Reducing Down(stream)time: Pretraining Molecular GNNs using Heterogeneous AI Accelerators

Jenna A. Bilbrey
Pacific Northwest National Laboratory
jenna.pope@pnnl.gov

Kristina M. Herman
Department of Chemistry, University of Washington
kmherman@uw.edu

Henry Sprueill
Pacific Northwest National Laboratory
henry.sprueill@pnnl.gov

Sotiris S. Xantheas
Pacific Northwest National Laboratory
Department of Chemistry, University of Washington
sotiris.xantheas@pnnl.gov

Payel Das
IBM Research
daspa@us.ibm.com

Manuel Lopez Roldan
Graphcore
manuelr@graphcore.ai

Mike Kraus
Graphcore
mikek@graphcore.ai

Hatem Helal
Graphcore
hatemh@graphcore.ai

Sutanay Choudhury
Pacific Northwest National Laboratory
sutanay.choudhury@pnnl.gov

Abstract

The demonstrated success of transfer learning has popularized approaches that involve pretraining models from massive data sources and subsequent finetuning towards a specific task. While such approaches have become the norm in fields such as natural language processing, implementation and evaluation of transfer learning approaches for chemistry are in the early stages. In this work, we demonstrate finetuning for downstream tasks on a graph neural network (GNN) trained over a molecular database containing 2.7 million water clusters. The use of Graphcore IPUs as an AI accelerator for training molecular GNNs reduces training time from a reported 2.7 days on 0.5M clusters to 1.2 hours on 2.7M clusters. Finetuning the pretrained model for downstream tasks of molecular dynamics and transfer to a different potential energy surface took only 8.3 hours and 28 minutes, respectively, on a single GPU.

1 Introduction

Pretraining models on massive datasets followed by finetuning towards specific downstream tasks is commonplace in natural language processing and computer vision approaches. The uptake of similar approaches for chemistry is lagging due to the limited number of large datasets and long training times involved. Training atomistic property prediction models from massive scientific datasets is a compute-intensive task, and much of the focus in recent literature has been on transformer-based models [1, 2]. The reduction of atomic positions to character strings via the SMILES notation has aided the generation of large datasets. For example, Wang et al. trained the BERT architecture on 18.7M SMILES strings from the ZINC database [1], while Chithrananda et al. trained a network based on the RoBERTa architecture, called ChemBERTa, on 77M SMILES strings obtained from PubChem [2]. ChemBERTa was later shown to perform well on downstream property prediction

Machine Learning and the Physical Sciences workshop, NeurIPS 2022.

tasks [3]. RNN-based generative models have been trained on a 1M-molecule subset of GDB-13 [4] and a 1.6M-molecule subset of ZINC [5]. More recently, Ross et al. developed a transformer-based encoder that uses linear attention, called Molformer, to efficiently train on >1000M SMILES strings, outperforming several graph- and geometry-based baselines on regression and classification tasks from benchmark datasets [6]. Subsequent work has explicitly incorporated spatial properties into Molformer for training on comparatively smaller datasets [7].

SMILES strings describe atom compositions and bonding configurations, but neglect information about the 3D geometry and long-range interactions, such as interactions between molecules. Here, we answer a benchmark challenge set forth in our prior work [8] of generating a predictive model that preserves intermolecular interactions. This benchmark is supported by an open-source dataset containing 4.95M unique hydrogen bonded clusters of water molecules [9]. All clusters in the dataset are minima on the potential energy surface (PES) computed using the TTM2.1-F potential, making them useful for static property prediction. For example, Bilbrey et al. [10] trained the SchNet neural network on a subset of 500,000 clusters and obtained a mean absolute error per water molecule of 0.002 kcal/mol on a 10,500-sample test set, which included cluster sizes outside of the range of those included in the training subset, indicating the ability of the network to extrapolate. Training on $\sim$10% of the full benchmark dataset took 2.7 days scaled over four NVIDIA V100 GPUs [10], making it impractical to train on larger subsets, much less the complete dataset, with traditional hardware. The computing power required for training neural networks on very large datasets restricts exploration of this area to researchers with ample access to GPU clusters or AI hardware accelerators [11, 12, 13].

