Scalable Geometric Deep Learning on Molecular Graphs

Anonymous Author(s)
Affiliation
Address
email

Abstract

Deep learning in molecular and materials sciences is limited by the lack of integration between applied science, artificial intelligence, and high-performance computing. Bottlenecks with respect to the amount of training data, the size and complexity of model architectures, and the scale of the compute infrastructure are all key factors limiting the scaling of deep learning for molecules and materials. Here, we present *LitMatter*, a lightweight framework for scaling molecular deep learning methods. We train four graph neural network architectures on over 400 GPUs and investigate the scaling behavior of these methods. Depending on the model architecture, training time speedups up to 60x are seen. Empirical neural scaling relations quantify the model-dependent scaling and enable optimal compute resource allocation and the identification of scalable molecular geometric deep learning model implementations.

1 Introduction

Many blockbuster results in deep learning are enabled by immense scale. Successes in natural language processing, computer vision, and applied scientific domains like protein folding rely on models that effectively scale to billions of parameters and tens or hundreds of GPUs. A distinguishing feature of these deep-learning-driven results compared to "vanilla" machine learning methods is their "unreasonable effectiveness" and ability to leverage increasing resources (data, model complexity, compute infrastructure) without quickly hitting asymptotic limits on performance [1]. Furthermore, large models hold the promise of emergence [2], wherein behaviors arise that cannot be anticipated by studying smaller models. To achieve this level of impact in scientific domains such as chemistry and materials science, there is a need to identify and investigate scalable model architectures that operate on molecular graphs.

In the field of molecular machine learning (ML), there is a staggering number of model architectures and methods to choose from [3], but most are never tested beyond the low-resource limit [4] (small benchmark datasets, small models, and single-GPU training). This makes it difficult to identify which methods are robust enough to be production ready for use cases beyond standard benchmarks [5]. For most physical scientists, it is difficult to justify the engineering effort and additional experimentation required to apply an existing method to their problem of interest. Moreover, in low-resource settings, deep learning methods struggle to outperform baselines of simple models (i.e., random forests, logistic regression) and human-crafted representations (i.e., chemical fingerprint vectors) on molecular ML tasks [4, 6]. To scale molecular ML beyond the low-resource limit, Figure 1 summarizes some of the key driving factors: training data availability, expressivity of models for learning molecular representations, and the availability of compute resources needed for hyperparameter optimization, architecture search, and model training. The availability of large-scale molecular and materials databases provides an unprecedented wealth of heterogeneous datasets and

Submitted to NeurIPS 2021 AI for Science Workshop.

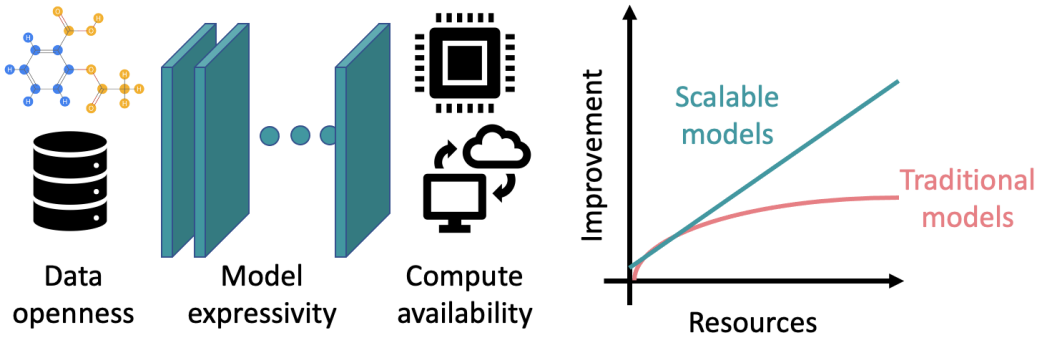

Figure 1: Bottlenecks to scaling geometric deep learning on molecular graphs.

learning tasks for molecular modeling. Due to the natural graph structure in condensed matter systems such as molecules, polymers, and crystalline materials [7], geometric deep learning models invoke the necessary geometric priors for representing molecular data manifolds [8, 9], in the same way that transformers and convolutional layers are natural network topologies for language and images, respectively. However, there still remains a need to investigate the scaling behavior of these models with respect to the available compute infrastructure.

