Journal of Machine Learning Research 25 (2024) 1-26 Submitted 5/23; Revised 8/23; Published 3/24

PhAST: Physics-Aware, Scalable, and Task-Specific GNNs for Accelerated Catalyst Design

Alexandre Duval* ALEXANDRE.DUVAL@MILA.QUEBEC
Mila, Inria, CentraleSupelec

Victor Schmidt* SCHMIDTV@MILA.QUEBEC
Mila, Universtité de Montréal

Santiago Miret SANTIAGO.MIRET@INTEL.COM
Intel Labs

Yoshua Bengio YOSHUA.BENGIO@MILA.QUEBEC
Mila, Université de Montréal, CIFAR Fellow

Alex Hernández-García ALEX.HERNANDEZ-GARCIA@MILA.QUEBEC
Mila, Université de Montréal

David Rolnick DAVID.ROLNICK@MILA.QUEBEC
Mila, McGill University

Editor: Shakir Mohamed

Abstract

Mitigating the climate crisis requires a rapid transition towards lower-carbon energy. Catalyst materials play a crucial role in the electrochemical reactions involved in numerous industrial processes key to this transition, such as renewable energy storage and electrofuel synthesis. To reduce the energy spent on such activities, we must quickly discover more efficient catalysts to drive electrochemical reactions. Machine learning (ML) holds the potential to efficiently model materials properties from large amounts of data, accelerating electrocatalyst design. The Open Catalyst Project OC20 dataset was constructed to that end. However, ML models trained on OC20 are still neither scalable nor accurate enough for practical applications. In this paper, we propose task-specific innovations applicable to most architectures, enhancing both computational efficiency and accuracy. This includes improvements in (1) the graph creation step, (2) atom representations, (3) the energy prediction head, and (4) the force prediction head. We describe these contributions, referred to as PhAST, and evaluate them thoroughly on multiple architectures. Overall, PhAST improves energy MAE by 4 to 42% while dividing compute time by 3 to 8× depending on the targeted task/model. PhAST also enables CPU training, leading to 40× speedups in highly parallelized settings. Python package: `https://phast.readthedocs.io`.

Keywords: climate change, scientific discovery, material modeling, graph neural networks, electrocatalysts.

1. Introduction

To mitigate climate change at a global scale, it is imperative to reduce the carbon emissions of ubiquitous industrial processes like cement production or fertiliser synthesis, as well as to develop infrastructures for storing low-carbon energy at scale, enabling to re-use it wherever and whenever needed. Since such processes rely on electrochemical reactions, they require the design of more efficient electrocatalysts (Zakeri and Syri, 2015) to become more environmentally and economically viable.

However, discovering easy-to-exploit low-cost catalysts that drive electrochemical reactions at high rates remains an open challenge. In fact, today's catalyst discovery pipeline mostly relies on expensive quantum mechanical simulations such as the Density Functional Theory (DFT) to approximate the behaviour of the materials involved in the targeted chemical reaction. Unfortunately, the high computational cost of these simulations limits the number of candidates that can be efficiently tested, and consequently stagnates further advances in the field.

Machine learning (ML) holds the potential to approximate these calculations while reducing the time needed to assess each candidate by several orders of magnitude (Zitnick et al., 2020). This capability could transform the search for new catalysts, by making it possible to sort through millions or even billions of possible materials to identify promising candidates for experimental inquiry(Zitnick et al., 2020).

To enable to use of ML for catalyst discovery, the Open Catalyst Project released OC20 (Chanussot et al., 2021), a large data set of pairs of catalyst and target molecule (known as *adsorbate*), along with the *relaxed energy* of the resulting system—a relevant metric to assess how good a catalyst is for a given chemical reaction—computed with DFT from the initial atomic structure. Despite recent progress (Gasteiger et al., 2021; Ying et al., 2021), major challenges remain. First, state-of-the-art models have not yet reached high enough performance for practical applications. Second, they are still too computationally expensive to allow the millions of inferences required to explore the large space of potential catalysts. Third, the graph neural networks (GNNs) typically used are designed for general 3D material modeling tasks rather than specifically for catalyst discovery, a complex task that may benefit from domain-specific architectures.

To address these challenges, we propose multiple model improvements to increase the accuracy and scalability of generic GNNs applied to catalyst discovery. In particular, our contributions are (1) a graph construction that is tailored to catalyst-adsorbate modeling, (2) richer physics-based atom representations, (3) an energy head that learns a weighted sum of per-atom predictions, and (4) a direct force prediction head encouraging energy conservation. We provide a broad evaluation of these contributions on OC20 and a thorough ablation study. In sum, the proposed PhAST improvements decrease energy MAE by 5–42 % while dividing compute time by 3–8$\times$ depending on the targeted task/model. These gains in model scalability enable efficient CPU training, with up to 40$\times$ speedups in highly parallelized pipelines using PhAST, making these models significantly more accessible to a wider community of researchers. We also believe that our work provides valuable insights for future research as it leverages domain-specific knowledge to improve parts of the pipeline that were not investigated up to now. Overall, the resulting performance and scalability gains open the door to a practical use of GNNs for new electrocatalyst design, the ultimate end goal of this line of research.

2. Background

The problem we address is the prediction of the relaxed energy $y \in \mathbb{R}$ of an adsorbate-catalyst system from its initial configuration in space $(\boldsymbol{X}, \boldsymbol{Z})$, where $\boldsymbol{X} \in \mathbb{R}^{N\times 3}$ is the matrix of 3D atom positions and $\boldsymbol{Z} \in \mathbb{N}^N$ contains atom characteristic numbers. This task is referred to as *Initial Structure to Relaxed Energy*, IS2RE, in Zitnick et al. (2020). This is commonly formulated as a graph regression task, where each sample is represented as a 3D graph $\mathcal{G}$ with node set $\mathcal{V}$ of dimension N and adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{N\times N}$. $\boldsymbol{H} \in \mathbb{R}^{N\times H}$ represents atom embeddings and $\boldsymbol{T} \in \{0, 1, 2\}^N$ corresponds to tag information (see 3.1). ML models designed for this task generally adopt graph neural networks as an architecture, as it naturally suits 3D material modeling. Such GNNs typically share a common pipeline for how they are applied, as depicted in Fig. 1.

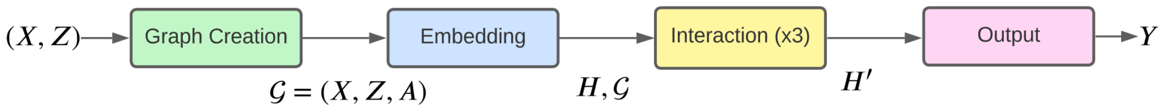


Figure 1: Common GNN inference pipeline for 3D material modeling. The graph creation step remains unchanged across all methods: it creates $\boldsymbol{A}$ using cutoff distances and periodic boundary conditions. The Embedding and Output blocks slightly differ across models but the underlying idea is the same. The Embedding block learns a representation for each chemical element and the Output block applies a global pooling of each node's representation to obtain the energy prediction. The key distinction between methods typically lies in the Interaction block, where the message passing schemes vary.

In material modeling tasks, it is desirable to endow ML models with relevant symmetry properties. In particular, we want predictions to be equivariant to translations, rotations and (often) reflections. Many models enforce these physical priors within the architecture, making it explicitly invariant or equivariant to the desired transformations. Formal definitions are included in Appendix A.1.

Many GNNs in prior work focus on enforcing equivariance, though it is not strictly required for relaxed energy prediction, which calls for invariance. Equivariant GNNs (Thomas et al., 2018; Anderson et al., 2019; Fuchs et al., 2020; Batzner et al., 2022; Brandstetter et al., 2021) are expressive and generalize well, but are very computationally expensive as they are constrained by equivariant filters built on spherical harmonics and the Clebsch-Gordan tensor product. Recent methods (Schütt et al., 2021; Satorras et al., 2021; Thölke and De Fabritiis, 2022) model equivariant interactions in Cartesian space using both invariant (scalar) and vector representations. While they are faster, their architectures are often very complex and lack theoretical guarantees. Alternatively, E(3)-invariant methods (Schütt et al., 2017; Unke and Meuwly, 2019; Shuaibi et al., 2021; Ying et al., 2021; Adams et al., 2021; Zitnick et al., 2022) do not use atom positions directly in their internal workings.

Instead, these methods extract and use quantities that remain invariant under rotations and reflections. DimeNet++ (Klicpera et al., 2020b,a), for example, includes a directional message passing (MP) mechanism that incorporates bond angles in addition to atom relative distances. However, distances and bond angles do not suffice to uniquely identify the graph 3D structure. This is achieved by SphereNet (Liu et al., 2021) and GemNet (Gasteiger et al., 2021), which additionally extract torsion information (between quadruplets of nodes). On the downside, these methods are very computationally expensive as they require considering 3-hop neighbourhoods for each update step. Importantly, all these Message Passing methods aim at broad applicability and do not leverage the specific constraints of individual tasks.

