Crystal Hypergraph Convolutional Networks

Alexander Heilman

May 7, 2024

1 Introduction

Machine learning has proven to be a computationally cost-effective and powerful predictive tool in the screening of large sets of material systems for certain material properties. Some of the most effective modern models applied to such invariant target predictions represent material systems, such as crystals or molecules, as graphs. These graphs encode physical properties in feature vectors associated with both the nodes and edges, and update or 'learn' these features via a graph neural network, described most generally with the message passing framework defined in *Gilmore, et al* [GSR+17].

One problem with such graphical representations, however, is the lack of representation of larger order geometrical structure, since the constructed crystal graphs are necessarily only defined in terms of pair-wise descriptors. This may make it hard, or often, impossible, for models to distinguish between compositionally similar, but structurally unique systems.

For example, higher order geometric information is often particularly important in the case of complex oxides, where compositions may vary subtly while structure varies substantially.

Here, we propose the concept of *crystal hypergraphs* to ail this lack of geometrical structure in the more restrictive graph representations. In a crystal hypergraph, we may define larger (than strictly pair-wise) hyperedges that correspond to higher order geometrical structures of material systems explicitly, such as triplets of neighboring atoms, first shells of neighbors or motifs, and entire unit cells. These different structures then may have different coordinate invariant features associated with them, such as angles, order parameters, and symmetry groups, respectively. Note that in that regard, crystal hypergraphs are naturally heterogeneous in their hyperedges, since there are different feature sets for different types.

Of course, the definition of a more general hypergraph representation requires the generalization of the message passing framework mentioned above. So here, we propose three possible approaches to such a generalization that handle the now-variable size of hyperedges. In a certain sense, these allow for the learning of a certain type of 'cluster-correlation expansion' by the model, where clusters of interest correspond to the hyperedges defined.

As a proof of concept, we implement the above in the first crystal hypergraph convolutional models (CHGCNN), with an eye towards extending to equivariant and different convolutional structures in future works. Even these first, basic, models give a novel application to machine learning in this field: we may compare the performance on different tasks' validation sets between models based on different hyperedge types. This allows us to investigate the importance of different order structures for these different tasks. Namely, here, we compare the performance of models based on atom, bond and triplet information against those incorporating atom, bond, and motif information (i.e. first shell hyperedges) on various predictive tasks with varying data sizes.

Results indicate that motif level information (encoded in terms of continuous symmetry measures and local order parameters) may be sufficient, if not more informative, than triplet information (encoded as bond angles) at much less computational cost in terms of message order scaling. A point which may guide further research in the area.

The structure of this work is as follows: first, we give a brief overview of related and motivating works; then, we give an overview of crystal graph construction and message passing networks. A motivating representation problem is then identified with our definitions and the concept of crystal hypergraphs is introduced, with a particular focus on different types of hyperedges and their corresponding feature sets. Three generalized message passing are then considered, and a specific model architecture is presented. Finally, this specific architecture is used on various datasets to compare performance of different sets of hyperedge types.

2 Related Works

The concept of a continuous-filter convolutional operator for atomic systems, one that may adequately learn from the nuances in interatomic distance, was introduced by *Schutt et al* in [SKSF$^+$17]. The popular CGCNN [XG18] then followed, utilizing a specific continuous-filter convolutional layer which we refer to as CGConv.

The limitations of graphical structures in the representation of atomistic systems is well known. This limitation is addressed in several modern approaches by the inclusion of bond angle information in an auxiliary *line graph* $L(G)$, derived from some graph G, as in M3GNet [CO22] and ALIGNN [CD21]. The edges in the derived line graph correspond to triplets in our work. This concept is further generalized by the concept of Nested Graph Networks (NGNs) to consecutive line graphs $L(L(...G)$. The edges in an n-line graph $L^n(G)$ then correspond to a certain subset of hyperedges of order $n+2$.

Here, we argue that an alternative approach is preferred to convolution performed on line graphs $L(G)$, since these methods require a quadratic increase in the number of additional messages with respect to the number of edges in G. Results suggest that a single first-shell neighborhood hyperedge for each atom provides comparable geometric resolution at much less computational cost (i.e. much fewer messages). At least, this is true for most tasks tested in this work.

3 Crystal Graphs

A common representation of crystalline systems in machine learning is via graphs, or, collections of nodes and binary connections between them. We may define a crystal graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ as a set of vertices $v_i \in \mathcal{V}$, corresponding to each atom i, and edges $e_{ij} \in \mathcal{E}$, where edges are determined by some physical criteria.

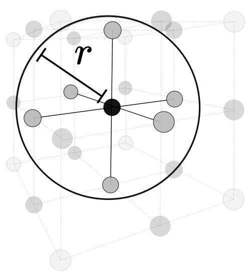
If we take the nodes to always represent atoms in crystal graphs, a specific implementation is then left to determine by what criteria we may determine connections between atoms, or edges, from the crystalline structure.