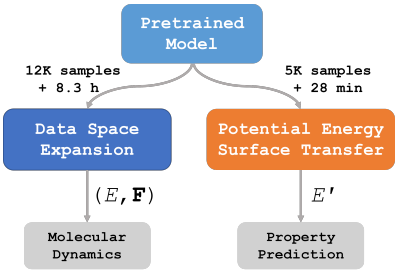

Figure 1: Downstream workflow.

In this work, we demonstrate the training and downstream inference of a graph neural network (GNN) built on a database of 2.7M molecular structures. We train the SchNet architecture [14, 15] on a 2.7M subset of the HydroNet benchmark dataset using Graphcore IPU hardware accelerators. We then finetune the pretrained network using much smaller datasets on a single V100 GPU for downstream tasks of molecular dynamics (MD) and property prediction. The reduced hardware and data requirements afforded by employing the pretrained model, which we make open source, open the door for experimentation by a larger number of researchers, all while decreasing energy resources consumed during experimentation. The two key factors underpinning this research are (1) the massive scale of a water cluster database that contains millions of 3D geometries and (2) the use of a novel AI accelerator that dramatically reduces training times.

2 Methods

Data Collection. The dataset of water cluster minima used for pretraining was obtained from an existing database generated using Monte Carlo temperature basin-paving (MC-TBP) simulations driven by the TTM2.1-F potential [16, 17]. Each cluster is associated with 3D coordinates $\mathbf{r}$ and energy E. The dataset of non-minima water clusters for downstream MD via data space expansion was obtained from MD simulations performed at 260K and 300K. Each non-minima includes atomic forces $\mathbf{F}$, $\mathbf{r}$, and E. To demonstrate transfer of the PES, E for a subset of 5,000 minima were obtained using the MB-pol interatomic potential [18, 19].

Hardware Accelerators. Accelerators designed specifically for AI/ML applications show improved processing speed, scalability, and energy efficiency, allowing faster training on larger datasets. In particular, Graphcore's IPU accelerators show a $4\times$ speedup over NVIDIA V100 GPUs for training of GNNs [20]. Graphcore's high-level development framework PopTorch was used to implement the Pytorch Geometric (PyG) library [21], which includes the SchNet framework. We trained SchNet on a dataset containing 2,726,710 water clusters, using hyperparameters reported in previous work [10].

Finetuning. Pretraining molecular GNNs on large datasets have been shown to improve generalization in downstream tasks [22]. Use of the PyG implementation of SchNet allows the model to be easily transferred between IPUs, GPUs, and CPUs. The saved model weights from the IPU-trained model constitute the pretrained network, which we update to obtain (1) drive MD simulations via

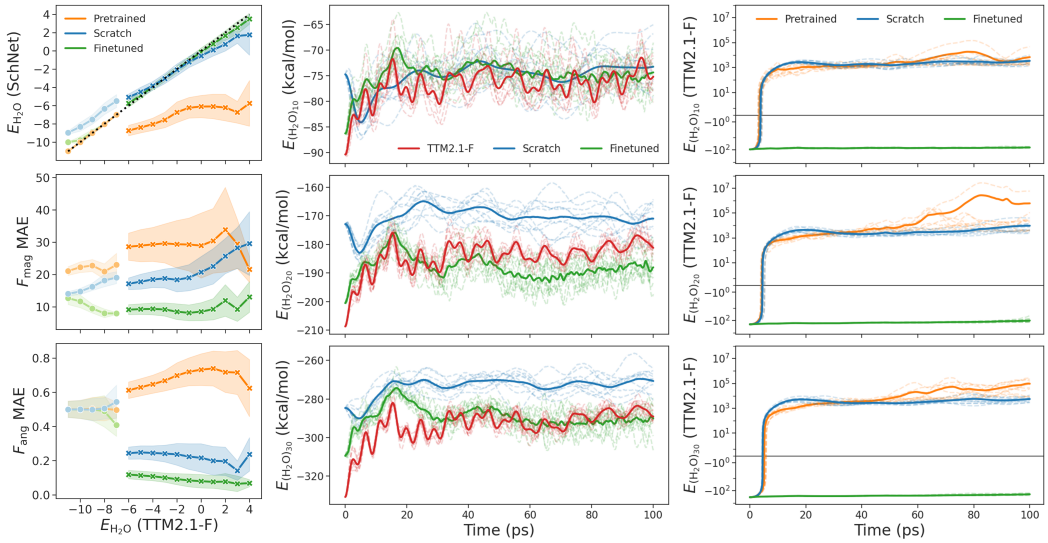

Figure 2: (Left) Static predictions of $E_{\mathrm{H_2O}}$, F_{mag}, and F_{ang} on minima (•) and non-minima (×) test sets. (Center) MD using the TTM2.1-F potential and NNPs for water clusters of size N=10, 20, 30. (Right) Static TTM2.1-F predictions on MD trajectories generated by the NNPs.

