem of the force-centric approach pointed out by the GNS work (Sanchez-Gonzalez et al., 2020), *i.e.*, model's prediction errors accumulate along the simulation trajectory, which forces the model to make increasingly erroneous prediction over structures that are far away from the simulation trajectory. The energy-centric models may suffer less from the problem since their built-in physical constraints allow them to make more well-behaved force prediction over the off-trajectory structures, which eventually leads to better simulation results despite the worse force prediction results.

Fortunately, OC20 additionally provides 94M off-trajectory structures obtained by either perturbing the on-trajectory structures or performing molecular dynamics from relaxed structures (Chanussot et al., 2020). We believe that these structures can be used to mitigate the compounding error problem of the force-centric approach by robustifying its off-trajectory force prediction. In fact, the original GNS work (Sanchez-Gonzalez et al., 2020) has demonstrated that training their force-centric models on perturbed off-trajectory structures significantly reduces the compounding error, thereby improving their simulation results. We leave this investigation to future work.

Table 8: Full IS2RS results. Inference time is in GPU hours and measured over 100K relaxations.

Model	**Inference time**	**AFbT (%)**					**ADwT (%)**				
		ID	OOD Ads.	OOD Cat.	OOD Both	**Average**	ID	OOD Ads.	OOD Cat.	OOD Both	**Average**
GNS	74.3h	2.22	0.66	1.44	0.62	1.24	30.60	23.13	30.92	31.15	28.95
SchNet	54.1h	4.90	2.66	2.75	2.90	3.30	35.54	29.80	26.86	28.39	30.15
DimeNet++	407.6h	17.41	14.41	14.19	14.55	15.14	48.75	45.19	48.59	53.14	48.92
DimeNet++-large	814.6h	**24.22**	**20.40**	**20.13**	**20.31**	**21.27**	**52.45**	**48.47**	**50.98**	**54.82**	**51.68**
ForceNet	75.1h	10.75	7.74	7.54	7.78	8.45	46.83	41.26	46.45	49.60	46.04
ForceNet-large	186.9h	14.77	12.23	12.16	11.46	12.66	50.59	45.16	49.80	52.94	49.62