Connectivity Optimized Nested Line Graph Networks for Crystal Structures

Robin Ruff
Institute of Theoretical Informatics
Karlsruhe Institute of Technology
Engler-Bunte-Ring 8,
76131 Karlsruhe, Germany

Patrick Reiser
Institute of Theoretical Informatics
Karlsruhe Institute of Technology
Engler-Bunte-Ring 8,
76131 Karlsruhe, Germany

Jan Stühmer
Institute for Anthropomatics and Robotics
Karlsruhe Institute of Technology
Engler-Bunte-Ring 8,
76131 Karlsruhe, Germany

Heidelberg Institute for Theoretical Studies
Schloß-Wolfsbrunnenweg 35,
69118 Heidelberg, Germany

Pascal Friederich
Institute of Theoretical Informatics
Karlsruhe Institute of Technology
Engler-Bunte-Ring 8,
76131 Karlsruhe, Germany

Institute of Nanotechnology
Karlsruhe Institute of Technology
Hermann-von-Helmholtz-Platz 1,
76344 Eggenstein-Leopoldshafen, Germany
pascal.friederich@kit.edu

Abstract

Graph neural networks (GNNs) have been applied to a large variety of applications in materials science and chemistry. Here, we systematically investigate the graph construction for crystalline (periodic) materials and investigate its impact on the GNNs model performance. We propose the asymmetric unit cell as a representation to reduce the number of nodes needed to represent periodic graphs by exploiting all symmetries of the system. Without any loss in accuracy, this substantially reduces the computational cost and thus time needed to train large graph neural networks. For architecture exploration we extend the original Graph Network framework (GN) of Battaglia et al. [1], introducing nested line graphs (Nested Line Graph Network, NLGN) to include more recent architectures. Thereby, with a systematically built GNN architecture based on NLGN blocks, we improve state-of-the-art results across all tasks within the MatBench benchmark. Further analysis shows that optimized connectivity and deeper message functions are responsible for the improvement. Asymmetric unit cells and connectivity optimization can be generally applied to (crystal) graph networks, while the suggested nested NLGN framework can be used to as a template to compare and build more GNN architectures.

1 Introduction

Since seminal work by Duvenaud et al. [2], graph neural networks (GNNs) developed into one of the most versatile and accurate classes of machine learning models for the prediction of molecular and material properties [3]. Consequently, GNNs find increasing application in materials sciences for structure-property predictions [4], materials screening [5] and high-throughput simulations [6]. Learning on experimental or simulated databases [7–9], GNNs show promising potential to develop new materials to tackle our society's growing demand for high-performance materials in the fields of catalysis, renewable energies, energy conversion or functional materials [10, 11].

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

Graph convolutional neural networks operate on the (spatial) graph structure to transform node embeddings and have been suggested for semi-supervised node classification [12, 13]. With the focus on molecular graphs, the message passing framework (MPNN) was proposed by Gilmer et al. [14] in order to group and generalize many GNN architectures that update node representations by iteratively exchanging information, or messages, across edges in the graph [15, 16]. Neural networks designed to predict the potential energy surface of molecules for molecular dynamics simulations [17], such as the continuous-filter convolutional network SchNet [18], can also be interpreted as MPNN graph networks. Significant progress was achieved by incorporating further geometric information such as bond [19] and dihedral angles [20] into the MPNN scheme. To compute explicit angle information, the graph network has to operate on higher order pathways or connect edge information in a so-called line graph, which sets up a graph on edges $L(G)$, i.e. on-top of the underlying graph G [21]. A state-of-the-art model that makes use of this principle and reaches good performance for materials is ALIGNN [22]. Recently, equivariant graph networks have been introduced [23], which build on irreducible representations of the Euclidean symmetry groups to achieve equivariance under e.g. rotation and translation [24].

2 Related work

Although many of the previously mentioned GNNs can or have been applied to materials, fewer architectures have been developed with a primary focus on crystalline systems. The crystal-graph convolution neural network (CGCNN) first introduced a GNN architecture on a crystalline system by constructing a multi-graph that correctly represents the atomic neighbors in a periodic system [25]. Its improved version iCGCNN incorporates information on the Voronoi tessellated crystal structure and explicit three-body correlations of neighboring constituent atoms [26]. MEGNet further leverages global state information and added edge updates in the convolution process [27]. With GeoCGNN a geometric GNN was introduced [28], which encodes the local geometrical information using an attention mask composed of Gaussian radial basis functions and plane waves from a k-point mesh of Monkhorst Pack special points [29]. Although multiple GNN model architectures have been proposed in this context [22, 25–28, 30, 31], there does not yet seem to be unanimous consent in the literature on which is the best method or the most decisive tool in geometric deep learning to process crystalline materials.

In most cases, newly introduced GNNs make specific improvements over previous architectures and propose multiple reasonable new design decisions inspired by chemical domain knowledge. While this approach has so far led to a consistent improvement of model accuracy, we choose a more systematic approach inspired by the work of You et al. [32]: First, we stake out a new design space based on a general extension of the original graph network (GN) framework [1], which we call nested line graph networks (NLGNs). Then we navigate through that design space to find suitable NLGN architectures.

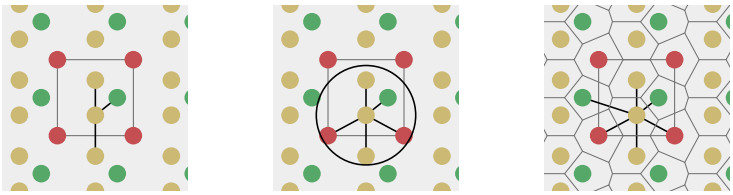
In this work, we re-evaluate edge selection methods to build a multi-graph as suggested by Xie and Grossman [25] and compare their performance on a wide set of parameters. In this context, we introduce the *asymmetric unit cell* as a representation to further exploit crystal symmetries and effectively reduce the number of edges. Next, we develop a connectivity-optimized crystal graph network (coGN/coNGN) from message passing and line-graph templates which are intentionally kept as general as possible. By optimizing within the generalized family of GNNs, we improve state-of-the-art results on 6 out of 9 of the MatBench benchmark datasets [33] and achieve parity results with the best models on the remaining 3 datasets, making our model the best general model on the entire set of diverse tasks in the benchmark.

3 Connectivity Optimized Nested Line Graph Networks (coNGN)

3.1 Connectivity Optimized Crystal Graph Construction

There are two main challenges when trying to build graph representations of crystal structures in contrast to organic molecules: (a) Bonds between atoms in crystals have more diverse types (covalent, ionic, metallic bonds), or are often not well defined at all. (b) Crystal structures have no fixed finite size, as they are defined as periodic repetitions of a unit cell.

Edge selection The first aspect raises the question of which edges to add to the graph that describes the crystal. To circumvent this problem one relies on the geometrical properties of the atom arrangement instead of chemically informed bonds.



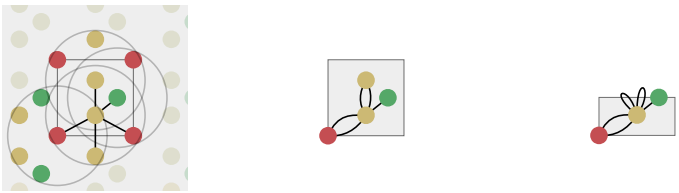
(a) kNN Edges (b) Radius Edges (c) Voronoi Edges

Figure 1: Edge selection for a single atom in a crystal structure. Red atoms mark the cubic unit cell.

Figure 1 schematically shows a two-dimensional crystal pattern and three different methods for selecting edges between atoms. The k-nearest-neighbors approach (Figure 1a) depends on the number of neighbors k, but can lead to largely different edge distances when the crystal density varies. The radius-based approach (Figure 1b) limits the distance between two nodes by a hyperparameter r, but the number of neighbors is unbounded and the method can lead to either disconnected or overly dense graphs if r is chosen inappropriately. The parameter-free Voronoi-based approach (Figure 1c) leads to an intuitive edge selection where edges are drawn between two atoms if there is a Voronoi cell ridge between them. However, at least in theory, the number of edges and their distances are also unbounded for this approach.

All three edge selection methods have been used previously in the context of crystals and GNNs [25–27, 34]. But to our knowledge, there is no detailed comparison between the methods and hyperparameters.

Exploiting crystal symmetries In contrast to molecules, crystal structures are modeled as (infinite) periodic repetitions of the unit cell atom motif. A direct approach to extracting a finite graph for a given crystal structure is to simply select all unit cell atoms with their respective neighbors (Figure 2a). In an information-theoretical sense, this representation captures the entire information of the crystal for a given edge selection method. However, from a naive message-passing perspective, only the nodes representing atoms inside of the central unit cell receive messages from all their neighboring atom nodes, which would model a finite graph rather than an infinite periodic graph.