accurate predictions of E and $\mathbf{F}$ on non-minima and (2) accurate predictions of E on a different PES. Inference using trained models was performed on a CPU, while finetuning was performed on a single NVIDIA V100 GPU.

Active Sampling. We adapt an active sampling strategy when finetuning for non-minima to minimize bias when compiling the small dataset. The training set was divided into a small training subset and large reserve set. During training, the mean absolute error (μ_ε) and standard deviation of errors (σ_ε) in $\mathbf{F}$ were computed over the validation set. Then, ε_s was calculated for a subset of reserve samples. A sample was moved from the reserve set to the training subset if $1 - \mathrm{erf}\left((\varepsilon_s - \mu_\varepsilon)/\sigma_\varepsilon\right) < p_{\mathrm{tol}}$, where p_{tol} is a chosen tolerance and $\mathrm{erf}(x)$ is the Gaussian error function.

3 Results and discussion

We present performance between three model variations: SchNet trained on 2.7M water cluster minima (PRETRAINED), the pretrained model further finetuned using a much smaller dataset (FINETUNED), and a SchNet model trained from scratch using only the smaller dataset (SCRATCH) – on two downstream tasks – data space expansion and PES transfer.

Model Pretraining. The SchNet model was trained on 2.7M water clusters of size N=3–25 with a 0.8:0.1:0.1 train-validation-test split. Training took 4.2, 2.2, or 1.2 hours scaled across 16, 32, or 64 IPUs, respectively. The validation loss increased with the number of IPUs to 0.0017, 0.0020, and 0.0030, respectively; therefore, we used the model trained over 16 IPUs to finetune for subsequent downstream tasks. This model showed a similarly low test-set error of 0.0018 kcal/mol.

Data Space Expansion. MD simulations explore non-minima on the PES and can be driven by neural network potentials (NNPs). The forces $\mathbf{F}$ used to drive atomic motions are obtained as

Table 1: Test-set accuracy, reported as mean absolute error (MAE) in $E_{\mathrm{H_2O}}$ (kcal/mol), F_{mag} (kcal/mol/Å), and F_{ang}. Test sets were derived from the specified dataset.

Hardware	Dataset	Initialization	N_{train}	$E_{\mathrm{H_2O}}$	F_{mag}	F_{ang}
IPU	minima	scratch	2.7M	0.0018	21.89	0.501
GPU	non-minima	scratch	11.8K	0.3548	18.76	0.238
GPU	non-minima	pretrain	11.6K	0.1316	8.94	0.098

the negative of the NNP gradient. A force term is typically added to the loss function to improve prediction accuracy [23]. Training without the force term, as was done for the pretrained model, leads to poor prediction of $\mathbf{F}$, even through the accuracy in E predictions on minima configurations is high. Moreover, the pretrained model produces poor E predictions on non-minima, necessitating the need for the data space covered by the model to be expanded. Finetuning the pretrained model on a much smaller subset of non-minima (>1% of the minima dataset) and including a force term in the loss produces a NNP with good predictions of non-minima $\mathbf{F}$ and E. Because of the roughly three-order-of-magnitude decrease in the size of the training set when using the pretrained model, training was accomplished on a single NVIDIA V100 GPU in 8.3 hours.