3. Proposed Method

In this section, we describe PhAST, a Physics-Aware, Scalable, and Task-specific GNN framework for catalyst design. Notably, the architectural innovations in our proposed framework are applicable to most current GNNs used in materials discovery. These include a novel graph creation step, richer atom representations, an advanced energy head for graph-level prediction, and a direct (energy-conserving) force-head for atom-wise predictions.

3.1 Graph creation

Although the graph construction step is critical in graph ML tasks, it has received little or no attention by previous work on the OC20 data set. Most methods reuse the original proposal by Chanussot et al. (2021). In OC20, each graph's atom is tagged as part of the adsorbate (tag 2), the catalyst's surface (tag 1), or its sub-surface volume (tag 0). Tag 0 atoms were originally added in DFT simulations to represent more explicitly the repeating pattern of the catalyst slab. They are, by definition, further away from the adsorbate and are fixed by construction, unlike tag 1 and tag 2 atoms, which can move during the relaxation. As a result, we hypothesise they contain redundant information, making them of lesser importance to predict the final relaxed energy. Besides, since they account for $\sim 65\%$ of the nodes B.5, and since SOTA GNNs often depend on multiple hops to compute bond and torsion angles, we propose to remove these nodes from the graph. This should greatly reduce inference time without impacting expressivity. As an alternative intermediate approach, we explore forming *super nodes* that aggregate tag 0 atoms to avoid a potential information loss caused by their total removal. We briefly describe these changes below, with more details in Appendix A.2.

remove-tag-0 removes all atoms with tag 0 (i.e. in $\mathcal{S} = \{i \in \mathcal{V} : t_i = 0\}$) from the graph, adapting correspondingly all graph attributes ($\boldsymbol{X}$, $\boldsymbol{A}$, $\boldsymbol{Z}$, $\boldsymbol{T}$, etc.).

one-supernode-per-graph aggregates all tag 0 nodes from $\mathcal{G}$ into a supernode s with position $\boldsymbol{x}_s = \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} \boldsymbol{x}_i$, adjacency $A_{is} = \max(A_{ij} : j \in \mathcal{S})$ and a new characteristic number z_s.

one-supernode-per-atom-type replicates the above strategy but creates one super node per distinct chemical element in the catalyst subsurface. Its attributes are defined based on its components, as previously.

3.2 Atom Embeddings

In all previously proposed GNN methods to solve energy-prediction tasks (e.g. IS2RE), atom representations are learned from scratch based on atomic number $\boldsymbol{H} = \boldsymbol{H_Z}$. We propose to leverage domain information to improve these representations. First, we hypothesise that whether a given atom belongs to the adsorbate, the catalyst surface, or its subsurface is important information. We therefore incorporate tag information into our model by utilizing a learnable embedding matrix $\boldsymbol{H_T}$ that encodes the tags as vectors. Second, we know from previous studies that some atomic properties (e.g. atomic radius or density) are useful for catalyst discovery (Takigawa et al., 2016; Ward et al., 2017). We leverage them as an additional embedding vector $\boldsymbol{H}_{\mathbf{F}}$ (see A.3 for the full list of properties). Lastly, we let the GNN learn embeddings for both the group and period information ($\boldsymbol{H}_{\mathbf{P}}, \boldsymbol{H}_{\mathbf{G}}$) since atoms belonging to the same group or period often share similar behaviours (Xie and Grossman, 2018). As a result, our proposed atom embedding $\boldsymbol{H}$ is a concatenation of all of the above: $\boldsymbol{H} = \boldsymbol{H}_Z || \boldsymbol{H}_T || \boldsymbol{H}_F || \boldsymbol{H}_P || \boldsymbol{H}_G$.

3.3 Energy head

In the literature, there is often limited focus on the energy head, which is the part of the output block responsible for the energy computation from final atom representations $\boldsymbol{h}_i^L$. To the best of our knowledge, all GNNs compute the relaxed energy using global pooling $\hat{y} = \sum_{i \in \mathcal{V}} h_i$, where node embeddings are reduced to a scalar h_i by linear layers. We identify two limitations in this procedure: First, all atoms are assigned the same importance, even though the properties of an atom are normally influenced by the properties of the element. Second, the graph topology is neglected by simply summing all atom encodings regardless of their 3D positions. To overcome these limitations, we explore alternative energy heads.

First, **a weighted sum of node representations**, which grants adaptive importance to each chemical element, expressed as $\hat{y} = \sum_{i \in \mathcal{V}} \alpha(\boldsymbol{h}_i^L) \cdot h_i$ or $\hat{y} = \sum_{i \in \mathcal{V}} \alpha(\boldsymbol{h}_i^0) \cdot h_i$, where the learnable importance weights $\alpha(\cdot)$ depend either on the embedding block initial encodings $\boldsymbol{h}_i^0$ or final ones $\boldsymbol{h}_i^L$.

Second, **a hierarchical pooling approach** endowed with the following energy head pipeline: $\boldsymbol{h}_i^L \rightarrow$ [Pooling $\rightarrow$ GCN] ($\times 2$) $\rightarrow$ Global Pooling $\rightarrow$ MLP $\rightarrow \hat{y}$. By applying a graph convolutional network (GCN, Kipf and Welling, 2016) on a coarsened graph, we propagate information differently, allowing us to capture hierarchical graph information. We implement HOSCPOOL (Duval and Malliaros, 2022): an end-to-end pooling operator that learns a cluster assignment matrix using a loss function inspired by motif spectral clustering.

3.4 Force head

A closely related task to IS2RE (i.e. energy prediction) consists in computing *forces* together with energy. This involves the additional prediction and training of atom-wise 3D vectors representing the forces currently applied on each atom by the rest of the system. This task is referred to as *Structure to Energy and Forces*, S2EF in Zitnick et al. (2020). In many previous works, atomic forces are directly computed as the predicted energy's gradient with respect to atom positions $\vec{y}_i = -\frac{\partial y}{\partial \boldsymbol{x}_i}$ (i.e. its definition in physics). While this guarantees energy-conserving forces[1], Kolluru et al. (2022) demonstrated the significant computational burden associated with this approach, which increases memory use by a factor of 2–4 and leads to decreased modeling performance for specific datasets. As a result, several recent works neglected this principle on OC20, proposing **direct force predictions** from final atom representations. Here, we extend this idea by proposing a plug-and-play force head architecture for traditional energy conserving GNNs (Schütt et al., 2017; Klicpera et al., 2020b). We thus use a shared backbone with two independent output heads: Φ^E denotes graph-level energy predictions and Φ^F denotes atom-level force predictions (both including the backbone).

We keep the traditional **energy loss** $\mathcal{L}_E = ||\hat{y} - y||_2$ and **force loss** $\mathcal{L}_F = \sum_i ||\hat{\vec{y}}_i - \vec{y}_i||_2$ to train our network, respectively pushing predicted energy $\hat{y} = \Phi^E(\mathcal{G})$ towards its ground truth value y and predicted forces $\hat{\vec{y}}_i = \Phi^F_i(\mathcal{G})$ towards their ground truth value $\vec{y}_i$, i.e. the negative energy gradient. However, using $\mathcal{L}_E$ and $\mathcal{L}_F$ does not guarantee that $\hat{\vec{y}}_i$ and $-\frac{\partial \hat{y}}{\partial \boldsymbol{x}_i}$ will be aligned since both are predicted separately.

To encourage energy conservation in the presence of a force head, we propose a new **gradient-target loss** term: the L_2 (squared) distance between atomic force predictions and the negative energy gradient with respect to atom positions. For a given graph:

$$\mathcal{L}_{Grad} = \frac{1}{|G|} \sum_{i \in \mathcal{G}} \left\| \hat{\vec{y}}_i - \left(-\frac{\partial \hat{y}}{\partial \boldsymbol{x}_i}\right) \right\|_2^2 \tag{1}$$

This term aims to correct for the possible misalignment between predicted forces and predicted energy, reinforcing the energy conserving character of our predictions. Alternatively, while the norm considers both the concept of direction and distance, we also study substituting the above term by a **cosine similarity loss** between predicted forces and the negative predicted energy gradient which mostly focuses on directional misalignment:

$$\mathcal{L}_{Cos} = \frac{1}{|G|} \sum_{i \in \mathcal{G}} \frac{\hat{\vec{y}}_i \cdot \left(-\frac{\partial \hat{y}}{\partial \boldsymbol{x}_i}\right)}{\|\hat{\vec{y}}_i\|_2 \cdot \|\left(-\frac{\partial \hat{y}}{\partial \boldsymbol{x}_i}\right)\|_2}. \tag{2}$$

While one could train with $\mathcal{L}_{Grad}$ or $\mathcal{L}_{Cos}$ all along, we empirically obtained slightly better performance by using this term at the end of training only, as fine-tuning. We hypothesize this is because we first want the GNN to properly predict energy and forces before making predictions more energy conserving.

3.5 PhAST: final components

In Section 5, we present the ablation study that lead us to the selection of the various components that make PhAST, across the four areas of improvements. All results displayed in Section 4 thus leverage these components. As an overview, we list them here:

1. Graph creation: *remove-tag-0*.

2. Atom embeddings: all embeddings $\boldsymbol{H} = \boldsymbol{H}_Z || \boldsymbol{H}_T || \boldsymbol{H}_F || \boldsymbol{H}_P || \boldsymbol{H}_G$.