We also generally want to associate physical information with the objects in these graphs. This is often done by way of feature vectors. These are vectors with components describing the physical characteristics of their corresponding graph element, and which may be further 'learned' or updated through convolution.

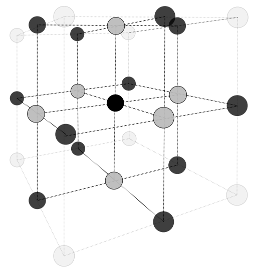
As such, crystal graphs are essentially defined by two (potentially task dependent) concerns: the criteria by which to derive edges from; and the features we choose to associate with the atoms and edges, which encode all of the physical information of the system. Below, we give the usual approaches for the construction of crystal graphs in terms of these concerns. Note, however, that while this section is meant to exemplify other works' approaches, we also use the following in the first step of crystal hypergraph generation.

3.1 Edge Criteria

Perhaps the simplest (and most often applied) criteria for the formation of edges between atoms is a combination of a maximum distance cutoff r_{max} and a maximum number of neighbors for each node N_{max}. That is, for each atom, we construct edges for up to N_{max}-th closest neighbors in the crystalline structure within a shell of radius r_{max}.



(within) Max radius r



N nearest neighbors

While other criteria may also be utilized, such as face-sharing in Voronoii Tesselations, and scaled maximum distance cutoffs based on nearest neighbor distances, these more advanced criteria are generally more computationally expensive in the data processing stage. Alternatively, most models [XG18, CD21, CYZ$^+$19] adopt the more basic criteria of maximum numbers of neighbors within cutoff shells, and adopt some continuous-filter convolutional function that can learn which connections are more important for the task at hand.

Often, the maximum number of neighbors considered is chosen to be 12, however recent tests in coNGN [RRSF24] suggest even this may be too restrictive, and show that a maximum set as large as 24 may improve performance.

3.2 Crystal Graph Features

With the means of edge determination settled, we now wish to incorporate more physical information with this derived graph representation.

Node features encode all of the atomic information of the sites they describe. Two usual techniques include: explicitly engineered feature vectors (as in [XG18]); and the learning of encodings for atomic sites based only on their atomic number (as in [CYZ$^+$19]), beginning with some random initialization.

Note that we could also, in principle, include an atomic position in the node features. However, this would require a special treatment of such information through convolution if we wished to maintain an *invariant* representation up to choice of coordinate system. As such, often only coordinate system (or more specifically, Euclidean group) invariant features are included in crystal graph representations. This restriction to invariant features also precludes any inclusion of unit vector orientation in edge features which are discussed below.

Edge features are further limited, to a certain extent, by their determination criteria. That is, if one determines edges based solely on their distance, and not at all on ionic or covalent structures, it is hard to justify the inclusion of specific bonding information in the corresponding edge features.

For edges determined exclusively by distance and a desired coordinate system invariance in edge features then, edge features are often derived exclusively from their distance. This distance is often expanded in dimensionality using some Radial Basis Function (RBF: $\mathbb{R} \to \mathbb{R}^{d_{RBF}}$ where d_{RBF} is the dimensionality of the radial basis function), such as a Gaussian or Bessel expansion.

These features are typical to modern machine learning models, with some notable exceptions and advancements (specifically those utilizing line graphs) to be considered later. We also note some limitations with these representations in the section after that which follows.

3.3 Crystal Graph Convolution

Crystal graphs are usually constructed for their sole use in some graph convolutional neural network. Perhaps the most general framework in which we may define graph convolution is the message passing framework, defined by Gilmore [GSR$^+$17]. A message passing network updates nodes based on 'messages' generated by the features of, and passed through, neighboring nodes (that is, nodes sharing an edge). Gilmore defines the message passing framework in terms of three functions: M, the message forming function; U, the node update function; and R, the readout function. These three functions act on graph representations as follows:

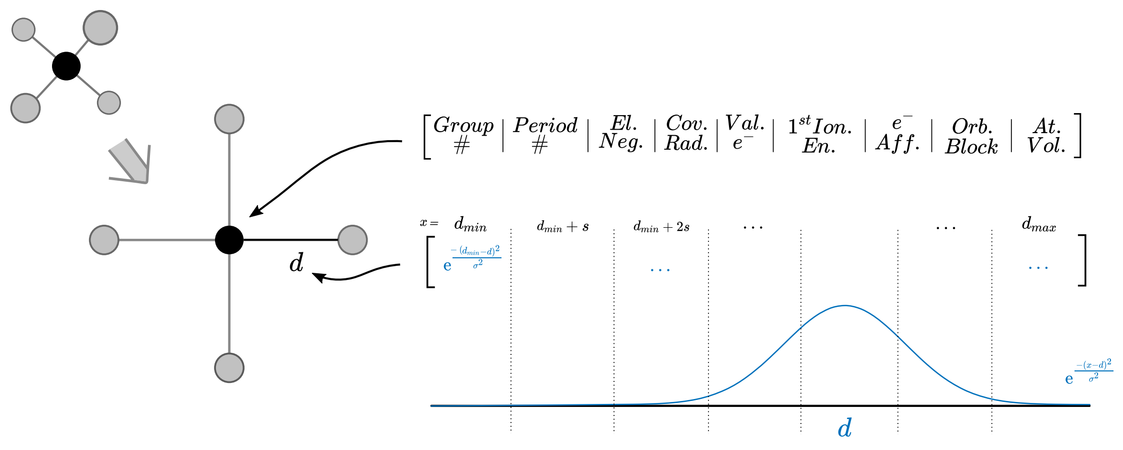


Figure 1: Schematic for a simple CGCNN-like graph encoding of some basic atomic structure. Note that the atomic properties are generally expanded by some one-hot encoding or RBF expansion and then concatenated. The edge feature here then is given, as an example, in terms of a Gaussian expansion of d, where d is the relevant interatomic spacing.

$$m_i^{t+1} = \sum_{n_j \in \mathcal{N}(i)} M_t(n_i^t, e_{ij}, n_j^t)$$
$$n_i^{t+1} = U_t(n_i^t, m_i^{t+1})$$
$$\hat{y} = R(\{n_v^T\})$$

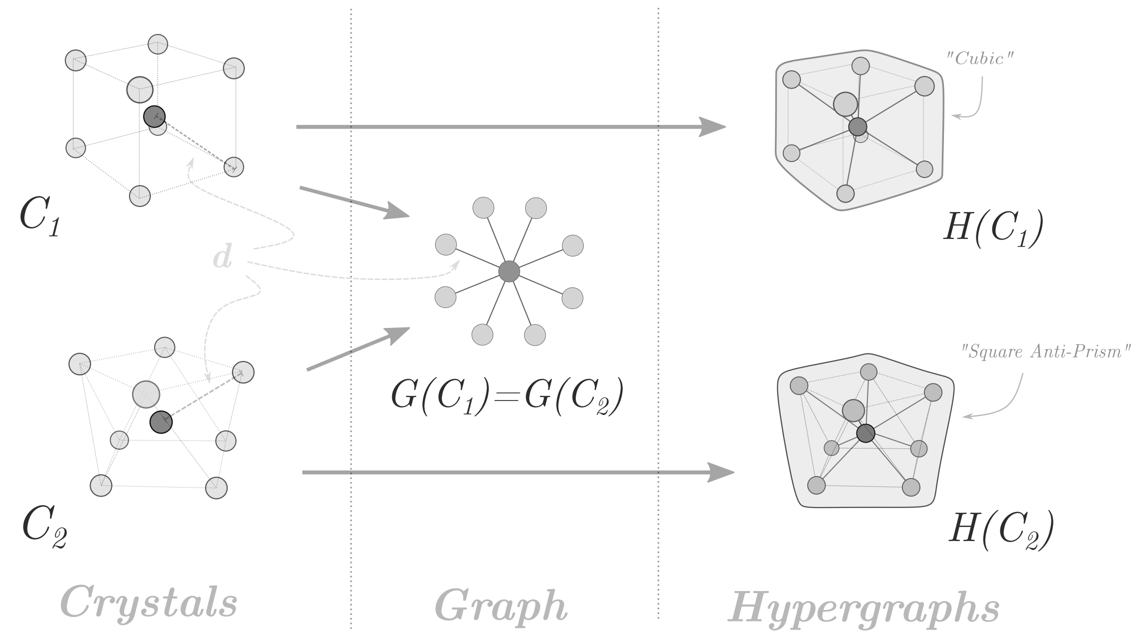
Where for each pair of connected nodes (or edge), a message is formed from each node's representation n_i^t, n_j^t, of layer t, and the connecting edge's feature e_{ij}. These node representations of layer t are updated to their $t+1$-th layer representation by update function U_t of layer t, which takes as input the set of messages for each node, and the nodes current representation. At the final layer T, the readout function R then predicts the target value from the updated node representations.

Message passing networks acting on crystal graphs have several advantages: message passing networks enjoy invariance under permutation of node indices; and, further, if the encoded features are coordinate system invariant, the output of the network is itself coordinate system invariant.

3.4 Disadvantages of Crystal Graphs

Perhaps the most poignant problem with our construction above though is the lack of higher-order geometrical information, i.e. local geometrical environments of atoms (that is, motifs) and global crystalline symmetries.

As a simple example of the low resolution manifest in crystal graphs, consider two atomic systems below: one with a local cubic symmetry, and another with a square anti-prism local environment; but both with the same bonding atoms.