Table 1 shows the test-set accuracy of the pretrained model and models trained on non-minima with and without pretraining. The test set corresponds to the specific training set, i.e., the pretrained model is tested on minima, while the models trained on non-minima are tested on non-minima. Following Chmiela et al. [23], we quantify the topological accuracy of atomic force predictions by the magnitude error $F_{\text{mag}} = \|\hat{\mathbf{F}}\| - \|\mathbf{F}\|$, which describes the extent to which the slope of the predicted and reference PES differ, and the angular error $F_{\text{ang}} = \cos^{-1}(\hat{\mathbf{F}}/\|\hat{\mathbf{F}}\| \cdot \mathbf{F}/\|\mathbf{F}\|)/\pi$, which describes the orientation of the predicted force direction relative to the reference force direction and ranges between 0 (aligned) and 1 (inverted). Figure 2 shows static errors for the three models on minima and non-minima. Training from scratch on non-minima greatly improves static predictions of E and provides some improvement in $\mathbf{F}$ predictions, while finetuning from the pretrained model greatly improves predictions of $\mathbf{F}$ and E for both minima and non-minima.

We then performed MD simulations driven by the NNPs (Fig. 2). Berendsen NVT dynamics of three clusters ($N = 10, 20, 30$) at 300K were simulated 10 times each with different random seeds. The mean (bold lines) and individual simulations (dashed lines) are shown for TTM2.1-F and the NNPs trained on non-minima. MD simulations using the pretrained model are not shown, as the predicted energies were several orders of magnitude below those from TTM2.1-F. The mean E of the finetuned model aligns for all cluster sizes, while that of the model without pretraining aligns only for $N = 10$. We then calculated the TTM2.1-F E on each point in the NNP-generated trajectories to validate the resulting molecular structures. Notably, only the finetuned model produced valid dynamics ($E < 0$ kcal/mol). Though the model without pretraining predicted E of similar magnitude to TTM2.1-F, the generated structures were calculated by TTM2.1-F to be highly unstable. In fact, the model trained from scratch did not outperform even the pretrained model in generating stable dynamics.

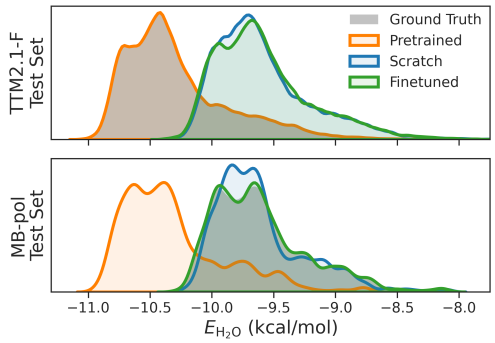

Figure 3: Predicted $E_{\text{H}_2\text{O}}$ distributions on test sets computed with the TTM2.1-F (top) and MB-pol (bottom) potentials.

PES Transfer. The PES of a physical system will have a different representation depending on the method of calculation, each of which can have different associated computational costs. Classical many-body force fields, such as TTM2.1-F, are fast enough to generate large datasets, though at reduced accuracy, while *ab initio* methods are highly accurate but prohibitively expensive, making the generation of comparably sized training sets intractable. The PES approximated by a model can be amended by updating a model trained on a large dataset collected by a specific method (here, using the TTM2.1-F potential) using a small dataset collected by an alternative method (here, the MB-pol potential). We perform such PES transfer by finetuning the pretrained model on 5,000 samples with E computed with the MB-pol potential. Because of the drastically reduced size of the dataset, the model was trained in under 28 minutes on 1 NVIDIA V100 GPU. Figure 3 shows $E_{\text{H}_2\text{O}}$ distributions for the TTM2.1-F and MB-pol test sets, with errors shown in Table 2. A shift in $E_{\text{H}_2\text{O}}$ towards higher values is seen for the MB-pol potential and is reproduced for both models trained on MB-pol data. However, the finetuned model showed a ∼17% lower error and more accurately reproduced the $E_{\text{H}_2\text{O}}$ distribution.

Table 2: Test-set errors in $E_{\mathrm{H_2O}}$ (kcal/mol).

