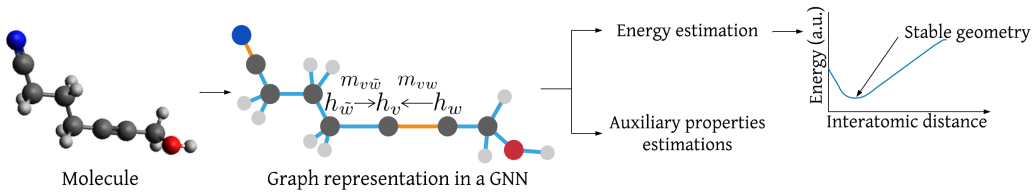
Domain-informed graph neural networks: a quantum chemistry case study

Jay Morgan, Adeline Paiement, Christian Klinke

Graphical Abstract

Domain-informed graph neural networks: a quantum chemistry case study

Jay Morgan, Adeline Paiement, Christian Klinke



Highlights

Domain-informed graph neural networks: a quantum chemistry case study

Jay Morgan, Adeline Paiement, Christian Klinke

- We leverage the existence of different edge types to modulate the information flow within graph neural networks. To this end, we formulate and compare two strategies, namely specialised message production and specialised state update.
- We leverage a multi-task learning framework to enforce learnt representations to be more related to quantities of interest. We explore the potential of this approach to better capture the underlying mechanisms behind the studied phenomenon.
- We evaluate our domain knowledge integration strategies on the case study of estimating the potential energy of chemical systems (molecules or crystals).
- To support these experiments, we release three new datasets of out-of-equilibrium molecules and crystals of various complexities.

Domain-informed graph neural networks: a quantum chemistry case study

Jay Morgan[a,b,**], Adeline Paiement[a,b,*], Christian Klinke[c,d,e,]

[a] *Université de Toulon, Aix Marseille Univ, CNRS, LIS, Marseille, France*
[b] *Department of Computer Science, Swansea University, Swansea, SA2 8PP, United Kingdom*
[c] *Institute of Physics, University of Rostock, Rostock, 18059, Germany*
[d] *Department "Life, Light & Matter", University of Rostock, Rostock, 18059, Germany*
[e] *Department of Chemistry, Swansea University, Swansea, SA2 8PP, United Kingdom*

Abstract

We explore different strategies to integrate prior domain knowledge into the design of graph neural networks (GNN). Our study is supported by a use-case of estimating the potential energy of chemical systems (molecules and crystals) represented as graphs. We integrate two elements of domain knowledge into the design of the GNN to constrain and regularise its learning, towards higher accuracy and generalisation. First, knowledge on the existence of different types of relations/graph edges (e.g. chemical bonds in our case study) between nodes of the graph is used to modulate their interactions. We formulate and compare two strategies, namely specialised message production and specialised update of internal states. Second, knowledge of the relevance of some physical quantities is used to constrain the learnt features towards a higher physical relevance using a simple multi-task learning (MTL) paradigm. We explore the potential of MTL to better capture the underlying mechanisms behind the studied phenomenon. We demonstrate the general applicability of our two knowledge integrations by applying them to three architectures that rely on different mechanisms to propagate information between nodes and to update node states. Our implementations are made publicly available. To support these experiments, we release three new datasets of out-of-equilibrium molecules and crystals of various complexities.

[*] Corresponding author, adeline.paiement@univ-tln.fr
[**] jay.morgan@univ-tln.fr
christian.klinke@uni-rostock.de

Keywords: Graph neural network, Domain knowledge integration, Quantum chemistry application

1. Introduction

We investigate the introduction of domain knowledge into the design of a graph neural network (GNN), to constrain and regularise its learning towards higher accuracy and generalisation. GNNs were first proposed in (Scarselli et al., 2009) to process data represented as graph. An internal state $\mathbf{h}_v$ is associated to each node v of the graph, and the ensemble of internal states serves to produce an output $\hat{y}$. Each internal state is iteratively updated, based on 1) input features $\mathbf{x}_v$ associated to the node, 2) input features $\bar{\mathbf{x}}_{vw}$ associated to the edges of the node, possibly complemented by internal states $\bar{\mathbf{h}}_{vw}$ associated to these edges, and 3) actions from neighbour nodes w, represented as a message $\mathbf{m}_v = \sum_w \mathbf{m}_{vw}$. The message $\mathbf{m}_{vw}$ is generated by a message function $M(\mathbf{h}_v, \mathbf{h}_w, \bar{\mathbf{x}}_{vw}, \bar{\mathbf{h}}_{vw})$[1] using the internal state $\mathbf{h}_w$ of the neighbouring node. The message function M is shared by all nodes. It is typically implemented by a perceptron, e.g. as in (Liu et al., 2021a) where a perceptron is applied to a concatenation of $\bar{\mathbf{x}}_{vw}$ and of $\mathbf{h}_v \circ \mathbf{h}_w$ ($\circ$ being element-wise multiplication). The update function $U(\mathbf{h}_v, \mathbf{x}_v, \mathbf{m}_v)$[2] is also shared by all nodes. It is originally implemented by a perceptron, but it can also benefit from a recurrent kernel such as Gated Recurrent Unit (GRU) (e.g. in (Gilmer et al., 2017)) or Long Short-Term Memory (LSTM). Depending on the task, the GNN's output is then computed from the node states by a readout function, which may be implemented by a perceptron or more complex kernels. Variants of the original GNN have been proposed to accommodate special graph types, such as directed, see e.g. (Wu et al., 2021; Zhou et al., 2020) for an overview.

We explore avenues for integrating domain knowledge into the design of GNNs, with the aim to constrain and simplify the GNN's learning to improve its generalisation and accuracy, and to allow training on smaller datasets. We consider two distinct aspects, namely 1) the information flow within the GNN, and 2) the relevance of learnt node states for the application domain.

[1] This is a general definition, and in some implementations M may only use a subset of its inputs.

[2] Here also this is a general definition, and in some implementations U may only use a subset of its inputs.

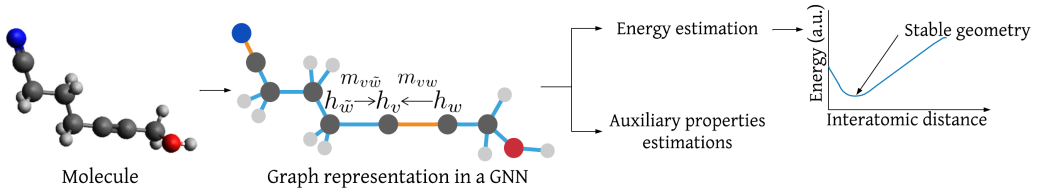


Figure 1: Graph representation of a molecule (from QM9) for GNN estimation of potential energy and stable geometry. Nodes of states $\mathbf{h}_i$ are atoms and colour denotes atom type (black: carbon, white: hydrogen, blue: nitrogen, red: oxygen). Edges link chemically bonded atoms and colour denotes bond type (blue: single bond, orange: double bond). Non-bonded atoms may also share edges in fully connected graphs, but these edges are not represented for readability. Messages $\mathbf{m}_{ij}$ are exchanged across edges. The GNN outputs an energy estimate, along with the estimates of one or several auxiliary (physical and chemical) properties in the case of multi-task learning. Energy estimates at different molecule geometries may be used to identify stable configurations at the minimum of energy. Left drawing is from (Glavatskikh et al., 2019).

Regarding point 1), it is of particular interest to account for the fact that nodes may share different types of relations, and therefore different information during inference. A few previous works proposed to specialise the message function M with regards to the type of relation between nodes e.g. (Schlichtkrull et al., 2018; Zhang et al., 2019). Other works, e.g. (Chen et al., 2021), used separate sub-graphs for different relation types. We expand and generalise on the initial proposition of (Schlichtkrull et al., 2018) through the exploration of different pathways for specialising the information flow within the GNN, namely by acting on the message generation M or on the update U of internal states. We propose a more general formulation for specialising M than the previously mentioned specialisation methods. To the best of our knowledge, the specialisation of U was not previously proposed. In addition, while previous methods are dependent on the GNN architecture, ours applies to all GNN architectures and their implementations of M and U.

With point 2), we also exploit the knowledge that some auxiliary quantities/qualities are closely related to the studied phenomena and should play a role in the internal representation of the Deep Neural Network (DNN).

We consider the case study of estimating *potential energies*[3] of a chemical system, either molecule or crystal, as a function of geometry (i.e. position of

[3]determined at the electronic ground-state at given positions of atoms (i.e. given geometry) for static systems

atoms). Such systems are represented as chemical graphs, with nodes denoting individual atoms, and edges the type of bond between them, as illustrated in Fig. 1. For *out-of-equilibrium (OoE)* systems, the atoms are at positions which are not at the minimum of potential energy. There is an interest in estimating the potential energy of OoE systems, as this allows searching for stable geometries through minimisation of the energy, with a possible aim to discovering new materials or simulating crystal growth. Such calculations typically require large amounts of resources and do not scale well to larger system sizes (i.e. number of atoms) (Jiang et al., 2003; Erba et al., 2017). This was addressed in the seminal work Message Passing Neural Network (MPNN) (Gilmer et al., 2017) and with successive improvements in SchNet (Schütt et al., 2017b) and DimeNet/DimeNet++ (Klicpera et al., 2020b,a) with GNNs that estimate energy in a matter of milliseconds. Indeed, GNNs and their message passing principle are well suited to capture the atomic interactions that underpin the system's chemical properties. Further recent works have built on these GNNs, e.g. (Liu et al., 2021b; Schütt et al., 2021; Satorras et al., 2021) improve on MPNN, (Unke and Meuwly, 2019; Klicpera et al., 2020b,a; Batzner et al., 2022) improve on SchNet, and (Gasteiger et al., 2021a,b) improve on DimeNet++.

We summarise our main contributions as follows:

[C1] We present methods to augment *existing* GNNs through integrating domain knowledge into their design. We are not aware of other works that follow this aim, as previous works focused on proposing new GNN architectures usually for some given applications. We demonstrate our method by augmenting three different existing GNN architectures, namely MPNN, SchNet, and DimeNet++.

[C2] We leverage the existence of different edge types to modulate the information flow within the GNN (Section 3.1). To this end, we formulate and compare two strategies, namely specialised message production and specialised state update. For the specialised state update, we provide different implementations for GRU and for dense layer-based update functions. While specialised message production was proposed in some previous works and for specific architectures, our formulation is more general since it does not depend on the GNN architecture. Our proposed principle of specialised state update is novel.

[C3] We leverage a multi-task learning framework to enforce learnt representations to be more related to quantities of interest (Section 3.2). We explore the potential of this approach to better capture the underlying mech-

anisms behind the studied phenomenon.

[C4] We evaluate our domain knowledge integration strategies both individually and jointly, on the case study of estimating the potential energy of chemical systems (molecules and crystals). Our methods are applied to the very different architectures of MPNN, SchNet, and DimeNet++ to demonstrate their flexibility to GNN type (Section 4). MPNN, SchNet, and DimeNet++ are the basis for most recent models within our case study. Therefore, our improved performances for these base models suggest that our methods may be applied to more recent models with similar improvements.