(a) No symmetries (b) Periodicity (c) All symmetries

Figure 2: Graph representations of crystal structures exploiting different levels of symmetries. The *asymmetric unit cell* is shown in (c)

To solve this issue, Xie and Grossman [25] proposed a multigraph crystal representation (Figure 2b) introducing periodic/cyclic boundary conditions, which we will refer to as a unit cell graph. In this representation, one node represents all the shift-equivalent atoms of the crystal, which usually results in multiple edges with distance information matching their translated lattice positions. As a consequence, GNNs will always provide equivalent node embedding for periodic atoms, consistent with Bloch's theorem.

The periodicity of the unit cell graph represents translation symmetry. Since crystals often exhibit more symmetries, we propose the asymmetric unit graph representation for crystals, which considers all symmetries of a crystal structure.[1] For more information about the asymmetric unit (ASU) and

[1]The set of symmetries for crystals and thus the asymmetric unit cell representation can be determined automatically based on the normal unit cell [35].

space groups please see for example [36–39]. In this representation, all symmetry-equivalent atoms are represented by a single node in the graph. The example crystal in Figure 2 exhibits a horizontal reflection symmetry. The two yellow atoms in the crystal unit cell are symmetry-equivalent and therefore merged into one node (with multiplicity two) for the asymmetric unit cell (Figure 2c). Just like in unit cell graphs, self-loops and multi-edges can occur in asymmetric unit cell graphs.

Since physical target properties in ML tasks are often invariant under E(3) symmetry operations, many GNNs are designed to be E(3)-invariant, but as consequence, yield equal node embeddings for symmetrical atoms in the unit cell graph, leading to redundant computations in the message passing step. The asymmetric unit graph representation can further remove these redundancies and yet maintain the same node embeddings x_v.[2] However, global readout operations have to be adapted to handle asymmetric unit cells, since atoms can have different symmetry-related multiplicities $\mathfrak{m}_v$ (the number of symmetry equivalent atoms for each equivalence class) A simple adaptation of the readout function can fix the issue and restore equal results for unit cell graphs and asymmetric unit graphs:

$$\underset{v\in V}{\text{agg}'}(x_v) = \begin{cases} \underset{v\in V}{\text{agg}}(x_v \cdot \mathfrak{m}_v) \cdot \frac{|V|}{\sum_v \mathfrak{m}_v} & \text{for mean or attention} \\ \underset{v\in V}{\text{agg}}(x_v \cdot \mathfrak{m}_v) & \text{for sum} \\ \underset{v\in V}{\text{agg}}(x_v) & \text{for min or max} \end{cases}$$

It is to note that for the ASU representation the structure needs to remain in the same space group and its atoms in their original Wyckoff sites, which means that for inference during e.g. molecular dynamics simulations and structure changes, the ASU has to be reevaluated which is cumbersome and expensive. Nonetheless, GNNs can be trained on ASU structures, which requires one ASU evaluation of the training set, and then used on the normal unit cell representation for inference.

3.2 Nested Line Graph Network Framework

In addition to the choice of input representation, the GNN model architecture has a substantial impact on the quality of crystal graph property predictions.

To find the best GNN architecture for a certain task, it is instructive to systematically explore (see e.g. [32]) the design space of GNN modules and building blocks. Moreover, a framework has to be chosen on how to define and process GNN modules. The Message Passing Framework by Gilmer et al. [14], for example, has shown that a framework can unify and accelerate the efforts of the scientific community.

For a combinatorial generalization of GNNs, we build upon the graph network (GN) framework of Battaglia et al. [1] in this work. In this framework, one GNN layer is described by a GN block, which transforms a generic attributed graph with edge, node, and global graph features via three update functions ϕ and three aggregation functions ρ.

However, many state-of-the-art models such as DimeNet [40, 40], GemNetčiteklicpera2021gemnet, ALIGNN [22] and M3GNet [31], which incorporate many-body interactions between atoms, are only indirectly captured by GNs. Therefore we propose an extension to the framework which we term nested line graph networks (NLGNs). In the following, we introduce NLGNs by first discussing how angle information is incorporated via the line graph concept [22] and then explaining the flow of NLGN calculations, before we will show concrete examples of implementations of NLGNs in Section 4.1. Note that the term *nested* refers here to stacking GNs on the line graph, not to confuse with nesting as introduced by Zhang and Li [41], which describes nesting and pooling GNNs on sub-graphs.

To achieve rotation invariance, popular models such as SchNet [18] or MEGNet [27] only include scalar distances between two atoms as edge features. However, to incorporate more geometric information, models such as DimeNet [40] or ALIGNN [22] additionally use E(3)-invariant angle information, represented by combinations of edges (see Figure 6). In the line graph $L(G)$, which can be constructed uniquely based on G (see Figure 3 and Harary and Norman [42]), there is an

[2] It is in principle also possible to adapt E(3)-equivariant GNNs to asymmetric unit graph representations, by specifying equivariant convolutional layers on the asymmetric unit graph and adapting message passing accordingly.

edge $e^{L(G)}_{e_{ij},e_{jk}}$ for every two incident edges e_{ij}, e_{jk} in G. This enables the assignment of angular information between three atoms to the edges of the line graph. The same applies to (generalized) dihedral angles (4-node or 3-edge objects) in the second-order line graph $L(L(G))$.

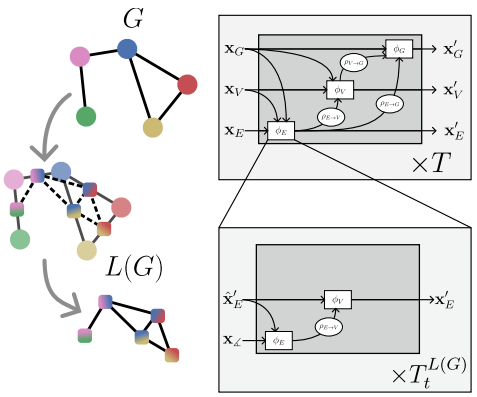


Figure 3: On the left: A graph G and the construction of its line graph $L(G)$. On the right: Nested line graph network architecture with GN blocks working on G and containing other GN blocks working on the line graph $L(G)$ as edge update function ϕ_E. The line graph is able to process multi-node geometric features, such as angles (see Table 1)

Table 1: Correspondence between entities in the crystal, the Crystal Graph G, its Line Graph $L(G)$, etc.

Entity in Crystal	G	$L(G)$	$L(L(G))$
Atoms	Nodes		
Bonds	Edges	Nodes	
Angles		Edges	Nodes
Dihedrals			Edges

NLGNs operate on the graph G as well as the line graph $L(G)$ (potentially also $L(L(G))$ etc.), exploiting the one-to-one mapping of edges in G and nodes in $L(G)$ (see Table 1). Each edge update function ϕ_E in GN blocks that operate on G can be instantiated as a nested GN (see Figure 3). A more detailed description of the algorithm can be found in Appendix. The NLGN framework extends the usual sequential compositionality of GN blocks with a hierarchical/nested compositionality, thereby increasing the expressiveness [43]. The composition of simple and well-known building blocks facilitates the implementation and the ease of understanding of the framework. Furthermore, NLGNs generalize existing models such as SchNet [18], DimeNet [40] and ALIGNN [22], making them more comparable and extensible (see Figure 4).

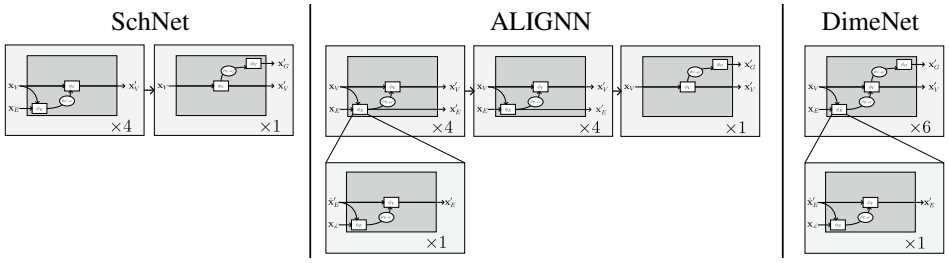


Figure 4: Other models in the NLGN framework.

4 Experiments and Evaluation

Datasets To evaluate the preprocessing methods and GN architectures we relied on the MatBench benchmark [33], which can be considered to be the ImageNet [44] for machine learning models in

materials science. The benchmark currently consists of 13 strictly standardized supervised crystal property prediction tasks curated from different data sources [45–48]. Out of the 13 tasks, four provide only the crystal composition as input and the other nine incorporate the crystal structure with the geometric arrangement of atoms. Since this work focuses on crystal structures specifically, we only use the nine datasets that rely on structural information for property prediction. The size of the datasets ranges from 636 to 132,752 crystal instances.

Implementations We used the Keras Graph Convolution Neural Networks (KGCNN) library [49] to implement the GN and NLGNs used in the experiments of this work. The code for crystal preprocessing and GNN models is available online[3]

4.1 GNN Architecture Search

To find a suitable GN architecture, we conduct an architecture search within the NLGN framework. In order to generalize previous models, our NLGN framework is flexible and spans a large hyperparameter space for architectural decisions which makes the exploration of the entire space infeasible. If no nesting is explicitly required, the NLGN framework falls back to the GN framework of Battaglia et al. [1]. For comparison, we search architectures with and without nesting, which lead to the connectivity-optimized graph architectures **coGN** and **coNGN**, respectively. The search procedure and architecture details are discussed in Appendix 6.5.