1. a desirable feature in molecular dynamics, as it improves the stability of the simulation and the ability to reach local minima (Chmiela et al., 2017)

3. Energy head: weighted sum from initial embeddings.

4. Force head: direct force prediction with gradient target loss.

4. Evaluation

In this section, we evaluate the performance and scalability of the PhAST framework for five well-known GNNs on the OC20 dataset (Chanussot et al., 2021). We first perform an in-depth study of the first three components of PhAST on the OC20 IS2RE energy prediction task, before looking at the gains of the last component (i.e. the force head) on the OC20 S2EF-2M energy-force prediction task.

4.1 Baselines

We target five well-known GNN baselines to study the impact of our contributions, including state-of-the-art method GemNet-OC Gasteiger et al. (2022). We have selected them based on their popularity and ease-of-implementation but note that PhAST improvements are applicable to all recent GNNs for 3D material modeling, to the best of our knowledge, because the changes are architecture-agnostic. We use the hyperparameters, training settings and model architectures provided in the original papers. As mentioned in Figure 1, they all follow a similar pipeline, mainly differing in their interaction blocks, which we briefly detail below.

SchNet (Schütt et al., 2017) is a simple message passing architecture that leverages relative distances to update atom representations via a continuous filter: $\boldsymbol{h}_i^{(l+1)} = \sum_j \boldsymbol{h}_j^l \odot W^l(\boldsymbol{x}_i - \boldsymbol{x}_j)$ where $W^l(\boldsymbol{x}_i - \boldsymbol{x}_j)$ is a radial basis function to encode distance between atom pairs.

DimeNet++ (Klicpera et al., 2020a) is an optimised version of DimeNet (Klicpera et al., 2020b), which proposes a directional message passing. In other words, they compute and update edge representations instead of atoms) using interatomic distances $\mathbf{e}_{RBF}$ (encoded via bessel functions) and bond angles $\mathbf{a}_{SBF}$ (encoded via 2D spherical Fourier-Bessel basis):

$$\mathbf{m}_{ij}^{(l+1)} = f_{update}\big(\mathbf{m}_{ij}^{(l)}, \sum_{k \in N_j \setminus i} f_{int}(\mathbf{m}_{ij}^{(l)}, \mathbf{e}_{RBF}^{(ij)}, \mathbf{a}_{SBF}^{(ki,ji)})\big)$$

ForceNet (Hu et al., 2021) is a scalable force-centric GNN that does not impose explicit physical constraints (energy conservation, rotational invariance). It attempts to encourage invariance by efficient rotation-based data augmentation. Model-wise, it adopts a node message passing approach that leverages node positions directly via a spherical harmonics basis.

GemNet (Gasteiger et al., 2021) builds on top of DimeNet++, but additionally incorporates torsion information between quadruplets of atoms. This grants it the ability to process more geometric information and thus to distinguish between a wider range of different graphs, at the cost of extra computational cost and model complexity.

GemNet-OC (Gasteiger et al., 2022) is an improved version of GemNet.

4.2 PhAST performance on IS2RE

Dataset. OC20 contains 1,281,040 DFT relaxations of randomly selected catalysts and adsorbates from a set of plausible candidates. In this section, we focus on the *Initial Structure to Relaxed Energy* (IS2RE) task (Zitnick et al., 2020), that is the direct prediction of the relaxed adsorption energy from the initial atomic structure. It comes with a pre-defined train/val/test split, 450,000 training samples and hidden test labels. Experiments are evaluated on the validation set, which has four splits of $\sim 25K$ samples: In Domain (ID), Out of Domain adsorbates (OOD-ads), Out of Domain catalysts (OOD-cat), and Out of Domain adsorbates and catalysts (OOD-both).

Metrics. We measure accuracy via the energy *Mean Average Error* (MAE) in meV on each validation split, and scalability by the *inference time* (in seconds) over the whole ID validation set. We also include *throughput* in Table 4, i.e. the number of samples processed per second at inference time[2]. Since the absolute time metrics are difficult to compare across hardware setups with respect to other works, we note that the most relevant metrics are the *relative* improvements we show using the exact same hardware and software for all models. In order to easily visualize the contributions of PhAST on the baseline models in terms of performance improvement, in Figure 2 (left) we plot the relative MAE improvement with respect to the baseline. Specifically, we compute the MAE improvement as

$$\text{MAE improvement} = 100 \times \frac{\text{MAE(baseline)} - \text{MAE(PhAST)}}{\text{MAE(baseline)}}. \tag{3}$$

Baselines. We study the enhancements brought by the PhAST components (see Section 3.5) to five key GNN architectures for material modeling: SchNet (Schütt et al., 2017), DimeNet++ (Klicpera et al., 2020a), ForceNet (Hu et al., 2021), GemNet (Gasteiger et al., 2021) and GemNet-OC (Gasteiger et al., 2022). We compare every baseline with their PhAST counterpart, incorporating the best components of each category detailed in Section 3.5, that is graph creation (3.1), enriched atom embedding (3.2) and advanced energy-head (3.3), as determined by the ablation study conducted in Section 5.

Results. From Table 1 and Figure 2, we conclude that our set of PhAST enhancements consistently improve both MAE and inference time upon the original baselines. More precisely, PhAST improves *Average* MAE over the four validation splits by ~ 6.2 % on average across baselines, while reducing model inference time by $\sim 4.5\times$ (on average across all baselines). Moreover, we observe an MAE improvement of 12.4 % for SchNet and 9.7 % for DimeNet++ on val OOD-both, compared to a 7.7 % and 5.2 % for *Average* MAE. This suggests that PhAST models generalise better than original baselines. From the ablation study conducted in Section 5, we conclude that this is due to the combination of our extensions, as they all contribute to significantly better performance on out-of-distribution adsorbate-catalyst systems (OOD-both). Note that inference time gains with PhAST are almost doubled from SchNet, a 1-hop message passing (MP) approach, to DimeNet++, a 2-hops MP approach

2. *Throughput* differs from *inference time* as it only measures the on-device forward pass of the model, neglecting data-loading, inter-device transfers etc. While more theoretically relevant, it is also less practically informative, which is why we report both.

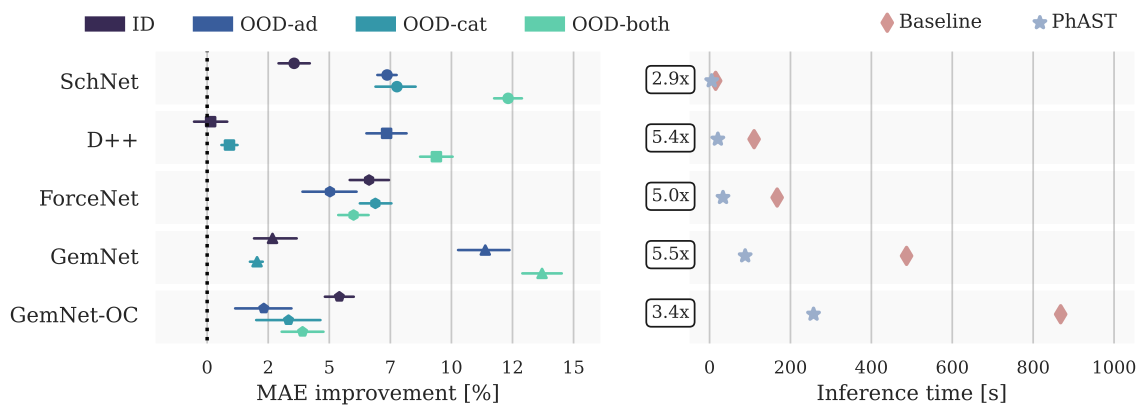


Figure 2: A comparison of the improvements brought by PhAST to the model's MAE (left) and inference times (right) on *OC20* IS2RE. The PhAST components are selected in Section 5 and summarised in Section 3.5. MAE improvements (left) are computed as in Equation 3 and values to the right of the dashed line at 0.0 denote an improvement of PhAST with respect to the baseline. The results are averaged over 3 runs, with bootstrapped confidence intervals represented by small horizontal bars. PhAST leads to a significant MAE improvement for each validation split, up to 13 %, in addition to decreasing inference time (s) by several factors (e.g. 3.4× or 5.4×). Absolute numerical values are provided in Table 1.

Baseline / MAE	ID	OOD-ad	OOD-cat	OOD-both	Average	Inference time (s)
SchNet	637	734	661	703	683	15 ± 0.49
PhAST-SchNet	618	677	611	616	**630**	$\mathbf{5 \pm 0.36}$
D++	571	722	561	661	628	110 ± 0.57
PhAST-D++	568	654	560	597	**595**	$\mathbf{20 \pm 0.63}$
ForceNet	658	701	632	628	654	167 ± 0.96
PhAST-ForceNet	612	664	592	597	**616**	$\mathbf{33 \pm 0.60}$
GemNet	573	808	571	744	674	487 ± 0.24
PhAST-GemNet	559	713	558	648	**619**	$\mathbf{88 \pm 0.10}$
GemNet-OC	593	658	605	584	610	868 ± 1.89
PhAST-GemNet-OC	564	636	587	562	**588**	$\mathbf{257 \pm 0.09}$

Table 1: MAE and inference time for various GNNs and their PhAST counterpart on *OC20* IS2RE, averaged over 3 runs. *Average* MAE is computed over all validation splits. PhAST models all show improved accuracy and drastic speedups. Note that PhAST-SchNet almost matches the original DimeNet++ while being 21 times faster. A graphical visualisation of the inference time and of relative MAE improvement is provided in Figure 2. Seamingly suboptimal MAE results for the two baseline GemNet models are explained in Appendix B.2.