While our knowledge integration strategies are demonstrated on potential energy estimation, they may be applied more generally to estimating other properties of chemical systems, such as different bioactivities and properties of molecules for drug discovery (Xiong et al., 2020), and even to other application domains where graph representations are relevant and where GNNs are used for prediction, for example circulation of goods and people in economics (Panford-Quainoo et al., 2020), traffic prediction (Diehl et al., 2019), E-commerce recommendations (Liu et al., 2021a), or biomedical knowledge discovery from knowledge graphs (Callahan et al., 2020).

[C5] We demonstrate that our proposed augmentation of existing GNNs through domain knowledge results in a improved performance on several points: the achieved accuracy on the task is increased, training is possible on smaller datasets, the GNN generalises better to new sizes of graphs, the GNN generalises better to new configurations of graphs such as new perturbations of node locations/properties.

[C6] To support these experiments, we release three new datasets of OoE molecules and crystals of various complexities (Section 4.1).

2. Previous works

2.1. GNNs for estimating potential energies of chemical systems

Gilmer et al. (2017) implement with their MPNN[4] a rather classical GNN with atom type as node feature, interatomic distance as edge feature, a perceptron as message function M, and GRU as update function U. The internal state $\mathbf{h}_v$ is initialised as the node feature $\mathbf{x}_v$. Then, messages and node states

[4]We use the implementation from: `https://github.com/priba/nmp_qc` (MIT license).

at iteration $t+1$ are computed as:

$$\mathbf{m}_v^{t+1} = \sum_{w \in \mathcal{N}_v} M\left(\mathbf{h}_w^t, \mathbf{x}_{vw}^e\right) \tag{1}$$

$$\mathbf{h}_v^{t+1} = U\left(\mathbf{h}_v^t, \mathbf{m}_v^{t+1}\right) \tag{2}$$

with $\mathcal{N}_v$ the set of neighbours of atom v. Three iterations are performed, then the set of node states at all timesteps is used to estimate the system's potential energy as the sum of individual nodes' contributions, computed by perceptrons, using a mean-squared error loss. In basic MPNN, only bonded atoms exchange messages ($\mathcal{N}_v$ only contains bonded neighbours). In a fully connected graph variant, which we denote as Extensive MPNN (E-MPNN), all pairs of atoms exchange messages to account for long-range interactions. Gilmer et al. (2017) also introduced bond type (BT) information as edge input feature. This additional feature improved energy estimates over using interatomic distance alone. We refer to this other variant as MPNN-BTF (MPNN with Bond Type Feature).

With SchNet[5], Schütt et al. (2017a) implement another GNN with 3 layers of interactions that do not share weights, as opposed to the recurrent interactions of MPNN, but weights are still shared between nodes/atoms for each layer. The interaction layers are followed by atom-wise dense layers, non-linearities, and a final sum pooling to obtain the energy estimate. Each interaction layer l considers the sum of actions of neighbours on atom v in a fully connected graph (Eq. 3). The atoms' actions (i.e. messages) are element-wise multiplication $\circ$ of their node states $\mathbf{h}_w^l$ (updated by a dense layer with parameters $\mathbf{W}^l$, $\mathbf{b}^l$) with a radial filter R^l that depends on the distance d_{vw} between the two atoms. R^l is implemented by two softplus dense layers. Node states are initialised based on atom type (i.e. node feature), then updated by a dense layer and non-linearity-based update function (Eq. 4).

$$\mathbf{m}_v^{l+1} = \sum_{w \in \mathcal{N}_v} \left(\mathbf{W}^l \mathbf{h}_w^l + \mathbf{b}^l\right) \circ R^l\left(d_{vw}\right) \tag{3}$$

$$\mathbf{h}_v^{l+1} = U^l\left(\mathbf{h}_v^l, \mathbf{m}_v^{l+1}\right) = \mathbf{h}_v^l + V^l\left(\mathbf{m}_v^{l+1}\right) \tag{4}$$

[5] `https://github.com/atomistic-machine-learning/schnetpack` (MIT license)

DimeNet and DimeNet++[6] (Klicpera et al., 2020b,a) are other GNNs with successive interaction layers that do not share weights. They each perform two steps of message passing across a chain of three atoms. This allows accounting explicitly for the angle a_{kwv} formed by these three atoms. This is achieved by first updating directional edge states $\bar{\mathbf{h}}_{vw}$ from actions of neighbour atoms of w (Eqs. 5 and 6). These edge states then serve to produce messages $\mathbf{m}_{vw}$ that represent the information flow from atom w to atom v (Eq. 7). For the first pass of message passing, an edge message $\bar{\mathbf{m}}_{vw}$ is produced by a message function $\bar{M}$, and a function $\bar{U}$ updates the edge state $\bar{\mathbf{h}}_{vw}$. For the second pass, a second message function M produces the messages $\mathbf{m}_{vw}$. The node state update function $U(\mathbf{m}_{vw})$ is simply the identity.

$$\bar{\mathbf{m}}_{vw}^{l+1} = \sum_{k \in \mathcal{N}_w \setminus \{v\}} \bar{M}^l \left(\bar{\mathbf{h}}_{wk}^l, d_{vw}, d_{wk}, a_{kwv} \right) \tag{5}$$

$$\bar{\mathbf{h}}_{vw}^{l+1} = \bar{U}^l \left(\bar{\mathbf{h}}_{vw}^l, \bar{\mathbf{m}}_{vw}^{l+1} \right) \tag{6}$$

$$\mathbf{m}_{v}^{l+1} = \sum_{w \in \mathcal{N}_v} M^l \left(\bar{\mathbf{h}}_{vw}^{l+1}, d_{vw} \right) \tag{7}$$

$$\mathbf{h}_{v}^{l+1} = \mathbf{m}_{v}^{l+1} \tag{8}$$

$\bar{M}$, $\bar{U}$, and M combine fully connected and residual operations. They use spherical Bessel functions and spherical harmonics to encode d_{wk} and a_{kwv}, and radial Bessel basis functions to encode d_{vw}. A readout function does dense transformations of the node states, for all layers, which are then summed across layers and nodes to produce the final output.

Further recent works have built on these GNNs, e.g. (Liu et al., 2021b; Schütt et al., 2021; Satorras et al., 2021) improve on MPNN, (Schütt et al., 2021; Unke and Meuwly, 2019; Klicpera et al., 2020b,a; Batzner et al., 2022) improve on SchNet, and (Gasteiger et al., 2021a,b) improve on DimeNet++. Choudhary and DeCost (2021) update node and edge states as in DimeNet++, using the formalism of two parallel and intertwined graphs. Similar to DimeNet++, Thürlemann and Riniker (2023) perform two steps of message passing with directional edge states based on Cartesian multipoles. Zitnick et al. (2022) implement a more classical one-step message passing, but define

[6]We use the implementation from: `https://github.com/pyg-team/pytorch_geometric/blob/master/torch_geometric/nn/models/dimenet.py` (Hippocratic License 2.1).

node states and messages as spherical functions, using spherical harmonics, to encode angular information. Kaba and Ravanbakhsh (2022) generalise the message passing idea to equivariance to the symmetry of the data, in order to handle and exploit the high degree of symmetry in crystals of various lattice types. This is achieved through appropriate specialisations of M and U to different symmetries. While we also implement specialised message and update functions, this pursues a different goal. Overall, the two ideas of message passing and of information embedding within node states remain central in all these works, and are the basis of our proposed augmentations in Section 3.

2.2. Domain-informed design of a deep neural network

In (Gilmer et al., 2017; Schütt et al., 2017a; Liu et al., 2021b; Schütt et al., 2021; Unke and Meuwly, 2019; Klicpera et al., 2020b,a; Batzner et al., 2022), the choice of a GNN architecture is a well-motivated inductive bias where the internal representation and information flow are designed to fit with the physics of the replicated phenomenon, namely the interactions of atoms producing the system's potential energy. Other similar examples of domain-informed choice of a GNN architecture include (Mrowca et al., 2018) that predicts future states of deformable objects represented by a hierarchical graph that decomposes them into particles at various scales. Convolution operations are defined on this graph to apply external forces to the system, and to exchange information on collisions and physical state change. Similarly, Salehi and Giannacopoulos (2022) use a GNN to align brain images while predicting physically meaningful soft tissue deformations, and (Kawahara et al., 2017) defined convolution operations on a brain connectivity graph to predict neurodevelopment scores. Diehl et al. (2019) found that accounting for interactions between neighbouring vehicles helps predict short-term behaviours of traffic participants. When predicting bilateral trade, Panford-Quainoo et al. (2020) hypothesised that "adoption of specific domestic trade policies in one country can influence similar policies to its neighbours", which was realised as a graph to represent trade relationships between countries. Liu et al. (2021a) also used a GNN architecture for E-commerce recommendation where missing relationships are inferred from a partially connected graph.

Some works exploited domain knowledge to constrain the information shared between nodes of a GNN. Wang et al. (2023) accounted for the attraction or repulsion of nodes, based on different/similar node types, in the

computation of messages. Schlichtkrull et al. (2018) considered the case of edges representing different types of relation between nodes, and proposed to specialise the message function M with regards to relation type r. This was achieved with distinct perceptrons. Zhang et al. (2019) obtained a similar specialisation of M to the relation type using specialised weights. Chen et al. (2021) took a different approach to account for relation type, with separate and specialised GNNs for each sub-graph of a given edge type, before final fusion of learnt representations. In this work, we are also interested in exploiting the relation type between nodes. First, we propose a generalisation of the specialised M of (Schlichtkrull et al., 2018) which does not depend on the GNN architecture. Second, we also experiment with specialising the embedding update function U, which is a new paradigm to the best of our knowledge.

In Yu and Gao (2022), the relation type between nodes (as well as types of nodes) are considered to stress some key edge types in the learning of graph representations. However, contrary to the previously cited works, the edge types are exploited in a motif GNN in parallel to the molecular GNN, before fusion of their representations. This added layer of complexity is not considered in this study, as we focus on single GNN processing.

Other works have adapted the training of DNN based upon prior physics knowledge. Raissi et al. (2018) constrained a dense-layer DNN that estimates physical quantities (e.g. velocity, pressure) by a loss term obtained from partial differential equations of fluid dynamics. These equations are implemented using automatic differentiation, and their parameters are learnable. Schütt et al. (2017a) used a similar approach to improve SchNet's predictions of both energies and their derivatives w.r.t. atom positions, using a loss term based on computed interaction forces. In the present work, we do not employ equations of atomic interactions, as they become intractable for larger systems. Instead, we focus on the domain-informed architecture avenue, i.e. how a GNN's design may further account for the known properties (e.g. known physics in our case study) of the problem.

3. Proposed domain knowledge integration

3.1. Specialising interactions based on edge type

The types of relation between nodes of a graph are an important factor when considering the nodes' interactions towards the GNN's inference.

As an illustration, in our case study, the types of bonds between atoms determine atomic interactions and their contribution to the system's energy. Therefore, when estimating the potential energy of a system as a function of geometry, it may be beneficial to account for the contribution of different BT. MPNN-BTF provide a clue towards confirming this assertion. By introducing BT information as edge input feature in their GNN, Gilmer et al. (2017) improved energy estimates over using distance alone. We take this domain knowledge integration principle further, and we propose to design the information exchange within the GNN to reflect the relations between nodes. In our case study, this results in the nodes' information exchange better reflecting the physics of atomic interactions.