4.2 Crystal Graph Connectivity

For GNNs in particular, there is a strong interdependency between input representation and model as the topology of the input graph also affects the computational graph. The interdependency between the preprocessing of crystals and model architecture also occurs with respect to crystal property predictions and should therefore be considered.

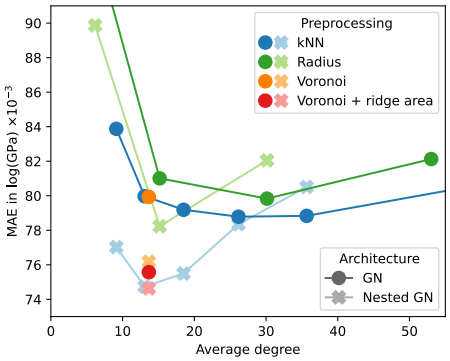


Figure 5: MAE for GNs, NLGNs, and different edge selection methods on the `log_gvrh` dataset.

Figure 5 shows the effect of different preprocessing methods for a non-nested GN. Again, results were obtained before the final ordinal hyperparameter optimization, which explains the discrepancy with Table 2. The MAE is plotted as a function of the average degree over the resulting graphs of the `log_gvrh` dataset for different edge selection methods. The accuracy of GNN models tends to fall for increasing graph connectivity up to an average degree of approximately 30. For a higher average degree the accuracy either gets worse or quickly converges depending on the dataset. Interestingly, for the kNN edge selection the minimum corresponds to $k = 24$, whereas other works use a value of 12 [22, 25]. The Voronoi-based edge selection results in graphs with an average degree of approximately 12. Adding the area of the Voronoi cell ridge to each edge as an additional edge feature can improve the predictive power of the GNN. We found, however, that this effect is much less pronounced after optimization of ordinal hyperparameters, making the $k = 24$ nearest-neighbor method the preferred choice for edge selection in our experiments.

[3] `https://github.com/matbench-submission-coGN/CrystalGNNs`

The observations for non-nested GNs do not apply to NLGNs, when comparing their behavior in Figure 5. For NLGNs, the minimum test error occurs at a lower average degree of approximately 12 edges per node and is followed by a steeper increase for higher connectivity. Higher-order nesting could potentially lead to further improvement at yet lower connectivity of the base graph, which should be systematically explored in the future.

At the same time, our observations raise the question of a trade-off between nesting and graph connectivity. This trade-off can also be explained from an analytical point of view. The reason for including angle information with NLGNs is shown in the example in Figure 6. The two geometric graphs are not distinguishable for GNNs from relative distance information alone. Incorporating angles between edges into the GNN architecture increases expressiveness and allows for the distinction of the graphs. Yet, similar enhancements of expressiveness can also be achieved by increasing graph connectivity. In the example, adding the single dashed edge between two red nodes makes the graphs distinguishable for GNNs, without any angle information.

Figure 6: Two different geometric graphs which are undistinguishable from distance edge features, but become distinguishable with angle information between edges or by adding the dashed edge to the graphs.

From our experiments, we can not conclude that NLGNs offer a systematic advantage in accuracy over simple GNs when used on graphs with high connectivity. Inspired by the results of Figure 5, we optimized an NLGN for less connected Voronoi (+ ridge area) preprocessed graphs and achieved the results displayed in Table 2. Although NLGNs showed the best performance on the specific dataset they have been (hyperparameter) optimized on, they can not maintain their advantage consistently on all other datasets without re-optimizing. Unfortunately, NLGNs require the construction of line graphs and tend to have significantly more trainable parameters. Consequently, training is more than three times as computationally expensive as for non-nested GNs.

Finally, we were able to demonstrate the effectiveness of densely connected crystal graphs with the coGN model, by surpassing current state-of-the-art models on most of the MatBench datasets as shown in Table 2. The results also support that hyperparameters, which originate from the optimization on the `log_gvrh` dataset, generalize to other datasets and tasks.

Table 2: Comparison with results of state-of-the-art model on MatBench structure datasets and splits [33]. Ordered by descending cardinality, these are `e_form` (meV/atom), `is_metal` (AUC(ROC)), `gap` (meV), `perovskites` (meV/unit cell), `log_kvrh` (log10(GPa)), `log_gvrh` (log10(GPa)), `dielectric` (unitless), `phonons` (1/cm) and `jdft2d` (meV/atom). Current benchmark holders on MatBench, namely, ALIGNN [22], MODNet [50] and CGCNN [25] are listed. Additionally, recent models M3GNet [31] and Matformer [51] are added, which have been published during the preparation of this work. Since Matformer was not trained on the official benchmark, we retrained the original model. The best results are indicated in bold font, while other results within one standard deviation are underlined. *For `is_metal` the classification metric is likely to change in future versions.

Dataset	coGN (ours)	coNGN (ours)	ALIGNN	MODNet	CGCNN	M3GNet	Matformer
`e_form` ↓	**17.0 ± 0.3**	17.8 ± 0.4	21.5 ± 0.5	44.8 ± 3.9	33.7 ± 0.6	19.5 ± 0.2	21.232 ± 0.302
`is_metal`* ↑	0.9124 ± 0.0023	0.9089 ± 0.0019	0.9128 ± 0.0015	0.9038 ± 0.0106	**0.9520 ± 0.0074**	0.958±0.001	0.812 ± 0.05
`gap` ↓	**155.9 ± 1.7**	169.7 ± 3.5	186.1 ± 3.0	219.9 ± 5.9	297.2 ± 3.5	183 ± 5	187.825 ± 3.817
`perovskites` ↓	**26.9 ± 0.8**	29.0 ± 1.1	28.8 ± 0.9	90.8 ± 2.8	45.2 ± 0.7	33 ± 1.0	31.514 ± 0.71
`log_kvrh` ↓	0.0535 ± 0.0028	**0.0491 ± 0.0026**	0.0568 ± 0.0028	0.0548 ± 0.0025	0.0712 ± 0.0028	0.058±0.003	0.063 ± 0.0027
`log_gvrh` ↓	0.0689 ± 0.0009	**0.0670 ± 0.0006**	0.0715 ± 0.0006	0.0731 ± 0.0007	0.0895 ± 0.0016	0.086±0.002	0.077 ± 0.0016
`dielectric` ↓	0.3088 ± 0.0859	0.3142 ± 0.0740	0.3449 ± 0.0871	**0.2711 ± 0.0714**	0.5988 ± 0.0833	0.312±0.063	0.634 ± 0.131
`phonons` ↓	29.712 ± 1.997	**28.887 ± 3.284**	29.539 ± 2.115	34.2751 ± 2.0781	57.7635 ± 12.311	34.1 ± 4.5	42.526 ± 11.886
`jdft2d` ↓	37.165 ± 13.683	36.170 ± 11.597	43.424 ± 8.949	**33.192 ± 7.343**	49.244 ± 11.587	50.1 ± 11.9	42.827 ± 12.281

Results of coGN/coNGN on other comparable materials benchmarks like JARVIS [52] or OC22 [9], which feature structure to property tasks, can be found in the appendix and yield similar top ranking performance.

4.3 Asymmetric Unit Graphs

In Section 3.1, we discussed different graph representations for crystals and found that asymmetric unit graphs are smaller than unit cell graphs and yet lead to identical predictions of GNNs with E(3)-invariant layers. The exact reduction factor for asymmetric unit graphs depends on the symmetries each specific crystal exhibits. Since the maximal space group order for crystals is 48, we can specify theoretical lower and upper bounds for the number of nodes n_{asu} in asymmetric unit graphs in relation to unit cell graphs:

$$n_{\text{unit}} \geq n_{\text{asu}} \geq \frac{1}{48} \cdot n_{\text{unit}}$$

We found that for the MatBench datasets, the empirical average reduction factor for the number of nodes (n), edges ($m = n^{L(G)}$) as well as line graph edges ($m^{L(G)}$) is approximately 2.1 [4]:

$$\frac{n_{\text{unit}}}{n_{\text{asu}}} \approx \frac{m_{\text{unit}}}{m_{\text{asu}}} \approx \frac{m_{\text{unit}}^{L(G)}}{m_{\text{asu}}^{L(G)}} \approx 2.1$$

Approximately the same factor can be observed for the GPU memory footprint during training. Due to parallelization and batching the acceleration of the training runtimes is somewhat smaller and depends on the selected batch size. For batch sizes of 32, 64, and 128, for example, we observed a speedup of 1.2, 1.3, and 1.8, respectively.