In the above constructions, both would map to the exact same crystal graph, but could be easily be distinguished with an additional descriptor describing the local geometry of each's central atom.

4 Crystal Hypergraphs

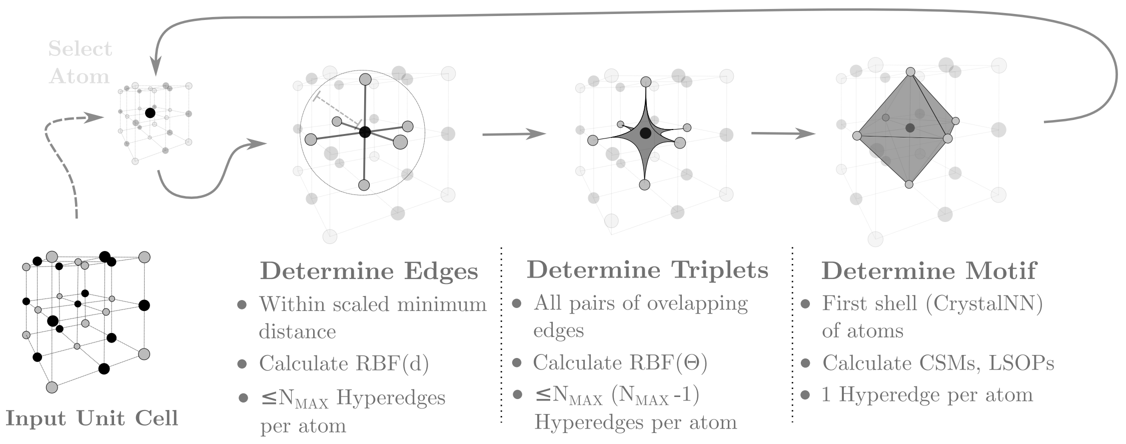
The extension proposed here solves the above problems by allowing for the explicit incorporation of this higher-order geometrical information in the form of hyperedges, which can be used to represent these higher-order structures explicitly.

A crystal hypergraph $\mathcal{H} = \{\mathcal{V}, H\}$ is a collection of nodes $v_i \in \mathcal{V}$ and hyperedges $h_j \in H$ (containing an arbitrary number of nodes), where the hyperedges are most generally heterogeneous (that is, we may wish to describe different types of structures in the same hypergraph). These objects further have associated feature vectors encoding relevant physical information, which we also refer to as v and h.

For the purpose of modeling material systems, we need first consider what different order structures are most important in their representation. Of course, atomic and bond level information is particularly important. However, higher order structures may also be of interest, such as: triplets of atoms, local environments of atoms (motifs in oxides), and entire crystalline unit cells of atoms.

Each of the aforementioned structures also has a natural set of distinct, coordinate-system invariant features that may be associated with them. At the triplet level (where two bonds share some common node), there is always a corresponding angle. At the motif level, order parameters [ZHJH17, ZJ20] or continuous symmetry measures [PA98, WGH$^+$20] may be used to describe 3D coordination environments quantitatively. And, at the unit cell level, we have the underlying crystal structure itself.

These different order structures may all be represented in a single crystal hypergraph. Below, we discuss the generation of, and association of features with, all of the above mentioned structures in crystalline solids.


Select Atom
Input Unit Cell
Determine Edges
Within scaled minimum distance
Calculate RBF(d)
≤N_MAX Hyperedges per atom
Determine Triplets
All pairs of ovelapping edges
Calculate RBF(Θ)
≤N_MAX (N_MAX -1) Hyperedges per atom
Determine Motif
First shell (CrystalNN) of atoms
Calculate CSMs, LSOPs
1 Hyperedge per atom

4.1 Bond Edges

Bonds, or pair-wise atomic connections, may be made in parallel to the approach detailed in the crystal graph section above. In the results below, for instance, we choose edges from a maximum number of neighbors $N_{max} = 12$ found within a shell of radius $r_{max} = 6\mathring{A}$.

The feature associated with such edges is a Gaussian expansion of the distance, with a range from 0 to 6 Angstrom and a dimensionality of 40.

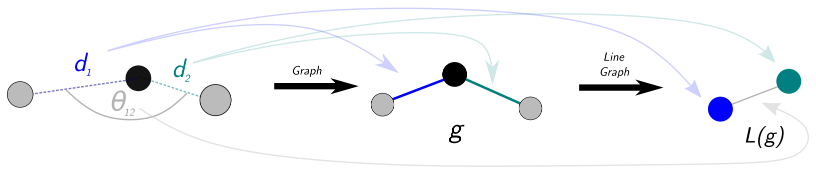
4.2 Triplet Hyperedges

Triplet hyperedges are then formed from the set of bonds, where for each set of bonds connected by one node, a triplet hyperedge is formed.