Similar to (Schlichtkrull et al., 2018), we introduce *specialised interactions based on relation type*, with relation type being BT[7] in our case study. While (Schlichtkrull et al., 2018) focused on the production of messages that are specialised to the relation, we explore two strategies of specialising either the message production, or the state update. These two latter processes operate based on their own and separate unit types (e.g. perceptron, GRU...) and learnt parameters. Thus, our two strategies act on different learning elements within the GNN. Furthermore, while (Schlichtkrull et al., 2018) focused on a single GNN type (namely Graph Convolutional Network, GCN), we provide more general formulations that are adapted to different architectures.

Since the DimeNet++ architecture implements two steps of message production and state update at each layer (see Section 2.1), we specialise interactions at both steps. However, given the extreme simplicity of the U function (identity), we only specialise $\bar{M}$, M, and $\bar{U}$.

3.1.1. Specialised message production

Separate messages $\mathbf{m}_{vw}^{r}$ for each relation type r between nodes are produced by a relation-specialised message function M_r, as illustrated in Fig. 2. In the specific case of a perceptron-based M_r, this is equivalent to (Schlichtkrull et al., 2018). For other implementations of the message function, the r-specialised version is simply obtained by having a separate set of learnt parameters per relation type r. For our three example GNNs, the r-specialised

[7]BT is predetermined using RDKit, `https://www.rdkit.org/` (BSD 3-Clause license), and Antechamber (Wang et al., 2001) (GPL-3 license), for molecules with Canonical SMILES representation and for others, respectively. BT is provided as an input and it is not determined by the GNN.

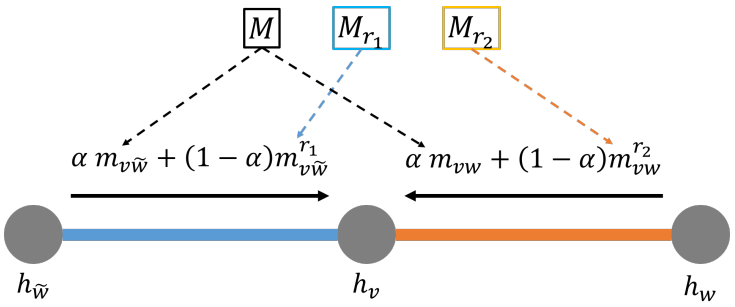


Figure 2: Illustration of the specialised message production method for specialising interactions of nodes based on edge type. In this example, two different edge types (blue and orange) are supported by their respective functions M_{r_1} and M_{r_2} for message production, in addition to the original generic message function M.

functions are denoted M_r (MPNN, see Eq. 1, and DimeNet++, see Eq. 7), R^l_r (SchNet, see Eq. 3), or $\bar{M}_r$ (DimeNet++, see Eq. 5) and become respectively:

$$\mathbf{m}_v^{t+1} = \alpha \sum_{w \in \mathcal{N}_v} M\left(\mathbf{h}_w^t, \bar{\mathbf{x}}_{vw}\right) + (1-\alpha) \sum_{r \in \mathcal{R}} \sum_{w \in \mathcal{N}_v^r} M_r\left(\mathbf{h}_w^t, \bar{\mathbf{x}}_{vw}\right) \tag{9}$$

$$\mathbf{m}_v^{l+1} = \alpha \sum_{w \in \mathcal{N}_v} \left(\mathbf{W}^l \mathbf{h}_w^l + \mathbf{b}^l\right) \circ R^l\left(d_{vw}\right) + (1-\alpha) \sum_{r \in \mathcal{R}} \sum_{w \in \mathcal{N}_v^r} \left(\mathbf{W}_r^l \mathbf{h}_w^l + \mathbf{b}_r^l\right) \circ R_r^l\left(d_{vw}\right) \tag{10}$$

$$\begin{cases} \bar{\mathbf{m}}_{vw}^{l+1} = \alpha \sum_{k \in \mathcal{N}_w \setminus \{v\}} \bar{M}^l\left(\bar{\mathbf{h}}_{wk}^l, \dots\right) + (1-\alpha) \sum_{r \in \mathcal{R}} \sum_{k \in \mathcal{N}_w^r \setminus \{v\}} \bar{M}_r^l\left(\bar{\mathbf{h}}_{wk}^l, \dots\right) \\ \mathbf{m}_v^{l+1} = \alpha \sum_{w \in \mathcal{N}_v} M^l\left(\bar{\mathbf{h}}_{vw}^{l+1}, d_{vw}\right) + (1-\alpha) \sum_{r \in \mathcal{R}} \sum_{w \in \mathcal{N}_v^r} M_r^l\left(\bar{\mathbf{h}}_{vw}^{l+1}, d_{vw}\right) \end{cases} \tag{11}$$

with α modulating the strength of the relation-specialised message production with regards to the original generic message production. $\mathcal{R}$ is the set of relation types. In our application scenario, it is the set of bond types, that may include a 'no bond' element to consider a fully connected graph as in SchNet and DimeNet++. $\mathcal{N}_v^r$ is the set of neighbour nodes that share a relation of type r with node v.

3.1.2. Specialised state update

Specialised interaction may also use generic messages $\mathbf{m}_{vw}$, but handling them in a relation-specialised manner when updating node states. We explore the following different implementations:

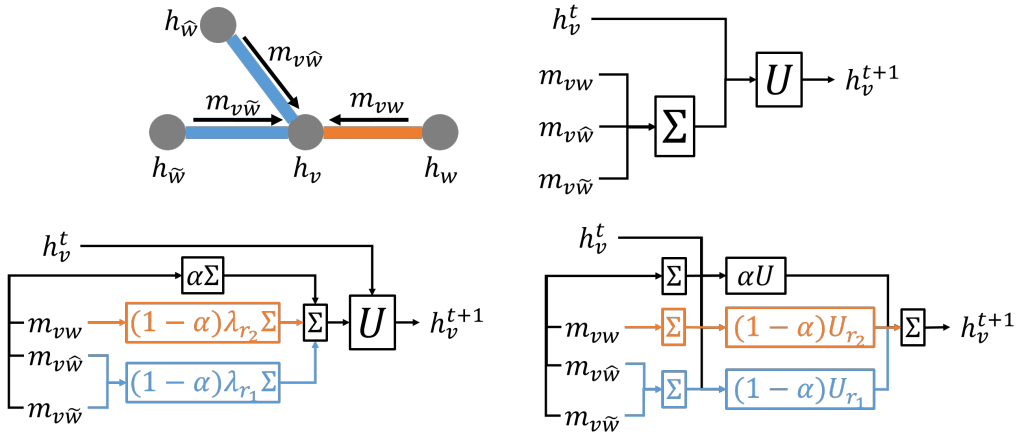


Figure 3: Illustration of the specialised state update method for specialising interactions of nodes based on edge type. In this examples, there are two different edge types (blue and orange). Top: classical state update process. Bottom: proposed specialised state update with (left) specialised weighting of the messages, and (right) specialised update functions.

Specialised weighting of the messages. Messages coming from nodes of different relation types are weighted by a relation-specialised and learnable (scalar or vector) weight λ_r:

$$\mathbf{m}_v = \alpha \sum_{w \in \mathcal{N}_v} \mathbf{m}_{vw} + (1-\alpha) \sum_{r \in \mathcal{R}} \lambda_r \sum_{w \in \mathcal{N}_v^r} \mathbf{m}_{vw} \tag{12}$$

This is illustrated on the bottom left of Fig. 3. In the specific case of a perceptron-based message function M, this approach is equivalent to the basis-decomposition regularisation proposed in (Schlichtkrull et al., 2018) as a mean to reduce the number of learnable parameters associated to specialised message production.

When there are two steps of message passing, as in DimeNet and DimeNet++, this principle may be applied at both steps, with either a unique or two different weights λ_r. In DimeNet++, vector weights need to be different in each step due to the messages having different sizes (Klicpera et al., 2020a):

$$\begin{cases} \bar{\mathbf{m}}_{vw} = \alpha \sum_{k \in \mathcal{N}_w \setminus \{v\}} \bar{\mathbf{m}}_{vwk} + (1-\alpha) \sum_{r \in \mathcal{R}} \lambda_r^1 \sum_{k \in \mathcal{N}_w^r \setminus \{v\}} \bar{\mathbf{m}}_{vwk} \\ \mathbf{m}_v = \alpha \sum_{w \in \mathcal{N}_v} \mathbf{m}_{vw} + (1-\alpha) \sum_{r \in \mathcal{R}} \lambda_r^2 \sum_{w \in \mathcal{N}_v^r} \mathbf{m}_{vw} \end{cases} \tag{13}$$

Specialised update functions. This approach implements separate relation-specialised update functions and sums their contributions, as illustrated on

the bottom right of Fig. 3. As for M_r, r-specialised update functions U_r of any type (e.g. perceptron, GRU, etc.) may be obtained through specialised sets of learnt parameters. State update for MPNN, SchNet, and DimeNet++ become respectively:

$$\mathbf{h}_v^{t+1} = \alpha U\left(\mathbf{h}_v^t, \mathbf{m}_v^{t+1}\right) + (1-\alpha)\sum_{r\in\mathcal{R}} U_r\left(\mathbf{h}_v^t, \sum_{w\in\mathcal{N}_v^r} \mathbf{m}_{vw}^{t+1}\right) \tag{14}$$

$$\mathbf{h}_v^{l+1} = \mathbf{h}_v^l + \alpha V^l\left(\mathbf{m}_v^{l+1}\right) + (1-\alpha)\sum_{r\in\mathcal{R}} V_r^l\left(\sum_{w\in\mathcal{N}_v^r} \mathbf{m}_{vw}^{l+1}\right) \tag{15}$$

$$\bar{\mathbf{h}}_{vw}^{l+1} = \alpha \bar{U}^l\left(\bar{\mathbf{h}}_{vw}^l, \bar{\mathbf{m}}_{vw}^{l+1}\right) + (1-\alpha)\sum_{r\in\mathcal{R}} \bar{U}_r^l\left(\bar{\mathbf{h}}_{vw}^l, \sum_{k\in\mathcal{N}_w^r\setminus\{v\}} \bar{\mathbf{m}}_{vwk}^{l+1}\right) \tag{16}$$

In DimeNet++, $U(\mathbf{m}_v)$ is the identity and is therefore not specialised.

In MPNN, the update function is implemented by a GRU unit, while in SchNet it is based on a dense layer, and in DimeNet++ $\bar{U}$ combines fully connected and residual operations. A straightforward implementation of the (additional) specialised update functions is to use the same type of unit (e.g. GRU for MPNN). Additionally, when augmenting MPNN (Eq. 14), we experiment with two levels of weight sharing when specialising U_r in order to limit the amount of added model complexity. This results in the following three implementations. A similar weight sharing strategy may be employed for other types of update function.

MPNN Implementation 1) We use separate GRU cells, one per relation type, to implement the different U_r.

MPNN Implementation 2) We use a single GRU cell, so that all relation channels share the GRU's internal state. The GRU's $\mathbf{W}_z$, $\mathbf{W}_r$ and $\mathbf{W}_h$ weight matrices contain weights that are specialised for the different relation types. In practice, this amounts to concatenating the different messages from each relation type $\left[\sum_{w\in\mathcal{N}_v^1} \mathbf{m}_{vw} \ldots \sum_{w\in\mathcal{N}_v^{|\mathcal{R}|}} \mathbf{m}_{vw}\right]^T$ before providing them to the GRU cell.