5 Conclusion

This paper discusses fundamental aspects of crystal property prediction with GNNs: the incorporation of symmetries in GNN models, the interdependence of input graph generation from crystal structures (i.e. preprocessing) with the systematic exploration of a suitable GNN architecture, as well as the generalization of model architectures as nested line graph networks. We conclude that these aspects cannot be considered separately, which is done in many other studies.

Our contribution to the first aspect includes the proposal of the *asymmetric unit graph* representation, which decreases training time and memory consumption without effects on predictive performance for E(3)-invariant GNNs. To use asymmetric unit graphs with equivariant GNNs some adaptations to message passing and readout operations are required, which will be addressed in future work. We furthermore compared different edge selection methods and discovered that graphs with higher connectivity (compared to previous works) can yield better performance. From this insight, we construct the coGN, which achieves state-of-the-art results on the MatBench benchmark.

To explore the space of GNN architectures, we introduced the *nested line graph network* (NLGN) framework, which subsumes state-of-the-art GNN models that incorporate angle information in line graph form. Although for the given dataset sizes in the MatBench benchmark we could not find an architecture for nested line graph networks that substantially outperforms graph networks without nesting, we still encourage further research in this direction, specifically into the scaling laws of accuracy as a function of data set size and complexity as well as the depth of line-graph nesting. Future work might overcome the mentioned trade-off between high connectivity and nesting and potentially explore specific nested line graph network architectures, which we did not consider due to hyperparameter space restrictions. Moreover, a practical but systematic comparison between recent equivariant GNs, which are more expressive than invariant GNs, and NLGNs could potentially offer further insight and gradual improvements.

Acknowledgement

The authors acknowledge support by the state of Baden-Württemberg through bwHPC.

[4] The `perovskites` dataset is a special outlier, where asymmetric graphs are on average not significantly smaller than unit cell graphs.

Data availability

Training code and results can be found on the official benchmark submission. Additionally, the code for the crystal GNNs can be found on github: `https://github.com/matbench-submission-coGN/CrystalGNNs`.

References

[1] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

[2] 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.

[3] Patrick Reiser, Marlen Neubert, André Eberhard, Luca Torresi, Chen Zhou, Chen Shao, Houssam Metni, Clint van Hoesel, Henrik Schopmans, Timo Sommer, et al. Graph neural networks for materials science and chemistry. *Communications Materials*, 3(1):93, 2022.

[4] Mohammadreza Karamad, Rishikesh Magar, Yuting Shi, Samira Siahrostami, Ian D. Gates, and Amir Barati Farimani. Orbital graph convolutional neural network for material property prediction. *Phys. Rev. Mater.*, 4:093801, Sep 2020. doi:10.1103/PhysRevMaterials.4.093801. URL `https://link.aps.org/doi/10.1103/PhysRevMaterials.4.093801`.

[5] Jonathan Schmidt, Love Pettersson, Claudio Verdozzi, Silvana Botti, and Miguel A. L. Marques. Crystal graph attention networks for the prediction of stable materials. *Science Advances*, 7(49): eabi7948, 2021. doi:10.1126/sciadv.abi7948. URL `https://www.science.org/doi/abs/10.1126/sciadv.abi7948`.

[6] Jörg Behler. Atom-centered symmetry functions for constructing high-dimensional neural network potentials. *The Journal of Chemical Physics*, 134(7):074106, 2011. doi:10.1063/1.3553717. URL `https://doi.org/10.1063/1.3553717`.

[7] Scott Kirklin, James E Saal, Bryce Meredig, Alex Thompson, Jeff W Doak, Muratahan Aykol, Stephan Rühl, and Chris Wolverton. The Open Quantum Materials Database (OQMD): assessing the accuracy of DFT formation energies. *npj Computational Materials*, 1(1):15010, December 2015. ISSN 2057-3960. doi:10.1038/npjcompumats.2015.10. URL `https://doi.org/10.1038/npjcompumats.2015.10`.

[8] Lowik Chanussot, Abhishek Das, Siddharth Goyal, Thibaut Lavril, Muhammed Shuaibi, Morgane Riviere, Kevin Tran, Javier Heras-Domingo, Caleb Ho, Weihua Hu, Aini Palizhati, Anuroop Sriram, Brandon Wood, Junwoong Yoon, Devi Parikh, C. Lawrence Zitnick, and Zachary Ulissi. Open catalyst 2020 (oc20) dataset and community challenges. *ACS Catalysis*, 2021. doi:10.1021/acscatal.0c04525.

[9] Richard Tran, Janice Lan, Muhammed Shuaibi, Brandon Wood, Siddharth Goyal, Abhishek Das, Javier Heras-Domingo, Adeesh Kolluru, Ammar Rizvi, Nima Shoghi, Anuroop Sriram, Zachary Ulissi, and C. Lawrence Zitnick. The open catalyst 2022 (oc22) dataset and challenges for oxide electrocatalysis. *arXiv preprint arXiv:2206.08917*, 2022.

[10] Dolf Gielen, Francisco Boshell, and Deger Saygin. Climate and energy challenges for materials science. 15(2):117–120. ISSN 1476-4660. doi:10.1038/nmat4545. URL `https://doi.org/10.1038/nmat4545`.

[11] Yuanjing Cui, Bin Li, Huajun He, Wei Zhou, Banglin Chen, and Guodong Qian. Metal–organic frameworks as platforms for functional materials. *Accounts of Chemical Research*, 49(3): 483–493, 2016. doi:10.1021/acs.accounts.5b00530. URL `https://doi.org/10.1021/acs.accounts.5b00530`. PMID: 26878085.

[12] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=SJU4ayYgl.

[13] Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In Aldo Gangemi, Roberto Navigli, Maria-Esther Vidal, Pascal Hitzler, Raphaël Troncy, Laura Hollink, Anna Tordai, and Mehwish Alam, editors, *The Semantic Web*, pages 593–607, Cham, 2018. Springer International Publishing. ISBN 978-3-319-93417-4.

[14] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.

[15] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ.

[16] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/5dd9db5e033da9c6fb5ba83c7a7ebea9-Paper.pdf.

[17] Jörg Behler. Neural network potential-energy surfaces in chemistry: a tool for large-scale simulations. *Phys. Chem. Chem. Phys.*, 13:17930–17955, 2011. doi:10.1039/C1CP21668F. URL http://dx.doi.org/10.1039/C1CP21668F.

[18] 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.

[19] Johannes Klicpera, Janek Groß, and Stephan Günnemann. Directional message passing for molecular graphs. *CoRR*, abs/2003.03123, 2020. URL https://arxiv.org/abs/2003.03123.

[20] Johannes Klicpera, Florian Becker, and Stephan Günnemann. Gemnet: Universal directional graph neural networks for molecules. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=HS_sOaxS9K-.

[21] Zhengdao Chen, Lisha Li, and Joan Bruna. Supervised community detection with line graph neural networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=H1g0Z3A9Fm.

[22] Kamal Choudhary and Brian DeCost. Atomistic line graph neural network for improved materials property predictions. *npj Computational Materials*, 7(1):1–8, 2021.

[23] 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, 2018. URL https://arxiv.org/abs/1802.08219.

[24] 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.

[25] 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.

[26] Cheol Woo Park and Chris Wolverton. Developing an improved crystal graph convolutional neural network framework for accelerated materials discovery. *Physical Review Materials*, 4(6): 063801, 2020.

[27] 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.

[28] Jiucheng Cheng, Chunkai Zhang, and Lifeng Dong. A geometric-information-enhanced crystal graph network for predicting properties of materials. *Communications Materials*, 2(1):92, September 2021. ISSN 2662-4443. doi:10.1038/s43246-021-00194-3. URL https://doi.org/10.1038/s43246-021-00194-3.

[29] Hendrik J. Monkhorst and James D. Pack. Special points for brillouin-zone integrations. *Phys. Rev. B*, 13:5188–5192, Jun 1976. doi:10.1103/PhysRevB.13.5188. URL https://link.aps.org/doi/10.1103/PhysRevB.13.5188.

[30] Takenori Yamamoto. Crystal graph neural networks for data mining in materials science. *Research Institute for Mathematical and Computational Sciences, LLC*, 2019.

[31] Chi Chen and Shyue Ping Ong. A universal graph deep learning interatomic potential for the periodic table. *Nature Computational Science*, 2(11):718–728, 2022. ISSN 2662-8457. doi:10.1038/s43588-022-00349-3. URL https://doi.org/10.1038/s43588-022-00349-3.

[32] Jiaxuan You, Rex Ying, and Jure Leskovec. Design space for graph neural networks. In *NeurIPS*, 2020.

[33] Alexander Dunn, Qi Wang, Alex Ganose, Daniel Dopp, and Anubhav Jain. Benchmarking materials property prediction methods: the matbench test set and automatminer reference algorithm. *npj Computational Materials*, 6(1):138, 2020. ISSN 2057-3960. doi:10.1038/s41524-020-00406-3. URL https://doi.org/10.1038/s41524-020-00406-3.