The feature of these triplet hyperedges is also a Gaussian expansion of the angle formed by the unit vectors of the two bonds.

4.2.1 Comparison to Line Graph

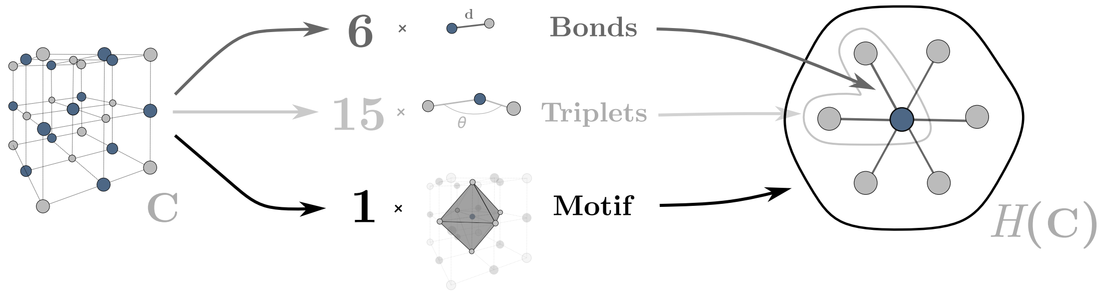
A more usual approach for the incorporation of bond angle information is via the construction of a line graph, as in [CD21, CO22].



These models generally first update the edge features of the crystal graph $\mathcal{G}$ by first applying some graph convolutional operator to the line graph $L(\mathcal{G})$ with angles encoded in $L(\mathcal{G})$'s initial edge features.

Our argument against such representation schemes here is that the order of messages grows combinatorically for derived line graphs as $\mathcal{O}(nm^2)$, where n is the number of nodes and m is the average number of edges per node in $\mathcal{G}$.

Here, we incorporate a similar level of higher order geometrical structure instead in a local environment, or 'motif', hyperedge (defined below). Note that these include only an extra number of messages on the order $\mathcal{O}(mn)$ if each node in a motif gets a message, or on the order $\mathcal{O}(n)$ if only center nodes are updated by it's own motif hyperedge(s).



4.3 Motif Hyperedges

Motif determination may be achieved by a wide range of functions, and is akin to an algorithmic determination of coordination number. Here, we use the method implemented as CrystalNN in pymatgen. Note this is a much stricter algorithm than that used to determine edges and triplets.

The features of these motifs include Zimmerman's 35 local structure order parameters [ZHJH17, ZJ20], and continuous symmetry measures [PA98] (e.g. 'distance to a perfect shape') calculated for 59 common coordination environments. Essentially, these are both just sets of quantitative measures designed to describe 3D physical shape.

5 Crystal Hypergraph Convolution

We now must consider a message passing framework analagous to *Gilmore, et al* [GSR+17] but applying to hypergraph structures. That is, we now have:

$$\begin{aligned} m_v^{t+1} &= \sum_{h_j \in \mathcal{N}(v)} M_t(n_v^t, h_j^t, \{n_w^t | n_w \in h_j\}), \\ n_v^{t+1} &= U_t(n_v^t, m_v^{t+1}), \\ \hat{y} &= R(\{n_v^T\}), \end{aligned}$$

so that each node is still updated according to layer-wise update function U_t, aggregating messages m^{t+1} formed from origin node features, hyperedge features h_j, and hyperedge neighborhood features $n_w \in h_j$, in analogy to the graph-based MPNN approach. This updating occurs node-wise and then after T layers, some readout function R is again used to output the corresponding predicted value $\hat{y}$, which utilizes the set of learned node features.

The biggest difference here is that we now need a message forming function M_i that accounts for a set of node features $\{n_w^t | n_w \in h_j\}$, that may vary in size between different hyperedges (even of the same type). This stands in opposition to the case of regular edges, where we are assured a fixed size of two nodes per edge.

One approach would be to fix the dimensionality of each type of hyperedge, or have a different convolutional operator for each different size hyperedge (as is effectively the approach taken in line graph networks). Here, however, we wish to maintain generality in edge size so we need not fix hyperedge orders for different types, as motifs with different coordination numbers, or different size unit cells, may be described by common sets of features.

Note that we may also wish to update hyperedge representations of one type according to other hyperedges of the same or other types. As an example, we may desire the two bonds forming a triplet to be updated from messages based on the angle they form. Or, we may hope to update a motif's representation based upon the unit cell it resides in.

These connections between the same or different orders of hyperedge may be defined based upon inclusion and/or contact. In the case of the bond-triplet-bond interaction, we may determine the relationship between these hyperedges by forming messages for bonds included in triplets. Furthermore, we may care to relate triplets that are touching each other, or mutually contain one or more nodes.