MPNN Implementation 3) We also use a single GRU cell with concatenated messages from different relation types, but we reduce the number of free GRU parameters in $\mathbf{W}_z$, $\mathbf{W}_r$ and $\mathbf{W}_h$ by sharing them between relation channels: $\mathbf{W}_{z/r/h} = \left[\mathbf{Q}_{z/r/h} \dots \mathbf{Q}_{z/r/h}\right]^T$ with $\mathbf{Q}_z$, $\mathbf{Q}_r$ and $\mathbf{Q}_h$ being matrices of weights. This achieves a drastic reduction in the number of learnable parameters, at the cost of a lower flexibility since all relation types are now handled in the same way (but still differently than the general interaction).

3.2. Relating learnt features to relevant physical quantities

Multi-task learning (MTL) may be seen as a way to introduce an inductive bias, where one or several auxiliary tasks further constrain the model and its learnt features (see e.g. (Ruder, 2017; Zhang and Yang, 2018) for a review, and Fig. 1 for illustration). We use this paradigm to encourage the GNN's node states to relate more to relevant (physical) properties for our problem. Drawing motivation from the fact that BT is characterised by the valence property of atoms, we hypothesise that a more physically relevant node state may better support BT-specialised interactions. We explore the potential of MTL for better relating learnt features to physical parameters that are relevant to a final task, and the effect on this task. Our exploration encompasses different GNN architectures and auxiliary tasks. We also evaluate separately the effects of individual tasks.

We experiment with the auxiliary estimation of three different low- and high-level system-wise properties: 1) The number of atoms of each type present in the system. This composition is directly relevant to the value of potential energy, therefore it may be beneficial to draw the attention of the DNN on composition. 2) The number of orbitals associated with each atom type. In addition to being physically relevant to the definition of potential energy, this is also particularly relevant to determine the BT between two atoms, in support for specialised interaction. 3) A probability distribution for the scaling to the stable geometry, estimated as a Gaussian function as in (Timoshenko et al., 2018). This quantity is less directly related to the physical properties of the chemical system, while still being related to potential energy. A chemical system at equilibrium being at its minimum of potential energy, the task of estimating how far the system is to equilibrium (i.e. to minimum potential energy) is an interesting task that considers the whole chemical system and how it relates to its potential energy. In future works, we may investigate auxiliary quantities linked to the physics of individual nodes/atoms, such as atomic forces or the medium-level descriptor

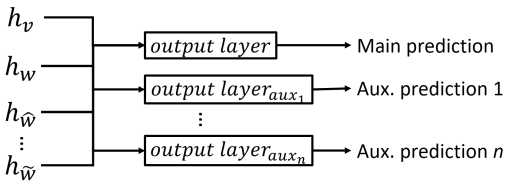


Figure 4: Illustration of multi-task learning for relating learnt features to relevant physical quantities.

atom-centred symmetry functions used in (Liu et al., 2021b). In other case studies, a purpose-designed set of relevant properties would need to be chosen using knowledge of the problem. While we focus on properties that are relevant to the whole graph, node-wise properties may also be considered in the future.

In practice, for each auxiliary estimation, a new output layer is added and a mean-squared error loss term is minimised during training, as illustrated in Fig. 4. This is identical to the energy output estimation, and the type of output layer depends on the GNN architecture (see Section 2.1 for details on the three GNNs used in our experiments). Each auxiliary loss term (three in our case study) is weighted by a β hyper-parameter. In our experiments we set β to 0.3 while the original potential energy loss keeps a weight of 1 so that the three combined auxiliary tasks weight on par with the main task in the total sum:

$$L_{total} = L_{MSE}(y, \hat{y}) + \beta \sum_{a \in \mathcal{A}} L_{MSE}(a, \hat{a}) \tag{17}$$

with y and $\hat{y}$ the ground-truth and predicted potential energies, and $\mathcal{A}$ the set of auxiliary quantities a being used. While we found that β values around 0.3 work well in our experiments, a more thorough parameter analysis would be recommended for any new usage of our method, as the optimal value may vary with the GNN architecture and the main and auxiliary tasks.

4. Experiments

We evaluate the impact of our knowledge integration on the three GNNs of MPNN, SchNet, and DimeNet++. We modify their respective codes[4,5,6] to introduce our augmentations, and we make the augmented versions publicly

available[8]. They are the bases to current state-of-the-art (SoTA) models for estimating potential energies e.g. (Liu et al., 2021b; Schütt et al., 2021; Satorras et al., 2021; Unke and Meuwly, 2019; Batzner et al., 2022; Gasteiger et al., 2021a,b; Choudhary and DeCost, 2021; Thürlemann and Riniker, 2023; Zitnick et al., 2022; Kaba and Ravanbakhsh, 2022), thus our methods may also be applied to current and future SoTA, likely with similar results. In addition, the fact that these three architectures are very different suggests that similar improvements may be obtained on other GNN types too.

The models are used in different conditions than in their original papers. In (Gilmer et al., 2017), MPNN was only evaluated on stable configurations of molecules from the QM9 dataset (Blum and Reymond, 2009; Montavon et al., 2013), that contains a diverse set of 134k molecules using 5 atom types. We further evaluate it on new datasets of OoE molecules and crystals (Section 4.1). SchNet was evaluated on OoE molecules in (Schütt et al., 2017a), however that was either on single molecules at a time (from the MD17 dataset (Chmiela et al., 2017; Schütt et al., 2017b; Chmiela et al., 2018) of 8 small organic molecules with independent perturbations of their atoms' positions) or on isomers (i.e. molecules of same size and atomic composition, from the ISO17 dataset (Schütt et al., 2017a,b; Ramakrishnan et al., 2014) of 129 isomers of $C_7O_2H_{10}$). We apply SchNet to our harder scenario of jointly modelling multiple molecules and crystals of various sizes and atomic compositions. DimeNet++ was evaluated both on QM9 and on small OoE organic molecules of the COLL dataset (Klicpera et al., 2020a). Our OoE perturbations differ from those of COLL, and our crystal datasets contain significantly more different system sizes. We also apply all three GNNs to unseen sizes and compositions.

A comparison of MPNN and E-MPNN is provided in the sup. materials, where MPNN does as well as E-MPNN on one dataset, and outperforms it on all others. Thus, we only work with MPNN. We do not use MPNN-BTF because we aim to demonstrate the effectiveness of accounting for physics knowledge in the design of the GNN, rather than merely in the input features. Results of the augmented MPNN-BTF are still provided in the sup. materials.

We use the base models' default hyper-parameters for both original and augmented versions. For (base and augmented) MPNN, we experimented

[8] `https://github.com/jaypmorgan/DGNNs`

with different numbers of iterations (see sup. materials). MPNN's default of 3 worked best. New hyper-parameters α are learnable and optimised during training of the DNN starting from a default initialisation of 0.5. These parameters modulate the contribution of the specialised interactions based on edge type, from 0 (only specialised interactions) to 1 (only general interaction). In our experiments, we found that the optimised value was consistently around 0.6 for MPNN and SchNet, which indicates that a significant contribution from the specialised interactions complements well the general interactions in support to a better modelling by the GNN. For DimeNet++, an α value close to 0 obtained the best results, indicating that the specialised interactions are particularly useful in this architecture. The sup. materials and Table 1 provide a list of the number of original and additional learnable parameters brought by our methods. Early stopping at 50 stable epochs ensures that each DNN trains for a suitable time.

4.1. Datasets

Extended QM9 (E-QM9) includes diverse sizes (i.e. number of atoms) and compositions of OoE molecules, through extending a subset of QM9 with OoE versions of 10k of its molecules.

Periodic crystals (PC) allows learning regular bonding patterns that arise in periodic structures by repeating the base crystal lattice. We use the Face-Centred Cubic (fcc) Bravais lattice (Fig. 5 left) for aluminium (Al) and copper (Cu) crystals. Unlike in Kaba and Ravanbakhsh (2022), we only consider the crystal lattice of an ideal crystal, assuming that no local symmetry breaking may happen.

Crystal Growth (CG) contains growing crystals of increasing size and complexity. Starting from a basic fcc crystal seed of 14 atoms (Fig. 5 centre), new systems are generated by iteratively placing atoms at a random location on the surface of the growing crystal following its lattice pattern (Fig. 5 right), with sizes ranging from 15 to 114 atoms. We use 20 random seeds for each atom type, thus creating 40 varied Al and Cu crystal growths and 4,000 stable systems. As a result, for a given crystal size and composition (atom type), there are 20 samples with differently located atoms. CG enables experimenting with large scale atomic interactions in non-regular systems, and enables evaluation of an ML method's ability to learn how each atom contributes to the final potential energy.

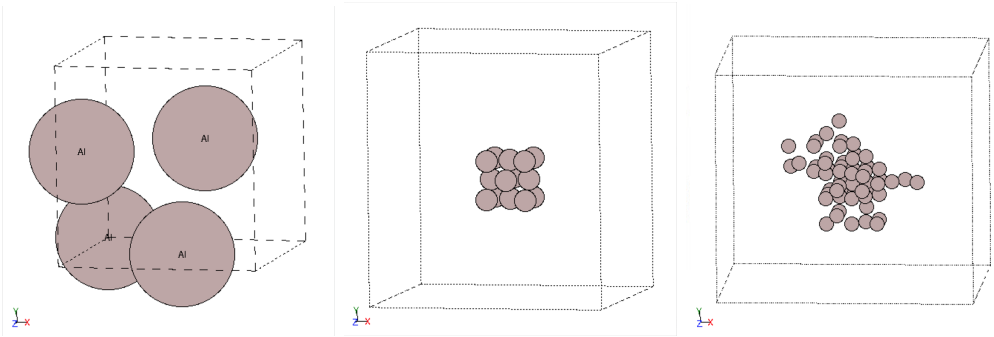


Figure 5: FCC Bravais crystal lattice (left) and growing crystal structures (centre: seed, right: intermediate).

In all datasets, OoE systems are obtained by compressing/dilating all interatomic distances (i.e. isometrically) at regular intervals within 90-150% of stable geometry, which we refer to as *'scaling'*. In other words, scaling is applied to the coordinates of all atoms within the system: $\hat{\mathbf{x}} = \lambda \mathbf{x}$. At each geometry, the ground-truth potential energy is calculated using CP2K[9]'s DFT.

We generate two subsets of CG: *Stable Crystal Growth* (SCG) with only stable geometries, and *Unstable Crystal Growth* (UCG) with scaling. Both contain Al and Cu crystals. SCG aims to evaluate an ML model at varying system sizes (i.e. number of atoms) without the added complexity of varying scale. For UCG, we select 5 of the random crystal growth seeds for each atom type to consider 1,000 stable geometries to be scaled. Statistics on all datasets are in the sup. materials. The datasets are publicly available at `https://doi.org/10.6084/m9.figshare.12360620`.

The datasets are split between training, testing, and validation in proportions appropriate for the data complexity as detailed in the sup. materials.

4.2. Evaluations of the individual domain knowledge integration strategies

We evaluate the individual effects of the different knowledge integration methods in comparison to non-augmented base GNNs, on E-QM9. Table 1 reports absolute error (AE) and relative error (RE = $\frac{|\hat{y}-y|}{|y|}$) between true y and predicted $\hat{y}$ energies in a.u. unit. Squared errors are provided in the sup. materials. We also report the Distance to Stable Geometry (DSG) which is

[9] `https://www.cp2k.org/` (under the GPL-2.0 license)

Table 1: Impact of each domain knowledge integration strategy on MPNN (top) and SchNet (bottom) for E-QM9. Results are in the format: mean(std). AE is provided as $1e^{-1}$, RE as $1e{-}3$ and DSG as $1e{-}2$. The specialised interaction methods that optimise at best energy and geometry are highlighted in bold for each GNN. We also show in the last column the added number of parameters in percentage of original model size.