Hardware	Initialization	Train Set	Test Set	MAE	RMSE
IPU	scratch	TTM2.1-F	TTM2.1-F	0.0018	0.0032
GPU	scratch	MB-pol	TTM2.1-F	0.6973	0.7009
GPU	pretrain	MB-pol	TTM2.1-F	0.7033	0.7071
IPU	scratch	TTM2.1-F	MB-pol	0.7071	0.7112
GPU	scratch	MB-pol	MB-pol	0.0719	0.0924
GPU	pretrain	MB-pol	MB-pol	0.0122	0.0158

4 Conclusions

We demonstrate that pretraining with a very large dataset of molecular structures improves downstream tasks such as driving MD simulations as well as transfer learning to a different PES. This paper is the first to demonstrate the effectiveness of the fine-grained parallelism of the Graphcore IPU architecture for training molecular GNNs. Initial training was accomplished over 16 IPUs in 4.2 hours, while finetuning with a small set of non-minima was accomplished in 8.3 hours on a single NVIDIA V100 GPU and transfer learning with a very small set of minima computed by a different method was accomplished in only 28 minutes. Pretaining was shown to decrease the amount of data and time required, as well as reduce hardware requirements, increasing training throughput and improving accessibility to researchers with limited resources. The pretrained model was not finetuned for downstream tasks on molecules other than water; moving to a separate area of chemical space, for example, organic small molecules, could be accomplished by following our workflow, i.e., training on an open dataset of minima and finetuning for the desired downstream task on a small bespoke dataset.

Data Availability

The full database of water cluster minima computed with the TTM2.1-F potential is available for download at `https://data.pnnl.gov/group/nodes/dataset/33224`. The preprocessed databases for training, including the database of nonminima computed with the TTM2.1-F potential and the database of minima computed with the MB-pol potential, dataset split files, trained model state dictionaries, ASE databases used for MD simulations, and results of data space expansion and PES transfer analyses are available for download at `https://data.pnnl.gov/group/nodes/dataset/33283`.

Code Availability

The codebase used to finetune the pretrained model for the tasks of data space expansion and PES transfer, along with hyperparameters used in this work, is available at `https://github.com/pnnl/downstream_mol_gnn`.

Impact statement

IPUs show reduced energy consumption compared with GPUs when training neural networks. In addition, pretraining reduces data and hardware requirements, leading to reduced energy consumption and increased accessibility to researchers with limited resources.

Acknowledgments and Disclosure of Funding

The authors thank Dr. Logan Ward for fruitful discussions on neural network potentials. J.A.B., K.M.H., H.S., S.S.X., and S.C. were supported by the DOE Exascale Computing Project, ExaLearn Co-design Center. The research was performed using resources available through Research Computing at Pacific Northwest National Laboratory (PNNL). PNNL is operated by Battelle for the U.S. Department of Energy under Contract DE-AC05-76RL01830.

References

[1] Sheng Wang, Yuzhi Guo, Yuhong Wang, Hongmao Sun, and Junzhou Huang. SMILES-BERT: large scale unsupervised pre-training for molecular property prediction. In *Proceedings of the 10th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics*, pages 429–436, 2019.

[2] Seyone Chithrananda, Gabriel Grand, and Bharath Ramsundar. ChemBERTa: large-scale self-supervised pretraining for molecular property prediction. *arXiv preprint arXiv:2010.09885*, 2020.

[3] Walid Ahmad, Elana Simon, Seyone Chithrananda, Gabriel Grand, and Bharath Ramsundar. ChemBERTa-2: Towards chemical foundation models. *arXiv preprint arXiv:2209.01712*, 2022.

[4] Josep Arús-Pous, Thomas Blaschke, Silas Ulander, Jean-Louis Reymond, Hongming Chen, and Ola Engkvist. Exploring the GDB-13 chemical space using deep generative models. *Journal of Cheminformatics*, 11(1):1–14, 2019.

[5] Vijil Chenthamarakshan, Payel Das, Samuel Hoffman, Hendrik Strobelt, Inkit Padhi, Kar Wai Lim, Benjamin Hoover, Matteo Manica, Jannis Born, Teodoro Laino, et al. Cogmol: target-specific and selective drug design for covid-19 using deep generative models. *Advances in Neural Information Processing Systems*, 33:4320–4332, 2020.

[6] Jerret Ross, Brian Belgodere, Vijil Chenthamarakshan, Inkit Padhi, Youssef Mroueh, and Payel Das. Do large scale molecular language representations capture important structural information? *arXiv preprint arXiv:2106.09553*, 2021.

[7] Fang Wu, Qiang Zhang, Dragomir Radev, Jiyu Cui, Wen Zhang, Huabin Xing, Ningyu Zhang, and Huajun Chen. Molformer: Motif-based transformer on 3d heterogeneous molecular graphs. *arXiv preprint arXiv:2110.01191*, 2021.