(from 3× to 5.5× speedup)[3]. Throughput scores provided in Table 4 support the results obtained from inference time.

3. A "limited" speedup of 3.4× on GemNet-OC can be explained by the fact that it is a bigger but more efficient version of the original GemNet architecture.

4.3 PhAST performance on S2EF

Dataset. In this section, we focus on the *Structure to Energy and Forces* (S2EF-2M) OC20 dataset, that is, the prediction of both the overall energy and atom forces, from a set of 2 million 3D material structures. According to the dataset creators, the 2M split closely approximates the much more expensive full S2EF dataset, making it suitable for model evaluation (Gasteiger et al., 2022). It also come with pre-defined train/val/test splits[4].

Metrics. Both Energy MAE (E-MAE) and Forces MAE (F-MAE) are used to measure model accuracy. Regarding scalability, we continue to use the inference time (seconds) over the ID validation set as well as the number of samples per seconds processed by the model at inference time (throughput).

Baselines. We re-use the same baselines as above, leaving aside GemNet and GemNet-OC given the increased computational scale of this new dataset and the size of those two models[5]. Since ForceNet already has a direct force prediction head, unlike SchNet and D++, we implemented ForceNet-FE which computes forces as the gradient of the energy with respect to atom positions (denoted *FE*, i.e. from energy) in order to assess the added value of the force head. *PhAST-FE* includes the components of the previous subsection (graph creation, atom embedding, energy head) and computes forces using the energy gradient while PhAST additionally contains the best performing force head, determined in Section 5.

Results. From Table 2, we conclude that (1) *PhAST-FE* improvements are also very significant on S2EF. They lead to better modeling accuracy (13% E-MAE improvement) and lower compute time (inference time divided by 4.4) across all three models. (2) Including the proposed PhAST force-head yields significantly better energy MAE than original *PhAST-FE* and it reduces memory usage by a factor of 2 to 4 as well as compute time by a smaller factor.

PhAST improves Energy MAE by 32% compared to base models and by 22% compared to *PhAST-FE* while suffering from a 7% drop in Force MAE (on average across all three GNNs). Compared to baselines, PhAST multiplies throughput by 10-15× and divides inference time by a factor of 4-8. Compared to energy-focused PhAST enhancements, it reduces inference time by 31% on average and increases throughput by a factor of 2.4×. These scalability gains arise both from avoiding to compute the gradient and from increasing batch size given saved memory space[6]. Lastly, we manage to make force prediction slightly more energy-conserving by using the gradient-target loss term, although the improvement is relatively small. A more detailed analysis can be found in Section 5.

5. Ablation study

In this section, we provide the results of a careful ablation study assessing the contribution to both accuracy and reduction in compute time of each of the proposed components of

4. Similarly to IS2RE, the S2EF validation dataset comes in 4 distinct splits with a cumulative total of 1M samples: ID, OOD-ad, OOD-cat, OOD-both.
5. For reference, GemNet-OC is trained for 2800 GPU hours, a computational budget we could not afford.
6. The enabled increase in batch size explains how throughput and inference time improve differently: while isolated forward passes can scale linearly with batch size due to GPU parallelism (throughput), the data loading of larger batches can be a bit slower (inference time). As explained before, we keep both figures because of the theoretical/practical gains trade-off.

Baseline / MAE	E-MAE	F-MAE	EC	Throughput (s/s)	Inference time (s)
SchNet	1014	70.6	0	519 ± 33	2050 ± 15
PhAST-FE-SchNet	890	**68.4**	0.27	2404 ± 125	593 ± 04
PhAST-SchNet	**595**	77.2	**0.22**	$\mathbf{6042 \pm 387}$	$\mathbf{477 \pm 03}$
D++	913	69.2	0	103 ± 14	10189 ± 114
PhAST-FE-D++	813	**67.9**	0.11	615 ± 40	1687 ± 05
PhAST-D++	**636**	83.6	**0.10**	$\mathbf{1522 \pm 205}$	$\mathbf{1259 \pm 26}$
ForceNet	721	68.6	0.25	227 ± 25	4524 ± 47
ForceNet-FE	765	190	0	105 ± 14	9191 ± 18
PhAST-ForceNet-FE	607	157	0	505 ± 40	2117 ± 03
PhAST-ForceNet	**542**	**63.9**	**0.19**	$\mathbf{1090 \pm 177}$	$\mathbf{1357 \pm 77}$

Table 2: A comparison of energy and forces prediction, throughput and inference time of *PhAST-FE* and PhAST (i.e. with force-head improvements) on the baseline GNN models on *OC20* S2EF. E-MAE and F-MAE denote respectively the Average Energy/Force MAE computed over all validation splits. Inference time (sec.) and Throughput (samples/sec. processed during inference) are averaged over 3 runs.

PhAST. Note that due to the size and computational cost of GemNet and GemNet-OC, we did not conduct a full ablation study on those two models.

5.1 Selecting IS2RE PhAST components

We study the accuracy and scalability gains of

– the graph creation step contributions: (1) remove-tag-0 (2) one-supernode-per-graph (denoted as *sn-graph* in the table) (3) one-supernode-per-atom-type (*sn-atom-type*); all described in Section 3.1.

– the atom embeddings' contributions, where we include in addition to $\mathbf{H_Z}$: (1) tag embeddings $\mathbf{H_T}$ (denoted as *tag-embed* in the table) (2) physics-aware embeddings $\mathbf{H_F}$ (*phys-embed*) (3) learned physics aware embeddings (*l-phys-embed*) (4) period and group embeddings $\mathbf{H_P}, \mathbf{H_G}$ (*pg*) (5) all of them (*all*, i.e. 1-2-4 concatenated), all described in Section 3.2.

– the energy-head contributions: (1) a weighted sum of node representation from initial encodings (denoted as *w-init* in the table) (2) a weighted sum of node representation from final encodings (*w-final*) (3) the hierachical pooling operator (*hoscpool*), all described in Section 3.3.

Figure 3 shows graphical results of the MAE improvement computed as in Equation 3 of the ablation study whose exact numerical values can be found in Table 5, Table 6, Table 7 in the appendix. From Figure 3, we derived the following observations:

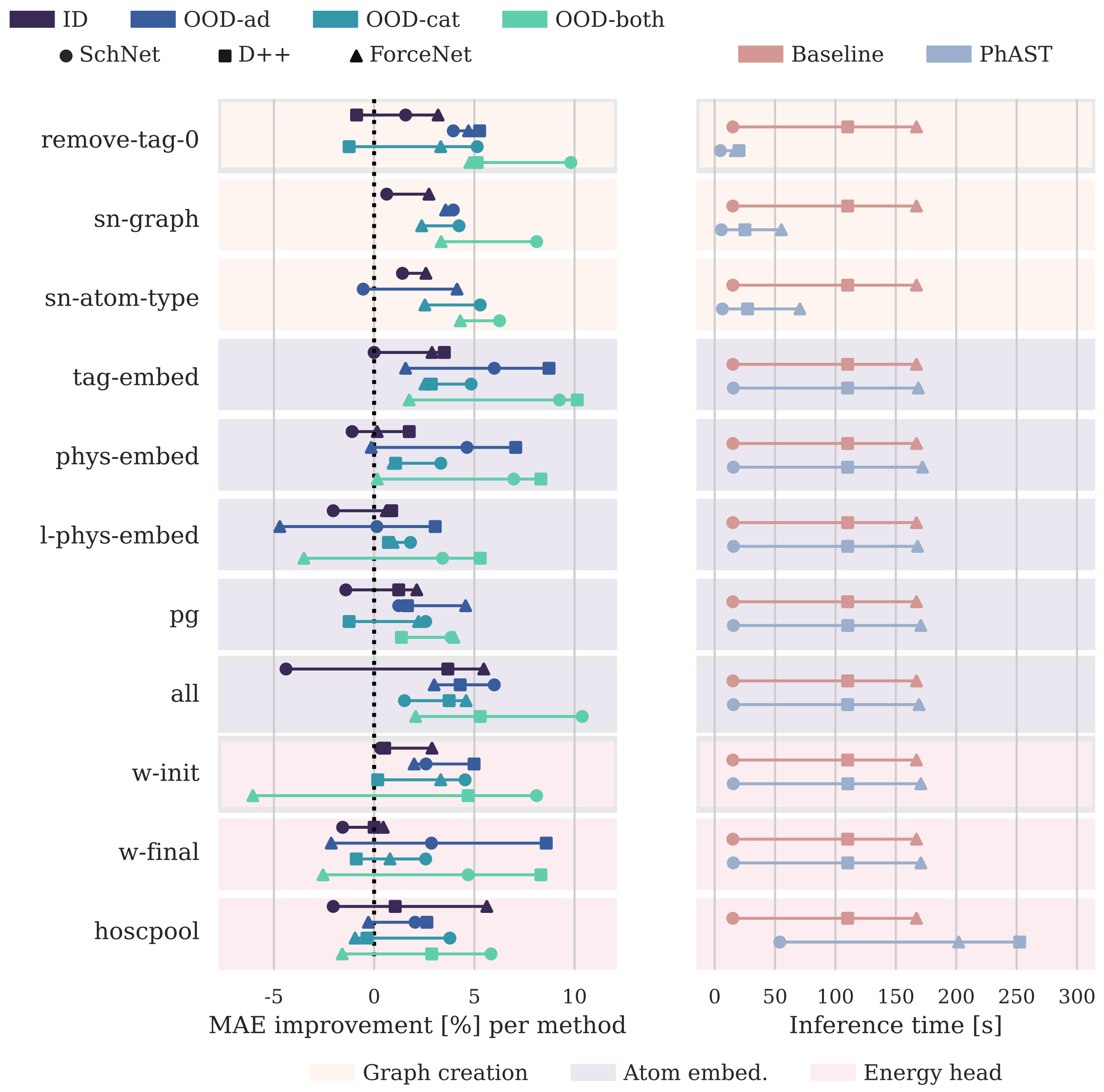


Figure 3: Ablation study results on IS2RE of our PhAST contributions about (1) the graph creation step, (2) atom embeddings and (3) the energy prediction head, all detailed in Section 3, for SchNet, DimeNet++ and ForceNet. All changes lead to significant improvements in model performance compared to the dash line denoting the baseline (Left). All contributions have little impact on inference time except from the graph rewiring steps which divides it by several factors (Right). The best technique from each family selected for PhAST are highlighted with a darker edge around the shaded background.

- Regarding our proposed improvements in the graph creation step, sub-surface atoms appear to contain redundant information as *remove-tag-0* does not cause performance drop and aggregating it into super nodes does not yield better results. We offer two potential explanations: (1) the data generation process of DFT simulations is not optimal and tag 0 does contain redundant information (2) ML models do not manage to extract meaningful information from this repeated pattern, in which case our approach could be used by future work to demonstrate a better usage of this long range context info. *remove-tag-0* also dramatically decreases compute time.

- Enriched atom embeddings improve performance and generalisation, especially when adding tag information (available in the data set). Notably, the combination of all embeddings provided the best results.
- Regarding the energy head, the hierarchical pooling approach is not very successful, either due to the difficulty of the task or the absence of hierarchical structures; but both energy-head weighted sums are beneficial.

In conclusion, we obtain the following best components for the PhAST version of our models: remove-tag-0, full concatenation of atom embeddings (all) and predicting the system energy as a learned weighted sum of per-atom predictions, from the initial embeddings (w-init).

5.2 Selecting S2EF PhAST components

Table 3 contains the result of an ablation study comparing the options described in Sections 3.4 and 4.3 to adapt PhAST to the OC20 S2EF data set. We study the accuracy / scalability trade-off of the following combinations:

– *FE*: original model, with forces predicted as the gradient of the energy prediction with respect to atom positions.

– *PhAST-FE*: PhAST enhancement of a baseline GNN model, with components selected in the IS2RE ablation study (Section 5.1).

– *PhAST-Direct*: PhAST model with the proposed direct force head.

– *PhAST-Grad*: PhAST model with direct force head and energy-grad loss from Eq.1.

– *PhAST-Cos*: PhAST model with direct force head and cosine similarity loss from Eq.2.

From Table 3, we draw the following observations:

- *PhAST-FE* yields significant compute time and MAE improvements for all baselines, similarly to what we saw on IS2RE in Sections 4.2 and 5.1. This confirms its relevance and generalization capabilities. To be more concrete, *PhAST-FE* leads to 13% and 4% improvements in E-MAE and F-MAE, respectively, while inference time is divided by 4.2 and throughput multiplied by 5.1$\times$ (averaged over all three baselines).
- *PhAST-Direct* yields additional computational gains compared to *PhAST-FE*. Indeed, direct force predictions avoids computing the energy gradient, which saves memory as we do not have to compute the energy's gradient in the *forward* pass. This extra memory space can be used to increase the batch size, leading to even lower total inference time.

Baseline / MAE	E-MAE	F-MAE	EC	Throughput	Inference time
SchNet-FE	1014	70.6	0	519 ± 33	2050 ± 15
PhAST-FE-SchNet	890	**68.4**	0	2404 ± 125	593 ± 04
PhAST-Direct-SchNet	716	81.2	0.27	6110 ± 390	455 ± 03
PhAST-Grad-SchNet	**619**	79.9	0.23	**6164 ± 411**	456 ± 03
PhAST-Cos-SchNet	667	84.9	−0.10	6078 ± 376	**435** ± 02
D++-FE	913	69.2	0	103 ± 14	10189 ± 114
PhAST-FE-D++	813	**67.9**	0	615 ± 40	1687 ± 05
PhAST-Direct-D++	663	83.7	0.11	1575 ± 184	1255 ± 28
PhAST-Grad-D++	**659**	83.7	0.10	**1621 ± 180**	1271 ± 12
PhAST-Cos-D++	713	83.9	−0.19	1607 ± 171	**1092 ± 94**
ForceNet	721	68.6	0.25	227 ± 25	4524 ± 47
ForceNet-FE	765	190	0	105 ± 14	9191 ± 18
PhAST-FE-ForceNet	607	157	0	505 ± 40	2117 ± 03
PhAST-Direct-ForceNet	**542**	**63.9**	0.19	**1090 ± 177**	1357 ± 77
PhAST-Grad-ForceNet	554	64.0	0.18	1066 ± 189	1294 ± 86
PhAST-Cos-ForceNet	700	80.1	−0.12	1034 ± 143	1262 ± 88

Table 3: Comparing model performance on *OC20* S2EF for baseline GNN, *PhAST-FE* GNN and the different PhAST force-head proposed enhancements whose description is provided in the Baselines paragraph above. E-MAE and F-MAE denote respectively the Average Energy/Force MAE computed over all val splits. Inference time (sec.) and Throughput (samples/sec. during inference) are averaged over 3 runs.

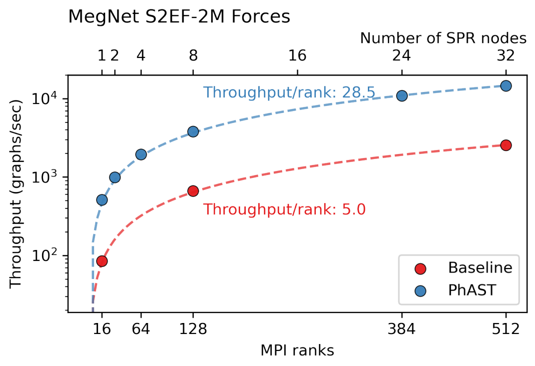
- *PhAST-Direct* also leads to significant gains in Energy MAE, at the cost of Force MAE points. We hypothesise that this happens because SchNet and DimeNet++ are invariant models and struggle to propagate equivariant information for accurate force predictions. This hypothesis is reinforced by the fact that it is not the case for ForceNet, which processes directional information.

- Adding a new loss term to encourage energy conservation (*Grad*, *Cos*) only has a small effect. Indeed, the difference between predicted forces and the energy gradient, given by the EC metric (see Equation (1)), only undergoes relatively small drops for *Grad* vs the *Direct* force head. However, it often leads to small performance gains, making it relevant nonetheless.

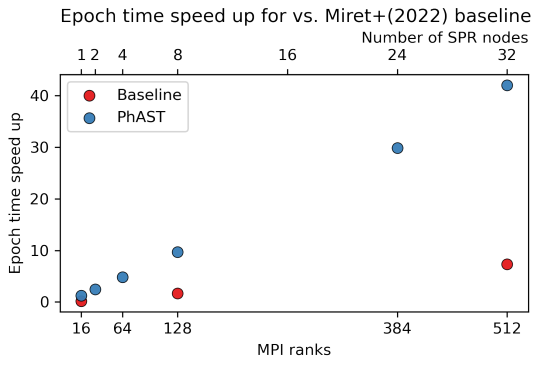
In conclusion, PhAST with direct force predictions using the gradient-target loss (*Grad*) is a desirable enhancement if one targets good energy prediction and/or high scalability. However, if interested in pure molecular dynamics, we suggest using PhAST without this force head.

6. CPU Training Enhancements

As described in previous sections, PhAST significantly improves the inference time of various models on the OpenCatalyst dataset. In this section, we'll also show how PhAST enables machine learning researchers to train their models on CPUs, thereby providing an additional hardware platform for different users. The greater abundance of CPUs compared to GPUs, especially in the computational chemistry community, makes the ability to effectively train models on CPUs highly desirable, as it unlocks the potential of training advanced ML models on OpenCatalyst for a larger set of users.



(a) MegNet throughput on S2EF-2M on Intel Sapphire Rapids (SPR) CPU. Parallization across multiple SPR nodes (2 CPUs per node) enables significant speedback in training throughput.