	Strategy			AE	RE	DSG	+% parameters
1			**MPNN base model** with no BT information	2.42(13.18)	2.9(15)	3.4(7.4)	–
2			MPNN-BTF	0.91(4.76)	1.2(5)	3.9(5.6)	–
3	r-specialised interactions	Specialised message Eq. 9 ($\alpha = 0.624$)		2.72(12.93)	3.4(14)	5.1(8.0)	398.20
4		Specialised state updates	Eq. 12 scalar λ_r ($\alpha = 0.628$)	1.22(4.31)	1.6(5)	3.2(5.7)	$5.28e^{-4}$
5			Eq. 12 vector λ_r ($\alpha = 0.627$)	1.05(5.02)	1.3(5)	5.0(4.9)	$3.85e^{-2}$
6			Eq. 14 impl. 1) ($\alpha = 0.615$)	1.06(2.53)	1.4(3)	2.3(4.8)	17.11
7			Eq. 14 impl. 2) ($\alpha = 0.635$)	**0.73(1.70)**	**1.0(2)**	**3.0(4.8)**	10.18
8			Eq. 14 impl. 3) ($\alpha = 0.618$)	1.31(4.36)	1.7(5)	2.5(5.5)	3.42
9	Auxiliary estimates of		# atoms of each type	1.71(9.86)	2.1(11)	3.1(4.6)	3.43
10			# orbitals	1.95(5.45)	2.5(6)	3.3(4.6)	3.43
11			distance to stable geometry	1.19(6.98)	1.5(8)	3.3(4.6)	3.40
12			**SchNet base model**	0.38(0.37)	0.5(0.5)	2.0(3.1)	–
13	r-specialised interactions	Specialised message Eq. 10 ($\alpha = 0.734$)		0.36(0.35)	0.5(0.5)	2.2(3.2)	220.43
14		Specialised state updates	Eq. 12 scalar λ_r ($\alpha = 0.529$)	**0.20(0.18)**	**0.3(0.2)**	**2.5(3.2)**	$8.01e^{-3}$
15			Eq. 12 vector λ_r ($\alpha = 0.570$)	0.24(0.28)	0.3(0.3)	3.4(3.4)	0.51
16			Eq. 15 ($\alpha = 0.727$)	0.31(0.32)	0.4(0.4)	1.5(2.8)	220.43
17	Auxiliary estimates of		# atoms of each type	0.38(0.36)	0.5(0.5)	3.2(3.3)	0.39
18			# orbitals	0.36(0.35)	0.5(0.5)	2.2(3.2)	0.39
19			distance to stable geometry	0.49(0.44)	0.7(0.6)	3.7(3.3)	0.36
20			**DimeNet++ base model**	4.17(3.83)	5.6(5.0)	6.8(5.6)	–
21	r-specialised interactions	Specialised message Eq. 11 ($\alpha = 0.012$)		0.72(0.58)	1.0(0.8)	4.4(3.2)	23.33
22		Specialised state updates	Eq. 13 scalar λ_r ($\alpha = 0.003$)	**0.60(0.49)**	**0.8(0.6)**	**2.7(3.3)**	$2.9e^{-4}$
23			Eq. 13 vector λ_r ($\alpha = 0.011$)	0.69(0.57)	0.9(0.7)	5.5(2.6)	$5.5e^{-2}$
24			Eq. 16 ($\alpha = 0.599$)	4.34(3.96)	5.9(5.2)	5.0(5.4)	55.08
25	Auxiliary estimates of		# atoms of each type	3.05(3.44)	4.1(4.2)	4.1(4.9)	40.46
26			# orbitals	7.61(7.05)	10.3(8.5)	11.2(8.5)	40.46
27			distance to stable geometry	7.44(7.74)	10.1(8.9)	12.1(13.1)	39.70

the absolute difference between the scaling where the predicted energy is minimal (optimised by multiple DNN queries for various scalings), and the ground-truth at-equilibrium scaling (i.e. 100%): DSG $= |\lambda - 1|$. Mean and std are computed over chemical systems, with std indicating the ability to handle a large variety of systems.

Results of (non-augmented) base models are provided in rows 1,12,20 of Table 1. When compared against these base models, and considering only the best performing methods for r-specialised interactions, all integrations of domain knowledge tend to improve energy estimation and/or finding stable geometries. Since all the base and augmented models are trained on the exact same data (no data augmentation), these improvements over base models cannot come from training on an extension of QM9 that contains OoE molecules. Instead, improvements reflect the effect of integrating domain knowledge into the design of the augmented GNNs.

The impact of BT information is the strongest for all three architecture types (rows 2,7,14,22). This confirms Gilmer et al. (2017)'s observation that BT is relevant for estimating energy. Furthermore, accounting for BT in the design of r-specialised interactions in MPNN (row 7) has a stronger positive impact than merely using it as an input edge feature as in (Gilmer et al., 2017) (row 2). This suggests that r-specialised interactions better capture the physics of atomic interactions.

When examining the effects of the different r-specialised interaction methods in parallel to the additional model complexity that they bring (last column of Table 1, see also sup. materials for more numbers of parameters), we notice that the very simple r-specialised weighting of the messages (Eqs. 12 and 13) performs generally well while bringing a very limited number of extra parameters. On the other hand, r-specialised message production (Eqs. 9 and 10) brings a very large number of new parameters for MPNN and SchNet (+220% and +398%), which may explain its lower performance, while the same method for DimeNet++ (Eq. 11) keeps the complexity reasonable (+23%) and obtains a decent performance. The same observation applies to r-specialised update functions for SchNet (Eq. 15, +220%) and DimeNet++ (EQ. 16, +55%). However, in the case of MPNN, r-specialised update functions (Eq. 14) provide a lower number of new parameters (+3% to 17%), and remain within a manageable complexity that is exploited to bring the best performance improvement for this architecture. In fact, the two implementations that bring the highest increase in complexity (+10-17%) provide the best performance for MPNN. Considering that MPNN has

Table 2: Ablation study and effect on handling different atom types in crystals of UCG

	Al		Cu	
Method	AE	DSG	AE	DSG
MPNN	0.643 (0.495)	0.096 (0.135)	6.250 (4.772)	0.214 (0.192)
Augm.-MPNN	0.150 (0.120)	0.031 (0.037)	1.160 (0.878)	0.108 (0.122)
Augm.-MPNN w/o r-spec. interactions	0.180 (0.144)	0.050 (0.036)	0.638 (0.582)	0.071 (0.071)
Augm.-MPNN w/o aux. # atoms	0.179 (0.129)	0.011 (0.026)	0.928 (0.760)	0.098 (0.125)
Augm.-MPNN w/o aux. # orbitals	0.190 (0.141)	0.020 (0.034)	0.906 (0.724)	0.080 (0.073)
Augm.-MPNN w/o aux. DSG	0.177 (0.151)	0.035 (0.038)	0.834 (0.680)	0.082 (0.102)
SchNet	6.357 (3.159)	0.117 (0.047)	5.355 (5.278)	0.066 (0.088)
Augm.-SchNet	0.333 (0.247)	0.069 (0.071)	0.368 (0.259)	0.002 (0.008)
Augm.-SchNet w/o r-spec. interactions	0.444 (0.358)	0.080 (0.073)	1.424 (1.490)	0.034 (0.060)
Augm.-SchNet w/o aux. # atoms	0.395 (0.169)	0.102 (0.025)	0.304 (0.238)	0.001 (0.009)
Augm.-SchNet w/o aux. # orbitals	0.241 (0.146)	0.006 (0.019)	0.286 (0.176)	0.054 (0.031)
Augm.-SchNet w/o aux. DSG	0.296 (0.203)	0.035 (0.049)	0.328 (0.220)	0.000 (0.000)

smaller node states than SchNet and DimeNet++ (73 vs. 128), it is also a possible explanation for MPNN benefiting more from an added model complexity.

The auxiliary estimation of physical properties (rows 9-11) also improves on base MPNN, but less so than BT information. On SchNet, estimating the number of orbitals also provides a slight improvement (row 18), and DimeNet++ benefits from estimating the number of atoms of each type (row 25). These improvements may come from the models implicitly discovering and encoding useful physics representations required for accurate predictions. Indeed, as discussed in Section 3.2, these quantities are chosen based on knowledge of the domain to help focus the GNN's attention on important aspects of the learning problem. In particular, the distance to stable geometry is related to a high level system optimisation and its improved performance may indicate that the GNN's features better capture the dependency of energy on geometry. It is interesting that different architectures benefit differently from different auxiliary estimations. This may be related to their respective ways of handling the relation between the global composition and geometry of the chemical system and its total energy. In future works, we may investigate auxiliary quantities linked to the physics of individual atoms/nodes.

We further explore the effect of the best method for r-specialised interactions (Eq. 14 impl. 2 for MPNN and Eq. 12 with scalar λ_r for SchNet) and of each auxiliary estimation in an ablation study using UCG (results are reported in Table 2). As previously, the two GNN architectures don't benefit

Table 3: Evaluation of the base (left) and augmented (right) GNNs. For each architecture type, best results between base and augmented GNNs are highlighted in bold.

		Base		Augmented	
Model	Dataset	AE	DSG	AE	DSG
MPNN	E-QM9	0.242 (1.318)	0.034 (0.074)	**0.065 (0.161)**	**0.030 (0.045)**
	Periodic Crystals	0.041 (0.032)	**0.073 (0.079)**	**0.039 (0.035)**	0.074 (0.079)
	Stable CG	2.796 (3.797)	–	**2.105 (2.624)**	–
	Unstable CG	2.892 (3.820)	0.344 (0.083)	**0.655 (0.805)**	**0.069 (0.098)**
SchNet	E-QM9	0.038 (0.037)	0.020 (0.031)	**0.026 (0.025)**	**0.016 (0.028)**
	Periodic Crystals	**13.444 (15.984)**	0.054 (0.012)	15.961 (19.424)	**0.039 (0.053)**
	Stable CG	2.021 (1.829)	–	**1.217 (1.586)**	–
	Unstable CG	5.866 (4.377)	0.091 (0.075)	**0.351 (0.254)**	**0.035 (0.060)**
DimeNet++	E-QM9	0.417 (0.383)	0.068 (0.056)	**0.101 (0.095)**	**0.036 (0.034))**
	Periodic Crystals	5.683 (14.656)	0.104 (0.043)	**1.819 (3.062)**	**0.101 (0.038)**
	Stable CG	**0.244 (0.254)**	–	0.249 (0.202)	–
	Unstable CG	27.475 (44.753)	0.990 (0.714)	**22.708 (34.180)**	**0.694 (0.663)**

equally from a same augmentation, with auxiliary estimates being mostly beneficial to MPNN. Furthermore, there are a few occurrences where some augmentations, which did help individually in Table 1, were better removed from fully augmented models. While these results have to be taken carefully given that they are based on different datasets, this may also suggest that 1) the weighting of the auxiliary tasks may need to be optimised, 2) some augmentations may interact and result in a more complex behaviour, and 3) total model complexity may also need to be accounted for to explain these behaviours. Future work is needed to further explore the cause for these varying behaviours and avenues to better balance the contributions of auxiliary estimations for example based on (Long et al., 2017; Cipolla et al., 2018) instead of the fixed $\beta = 0.3$. In addition, since the auxiliary tasks that we consider in this study are related to the global composition and geometry of the chemical system, future study may benefit in also exploring the use and effect of auxiliary tasks that are related to individual atoms. For the rest of the paper, since the combined four augmentations still outperform base models by a significant margin, we use this simple configuration to assess the properties of fully augmented GNNs.