[34] Olexandr Isayev, Denis Fourches, Eugene N Muratov, Corey Oses, Kevin Rasch, Alexander Tropsha, and Stefano Curtarolo. Materials cartography: representing and mining materials space using structural and electronic fingerprints. *Chemistry of Materials*, 27(3):735–743, 2015.

[35] Atsushi Togo and Isao Tanaka. Spglib: a software library for crystal symmetry search. *arXiv preprint arXiv:1808.01590*, 2018.

[36] Ralf W. Grosse-Kunstleve, Buddy Wong, Marat Mustyakimov, and Paul D. Adams. Exact direct-space asymmetric units for the 230 crystallographic space groups. *Acta Crystallographica Section A*, 67(3):269–275, May 2011. doi:10.1107/S0108767311007008. URL https://doi.org/10.1107/S0108767311007008.

[37] F. Hoffmann. *Introduction to Crystallography*. Springer International Publishing, 2020. ISBN 9783030351106. URL https://books.google.de/books?id=16H0DwAAQBAJ.

[38] M. O'Keeffe and B.G. Hyde. *Crystal Structures*. Dover Publications, 2020. ISBN 9780486836546. URL https://books.google.de/books?id=_MjPDwAAQBAJ.

[39] T. Hahn, U. Shmueli, A.J.C. Wilson, and International Union of Crystallography. *International Tables for Crystallography*. Number v. 1 in International Tables for Crystallography. D. Reidel Publishing Company, 1984. ISBN 9789027722805. URL https://books.google.de/books?id=W3NNAQAAIAAJ.

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

[41] Muhan Zhang and Pan Li. Nested graph neural networks. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 15734–15747. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/8462a7c229aea03dde69da754c3bbcc4-Paper.pdf.

[42] Frank Harary and Robert Z Norman. Some properties of line digraphs. *Rendiconti del circolo matematico di palermo*, 9(2):161–168, 1960.

[43] Haggai Maron, Heli Ben-Hamu, Hadar Serviansky, and Yaron Lipman. Provably powerful graph networks. *Advances in neural information processing systems*, 32, 2019.

[44] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi:10.1007/s11263-015-0816-y.

[45] Anubhav Jain, Shyue Ping Ong, Geoffroy Hautier, Wei Chen, William Davidson Richards, Stephen Dacek, Shreyas Cholia, Dan Gunter, David Skinner, Gerbrand Ceder, and Kristin A. Persson. Commentary: The materials project: A materials genome approach to accelerating materials innovation. *APL Materials*, 1(1):011002, 2013. doi:10.1063/1.4812323. URL `https://doi.org/10.1063/1.4812323`.

[46] Maarten de Jong, Wei Chen, Thomas Angsten, Anubhav Jain, Randy Notestine, Anthony Gamst, Marcel Sluiter, Chaitanya Krishna Ande, Sybrand van der Zwaag, Jose J Plata, Cormac Toher, Stefano Curtarolo, Gerbrand Ceder, Kristin A. Persson, and Mark Asta. Charting the complete elastic properties of inorganic crystalline compounds. *Scientific Data*, 2(1):150009, 2015. ISSN 2052-4463. doi:10.1038/sdata.2015.9. URL `https://doi.org/10.1038/sdata.2015.9`.

[47] Kamal Choudhary, Irina Kalish, Ryan Beams, and Francesca Tavazza. High-throughput identification and characterization of two-dimensional materials using density functional theory. *Scientific Reports*, 7(1):5179, 2017. ISSN 2045-2322. doi:10.1038/s41598-017-05402-0. URL `https://doi.org/10.1038/s41598-017-05402-0`.

[48] Ivano E. Castelli, David D. Landis, Kristian S. Thygesen, Søren Dahl, Ib Chorkendorff, Thomas F. Jaramillo, and Karsten W. Jacobsen. New cubic perovskites for one- and two-photon water splitting using the computational materials repository. *Energy Environ. Sci.*, 5:9034–9043, 2012. doi:10.1039/C2EE22341D. URL `http://dx.doi.org/10.1039/C2EE22341D`.

[49] Patrick Reiser, André Eberhard, and Pascal Friederich. Graph neural networks in tensorflow-keras with raggedtensor representation (kgcnn). *Software Impacts*, 9:100095, 2021.

[50] Pierre-Paul De Breuck, Geoffroy Hautier, and Gian-Marco Rignanese. Materials property prediction for limited datasets enabled by feature selection and joint learning with MODNet. *npj Computational Materials*, 7(1):83, 2021. ISSN 2057-3960. doi:10.1038/s41524-021-00552-2. URL `https://doi.org/10.1038/s41524-021-00552-2`.

[51] Keqiang Yan, Yi Liu, Yuchao Lin, and Shuiwang Ji. Periodic graph transformers for crystal material property prediction. *arXiv preprint arXiv:2209.11807*, 2022.

[52] Kamal Choudhary, Kevin F. Garrity, Andrew C. E. Reid, Brian DeCost, Adam J. Biacchi, Angela R. Hight Walker, Zachary Trautt, Jason Hattrick-Simpers, A. Gilad Kusne, Andrea Centrone, Albert Davydov, Jie Jiang, Ruth Pachter, Gowoon Cheon, Evan Reed, Ankit Agrawal, Xiaofeng Qian, Vinit Sharma, Houlong Zhuang, Sergei V. Kalinin, Bobby G. Sumpter, Ghanshyam Pilania, Pinar Acar, Subhasish Mandal, Kristjan Haule, David Vanderbilt, Karin Rabe, and Francesca Tavazza. The joint automated repository for various integrated simulations (JARVIS) for data-driven materials design. 6(1):173, 2020. ISSN 2057-3960. doi:10.1038/s41524-020-00440-1. URL `https://doi.org/10.1038/s41524-020-00440-1`.

[53] 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.

[54] Michaël Defferrard, Martino Milani, Frédérick Gusset, and Nathanaël Perraudin. Deepsphere: a graph-based spherical cnn. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=B1e3OlStPB`.

[55] Tim Hsu, Nathan Keilbart, Stephen Weitzner, James Chapman, Penghao Xiao, Tuan Anh Pham, S Roger Qiu, Xiao Chen, and Brandon C Wood. Efficient, interpretable atomistic graph neural network representation for angle-dependent properties and its application to optical spectroscopy prediction. *arXiv preprint arXiv:2109.11576*, 2021.

[56] 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*, 2020.

[57] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

[58] Tianle Cai, Shengjie Luo, Keyulu Xu, Di He, Tie-yan Liu, and Liwei Wang. Graphnorm: A principled approach to accelerating graph neural network training. In *International Conference on Machine Learning*, pages 1204–1215. PMLR, 2021.

[59] 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.

[60] Kangming Li, Brian DeCost, Kamal Choudhary, Michael Greenwood, and Jason Hattrick-Simpers. A critical examination of robustness and generalizability of machine learning prediction of materials properties. 9(1):55. ISSN 2057-3960. doi:10.1038/s41524-023-01012-9. URL `https://doi.org/10.1038/s41524-023-01012-9`.

[61] Logan Ward, Alexander Dunn, Alireza Faghaninia, Nils E.R. Zimmermann, Saurabh Bajaj, Qi Wang, Joseph Montoya, Jiming Chen, Kyle Bystrom, Maxwell Dylla, Kyle Chard, Mark Asta, Kristin A. Persson, G. Jeffrey Snyder, Ian Foster, and Anubhav Jain. Matminer: An open source toolkit for materials data mining. *Computational Materials Science*, 152:60–69, September 2018. ISSN 0927-0256. doi:10.1016/j.commatsci.2018.05.018.

[62] Kamal Choudhary, Brian DeCost, and Francesca Tavazza. Machine learning with force-field-inspired descriptors for materials: Fast screening and mapping energy landscape. *Phys. Rev. Mater.*, 2:083801, Aug 2018. doi:10.1103/PhysRevMaterials.2.083801. URL `https://link.aps.org/doi/10.1103/PhysRevMaterials.2.083801`.