(b) Speedup compared to Miret et al. (2022) on Intel Sapphire Rapids (SPR). At the largest degree of parallezation with 512 MPI ranks containing individual processes, we achieve 40x speedup.

Figure 4: CPU-based training of MegNet (Chen et al., 2019) using 4th Gen Intel Xeon Scalable Processors known as Sapphire Rapids (SPR). The top of the x-axis specifies the number of CPU nodes with each node including 2 SPR CPUs, meaning that at the largest degree of parallization we run on 32 SPR with 64 CPUs. The bottom x-axis outlines the number of MPI ranks, which specifies the number of parallel MPI processes occuring at a given time. Using 32 SPR nodes, we can scale up to 512 MPI ranks which provide significant speedup in model trianing.

To implement PhAST CPU training, we leverage the Open MatSci ML Toolkit by Miret et al. (2022) which provides a unified platform for training deep learning on the OpenCatalyst dataset across different hardware platforms. Additionally, Miret et al. (2022) utilize the Deep Graph Library (DGL) as the platform for GNN development, which provides an additional proofpoint given that all prior experiments were performed using PyTorch Geometric. In this set of experiments, we focus on a distinct baseline architecture, MegNet (Chen et al., 2019), a GNN that was constructed for chemical modeling featuring node attributes, bond attributes and graph level attributes. We performed our training on fourth generation Intel Xeon Scalable processors (named Sapphire Rapids - SPR) and investigated the scalibility and compute of PhAST across multiple CPU nodes. All such experiments perform MegNet training on S2EF-2M with hyperparameters described in Miret et al. (2022).

The first set of observations we can make from our study is that applying PhAST significantly increases the throughput of processed graphs for training experiments as shown in Figure 4a. The throughput achieved with PhAST is greater than 5× the throughput achieved without PhAST and scales more favorably up 128 MPI ranks. This confirms the general trend observed in previous experiments providing further evidence that PhAST significantly increases the compute efficiency of both inference and training on the OpenCatalyst dataset.

The second set of observations we can make relates to the training speedup achieved by PhAST using advanced CPUs seen in Figure 4b, which shows that we can achieve up 40× speedup in training time combining PhAST with advanced CPUs compared to the GPU based machines used in Miret et al. (2022). Additionally, the increase in compute gained from PhAST is clearly shown when comparing to running the same training experiment without PhAST (shown in red on Figure 4b). At the largest degree of parallelization, regular training achieves 5× performance gain which can attributed to more advanced hardware, which is minimal compared to the ∼40× speedup achieved using PhAST.

7. Conclusion

In this work, we presented several enhancements targeted to catalyst discovery and applicable across a variety of existing GNN models. We showed that (1) enriching atom representations with physics-based properties, (2) tailoring the graph creation to the specific task at hand, (3) weighting atoms' importance when computing the system energy, (4) making direct force predictions with a energy conserving loss term, all reduce inference time significantly while leading to better accuracy. Besides, these gains in memory and running time make it possible to run models on CPUs, achieving up to 40× speedups and making these algorithms accessible to a greater number of researchers. Overall, our results also suggest that complex practical applications like catalyst discovery benefit from task-specific methods rather than general 3D material modeling GNNs and that performance and scalability gains can be achieved by focusing on all aspects of the pipeline instead of only the message passing block.

Additionally, we expect generative models to play a prominent role in catalyst discovery, replacing manual suggestion of promising new catalyst. In this paradigm, generative models require millions of calls to a GNN oracle to assess how good each catalyst is and explore the space of potential candidates accordingly. Due to its significant computational and accuracy gains, we believe that PhAST holds the potential to make a real difference, enabling the discovery of superior catalysts. This could lead to more efficient electrochemical reactions and thus contribute to reducing carbon emissions in industrial processes like fertilizer, cement, and green hydrogen production.

Finally, despite being designed for catalysis discovery, we anticipate that PhAST components will yield benefits in other application domains, such as QM9 (Ramakrishnan et al., 2014), QM7X(Hoja et al., 2021) and MD17 (Duvenaud et al., 2015), as well as generalizing to other GNN architectures.

Acknowledgments

This research is supported in part by ANR (French National Research Agency) under the JCJC project GraphIA (ANR-20-CE23-0009-01) as well as Samsung and Intel and was made possible thanks to Fragkiskos D. Malliaros, who also provided valuable feedback all along. Alexandre Duval acknowledges support from a Mitacs Globalink Research Award. Alex Hernandez-Garcia acknowledges the support of IVADO and the Canada First Research Excellence Fund. David Rolnick acknowledges support from the Canada CIFAR AI Chairs Program. The authors also acknowledge material support from NVIDIA and Intel in the form of computational resources, and are grateful for technical support from the Mila IDT team in maintaining the Mila Compute Cluster. The authors acknowledge the support of Kin Long Kelvin Lee in performing relevant training experiments on 4th Gen Intel Xeon Scalable Processors known as Sapphire Rapids.

References

Keir Adams, Lagnajit Pattanaik, and Connor W Coley. Learning 3d representations of molecular chirality with invariance to bond rotations. *arXiv preprint arXiv:2110.04383*, 2021.

Brandon Anderson, Truong Son Hy, and Risi Kondor. Cormorant: Covariant molecular neural networks. *Advances in neural information processing systems*, 32, 2019.

Simon Batzner, Albert Musaelian, Lixin Sun, Mario Geiger, Jonathan P Mailoa, Mordechai Kornbluth, Nicola Molinari, Tess E Smidt, and Boris Kozinsky. E (3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *Nature communications*, 13(1):1–11, 2022.

Johannes Brandstetter, Rob Hesselink, Elise van der Pol, Erik Bekkers, and Max Welling. Geometric and physical quantities improve e (3) equivariant message passing. *arXiv preprint arXiv:2110.02905*, 2021.

Lowik Chanussot, Abhishek Das, Siddharth Goyal, Thibaut Lavril, Muhammed Shuaibi, Morgane Riviere, Kevin Tran, Javier Heras-Domingo, Caleb Ho, Weihua Hu, et al. Open catalyst 2020 (oc20) dataset and community challenges. *ACS Catalysis*, 11(10):6059–6072, 2021.

Chi Chen, Weike Ye, Yunxing Zuo, Chen Zheng, and Shyue Ping Ong. Graph networks as a universal machine learning framework for molecules and crystals. *Chemistry of Materials*, 31(9):3564–3572, 2019.

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.

Alexandre Duval and Fragkiskos Malliaros. Higher-order clustering and pooling for graph neural networks. *arXiv preprint arXiv:2209.03473*, 2022.

David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. *Advances in neural information processing systems*, 28, 2015.

Fabian Fuchs, Daniel Worrall, Volker Fischer, and Max Welling. Se (3)-transformers: 3d roto-translation equivariant attention networks. *Advances in Neural Information Processing Systems*, 33:1970–1981, 2020.

Johannes Gasteiger, Florian Becker, and Stephan Günnemann. Gemnet: Universal directional graph neural networks for molecules. *Advances in Neural Information Processing Systems*, 34:6790–6802, 2021.

Johannes Gasteiger, Muhammed Shuaibi, Anuroop Sriram, Stephan Günnemann, Zachary Ulissi, C Lawrence Zitnick, and Abhishek Das. Gemnet-oc: developing graph neural networks for large and diverse molecular simulation datasets. *arXiv preprint arXiv:2204.02782*, 2022.

Johannes Hoja, Leonardo Medrano Sandonas, Brian G. Ernst, Alvaro Vazquez-Mayagoitia, Robert A. DiStasio Jr., and Alexandre Tkatchenko. Qm7-x, a comprehensive dataset of quantum-mechanical properties spanning the chemical space of small organic molecules. *Scientific Data*, 2021. doi: 10.1038/s41597-021-00812-2. URL `https://doi.org/10.1038/s41597-021-00812-2`.

Weihua Hu, Muhammed Shuaibi, Abhishek Das, Siddharth Goyal, Anuroop Sriram, Jure Leskovec, Devi Parikh, and C Lawrence Zitnick. Forcenet: A graph neural network for large-scale quantum calculations. *arXiv preprint arXiv:2103.01436*, 2021.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

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

Johannes Klicpera, Janek Groß, and Stephan Günnemann. Directional message passing for molecular graphs. *arXiv preprint arXiv:2003.03123*, 2020b.

Adeesh Kolluru, Muhammed Shuaibi, Aini Palizhati, Nima Shoghi, Abhishek Das, Brandon Wood, C Lawrence Zitnick, John R Kitchin, and Zachary W Ulissi. Open challenges in developing generalizable large scale machine learning models for catalyst discovery. *Preprint arXiv:2206.02005*, 2022.

Yi Liu, Limei Wang, Meng Liu, Xuan Zhang, Bora Oztekin, and Shuiwang Ji. Spherical message passing for 3d graph networks. *arXiv preprint arXiv:2102.05013*, 2021.

Łukasz Mentel. mendeleev – a python resource for properties of chemical elements, ions and isotopes, 2014. URL `https://github.com/lmmentel/mendeleev`.