Of course, there should be different message and update functions for each different order structure (bonds, triplets, motifs, etc.) with different features. This is accounted for by treating the data as a heterogeneous graph, with different hyperedge types. Below, we consider three strategies that allow us to apply our convolutional operator to hyperedges of arbitrary size.

5.1 Three Possible Approaches to Hyperedge Convolution

Three general approaches for message passing that account for this multi-order nature and which can be used to incorporate inter-order hyperedge message passing have been considered in this work: **1.** the construction of a hyperedge relatives graph, upon which regular graph convolution may be applied; **2.** total exchange hyperedge message convolution, which completely generalizes the CGCNN [XG18] and ALIGNN [CD21] models to hypergraphs; and **3.** neighborhood aggregation, which balances performance of the former approach by forming a single neighborhood feature for each hyperedge.

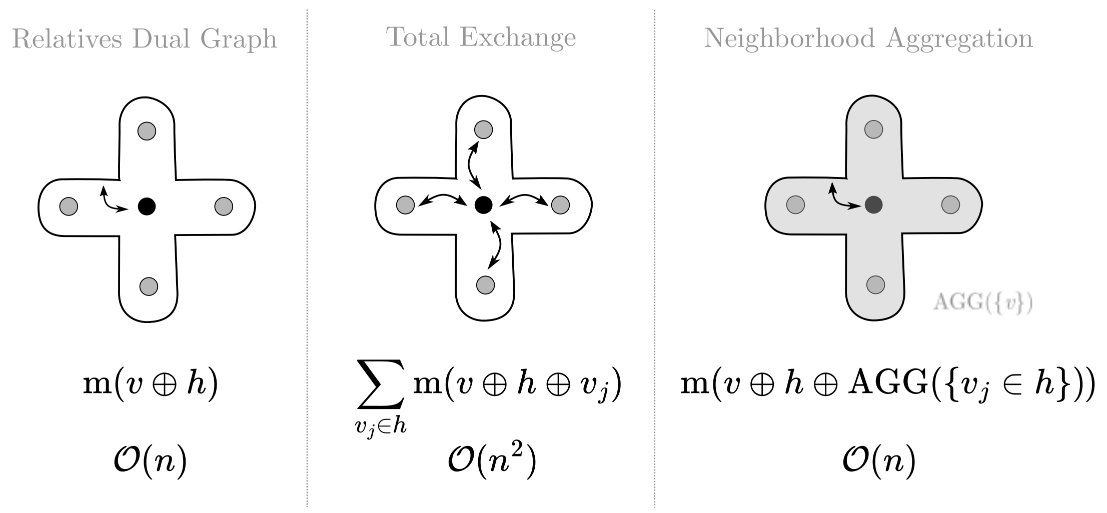
Each approach has a different computational cost in terms of total number of messages, along with a potentially different practical definition of a hypergraph. These considerations are presented below, with a specific convolutional structure and empirical results on common test datasets then presented after.

5.1.1 Hyperedge Dual Graph

We may define a dual graph $\mathcal{D}(h)$ to a hypergraph h to be a graph in which nodes represent the hyperedges of the hypergraph, and connections represent the overlap of respective hyperedge neighborhoods.

In the case of a crystal hypergraph with heterogeneous hyperedges, this dual graph is a graph with heterogeneous nodes. We term this heterogeneous dual graph of a crystal hypergraph the relatives graph for simplicity. Note that atomic features may be included by adding a singleton hyperedge for each node.

The definition of the relatives graph allows us to perform the usual methods of graph convolution on hyperedge features. And thus, requires we only define an edge index describing the connection between different hyperedges and nodes.



However, this approach lacks the interaction of neighboring features in convolution via the connecting hyperedge. That is, without a clear definition of the edge attribute, messages are generally of the form below:

$$m_v^t = \sum_{h_j \in \mathcal{N}(v)} M_t(n_v^t, h_j^t)$$

in which we simply discard the neighborhood of other node features contained in the hyperedge.

Computationally, this approach has a total number of messages that scales linearly with average hyperedge size, since each hyperedge only contributes one message to each node it contains. Accounting only for node-hyperedge connections in a relatives graph derived from a hypergraph with m hyperedges of average order n, the total number of messages per convolution will scale as $\mathcal{O}(nm)$.

5.1.2 Total Exchange

Of course, we may wish to incorporate the neighboring features of some representation via their connecting hyperedge. This may be accomplished by simply forming a message for every pair of connected representations along with their connecting hyperedge's representation.

$$m_v^t = \sum_{h_j \in \mathcal{N}(v)} \sum_{n_w \in h_j} M_t(n_v^t, h_j^t, n_w^t),$$

Note that this method completely generalizes previous approaches based on line graph convolution, or nested graph networks.