6 Appendix

6.1 Crystal Preprocessing: Edge Selection

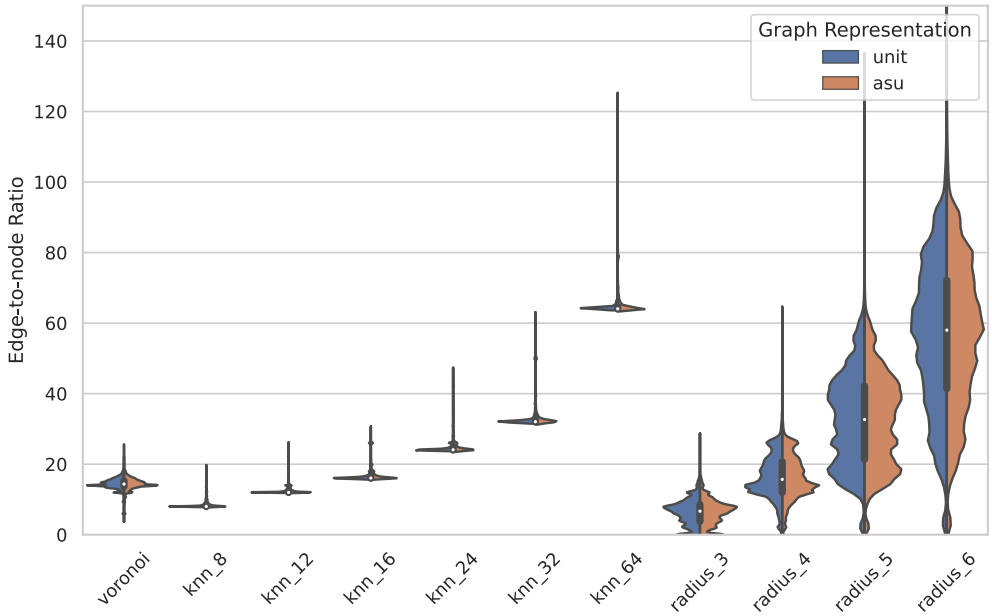


Figure 7: Distributions of edge-to-node ratios per graph (average degree) in the `mp_e_form` dataset for different preprocessing methods. Due to the symmetrical and regular structure of crystals, it often happens that there are many neighbors with equal distances. Therefore our kNN-based edge selection implementation, also allows to toggle whether edges with a ϵ-distance difference as the k-th nearest neighbor are also added as connections. We included an $\epsilon = 10^{-9}$ Åthreshold to make up for numerical inaccuracies.

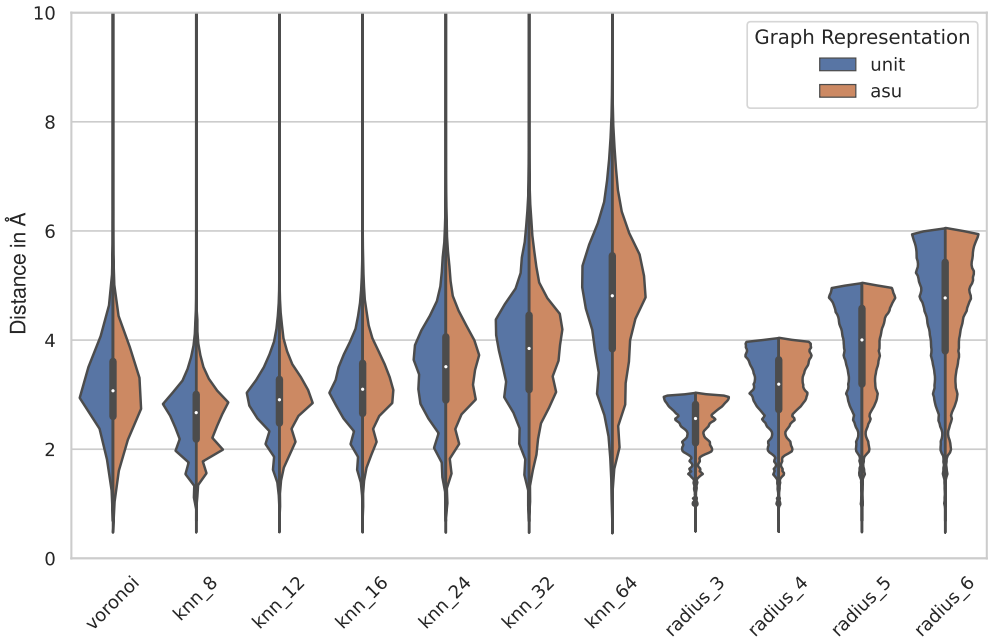


Figure 8: Distribution of edge distances in the `mp_e_form` dataset for different edge selection methods.

6.2 Crystal Preprocessing: Symmetries

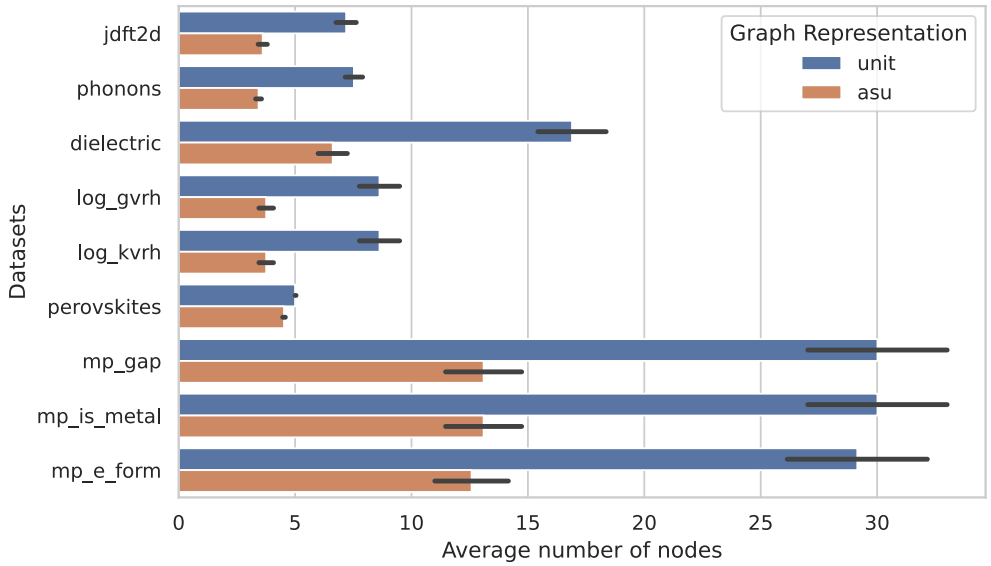


Figure 9: Average number of nodes in each dataset for unit cell graphs (unit) and asymmetric unit graphs (asu).

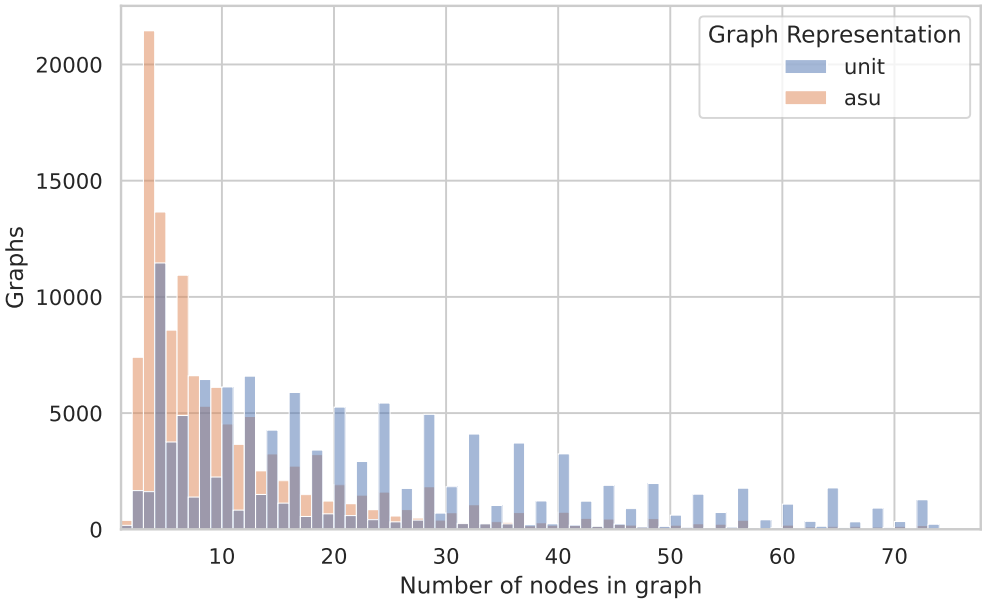


Figure 10: Histogram of number of nodes in `mp_e_form` dataset for unit cell graphs (unit) and asymmetric unit graphs (asu). The distribution for asymmetric unit graphs is skewed towards lower number of nodes. Clearly pronounced periodic peaks at even numbers for the unit cell graph distribution can be explained with symmetric atom pairs.

6.3 Nested Line Graph Network Algorithm

6.4 Line Graph Construction

Mathematically a line graph for a directed graph is defined as follows: An edge exists in $L(G)$ for each corresponding edge pair (e_{ij}, e_{jk}), which forms a path of length two in G (Figure 11a). This definition coincides with the angles $\measuredangle e_{ij}e_{jk}$ as used in in DimeNet [40], GemNet [53] and ALIGNN [22]. In this work, we propose a deviation from the original line graph definition and use angles

(a) Original line graph. (b) Our line graph variant.

Figure 11: Line graph variants.

$\measuredangle e_{ij}e_{kj}$ between edges that have the same destination node, instead of edges that form a path of length two (see Figure 11b). The intuition behind this deviation is based on the interpretation of edge messages in the GNN as force contributions that act on an atom from different directions [54].

Note that we will use the term line graph very loosely in this work. Even though the variant is strictly speaking not a line graph we will still refer to it as such. In general, we will use the term line graph for every graph $L(G)$ that has a bijection between edges in G and nodes in $L(G)$ and a deterministic method for constructing edges in $L(G)$ based on the topology of G.