[8] Sutanay Choudhury, Jenna A. Bilbrey, Logan T. Ward, Sotiris S. Xantheas, Ian T. Foster, Joseph P. Heindel, Ben Blaiszik, and Marcus E. Schwarting. HydroNet: Benchmark tasks for preserving intermolecular interactions and structural motifs in predictive and generative models for molecular data. *Machine Learning and the Physical Sciences Workshop at the 34th Conference on Neural Information Processing Systems (NeurIPS)*, arXiv:2012.00131, 2020.

[9] Database of water clusters, Accessed September 2022. `https://sites.uw.edu/wdbase`.

[10] Jenna A Bilbrey, Joseph P Heindel, Malachi Schram, Pradipta Bandyopadhyay, Sotiris S Xantheas, and Sutanay Choudhury. A look inside the black box: Using graph-theoretical descriptors to interpret a continuous-filter convolutional neural network (CF-CNN) trained on the global and local minimum energy structures of neutral water clusters. *The Journal of Chemical Physics*, 153(2):024302, 2020.

[11] Kaiyuan Guo, Shulin Zeng, Jincheng Yu, Yu Wang, and Huazhong Yang. A survey of FPGA-based neural network accelerator. *arXiv preprint arXiv:1712.08934*, 2017.

[12] Maurizio Capra, Beatrice Bussolino, Alberto Marchisio, Muhammad Shafique, Guido Masera, and Maurizio Martina. An updated survey of efficient hardware architectures for accelerating deep convolutional neural networks. *Future Internet*, 12(7):113, 2020.

[13] Nathan C Frey, Siddharth Samsi, Joseph McDonald, Lin Li, Connor W Coley, and Vijay Gadepally. Scalable geometric deep learning on molecular graphs. In *NeurIPS 2021 AI for Science Workshop*, 2021.

[14] KT Schütt, Pan Kessel, Michael Gastegger, KA Nicoli, Alexandre Tkatchenko, and K-R Müller. SchNetPack: A deep learning toolbox for atomistic systems. *Journal of Chemical Theory and Computation*, 15(1): 448–455, 2018.

[15] Kristof T Schütt, Huziel E Sauceda, P-J Kindermans, Alexandre Tkatchenko, and K-R Müller. SchNet–a deep learning architecture for molecules and materials. *The Journal of Chemical Physics*, 148(24):241722, 2018.

[16] Avijit Rakshit, Pradipta Bandyopadhyay, Joseph P Heindel, and Sotiris S Xantheas. Atlas of putative minima and low-lying energy networks of water clusters n=3–25. *The Journal of Chemical Physics*, 151 (21):214307, 2019.

[17] George S Fanourgakis and Sotiris S Xantheas. The flexible, polarizable, Thole-type interaction potential for water (TTM2-F) revisited. *The Journal of Physical Chemistry A*, 110(11):4100–4106, 2006.

[18] Volodymyr Babin, Claude Leforestier, and Francesco Paesani. Development of a "first principles" water potential with flexible monomers: Dimer potential energy surface, VRT spectrum, and second virial coefficient. *Journal of Chemical Theory and Computation*, 9(12):5395–5403, 2013.

[19] Volodymyr Babin, Gregory R Medders, and Francesco Paesani. Development of a "first principles" water potential with flexible monomers. II: Trimer potential energy surface, third virial coefficient, and small clusters. *Journal of Chemical Theory and Computation*, 10(4):1599–1607, 2014.

[20] Johannes Moe, Konstantin Pogorelov, Daniel Thilo Schroeder, and Johannes Langguth. Implementating spatio-temporal graph convolutional networks on Graphcore IPUs. In *2022 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pages 45–54. IEEE, 2022.

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

[22] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. In *International Conference on Learning Representations*, arXiv:1905.12265, 2020.

[23] Stefan Chmiela, Alexandre Tkatchenko, Huziel E Sauceda, Igor Poltavsky, Kristof T Schütt, and Klaus-Robert Müller. Machine learning of accurate energy-conserving molecular force fields. *Science Advances*, 3(5):e1603015, 2017.

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 (OPEN SOURCE)
   (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? N/A
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? N/A
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? N/A
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? N/A
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? N/A