Here, though, we've introduced a new summation which may drastically increase the number of messages for larger hyperedges. If each hyperedge contains an average of n nodes, and there are m hyperedges in some hypergraph, the total number of messages exchanged per node-wise convolution will scale as $\mathcal{O}(n^2 m)$.

5.1.3 Neighborhood Aggregation

Since the number of messages will scale tremendously with larger hyperedges in the framework described above, we may seek a way to incorporate the neighborhood of features of a hyperedge into a single message.

In this case, we may essentially form a 'neighborhood feature' representative of all a hyperedge's contained nodes. Typical aggregation methods may be used and trained to perform this neighborhood feature generation. Here then, we deal with message functions of the form:

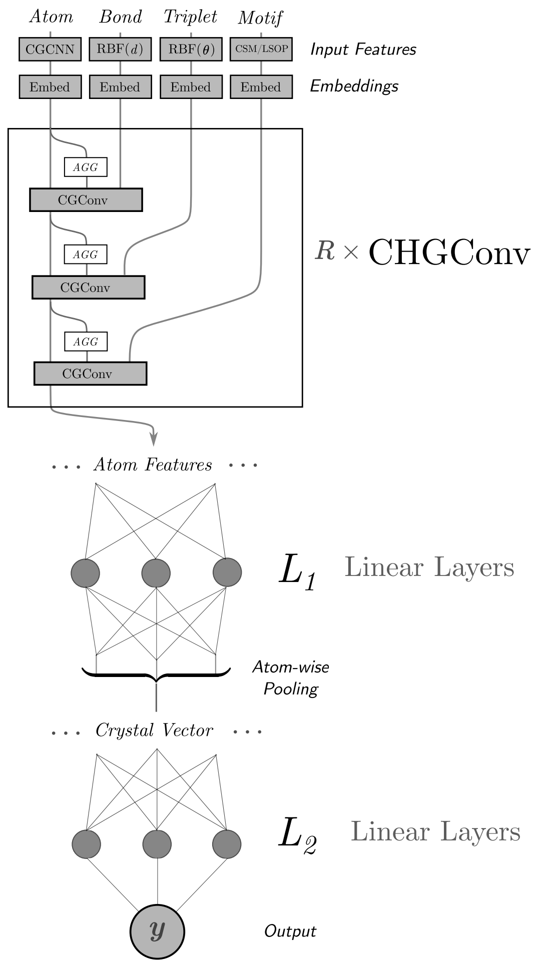
$$m_v^{t+1} = \sum_{h_j \in \mathcal{N}(v)} M_t\big(n_v^t, h_j^t, \mathrm{AGG}\big(\{n_w^t | n_w \in h_j\}\big)$$

This results in a number of node-wise messages that scales linearly with the average size of hyperedges, so that we now have a relationship of order $\mathcal{O}(nm)$ again, while still incorporating the features of neighboring nodes.

This approach also reduces the definition of the hypergraph back to a two dimensional, though now hyperedge, index set; where the first index is the node contained and the second index is the containing hyperedge (as in [BZT21]).

5.2 Example Model Architecture

Node and hyperedge features were first passed into a linear embedding layer (with no activation function), these were then updated via CHGConv layers applied in alternating order to bond and motif features (bond, motif, bond, motif). The learned node features were then mean pooled to form a crystal vector, which was passed to a fully connected layer and then projected down to a one-dimensional (scalar) output for regression. In the case of classification tasks, the fully connected layer, after mean pooling, utilized a dropout mechanism and output a probability distribution of classes via a softmax activation function.



6 Training and Results

Even with a basic convolutional structure (i.e. currently bad results overall), crystal hypergraph networks provide a unique opportunity to investigate the importance of different order correlations for different physical properties of the same systems. That is, by comparing validation results for different types of hyperedges, one may probe the importance of certain structures (i.e. motifs vs. triplets) by comparing such predictive performance of these different order models.

As such, a basic continuous filter hypergraph convolution is adopted here, with the specific details given in **??**; leaving performance improvements to be sought in different convolutional structures in future works.

Thus, we here focus on the comparative performance of models with different types of hyperedges on a common set of targets. Namely, we test models with and without bond, triplet, and motif level hyperedges on several invariant targets including formation energy, band gap, dielectric constants and elastic constants from both Materials Project (Table **??**) at large, and the more curated MatBench datasets (Table **??**).

Perhaps the strongest point to be made in regards to the results is that for most tasks, motif information contributes to comparable or better performance than triplet-level results. This is at much less computational cost in terms of the total number of messages exchange through convolution.

Future works may investigate more powerful hypergraph convolutional operators for improved performance. Other order structures (beyond motif-level) or hyperedges representing defects may also be of interest for future works. The inclusion of equivariant features and equivariant convolution may also be desired for certain tasks on hypergraphs.