Here, we investigate the scaling behavior of molecular graph neural networks via distributed training on many GPUs. We present the *LitMatter* template, a lightweight framework for rapid prototyping and scaling deep learning on molecular graphs. We train four state-of-the-art GNN models using up to 416 GPUs. We find that different standard model implementations achieve training time speedups as dramatic as 60x and as low as 3.7x, when scaling the number of GPUs from 2 to 416, depending on the model's ability to leverage increasing compute resources. Neural scaling relations [10, 11, 12] are proposed to quantify the model-dependent scaling. These empirical scaling relations can be used to identify models that benefit from increasing compute and to optimally allocate compute resources for efficient training of large models. We envision that the results presented here will help to "bridge the gaps" between high-performance computing, AI, and molecular science.

2 LitMatter

To facilitate easy distributed training of GNNs and other model architectures relevant for molecular and materials science, we present *LitMatter*, a lightweight template for rapid experimentation and scaling. The main features of *LitMatter* are summarized in the schematic in Figure 2. *LitMatter* uses *PyTorch Lightning* [13] to organize code so that engineering best practices are enabled by default and to reduce the overhead required to enable distributed training. *LitMatter* uses the the Distributed Data Parallel [14] accelerator for multi-GPU training. The template is designed to span an entire deep learning workflow, from rapid prototyping to end-to-end training/inference pipeline on hundreds of GPUs. Arbitrary PyTorch models and datasets/dataloaders can be "injected" into the template; and training can be launched interactively in a Jupyter notebook, through the command line, or scheduled through a SLURM job *without modifying or adding any code*. Using the *LitMatter* template allows any researcher to run the experiments presented here, or investigate the scaling behavior of other models of interest.

3 Scaling Experiments

We are interested in identifying GNN model architectures that leverage increasing amounts of compute. For reproducibility and simplicity, we choose four state-of-the-art GNN architectures implemented in *PyTorch Geometric* [15] - DimeNet [16], NNConv [17], SchNet [18], and Principal Neighborhood Aggregation (PNA) [19]. DimeNet, NNConv, and SchNet were trained to predict quantum chemical properties of small molecules from the QM9 [20] dataset, while PNA was trained on regression tasks for 250K compounds from the ZINC15 [21] database. All model implementations

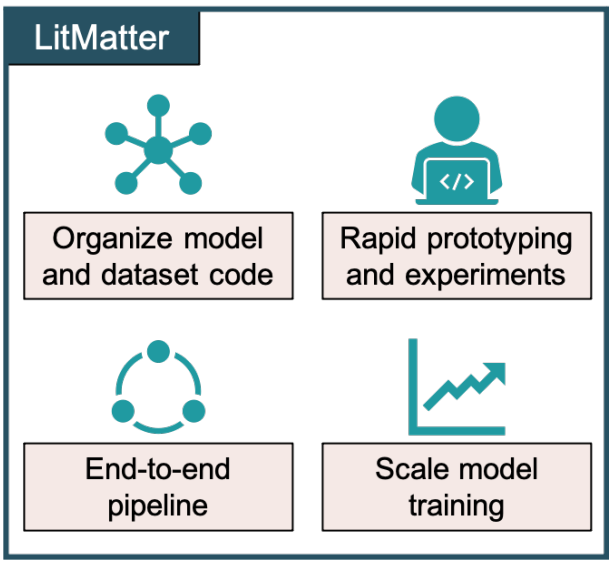

Figure 2: LitMatter: a template for rapid experimentation and scaling deep learning models on molecular graphs.

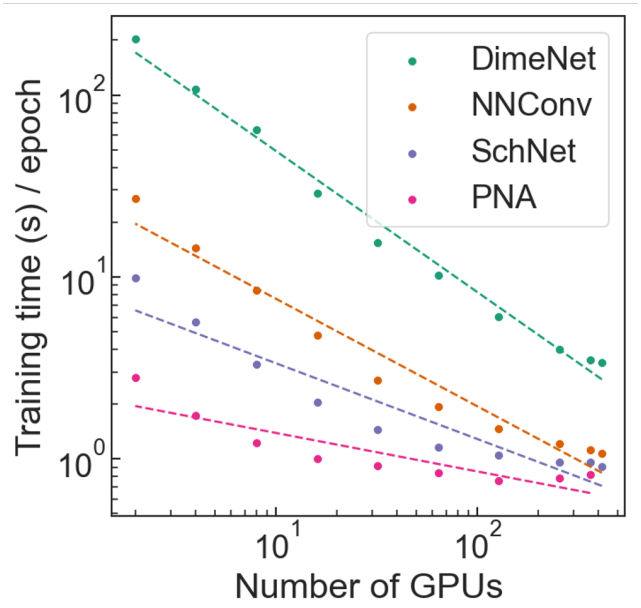

Figure 3: Training time per epoch versus number of GPUs for various GNN models.

and datasets were taken directly from *PyTorch Geometric*. All hyperparameters were set to the values provided in the *PyTorch Geometric* examples.