4.3. Evaluation of the fully augmented models on OoE molecule and crystal data

We combine all our domain integration methods (i.e. the best performing method for r-specialised interactions and all auxiliary estimates)

into Augmented-MPNN, Augmented-SchNet, and Augmented-DimeNet++. For r-specialised interactions, we retain Eq. 14 impl. 2 for MPNN, Eq. 12 with scalar λ_r for SchNet, and Eq. 13 also with scalar λ_r for DimeNet++. Given the complexity discussion of the previous section, we will also present some results using the simpler (and still reasonably well performing) Eq. 12 with scalar λ_r for MPNN. Although r-specialised message production and r-specialised state update are not mutually exclusive, their combination will be explored in future work. For CG, we consider system sizes of 15 to 75 atoms instead of the available 114 due to memory restrictions. Results are reported in Table 3 and with more details and metrics in the sup. materials. We further illustrate the performance of the fully augmented GNNs by plotting their energy estimates for all scaled versions of randomly picked systems of E-QM9 in Fig. 6, and of CG in Fig. 7.

We first note that it is generally beneficial to combine r-specialised interactions and auxiliary estimations, with AE and/or DSG results being further improved in Table 3 as compared to individual augmentations in Table 1. The fact that some auxiliary estimations (number of orbitals) are chosen to support r-specialised interactions may be a reason for these further improvements. An exception is DimeNet++ for which the best results (on E-QM9) were obtained with r-specialised interactions only (no auxiliary estimations). For this architecture, the auxiliary estimation of number of orbitals was hindering results in Table 1. Nevertheless, even if the full augmentation is not the best performing for DimeNet++, it still improves on the base model. Therefore, as discussed at the end of Section 4.2, we do not optimise the contributions of auxiliary estimations and we present all results with the default $\beta = 0.3$ for consistency.

All three GNN types benefit from our domain knowledge integration, with overall more accurate energy estimates and better performance at finding stable geometries (DSG). The latter point is also illustrated in the example energy plots of Figs. 6 and 7 where the augmented GNNs produce curves with a correct minimum while base GNNs sometimes had physically-unrealistic curves with incorrect minimum. The improvement is particularly strong on CG, hinting that our augmented models can generalise better to new geometries, that are a particularity of this dataset. Consequently, on Periodic Crystals, which contains only two mono-atomic crystals (with various scalings), we expect to see further improvements in future works by considering structures outside of the fcc lattice and a wider variety of atom types.

The absolute errors on the auxiliary tasks are provided in the sup. mate-

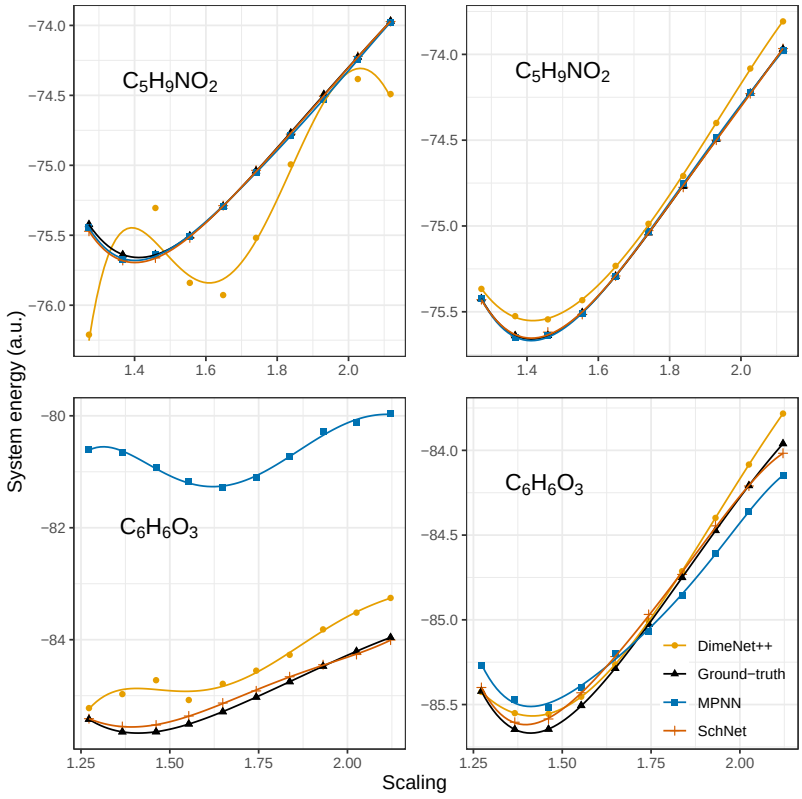


Figure 6: Energy estimations at different scalings of two examples of molecules from E-QM9 for base (left) and augmented (right) GNNs.

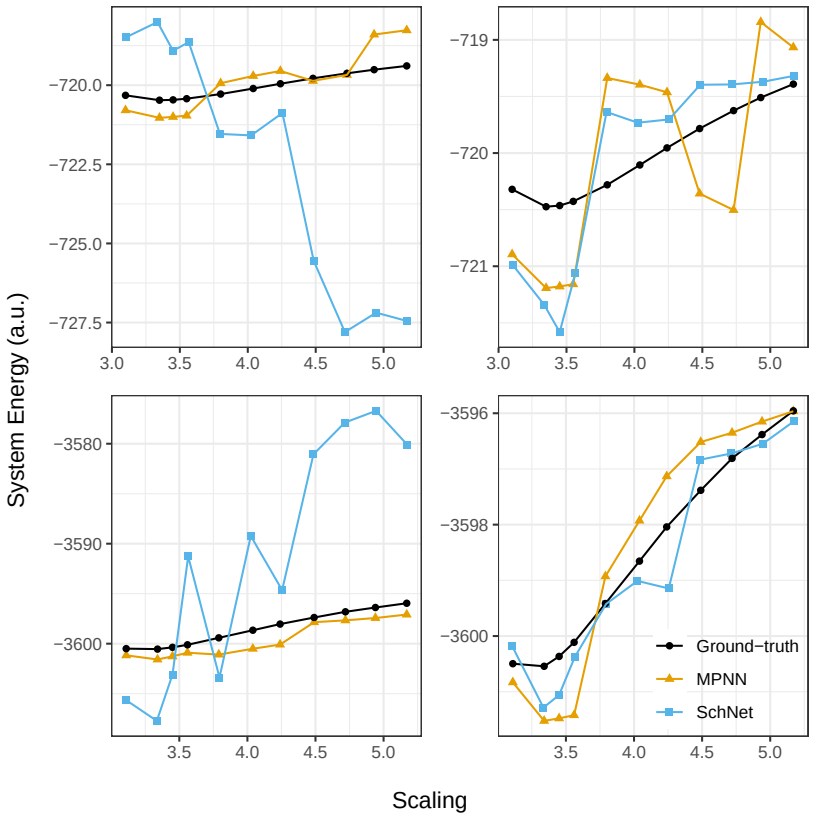


Figure 7: Example energy estimations at different scalings for a small (15 atoms, top) and large (75 atoms, bottom) crystalline system of the UCG dataset. Left: base models, right: augmented models. As DimeNet++ performs poorly on UCG, it is removed from this figure to preserve the legibility of the plot (axis scaling).

rials. While these tasks are not the end goals for the GNNs and have no other use than guiding the learnt representations to be more physically relevant, we verify that these predictions have an accuracy that matches the accuracy level of the potential energy estimations (Tables 1 and 3).

4.4. Interpretation of the nodes' hidden states

We examine the node states through visualising the contribution of each node to the final estimate. In MPNN, perceptrons combine the states of a node at all timesteps. In SchNet, the last atom-wise layer has output size of 1. In DimeNet++, the output size is also 1 for each node, and outputs for all layers are summed. In all three networks, these reduced values are summed across nodes to obtain the energy estimate. After normalising across all nodes, they may be considered as the *atoms' contributions* to the energy estimate.

After training the GNNs on UCG, we consider an atom that is newly added to a (stable) crystal seed of 14 atoms. When scaling the distance of this single atom to the rest of the crystal, we examine its contribution and that of the rest of the (static) atoms. This scenario differs from the training scenario of whole system scaling, but Augm.-SchNet in Fig. 8 still produces plausible energy estimates with correct minimum (although some offsets in energy sometimes happen and will be investigated in future work).

As shown in the sup. materials, base GNNs were not successful, including SchNet and DimeNet++ although this 'freely' moving atom setup is closer to their original experimental setups (Schütt et al., 2017a; Klicpera et al., 2020a), possibly due to not being able to learn on scaled systems and many sizes at once. Thus, our augmentations increased the generalisation ability of SchNet through a better capture of the atomic interactions.

Augm.-MPNN could not obtain a plausible energy curve with correctly located minimum either (see sup. materials). We tested both the best performing specialised interaction method of Eq. 14, that introduces a higher model complexity, and the simpler specialised interaction method of Eq. 12 with scalar λ_r, used in Augm.-SchNet. Neither methods succeeded in handling the single moving atom, which indicates that the lower generalisation to new scenarios does not come from the higher complexity introduced by Eq. 14, but rather would be inherent to the MPNN model, and could not be overcome by our augmentations as in SchNet. DimeNet++ does very poorly on UCG. While Augm.-DimeNet++ does better, its performance still

remains poor. It is therefore not surprising that it fails to handle our single moving atom scenario after training on UCG (see sup. materials).

In Augm.-SchNet, the moving atom's contribution (left red curve) is minimal at its stable location (i.e. minimum of energy, black curve in Fig. 8). At the same time, static atoms (right red curve) contribute maximally, in line with the crystal being at its minimum energy driven by its regular structure. As the atom is moved closer or pulled away, its contribution increases simultaneously with energy. At the same time, the relative contributions of static atoms decrease to give way to the perturbation of the moving atom. This strongly suggests that the GNN learnt to pay attention to the location of individual atoms, although it was trained on systems that are isometrically scaled.

The same experiment, when performed after training on SCG, is not as successful, with Augm.-SchNet producing an implausible energy curve with no minimum. This indicates that the different geometries provided by the 20 varied locations of atoms for each crystal size were not enough to learn the importance of fine location of individual atoms. When training on UCG, the (global) scaling was complementary to the varied occupations of atom sites in learning the importance of precise location of individual atoms. It is worth noting that the same training did not allow original SchNet to learn this principle and to produce plausible energy curves. Thus, it is the combination of our augmentations and of a sufficiently diverse training data that achieved this result.

4.5. Generalisation to larger graphs

Fig. 9 presents energy AE on UCG as a function of system size for base and augmented GNNs when trained on all system sizes (up to 75 atoms) or on small systems only (up to 25 atoms), keeping the testing set equal. For Augm.-SchNet, the better accuracy from the augmentations maintains the AE in the best achieved range when training on small systems only. For Augm.-MPNN, when training on small systems only, the range of AE is also improved by a factor $\sim$100 compared to base MPNN. An improvement is also observed for Augm.-DimeNet++, with a range of AE reduced from [0,500] to [0,300], in spite of the aforementioned difficulty of DimeNet++ with the UCG dataset.