This generalized definition would also allow for incorporating dihedral angles by adding edges $\measuredangle e_{ij}e_{kl}$ to the line graph for every path (e_{ij}, e_{jk}, e_{kl}) of length three in the graph G [20, 55, 56].

Function NGN-Block($\mathbf{X}_E, \mathbf{X}_V, \mathbf{x}_G, G; \theta$):

```
// X_E = {x_{e_1}, ..., x_{e_m}}
// X_V = {x_{v_1}, ..., x_{v_n}}
for e_ij ∈ E do
    x'_{e_ij} ← φ̂_E(x_{e_ij}, x_{v_i}, x_{v_j}, x_G)                    // Edge update
end
X_∠ ← get_line_graph_edge_features(X_E, X_V, x_G, G)                  // i.e. angles between edges
for t ← 1 to T^{L(G)} do
    _, X'_E, _, _ ← GN-Block(X_∠, X'_E, x_G, L(G); θ^t)               // Treat X'_E as node features of L(G)
end
for v_k ∈ V do
    x̂_{v_k} ← ρ_{E→V}({x'_{e_ij} | e_ij ∈ E ∧ j = k})                 // Local edge aggregation
    x'_{v_k} ← φ_V(x_{v_k}, x̂_{v_k}, x_G)                             // Node update
end
x̂_G ← ρ_{V→G}({x'_v | v ∈ V})                                         // Node aggregation
x̃_G ← ρ_{E→G}({x'_e | e ∈ E})                                         // Global edge aggregation
x'_G ← φ_G(x_G, x̂_G, x̃_G)                                             // Global update
// X'_E = {x'_{e_1}, ..., x'_{e_m}}
// X'_V = {x'_{v_1}, ..., x'_{v_n}}
return X'_E, X'_V, x'_G, G

// Iterate over sequentially composed NGN blocks
for t ← 1 to T do
    X_E, X_V, x_G, G ← NGN-Block(X_E, X_V, x_G, G; θ_t)
end
```

Algorithm 1: Graph network algorithm adapted from [1] with additional blue parts that indicate the modifications for the nested line graph networks. GN blocks are parameterized by parameters θ and receive edge features $\mathbf{X}_E$, node features $\mathbf{X}_V$, graph level features $\mathbf{x}_G$ as well as the graph topology G as input.

Another way to incorporate information of dihedral edges into the model would be to construct the second-order line graph $L(L(G))$ and add another layer of nesting to the model architecture.

6.5 Graph Network Exploration

Graph Network Exploration We start with a simple non-nested architecture shown in Figure 12, which consists of sequentially connected GN blocks, which can be divided into three phases.

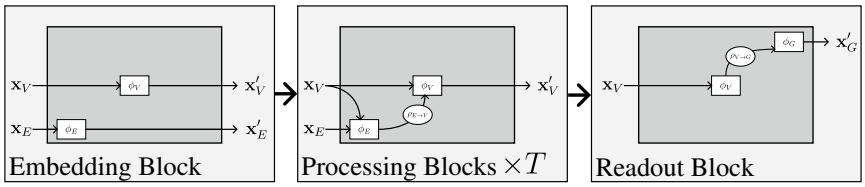


Figure 12: Basic graph network structure.

Moreover, we optimize the hyperparameter space on one dataset only, namely the `log_gvrh` MatBench dataset, and tested the model on all other datasets of MatBench without changing hyperparameters to verify that the found architecture is indeed a suitable candidate for multiple diverse tasks on crystal graphs.

The first block embeds atom/node features and bond/edge features independent from one another. In the second phase, we sequentially connect T processing blocks with identical architecture, but

independent learnable parameters. To restrict the search space we do not include global graph features and do not allow edge updates between blocks. This resembles a conventional message-passing architecture. In the third phase, we have a single readout block, which aggregates node features into graph-level features to make the crystal property prediction.

Figure 12 only specifies the high-level architecture of the GNN and not the concrete instantiations of update functions ϕ_E, ϕ_V, ϕ_G and aggregation functions $\rho_{E\to V}, \rho_{V\to G}$. To narrow down suitable concrete implementations, we conducted a two-part hyperparameter search on the `log_gvrh` MatBench dataset. First, we searched for categorical hyperparameters, in a greedy stepwise search. In the second step, we used the Optuna hyperparameter optimization framework [57] and its implementation of the Tree-structured Parzen Estimator (TPE) to find ordinal hyperparameters.[5]

We assume independence between categorical hyperparameters and optimize for each parameter individually, to keep runtimes within limits. We initially instantiated all update functions ϕ_E, ϕ_V, ϕ_G with MLPs of depth 3. Extending ϕ_V with residual $\phi_V(\mathbf{x}_v, \hat{\mathbf{x}}_v, \mathbf{x}_G) = \mathbf{x}_v + \text{MLP}_V(\hat{\mathbf{x}}_v)$ and gated node updates $\phi_V(\mathbf{x}_v, \hat{\mathbf{x}}_v, \mathbf{x}_G) = \text{GRU}(\mathbf{x}_v, \text{MLP}_V(\hat{\mathbf{x}}_v))$ both improve performance. For aggregation function ρ_V, ρ_G we tried mean, sum, and attention-based functions. Batch and graph normalization [58] increased training time significantly without considerable benefits for predictive performance. Prediction accuracy further increases when including atom features (atomic mass, radius, electronegativity, ionization, and oxidation states) in addition to the atomic number. The influence of the choice of edge selection methods shown in Figure 5 is discussed in detail in Section 4.2.

For the kNN-based edge selection preprocessing with $k = 24$, we searched for ordinal hyperparameters with a hyperparameter optimization.

Embedding Block

ϕ_E: Gauss basis expansion of edge distances with 32 Gaussians evenly spread on the $[0, 8]$ Å interval. As only distance information is used, this embedding is E(3)-invariant. The initial representation is projected into a 128-dimensional embedding space with a single perceptron layer.

ϕ_V: Embedding of atom features (atomic number, atomic mass, atomic radius, electronegativity, ionization states, and oxidation states) into a 128-dimensional space.

Processing Blocks Five processing blocks are concatenated with identical configurations ($T = 5$).

$\phi_E(\mathbf{x}_{e_{ij}}, \mathbf{x}_{v_i}, \mathbf{x}_{v_j}) = \text{MLP}_E(\mathbf{x}_{e_{ij}} || \mathbf{x}_{v_i} || \mathbf{x}_{v_j}) = \mathbf{x}'_{e_{ij}}$ The edge update function, which constructs the message between two nodes, is a five-layer MLP and takes the concatenation of edge, receiver, and sender node features.

$\rho_{E\to V}(\{\mathbf{x}'_{e_{ij}} | j = k\}) = \underset{j=k}{\text{sum}}(\mathbf{x}'_{e_{ij}}) = \hat{\mathbf{x}}_{v_k}$ For each node the incoming messages are sum-aggregated.

$\phi_V(\mathbf{x}_v, \hat{\mathbf{x}}_v) = \mathbf{x}_v + \text{MLP}_V(\hat{\mathbf{x}}_v)$ A residual node update function transforms the aggregated messages with a single-layer perceptron.

Readout Block

$\phi_V(\mathbf{x}_v) = \mathbf{x}_v$ The readout block does not update node features. Its node update function is the identity function.

$\rho_{V\to G}(\{\mathbf{x}'_v | v \in V\}) = \underset{v\in V}{\text{mean}}(\mathbf{x}'_v) = \hat{\mathbf{x}}_G$ The mean of the node features is aggregated to compute the graph-level prediction.

$\phi_G(\hat{\mathbf{x}}_G) = \text{MLP}_G(\hat{\mathbf{x}}_G) = \mathbf{x}'_G$ A single-layer MLP with a linear activation function creates the final prediction for the crystal property.

Below we give a complete description of our model, which we name coGN, after categorical and ordinal hyperparameter optimization:

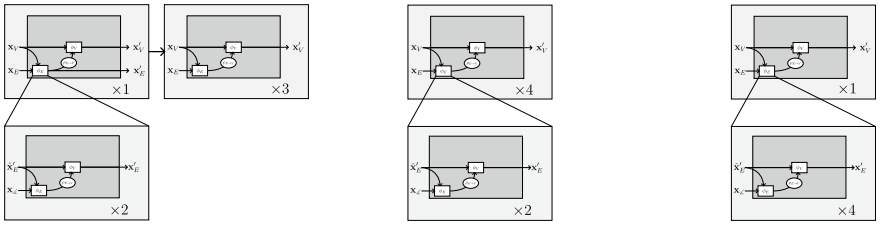
We used the same dimensionality (128) for all hidden representations of edges, nodes, and graphs. Unless mentioned otherwise we use the commonly used swish activation function in MLPs. The GNN is trained with an Adam optimizer with a linear learning rate scheduler for 800 epochs.