Multi-node, multi-GPU training was achieved using the *LitMatter* framework. Each model was trained for a fixed number of epochs (200 for DimeNet and 1000 for all other models) and with a fixed batch size of 128 on a variable number of GPUs. The total training time was logged for each model on 2, 4, 8, 16, 32, 64, 128, 256, 364, and 416 GPUs (with the exception of PNA on 416 GPUs). Then the training time per epoch was calculated for each model for each training run. Models were trained on NVIDIA Volta V100 Tensor Core GPUs.

The results of the scaling experiments are shown in Figure 3. The most extreme speedup is a 50x performance improvement for DimeNet, which has a training time of over 200 s / epoch on two GPUs

Table 1: Power law exponents

Model	Number of Parameters	β	R^2
DimeNet	$2.1 * 10^6$	0.78	0.99
NNConv	$6.2 * 10^5$	0.59	0.96
SchNet	$4.6 * 10^5$	0.42	0.90
PNA	$6.8 * 10^5$	0.21	0.79

that decreases to less than 4 s / epoch on 416 GPUs. NNConv training time decreases from 27 s / epoch (2 GPUs) to 1.08 s / epoch (416 GPUs) for a 25x speedup. SchNet's training time improves from 9.9 s / epoch (2 GPUs) to 0.91 s / epoch (416 GPUs) for an 11x speedup. PNA shows the least dramatic performance improvement, with a training time of 2.8 s / epoch on 2 GPUs and a minimum training time of 0.76 s / epoch on 128 GPUs, corresponding to a 3.7x speedup. PNA is the only model that does not exhibit a monotonic decreases in training time per epoch with increasing number of GPUs, due to inter-node communication bottlenecks at higher counts of compute nodes. For all four models, we observe that there is a clear inflection point at 32 GPUs where allocating additional compute yields diminishing returns. This could reflect a limitation of the dataset sizes (~100K samples for QM9 and 250K for ZINC15), bottlenecks in the inter-node communications, and/or constraints on model complexity.

4 Power Law Scaling

To further quantify the scaling behavior of the GNN architectures, we fit the scaling data to power laws of the form

$$t = \alpha n^{-\beta} \tag{1}$$

where t is the training time per epoch and n is the compute (number of GPUs). α is a constant and β is the scaling exponent that determines the slope of the power law scaling.

The DimeNet data exhibits almost a perfect power law fit ($R^2 = 0.99$) with $\beta = 0.78$. The magnitude of the scaling exponent reflects the speedup achievable for DimeNet. Table 1 summarizes the model size, scaling exponent, and goodness of fit for the power law fits for all GNN architectures.

The decreasing magnitude of β going from DimeNet to NNConv, SchNet, and PNA indicates the relative speedup each model achieves through multi-GPU training. All power law fits have $R^2 >= 0.9$ except PNA, which shows a deviation from power law scaling above 128 GPUs due to the previously mentioned inter-node communication bottleneck.

5 Discussion

In this paper we presented the *LitMatter* framework to overcome difficulties in scaling molecular deep learning, enabling researchers to rapidly experiment with and scale deep learning methods on molecular graphs. We showed the scaling behavior of molecular graph neural networks with respect to compute. DimeNet, a GNN with continuous equivariance to rotations (the SO(3) group), exhibits the most dramatic reduction in training time with increased compute. Because equivariant networks like DimeNet use spherical Bessel functions and spherical harmonics to construct representations, they have significant memory requirements and can be difficult to train. SOTA benchmark results [16] and the attractive properties of equivariance suggest that equivariant GNNs may be the "ideal" network topology for molecular ML. The scaling behavior of GNNs was quantified by fitting the data to power laws and extracting the scaling exponents. The empirical scaling relations succinctly encapsulate the results of our experiments and can be used in a practical setting to optimally allocate compute resources during training. Integrating our scaling results with previous work on GNN pre-training strategies [22] and demonstrations of the sample-efficiency and ability of large models to avoid overfitting through early stopping [2, 12] suggests a recipe for large GNNs that are trained at scale and fine-tuned to downstream molecular ML tasks. Next steps for this work include investigating molecular graph neural scaling behavior with respect to other limiting factors (dataset and model size) and identifying a candidate model architecture with optimal scaling behavior that can serve as a molecular graph "foundation model."