Furthermore, for Augm.-MPNN, accuracy correlates with system size only marginally stronger than when training on all sizes (Pearson coef. 0.200 against 0.193). This is an improvement from base MPNN, where Pearson

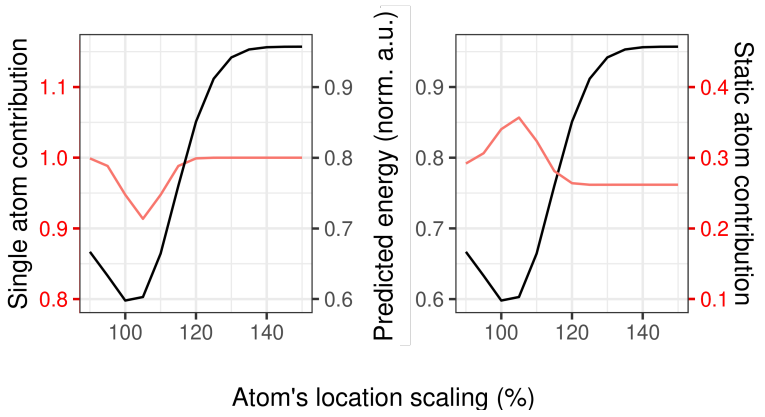


Figure 8: Contribution of a moving atom (left red curve) and mean contribution of static atoms (right red curve) toward the energy estimate of a crystal (black curves) by Augmented-SchNet. Note that the curves cover different ranges of values, with the red axis labels being for their corresponding red curves (contributions), and the black axis labels being for the black curves (energy estimate).

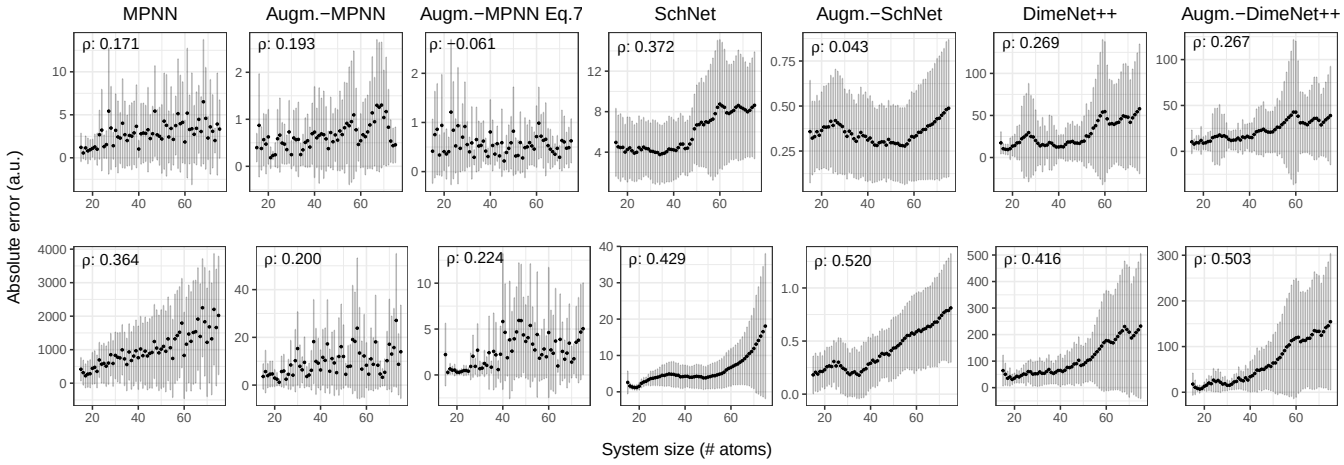


Figure 9: Impact of system size on accuracy when trained on systems of up to 75 (top) and 25 (bottom) atoms from UCG.

correlation increases from 0.171 to 0.364. Therefore, our augmentations allow MPNN for learning of some basic principles about the atomic interactions that are applicable to larger, unfamiliar systems. When using the specialised interaction method of Eq. 12 with scalar λ_r, Pearson correlation is similar at 0.224, and the range of AE is improved by a factor $\sim$400. We explain this better AE by the different complexities of Eqs. 14 and 12, which make the model more or less robust to smaller training sets.

4.6. Generalisation to smaller training sets

We test whether the domain knowledge integration brings a higher robustness to small training sets through a better generalisation, by reducing progressively the size of the training set (keeping the testing set equal). Figs. 10, 11, and 12 present results respectively on E-QM9 when decreasing the variety of molecules and when decreasing the variety of scalings, and on SCG. For E-QM9, we vary the size of the training sets by intervals of 10%, followed by intervals of 1%[10] to test even smaller training sets. Since E-QM9 comprises OoE molecules, in a first experiment the n^{th} training set includes all scalings of n% of the (original) training molecules. In a second experiment, the n^{th} training set includes n% of the scalings of all molecules (with a random selection of scalings for each molecule). For SCG, we vary the number of crystals in the SCG subset.

When considering extremely small training sets, there are a few cases where the augmented GNNs obtain worse results than base GNNs. This might be explained by the increased model complexity due to additional learnt parameters and auxiliary tasks, requiring a minimal amount of data to train effectively.

More generally, on E-QM9, while augmented GNNs tend to outperform their base counterparts, they are not more robust to decreasing the number of molecules to be trained on as performances decrease with similar rates (Fig. 10). Thus, the augmentations could not compensate for the loss of composition diversity.

However, both augmented GNNs are quite stable to varying numbers of scalings, and more so than their base counterparts (Fig. 11). This indicates that they are better able to account for the effect of varying geometries.

The most dramatic improvement is on SCG, where both augmented GNNs

[10] when numbers of samples were sufficient

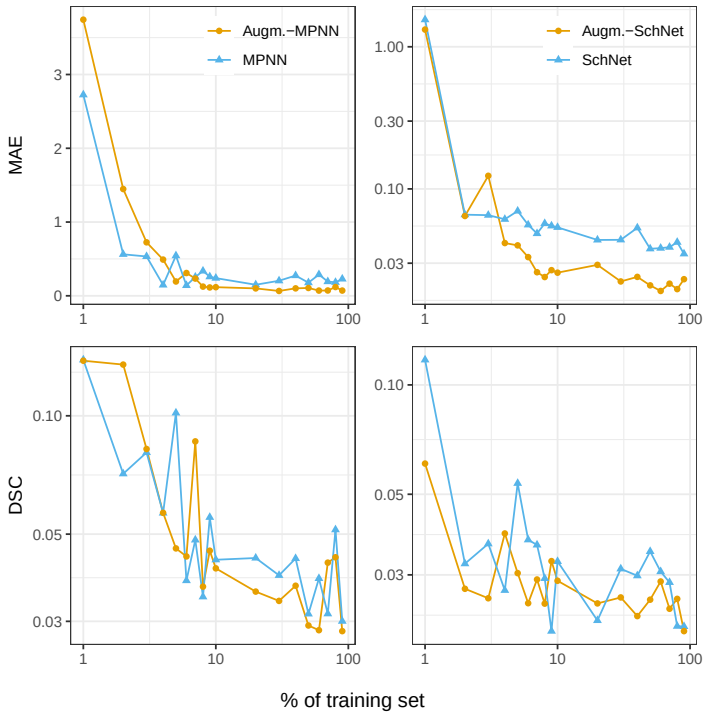


Figure 10: Robustness of base and augmented GNNs to smaller training sets (i.e. subsets of molecules) for E-QM9. Top: mean AE, bottom: mean DSG. Left: MPNN, right: SchNet. Blue: base, orange: augmented.

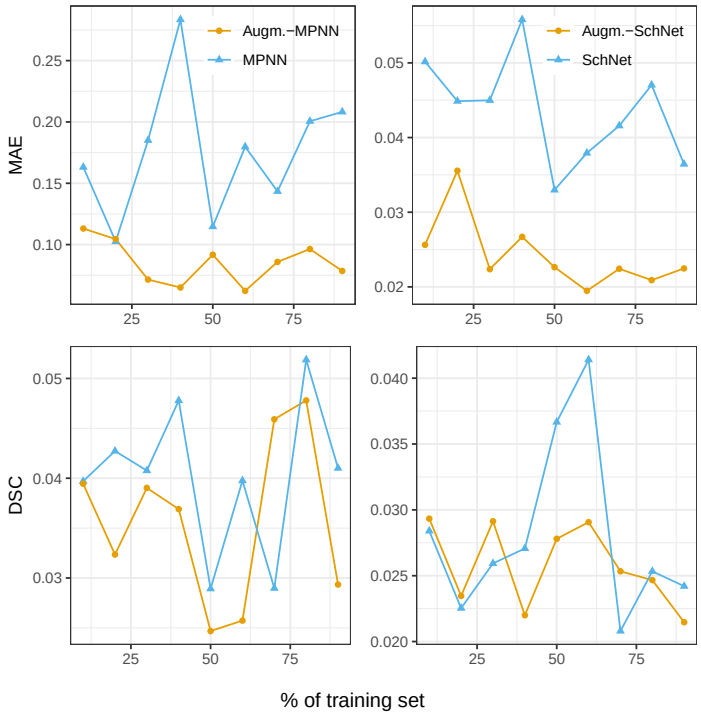


Figure 11: Robustness of base and augmented GNNs to smaller training sets (i.e. subsets of scalings) for E-QM9. Top: mean AE, bottom: mean DSG. Left: MPNN, right: SchNet. Blue: base, orange: augmented.

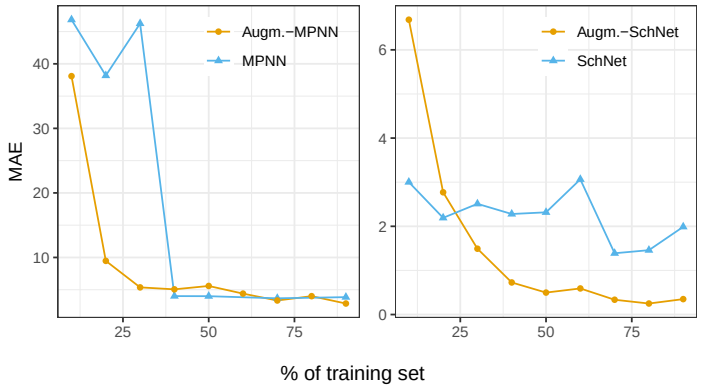


Figure 12: Robustness of base and augmented GNNs to smaller training sets for SCG. Left: MPNN, right: SchNet. Blue: base, orange: augmented.

see a more stable, then a later rise in mean AE when decreasing the dataset size and the associated diversity in crystal geometries (Fig. 12). We conclude that the augmentations provided insights on geometry and scaling that did not need to be learnt from data, hence reducing the need for exhaustively covering these aspects in training samples.

5. Conclusion

We integrate domain knowledge into GNNs to improve accuracy and generalisation, with two proposed strategies: (1) specialised information flow within the GNN to better account for relation types between nodes, and (2) further relating the learnt representations to the studied phenomenon through MTL. We explore the different means for specialising information flow by acting on either message production or state update, and we provide general formulations that are adapted to different architectures. We demonstrate them on three architectures: MPNN, SchNet, and DimeNet++.