[5] Ordinal hyperparameters: depth of GNN (T), depth of MLPs $\text{MLP}_E, \text{MLP}_V, \text{MLP}_G$, dimensionality of features

Overall, coGN is comparably simple, as it only contains MLPs as update functions, mean or sum aggregation functions and no sophisticated message-passing scheme, such as edge-gated or attention-based message passing used in CGCNN, ALIGNN or GeoCGNN. In this regard, coGN also does not incorporate any domain-specific design decisions, which are not justified by the hyperparameter optimization.

When compared with previous models, we find that along with a connectivity optimization, discussed in section 4.2, a much deeper edge update network ϕ_E, in our case five layers, yields better results. We attribute this observation to the increased complexity of edge information in a periodic multi-graph that in practice exhibits a large number of edges per node.

Nested Line Graph Network Exploration Based on the results of the first part of the hyperparameter search for parameters with categorical values, we augment the processing blocks with nested GN blocks shown in Figure 13.



(a) NLGN variant 1. (b) NLGN variant 2. (c) NLGN variant 3.

Figure 13: Nested line graph network variants.

The first variant (Figure 13a) has a single nested GN block at the beginning, which updates node and edge features, followed by non-nested GN blocks. The reasoning behind this architecture is that the Nested GN block might be able to encode geometrical constellations of edge angles into edge features. The second variant (Figure 13b) is a generalized version of ALIGNN-d [55] and DimeNet(++) [40, 56]. Each GN block on the graph level (G) has two GN blocks on the line graph level ($L(G)$). Only node features are updated between GN blocks. The third variant (Figure 13c) moves most of the computation to the line graph level ($L(G)$). It consists of only one GN block on the graph level with 4 consecutive nested GN blocks ($T^{L(G)} = 4$).

E(3)-invariant angle features between two edges are encoded with a 16-dimensional Gauss basis expansion on the $[0, \pi]$ rad interval and attached to line graph edges.

Table 3: Results on the `log_gvrh` dataset with NLGN variants.

Nested GN	Line Graph	MAE (`log_gvrh`)
Variant 1	$\measuredangle e_{ij}e_{jk}$	0.0809 ± 0.0022
	$\measuredangle e_{ij}e_{kj}$	0.0787 ± 0.0019
Variant 2	$\measuredangle e_{ij}e_{jk}$	0.0801 ± 0.0020
	$\measuredangle e_{ij}e_{kj}$	0.0783 ± 0.0033
Variant 3	$\measuredangle e_{ij}e_{jk}$	0.0805 ± 0.0015
	$\measuredangle e_{ij}e_{kj}$	0.0799 ± 0.0016

Training on preprocessed crystals with kNN edge selection and $k = 24$ and both line graph variants from Section **??**, we obtain the results displayed in Table 3. The results were obtained before the final ordinal hyperparameter optimization, which explains the discrepancy with Table 2. For comparison, the MAE for the corresponding non-nested GN is 0.0788.

Despite the theoretically greater expressiveness of NLGNs, we do not achieve substantially better prediction results. The proposed line graph variant with angles between edges with the same target node ($\measuredangle e_{ij}e_{kj}$) leads to a small but consistent improvement across all line graph variants.

We optimized the ordinal hyperparameters of the DimeNet-like architecture (Variant 2) with TPE and reached a MAE of 0.0705 on the `log_gvrh` MatBench dataset which yields better performance than the current leader on MatBench [22], even though the graph preprocessing, i.e. the connectivity was not optimized yet. Hyperparameter optimization of plain GNs yields a similar error of 0.0693, making them comparable to nested GNs.

6.6 Open Catalyst Project: OC22

Table 4: Test error for the *initial structure to relaxed energies* (IS2RES) task of the OC22 challenge [9] (status 2023-08-08). Note that Models trained additionally on OC20 [8] and other data sources or indirect predictions using relaxations yield lower errors and better performance. Baseline models are SchNet [18], DimeNet++ [19], PaiNN [59] and GemNet [20]. The best Direct OC22-only predictions is marked by underscores.

Model	MAE (ID)	MAE (OOD)	Average
EquiformerV2 (122M, λ_E=1, λ_F=1, OC22-only)	**1.0837**	1.4443	**1.264**
EquiformerV2 (122M, λ_E=4, λ_F=100, OC22-only)	1.1181	**1.4398**	1.279
GemNet-OC (Relaxation OC20-All+OC22)	1.2007	1.5339	1.3673
GemNet-OC (Relaxation OC22-only)	1.3294	1.5841	1.4567
GemNet-dT (Relaxation OC22-only)	1.8129	2.0439	1.9284
coGN (Direct OC22-only, r=5.0)	<u>1.6183</u>	<u>2.8058</u>	<u>2.2121</u>
coGN (Direct OC22-only, k=32)	1.6278	2.9706	2.2992
GemNet-dT (Direct OC22-only)	1.6771	3.0837	2.3804
PaiNN (Direct OC22-only)	1.716	3.6835	2.6997
DimeNet++ (Direct OC22-only)	1.96	3.5186	2.7393
SchNet (Direct OC22-only)	2.0012	4.8468	3.424

6.7 JARVIS-Tools

Table 5: Test error for multiple tasks of the JARVIS benchmark [52]. Values are copied from the JARVIS leaderboard (status 2023-08-08). Models for comparison are DimeNet++ (DN++) [19] , ALIGNN [22], CGCNN [25], MatMiner (MM) [60, 61] and CFID [62].

Task	coGN	coNGN	DN++	ALIGNN	CGCNN	MM	CFID
dft_3d_mepsz	24.1081	**22.842**	30.3644	23.7313	36.0538	24.6651	29.3445
dft_3d_exfoliation_energy	47.6979	46.272	46.1517	52.7033	52.7033	**40.887**	62.1169
dft_3d_shear_modulus_gv	8.6612	**8.4881**	26.0817	9.476	16.0459	10.5415	11.9164
dft_3d_spillage	0.3609	**0.3463**	0.4137	0.351	0.3965	0.3592	0.3867
dft_3d_optb88vdw_total_energy	**0.0262**	0.0273	0.051	0.0367	0.0815	0.0936	0.2436
dft_3d_mepsx	24.2289	**23.3801**	31.9568	24.0458	33.6597	25.2932	30.261
dft_3d_epsz	19.6192	**17.8104**	33.8379	19.5678	33.6597	25.2932	24.781
dft_3d_dfpt_piezo_max_dij	15.2235	**13.8868**	13.9889	20.5705	16.0135	21.5729	-
dft_3d_mepsy	24.1891	**23.3299**	31.0215	23.6482	32.4577	25.0706	30.0578
qe_tb_indir_gap	0.0474	-	-	0.1167	-	**0.0351**	-
dft_3d_kpoint_length_unit	9.5722	9.3459	11.8875	9.5146	13.2145	**9.047**	9.7085
dft_3d_n_powerfact	452.235	456.6118	568.8357	**442.2993**	-	469.6279	-
dft_3d_ph_heat_capacity	6.1125	7.8127	23.3618	9.6064	-	**5.2757**	-
dft_3d_formation_energy_peratom	**0.0271**	0.0291	0.0528	0.0331	0.0625	0.0734	0.1419
dft_3d_epsx	20.0004	**18.5738**	27.2511	20.3942	31.4744	21.2597	24.8408
dft_3d_optb88vdw_bandgap	**0.1219**	0.1267	0.2247	0.1423	0.1908	0.1873	0.299
qe_tb_energy_per_atom	**0.0636**	-	0.7515	-	-	1.5049	-
dft_3d_max_efg	20.4417	19.5495	26.9552	**19.1211**	24.6695	19.4382	-
dft_3d_epsy	24.1891	**23.3299**	31.0215	23.6482	32.4577	25.0706	30.0578
dft_3d_encut	133.8915	**129.8266**	164.315	133.7962	190.3857	138.2769	139.4357
dft_3d_n_Seebeck	**39.2692**	40.0977	54.2759	40.9214	49.3172	44.2229	-
dft_3d_ehull	**0.0466**	0.0485	0.3685	0.0763	0.173	0.0601	-
dft_3d_bulk_modulus_kv	8.992	**8.7022**	13.3743	10.3988	19.3028	12.7411	14.1999
dft_3d_avg_hole_mass	0.1372	0.1285	0.1709	**0.1239**	-	0.1529	-
dft_3d_avg_elec_mass	0.0917	0.0876	0.112	**0.0853**	-	0.107	-
dft_3d_mbj_bandgap	**0.264**	0.2719	0.4764	0.3104	0.4067	0.3392	0.5313
dft_3d_dfpt_piezo_max_dielectric	30.2923	**25.5553**	30.3358	28.1514	32.5589	36.6913	-
dft_3d_slme	4.4507	**4.4428**	5.6403	4.5207	5.6603	4.9255	6.2607
qe_tb_f_enp	**0.0956**	-	-	0.1016	-	0.3219	-
dft_3d_magmom_oszicar	0.2502	**0.2437**	0.3995	0.2574	0.3543	0.3645	0.4748
qe_tb_final_energy	**1.3185**	-	-	-	-	1.4714	-