Santiago Miret, Kin Long Kelvin Lee, Carmelo Gonzales, Marcel Nassar, and Matthew Spellings. The open matsci ml toolkit: A flexible framework for machine learning in materials science. *arXiv preprint arXiv:2210.17484*, 2022.

Raghunathan Ramakrishnan, Pavlo O Dral, Matthias Rupp, and O Anatole Von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific data*, 1(1): 1–7, 2014.

Vıctor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E (n) equivariant graph neural networks. In *International conference on machine learning*, pages 9323–9332. PMLR, 2021.

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. *Advances in neural information processing systems*, 30, 2017.

Kristof Schütt, Oliver Unke, and Michael Gastegger. Equivariant message passing for the prediction of tensorial properties and molecular spectra. In *International Conference on Machine Learning*, pages 9377–9388. PMLR, 2021.

Muhammed Shuaibi, Adeesh Kolluru, Abhishek Das, Aditya Grover, Anuroop Sriram, Zachary Ulissi, and C Lawrence Zitnick. Rotation invariant graph neural networks using spin convolutions. *arXiv preprint arXiv:2106.09575*, 2021.

Ichigaku Takigawa, Ken-ichi Shimizu, Koji Tsuda, and Satoru Takakusagi. Machine-learning prediction of the d-band center for metals and bimetals. *RSC advances*, 6(58):52587–52595, 2016.

Philipp Thölke and Gianni De Fabritiis. Torchmd-net: Equivariant transformers for neural network based molecular potentials. *arXiv preprint arXiv:2202.02541*, 2022.

Nathaniel Thomas, Tess Smidt, Steven Kearnes, Lusann Yang, Li Li, Kai Kohlhoff, and Patrick Riley. Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds. *arXiv preprint arXiv:1802.08219*, 2018.

Oliver T Unke and Markus Meuwly. Physnet: A neural network for predicting energies, forces, dipole moments, and partial charges. *Journal of chemical theory and computation*, 15(6):3678–3693, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Logan Ward, Ruoqian Liu, Amar Krishna, Vinay I Hegde, Ankit Agrawal, Alok Choudhary, and Chris Wolverton. Including crystal structure attributes in machine learning models of formation energies via voronoi tessellations. *Physical Review B*, 96(2):024104, 2017.

Tian Xie and Jeffrey C Grossman. Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties. *Physical review letters*, 120 (14):145301, 2018.

Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? *Advances in Neural Information Processing Systems*, 34:28877–28888, 2021.

Behnam Zakeri and Sanna Syri. Electrical energy storage systems: A comparative life cycle cost analysis. *Renewable and sustainable energy reviews*, 42:569–596, 2015.

C Lawrence Zitnick, Lowik Chanussot, Abhishek Das, Siddharth Goyal, Javier Heras-Domingo, Caleb Ho, Weihua Hu, Thibaut Lavril, Aini Palizhati, Morgane Riviere, et al. An introduction to electrocatalyst design using machine learning for renewable energy storage. *arXiv preprint arXiv:2010.09435*, 2020.

Larry Zitnick, Abhishek Das, Adeesh Kolluru, Janice Lan, Muhammed Shuaibi, Anuroop Sriram, Zachary Ulissi, and Brandon Wood. Spherical channels for modeling atomic interactions. *Advances in Neural Information Processing Systems*, 35:8054–8067, 2022.

Appendix A. Method

A.1 Invariance and equivariance to symmetries

Let $\phi : V \to \mathbb{R}$ and $\Phi : V \to W$ be arbitrary functions where W,V are linear spaces. Let G be a group describing a symmetry which we want to incorporate into ϕ, Φ (e.g. euclidean symmetries $E(3)$). We use group representations $\rho_1 : G \to GL(V)$ and $\rho_2 : G \to GL(W)$, where $GL(V)$ is the space of invertible linear maps $V \to V$ to represent how the symmetries $g \in G$ are applied to vectors $X \in V, W$. ϕ is an G-invariant function if it satisfies $\phi(\rho_1(g)X) = \phi(X)$, $\forall g \in G$ and $X \in V$.
Φ is an G-equivariant function if it satisfies $\Phi(\rho_1(g)X) = \rho_2(g)\Phi(X)$, $\forall g \in G$ and $X \in V$.

In this paper, we focus on accelerated catalysis and thus on adslab relaxed adsorption energy prediction. Like for most 3D molecular prediction tasks, we want GNNs to predict the same energy for two rotated, translated or reflected versions of the same system, since their energy is equal in real-life. Hence, we target $E(3)$-invariant models, where $E(3)$ is the Euclidean group in a 3D space (we have 3D atom positions), that is, the transformations of that 3D space that preserve the Euclidean distance between any two points (i.e. rotations, reflections, translations). Note that we do desire reflection invariance because we rotate the whole adsorbate-catalyst system and not just the adsorbate, in which case chiral molecules may have a different behaviour and shall be considered distinctly.

A.2 Graph creation

A.2.1 OC20

Chanussot, Lowik, et al. Chanussot et al. (2021) create each OC20 sample by choosing a bulk material from the Materials Project database[7]. Then, they select a surface from the bulk using Miller indices (at random) and replicate it at depth of at least 7 Å and a width of at least 8 Å. The final slab is defined by a unit cell that is periodic in all directions with a vacuum layer of at least 20 Å applied in the z direction. Next, they pick a binding site on this surface to attach the adsorbate onto the catalyst. The graph is now a set of atoms with their 3D positions. Last but not least, edges are created between any two nodes within a cutoff distance $c = 6$Å of each other (considering periodic boundary conditions).

7. `https://materialsproject.org/`

A.2.2 PhAST graph creation process

Although well grounded, the assumptions of this graph creation process are rarely questioned. We do, with the objective of making the graph sparser and more informative for subsequent GNNs. We describe more formally the three proposals evoked in 3.1.

remove-tag-0. Let $\mathcal{S} = \{i \in \mathcal{V} : t_i = 0\}$ denote the set of tag 0 atoms in the atomic system. The new graph we derive has attributes $\boldsymbol{X} = \boldsymbol{X}_{\mathcal{S}}$ where $\boldsymbol{X}_{\mathcal{S}}$ is the position of all atoms except those in $\mathcal{S}$. Similarly, $\boldsymbol{Z} = \boldsymbol{Z}_{\mathcal{S}}$ and $\boldsymbol{T} = \boldsymbol{T}_{\mathcal{S}}$. The new adjacency matrix $\boldsymbol{A}_{\mathcal{S}}$ is still defined based on cutoff distance and periodic boundary conditions: $A_{ij} = 1$ if $||\boldsymbol{x}_i - \boldsymbol{x}_j|| < c$, 0 otherwise. But it focuses on $\boldsymbol{X}_{\mathcal{S}}$, thus only containing edges which do not involve atoms in $\mathcal{S}$. Same for cell offsets $\mathbf{O}$.

one-supernode-per-graph. The position of the created super node is the mean of its components: $\boldsymbol{x}_s = \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} \boldsymbol{x}_i$ (with $\mathcal{S}$ as defined above). We associate it to a new characteristic number z_s (corresponding to a new element in atomic table) and adjacency $A_{is} = \max(a_{ij} : j \in \mathcal{S})$. We now remove all tag-0 atoms using the *remove-tag-0* method, and finally add a tag-0 attribute $t_i = 0$ to the supernode. Note that we also remove self-loop for the supernode.

one-supernode-per-atom-type. This extension is similar to the previous one, except that we create one supernode for each chemical element in the sub-surface catalyst. This complexify a bit the graph definition. Let a be the number of distinct elements with tag 0 in the graph ($a = 1, 2, 3$ by construction) and a_k be their characteristic number. Let $\mathcal{S}_{a_k} = \{i \in \mathcal{V} | t_i = 0 \text{ and } z_i = a_k\}$ be the set of atoms corresponding to each distinct tag-0 element a_k. Each supernode s_k is defined with $x_{s_k} = \frac{1}{|\mathcal{S}_{a_k}|} \sum_{i \in \mathcal{S}_{a_k}} \boldsymbol{x}_i$, $z_k = a_k$, $A_{is_k} = \max(A_{ij} : j \in \mathcal{S}_{a_k})$ $(\forall i \in \mathcal{V})$ and $A_{s_{k'} s_k} = 1$, $A_{s_k s_k} = 0$.

For both super-node methods, we encode the number of tag-0 nodes aggregated into each super node with **Positional Encodings** ((Vaswani et al., 2017)) to represent their "cardinal".

A.3 Atom properties for the Embedding block

In atom embeddings, we use the following properties from the `mendeleev` Python package (Mentel (2014)):

1. atomic radius,
2. atomic volume
3. atomic density
4. dipole polarizability
5. electron affinity
6. electronegativity (allen)
7. Van-Der-Walls radius
8. metallic radius

9. covalent radius

10. ionization energy (first and second order).

Appendix B. Results

B.1 Hyperparameters

Hyperparameters. We use each method's optimal set of parameters, provided in the config folder of the OCP repository for the IS2RE task, for the full dataset: `https://github.com/Open-Catalyst-Project/ocp/tree/main/configs/is2re/all`. Since ForceNet was not applied to IS2RE before, we adapted its S2EF configuration file to fit the IS2RE task. The only change is the smaller number of epochs used for DimeNet++ (10 instead of 20) and SchNet (20 instead of 30), as these additional epochs only lead to a small performance gain for a large amount of additional compute time. For PhAST models, we fine-tuned its hyperpameters to reach optimal performance.