Our proposed domain knowledge integration is tested on a quantum chemistry case study where potential energy of chemical systems (molecules or crystals) is estimated as a function of geometry. For this application, the elements of domain knowledge that we use are specifically related to atomic interaction and underlying physics of potential energy.

The added model complexity varies widely with the augmentation methods, and a too high complexity for a given GNN and application may be detrimental. However, in many cases, the (reasonable) added complexity proved helpful in learning complex and more general concepts. It allowed the GNNs to better learn principles of atomic interactions, with improved handling of unseen geometries in graphs, including unseen sizes of graphs and unseen perturbations of their nodes' locations, and the ability to train on smaller datasets.

Our domain knowledge integration methods add to the toolbox of GNN augmentation strategies. They are generally applicable to non-physics domains, such as economics (Panford-Quainoo et al., 2020), traffic prediction (Diehl et al., 2019), biomedical knowledge discovery from knowledge graphs (Callahan et al., 2020), or predicting relationships for E-commerce recommendation (Liu et al., 2021a), where problems are naturally represented as graphs. Our proposed specialised information flow may enhance GNN analysis where it is known that nodes share different relation types. While in our case study, the relation types are chemical BTs, in other applications

they may be of different and very diverse nature, such as priority relations between traffic participants, semantically and biologically meaningful relationships between different types of biomedical data, or political or national affinities in country trade relationships, for example. These new applications of our augmentations would require an expert-based identification of such known relations and/or of auxiliary tasks of interest, following the example of our case study.

From the viewpoint of our case study, our augmentations may enhance current and future SoTA estimating potential energies and stable geometries of chemical systems. Indeed, current SoTA are based on the GNNs used in our case study, and we expect similar results may be obtained from augmenting their more recent variations. Furthermore, our results on three very different GNN architectures suggest that similar improvements may also be obtained on other GNN types, including future GNN-based SoTAs. Other aspects of chemical systems may also be considered, such as drug discovery (Xiong et al., 2020), nucleation and growth of crystals, mechanical properties of crystals (in physics and engineering), molecular structure (in chemistry), doping (in electronic materials). In any case, it is well-known that DNNs cannot offer guarantees on results and suffer from well documented problems such as adversarial examples. Therefore, although our methods improve their accuracy and robustness, numerical simulations remain necessary for a final verification of an optimal geometry found through a DNN's predictions.

References

Batzner, S., Musaelian, A., Sun, L., Geiger, M., Mailoa, J., Kornbluth, M., Molinari, N., Smidt, T., Kozinsky, B., 2022. SE(3)-Equivariant graph neural networks for data-efficient and accurate interatomic potentials. Nature Communications 13.

Blum, L., Reymond, J., 2009. 970 Million Druglike Small Molecules for Virtual Screening in the Chemical Universe Database {GDB-13}. J. Am. Chem. Soc. 131, 8732.

Callahan, T., Tripodi, I., Pielke-Lombardo, H., Hunter, L., 2020. Knowledge-based biomedical data science. Annu Rev Biomed Data Sci. 3, 23–41.

Chen, D., Liu, R., Hu, Q., Ding, S.X., 2021. Interaction-aware graph neural

networks for fault diagnosis of complex industrial processes. IEEE Transactions on Neural Networks and Learning Systems , 1–14.

Chmiela, S., Sauceda, H.E., Müller, K.R., Tkatchenko, A., 2018. Towards exact molecular dynamics simulations with machine-learned force fields. Nature Communications 9, 1–12.

Chmiela, S., Tkatchenko, A., Sauceda, H.E., Poltavsky, I., Schütt, K.T., Müller, K.R., 2017. Machine learning of accurate energy-conserving molecular force fields. Science Advances 3.

Choudhary, K., DeCost, B., 2021. Atomistic line graph neural network for improved materials property predictions. npj Comput Mater 7.

Cipolla, R., Gal, Y., Kendall, A., 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics, in: CVPR.

Diehl, F., Brunner, T., Truong Le, M., Knoll, A., 2019. Graph neural networks for modelling traffic participant interaction, in: IEEE Intelligent Vehicles Symposium, pp. 695–701.

Erba, A., Baima, J., Bush, I., Orlando, R., Dovesi, R., Erba, A., 2017. Large-Scale Condensed Matter DFT Simulations: Performance and Capabilities of the CRYSTAL Code. Journal of Chemical Theory and Computation 13, 5019–5027.

Gasteiger, J., Becker, F., Günnemann, S., 2021a. Gemnet: Universal directional graph neural networks for molecules, in: NeurIPS.

Gasteiger, J., Yeshwanth, C., Günnemann, S., 2021b. Directional message passing on molecular graphs via synthetic coordinates, in: NeurIPS.

Gilmer, J., Schoenholz, S., Riley, P., Vinyals, O., Dahl, G., 2017. Neural Message Passing for Quantum Chemistry, in: International Conference on Machine Learning, pp. 1263–1272.

Glavatskikh, M., Leguy, J., Hunault, G., Cauchy, T., Da Mota, B., 2019. Dataset's chemical diversity limits the generalizability of machine learning predictions. Journal of Cheminformatics 11.

Jiang, H., Baranger, H., Yang, W., 2003. Density-functional theory simulation of large quantum dots. Physical Review B - Condensed Matter and Materials Physics 68, 1–9.

Kaba, O., Ravanbakhsh, S., 2022. Equivariant Networks for Crystal Structures , in: NeurIPS.

Kawahara, J., Brown, C., Miller, S., Booth, B., Chau, V., Grunau, R., Zwicker, J., Hamarneh, G., 2017. BrainNetCNN: Convolutional neural networks for brain networks; towards predicting neurodevelopment. NeuroImage 146, 1038–1049.

Klicpera, J., Giri, S., Margraf, J., Günnemann, S., 2020a. Fast and uncertainty-aware directional message passing for non-equilibrium molecules, in: ML for Molecules workshop, NeurIPS.

Klicpera, J., Groß, J., Günnemann, S., 2020b. Directional message passing for molecular graphs, in: ICLR.

Liu, W., Zhang, Y., Wang, J., He, Y., Caverlee, J., Chan, P.P.K., Yeung, D.S., Heng, P.A., 2021a. Item relationship graph neural networks for E-Commerce. IEEE Transactions on Neural Networks and Learning Systems , 1–15.

Liu, Z., Lin, L., Jia, Q., Cheng, Z., Jiang, Y., Guo, Y., Ma, J., 2021b. Transferable multi-level attention neural network for accurate prediction of quantum chemistry properties via multi-task learning. Journal of Chemical Information and Modeling 61.

Long, M., Cao, Z., Wang, J., Yu, P.S., 2017. Learning multiple tasks with multilinear relationship networks, in: NIPS.

Montavon, G., Rupp, M., Gobre, V., Vazquez-Mayagoitia, A., Hansen, K., Tkatchenko, A., Müller, K.R., Anatole Von Lilienfeld, O., 2013. Machine learning of molecular electronic properties in chemical compound space. New Journal of Physics 15, 095003.

Mrowca, D., Zhuang, C., Wang, E., Haber, N., Fei-Fei, L., Tenenbaum, J., Yamins, D., 2018. Flexible Neural Representation for Physics Prediction, in: Advances in Neural Information Processing Systems, pp. 8799–8810.

Panford-Quainoo, K., Bose, A.J., Defferrard, M., 2020. Bilateral trade modelling with graph neural networks, in: ICLR Workshop on Practical ML for Developing Countries.

Raissi, M., Yazdani, A., Karniadakis, G., 2018. Hidden Fluid Mechanics: A Navier-Stokes Informed Deep Learning Framework for Assimilating Flow Visualization Data. arXiv preprint arXiv:1808.04327 .

Ramakrishnan, R., Dral, P.O., Rupp, M., Von Lilienfeld, O.A., 2014. Quantum chemistry structures and properties of 134 kilo molecules. Scientific Data 1, 1–7.

Ruder, S., 2017. An overview of multi-task learning in deep neural networks. arXiv preprint arXiv:1706.05098 .

Salehi, Y., Giannacopoulos, D., 2022. PhysGNN: A Physics–Driven Graph Neural Network Based Model for Predicting Soft Tissue Deformation in Image–Guided Neurosurgery , in: NeurIPS.

Satorras, V., Hoogeboom, E., Welling, M., 2021. E(n) equivariant graph neural networks, in: ICML.

Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G., 2009. The graph neural network model. IEEE Transactions on Neural Networks 20, 61–80.

Schlichtkrull, M., Kipf, T., Bloem, P., van den Berg, R., Titov, I., Welling, M., 2018. Modeling relational data with graph convolutional networks, in: European Semantic Web Conference, pp. 593–607.

Schütt, K., Kindermans, P., Felix, H.E.S., Chmiela, S., Tkatchenko, A., Müller, K., 2017a. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions, in: Advances in neural information processing systems, pp. 991–1001.

Schütt, K., Unke, O., Gastegger, M., 2021. Equivariant message passing for the prediction of tensorial properties and molecular spectra, in: International Conference on Machine Learning.

Schütt, K.T., Arbabzadah, F., Chmiela, S., Müller, K.R., Tkatchenko, A., 2017b. Quantum-chemical insights from deep tensor neural networks. Nature communications 8, 1–8.

Thürlemann, M., Riniker, S., 2023. Anisotropic Message Passing: Graph Neural Networks with Directional and Long-Range Interactions , in: ICLR.

Timoshenko, J., Wrasman, C., Luneau, M., Shirman, T., Cargnello, M., Bare, S., Aizenberg, J., Friend, C., Frenkel, A., 2018. Probing atomic distributions in mono- and bimetallic nanoparticles by supervised machine learning. Nano Letters 19, 520–529.

Unke, O., Meuwly, M., 2019. PhysNet: A neural network for predicting energies, forces, dipole moments, and partial charges. J. Chem. Theory Comput. 15.

Wang, J., Wang, W., Kollman, P.A., Case, D.A., 2001. Antechamber: an accessory software package for molecular mechanical calculations. J. Am. Chem. Soc 222, U403.

Wang, Y., Yi, K., Liu, X., Wang, Y., Jin, S., 2023. ACMP: Allen-Cahn Message Passing with Attractive and Repulsive Forces for Graph Neural Networks , in: ICLR.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Yu, P., 2021. A comprehensive survey on graph neural networks. IEEE Transactions on Neural Networks and Learning Systems 32, 4–24.

Xiong, Z., Wang, D., Liu, X., Zhong, F., Wan, X., Li, X., Li, Z., Luo, X., Chen, K., Jiang, H., Zheng, M., 2020. Pushing the boundaries of molecular representation for drug discovery with the graph attention mechanism. Journal of Medicinal Chemistry 63, 8749–8760.

Yu, Z., Gao, H., 2022. Molecular Representation Learning via Heterogeneous Motif Graph Neural Networks, in: ICML.

Zhang, C., Song, D., Huang, C., Swami, A., Chawla, N.V., 2019. Heterogeneous graph neural network, in: International Conference on Knowledge Discovery & Data Mining.

Zhang, Y., Yang, Q., 2018. An overview of multi-task learning. National Science Review 5, 30–43.

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., Sun, M., 2020. Graph neural networks: A review of methods and applications. AI Open 1, 57–81.

Zitnick, L., Das, A., Kolluru, A., Lan, J., Shuaibi, M., Sriram, A., Ulissi, Z., Wood, B., 2022. Spherical Channels for Modeling Atomic Interactions, in: NeurIPS.