References

[1] Terrence J. Sejnowski. The unreasonable effectiveness of deep learning in artificial intelligence. *Proceedings of the National Academy of Sciences*, 117(48):30033–30038, 2020. ISSN 0027-8424. doi: 10.1073/pnas.1907373117. URL https://www.pnas.org/content/117/48/30033.

[2] Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Quincy Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Kohd, Mark Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Ben Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, Julian Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Rob Reich, Hongyu Ren, Frieda Rong, Yusuf Roohani, Camilo Ruiz, Jack Ryan, Christopher Ré, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishnan Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. On the opportunities and risks of foundation models, 2021.

[3] Keith T. Butler, Daniel W. Davies, Hugh Cartwright, Olexandr Isayev, and Aron Walsh. Machine learning for molecular and materials science. *Nature*, 559, 2018. ISSN 14764687. doi: 10.1038/s41586-018-0337-2.

[4] Aneesh Pappu and Brooks Paige. Making graph neural networks worth it for low-data molecular machine learning, 2020.

[5] Vijay Prakash Dwivedi, Chaitanya K. Joshi, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking graph neural networks, 2020.

[6] Andreas Mayr, Günter Klambauer, Thomas Unterthiner, Marvin Steijaert, Jörg K. Wegner, Hugo Ceulemans, Djork-Arné Clevert, and Sepp Hochreiter. Large-scale comparison of machine learning methods for drug target prediction on chembl. *Chem. Sci.*, 9:5441–5451, 2018. doi: 10.1039/C8SC00148K. URL http://dx.doi.org/10.1039/C8SC00148K.

[7] Steven Kearnes, Kevin McCloskey, Marc Berndl, Vijay Pande, and Patrick Riley. Molecular graph convolutions: moving beyond fingerprints. *Journal of Computer-Aided Molecular Design*, 30(8):595–608, Aug 2016. ISSN 1573-4951. doi: 10.1007/s10822-016-9938-8. URL http://dx.doi.org/10.1007/s10822-016-9938-8.

[8] Michael M. Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: Going beyond euclidean data. *IEEE Signal Processing Magazine*, 34 (4):18–42, 2017. doi: 10.1109/MSP.2017.2693418.

[9] Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges, 2021.

[10] Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md. Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou. Deep learning scaling is predictable, empirically, 2017.

[11] Yasaman Bahri, Ethan Dyer, Jared Kaplan, Jaehoon Lee, and Utkarsh Sharma. Explaining neural scaling laws, 2021.

[12] Tom Henighan, Jared Kaplan, Mor Katz, Mark Chen, Christopher Hesse, Jacob Jackson, Heewoo Jun, Tom B. Brown, Prafulla Dhariwal, Scott Gray, Chris Hallacy, Benjamin Mann, Alec Radford, Aditya Ramesh, Nick Ryder, Daniel M. Ziegler, John Schulman, Dario Amodei, and Sam McCandlish. Scaling laws for autoregressive generative modeling, 2020.

[13] WA Falcon et al. Pytorch lightning. *GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning*, 3, 2019.

[14] Shen Li, Yanli Zhao, Rohan Varma, Omkar Salpekar, Pieter Noordhuis, Teng Li, Adam Paszke, Jeff Smith, Brian Vaughan, Pritam Damania, and Soumith Chintala. Pytorch distributed: Experiences on accelerating data parallel training, 2020.

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

[16] Johannes Klicpera, Janek Groß, and Stephan Günnemann. Directional message passing for molecular graphs, 2020.

[17] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. 4 2017. URL `http://arxiv.org/abs/1704.01212`.

[18] Kristof T. Schütt, Pieter-Jan Kindermans, Huziel E. Sauceda, Stefan Chmiela, Alexandre Tkatchenko, and Klaus-Robert Müller. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions, 2017.

[19] Gabriele Corso, Luca Cavalleri, Dominique Beaini, Pietro Liò, and Petar Velickovic. Principal neighbourhood aggregation for graph nets. volume 2020-December, 2020.

[20] Raghunathan Ramakrishnan, Pavlo O Dral, Matthias Rupp, and O Anatole von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data*, 1, 2014.

[21] Teague Sterling and John J. Irwin. Zinc 15 - ligand discovery for everyone. *Journal of Chemical Information and Modeling*, 55:2324–2337, 11 2015. ISSN 15205142. doi: 10.1021/acs.jcim.5b00559. URL `https://ClinicalTrials.gov`.

[22] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks, 2020.