B.2 GemNet IS2RE results

Results provided in Table 1 should appear surprising: both GemNet and GemNet-OC report much better results on IS2RE than we show. This is due to the fact that there are, in general, two ways to obtain the relaxed energy: either through direct prediction as we have explained in the paper, or through *relaxation.* The latter relies on an S2EF model that is iteratively applied to relax the system (positions are updated for the next step according to the current atom positions and associated predicted forces) until convergence, and the energy of the final (relaxed) structure is considered the model's output energy. This procedure is both more precise (yields better Energy MAE) but also much more computationally expensive. Both GemNet and GemNet-OC report the iterative *relaxation*-based relaxed energy prediction method to evaluate the performance of the models on IS2RE, while training on the much larger S2EF data set.

In addition, as explained in Section 4.2 we use the published hyper parameters. We could not afford the cost of a direct-IS2RE hyper parameter search on GemNet and GemNet-OC and therefore resulted to use their S2EF hyper parameters.

All in all, while the absolute values of Energy MAE may seem surprising, the point of Figure 2 and Table 1 is mainly to measure the *relative* effects of PhAST on the individual models. And on this aspect, PhAST improves significantly the performance of all methods. We expect these benefits to translate to other training configurations and state-of-the-art architectures.

B.3 Throughput results on IS2RE

In Table 4, we include throughput results for IS2RE models.

B.4 Numerical results of the ablation study on IS2RE

Notation. For the embedding block, *tag-embed* defines atom embeddings $\boldsymbol{H}$ using atom tag information and characteristic number: $\boldsymbol{H} = \boldsymbol{H}_Z || \boldsymbol{H}_T$, where $||$ denotes concatenation. Similarly, *phys-embed* defines $\boldsymbol{H} = \boldsymbol{H}_Z || \boldsymbol{H}_F$. *l-phys-embed* is a learnable alternative $\boldsymbol{H} =$

Baseline / MAE	Average	Throughput (s/s)	Inference time (s)
SchNet	683	3190 ± 302	15 ± 0.49
PhAST-SchNet	**630**	**16109 ± 2000**	**5 ± 0.36**
D++	628	191 ± 30	110 ± 0.57
PhAST-D++	**595**	**1021 ± 130**	**20 ± 0.63**
ForceNet	654	147 ± 14	167 ± 0.96
PhAST-ForceNet	**616**	**734 ± 139**	**33 ± 0.60**
GemNet	674	52 ± 06	487 ± 0.24
PhAST-GemNet	**619**	**306 ± 31**	**52 ± 0.10**
GemNet-OC	654	29 ± 04	858 ± 1.89
PhAST-GemNet-OC	**616**	**88 ± 06**	**257 ± 0.09**

Table 4: Comparing model performance on *OC20* IS2RE. *Average* MAE is computed over all validation splits. PhAST models all show improved performance and drastic speedups (e.g. throughput is roughly 5× higher). Inference times and Throughput (number of samples per second processed at inference time) are averaged over 3 runs.

$\boldsymbol{H}_Z||MLP(\boldsymbol{H}_F)$. *pg* refer to period and group embeddings: $\boldsymbol{H} = \boldsymbol{H}_Z||\boldsymbol{H}_P||\boldsymbol{H}_G$. *All* is a concatenation of all five embeddings: $\boldsymbol{H} = \boldsymbol{H}_Z||\boldsymbol{H}_T||\boldsymbol{H}_F||\boldsymbol{H}_P||\boldsymbol{H}_G$. For the graph creation step, we have defined these approaches in A.2.2 (note: *sn* stands for supernode). For the energy head part, *w-init* (*w-final*) denote the weighted sum of initial (final) atom embeddings. *graclus* and *hoscpool* refer to the two hierarchical pooling approaches.

See Tables 5, 6, 7. The results are reported in a similar fashion as for Table 1. The symbol † indicates that a result is better than the baseline model. **Bold** font shows the best extension for each PhAST improvement category.

B.5 Graph-Rewiring: impact on the number of edges and nodes

Method / **MAE**	Average	ID	OOD-ad	OOD-cat	OOD-both	Inference time (s)
tag-embed	**648**†	**637**	**690**†	**629**†	638†	15.39 +/- 0.33
phys-embed	662†	644	700†	639†	654†	15.48 +/- 0.38
l-phys-embed	678†	650	733†	649†	679†	15.59 +/- 0.37
pg	673†	646	725†	644†	676†	15.44 +/- 0.52
All	659†	665	**690**†	651†	**630**†	15.46 +/- 0.46
remove-tag-0	**648**†	**627**†	**705**†	627†	**634**†	**4.74** +/- **0.50**
sn-graph	654†	633†	**705**†	633†	646†	5.54 +/- 0.68
sn-atom-type	663†	628†	738	**626**†	659†	6.38 +/- 0.45
w-init	**657**†	**635**†	715†	**631**†	**646**†	15.37 +/- 0.48
w-final	668†	647	**713**†	644†	670†	15.41 +/- 0.38
hoscpool	667†	650	719†	636†	662†	53.93 +/- 1.36
SchNet	683	637	734	661	703	15.40 +/- 0.49

Table 5: SchNet ablation study on *OC20* IS2RE.

Method / **MAE**	Average	ID	OOD-ad	OOD-cat	OOD-both	Inference time (s)
tag-embed	**579**†	551†	**659**†	545†	**594**†	110.04 +/- 0.89
phys-embed	590†	561†	671†	555	606†	110.08 +/- 0.74
l-phys-embed	612†	566†	700†	557†	626†	110.12 +/- 0.75
pg	624†	564†	710†	568	652†	110.18 +/- 0.71
All	602†	**550**†	691†	**540**†	626†	110.03 +/- 0.77
remove-tag-0	610†	576	684†	568	627†	**20.15** +/- **0.55**†
sn-graph	-	-	-	-	-	25.04 +/- 1.54
sn-atom-type	-	-	-	-	-	27.22 +/- 0.94
w-init	611†	568†	686†	**560**†	630†	110.23 +/- 0.81
w-final	**601**†	571	660†	566	**606**†	110.16 +/- 0.76
hoscpool	618†	**565**†	703†	563	642†	252.55 +/- 0.45
D++	628	571	722	561	661	110.11 +/- 0.57

Table 6: Dimenet++ ablation study on *OC20* IS2RE*

* Unfortunately we did not manage to train DimeNet++ with the super node extensions. For a very wide range of learning rates, the training loss consistently reached NaN values.

Method / MAE	Average	ID	OOD-ad	OOD-cat	OOD-both	Inference time (s)
tag-embed	640†	639†	690†	616†	617†	168.64 +/- 0.73
phys-embed	653†	657†	702	626†	627†	172.04 +/- 0.98
l-phys-embed	667	654†	734	626†	650	167.98 +/- 0.71
pg	**634†**	644†	**669†**	618†	**603†**	170.70 +/- 0.96
All	637†	**622†**	680†	**603**	615†	169.23 +/- 0.84
remove-tag-0	**628†**	**637†**	**668†**	**611†**	**598†**	**17.06** +/- **0.58**
sn-graph	635†	640†	676†	617†	607†	55.30 +/- 1.86
sn-atom-type	632†	641†	672†	616†	601†	70.55 +/- 0.15
w-init	**639†**	639†	687†	**611†**	616†	170.71 +/- 0.60
w-final	660	655†	716	627†	644	170.72 +/- 1.61
hoscpool	655	**621†**	703	638	**638**	202.03 +/- 1.23
ForceNet	654	658	701	632	628	167.08 +/- 0.52

Table 7: ForceNet ablation study on *OC20* IS2RE.

Rewiring	Atoms	Edges
Train		
Full graph	35 789 459	1 309 308 840
remove-tag-0	32.53%	16.61%
one-supernode-per-graph	33.81%	17.76%
one-supernode-per-atom-type	35.45%	19.09%
ID		
Full graph	1 939 553	70 825 106
remove-tag-0	32.54%	16.65%
one-supernode-per-graph	33.83%	17.80%
one-supernode-per-atom-type	35.47%	19.13%
OOD-ads		
Full graph	1 918 704	69 877 652
remove-tag-0	32.42%	16.50%
one-supernode-per-graph	33.72%	17.64%
one-supernode-per-atom-type	35.38%	18.97%
OOD-cat		
Full graph	1 917 954	70 314 085
remove-tag-0	32.88%	16.78%
one-supernode-per-graph	34.18%	17.95%
one-supernode-per-atom-type	35.90%	19.34%
OOD-both		
Full graph	2 094 709	80 074 123
remove-tag-0	31.12%	15.33%
one-supernode-per-graph	32.31%	16.37%
one-supernode-per-atom-type	34.17%	17.83%

Table 8: Comparison of the number of nodes and edges in the original 5 datasets (training and 4 validation splits) and the remaining number of nodes and edges after the various rewiring strategies are performed. We can see that our rewiring methods generally remove 65+% of the atoms and 80+% of the edges.