	Phonons	Refractive Indices	Perovskites	$\text{Log}_{10}(G_{vrh})$	$\text{Log}_{10}(K_{vrh})$
	(1,265)	*(4,764)*	*(18,829)*	*(10,987)*	*(10,987)*
	MAE (cm^{-1})	MAE	MAE (eV/Atom)	MAE (Log_{10}GPa)	MAE (Log_{10}GPa)
Bond-only	84.1	0.497	0.0584	0.099	0.083
Triplet-only	71.3	0.520	0.0566	0.094	0.073
Motif-only	77.6	**0.485**	0.0611	0.103	0.077
Bond & Triplet	69.2	0.550	0.0550	**0.088**	**0.071**
Bond & Motif	**64.5**	0.510	**0.0488**	0.095	0.073

Figure 2: Validation dataset results for several MatBench target sets, note that the italicized numbers next to the target name correspond to the total size of each dataset. Best results are indicated in bold.

Hyperedge	Form. Energy	Band Gap	Metal/Non-metal
Types	Best MAE (eV/Atom)	Best MAE (eV)	Best Accuracy
Bond-only	0.177	0.315	.837
Motif-only	0.088	0.387	.853
Bond & Motif	**0.074**	**0.301**	**.860**

Figure 3: Validation dataset results for several Materials Project target sets. Here, each dataset included a total of 152,605 materials. Best results are indicated in bold.

References

[BHZ+21] Huta R Banjade, Sandro Hauri, Shanshan Zhang, Francesco Ricci, Weiyi Gong, Geoffroy Hautier, Slobodan Vucetic, and Qimin Yan. Structure motif–centric learning framework for inorganic crystalline systems. *Science advances*, 7(17):eabf1754, 2021.

[BZT21] Song Bai, Feihu Zhang, and Philip HS Torr. Hypergraph convolution and hypergraph attention. *Pattern Recognition*, 110:107637, 2021.

[CD21] Kamal Choudhary and Brian DeCost. Atomistic line graph neural network for improved materials property predictions. *npj Computational Materials*, 7(1):1–8, 2021.

[CO22] 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.

[CYZ+19] 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.

[GSR+17] 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.

[JBL+21] Jaehyeong Jo, Jinheon Baek, Seul Lee, Dongki Kim, Minki Kang, and Sung Ju Hwang. Edge representation learning with hypergraphs. *Advances in Neural Information Processing Systems*, 34:7534–7546, 2021.

[PA98] Mark Pinsky and David Avnir. Continuous symmetry measures. 5. the classical polyhedra. *Inorganic chemistry*, 37(21):5575–5582, 1998.

[RRSF24] Robin Ruff, Patrick Reiser, Jan Stühmer, and Pascal Friederich. Connectivity optimized nested line graph networks for crystal structures. *Digital Discovery*, 2024.

[SKSF+17] 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.

[WGH+20] David Waroquiers, Janine George, Matthew Horton, Stephan Schenk, Kristin A Persson, G-M Rignanese, Xavier Gonze, and Geoffroy Hautier. Chemenv: a fast and robust coordination environment identification tool. *Acta Crystallographica Section B: Structural Science, Crystal Engineering and Materials*, 76(4):683–695, 2020.

[XG18] 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.

[ZHJH17] Nils E. R. Zimmermann, Matthew K. Horton, Anubhav Jain, and Maciej Haranczyk. Assessing local structure motifs using order parameters for motif recognition, interstitial identification, and diffusion path characterization. *Frontiers in Materials*, 4, 2017.

[ZJ20] Nils ER Zimmermann and Anubhav Jain. Local structure order parameters and site fingerprints for quantification of coordination environment and crystal structure similarity. *RSC advances*, 10(10):6063–6081, 2020.

A Motif Determination

Motifs are determined according to criteria similar to that used in bond determination. Namely, interatomic distance, Voronoii tesselations, and solid-angles of Voronoii faces are still relevant. Despite this similarity in potential criteria, testing showed more restrictive approaches to motif determination resulted in better performance on validation sets. This stands in contrast to edge determination, where the continuous order convolution operators may be used to 'pick' the more important bonds from a large, loosely defined set of bonds, by the corresponding distance feature.

Several different schemes were tested for a single task (band gap) with a relatively large amount of samples (150,000) and compared. First, the same criteria as bonds was applied (that is, all within a certain radius, up to a maximum number of neighbors), which performed worst. Second, a scaled minimum radius cutoff was applied, where for each site, the motif was determined to be all those neighbors within a factor of $\alpha = 1.2$ times the closest neighbor's interatomic distance (for that site) r_{min}. Finally, the more restrictive criteria of CrystalNN, as defined in [?], was applied, which applies a scaled distance cutoff as well as a scaled solid angle cutoff. Results are displayed below for comparison. Note that all models used both bond and motif-level information (where bond criteria was held constant through all)

Motif Criteria	Naive	Scaled r_{min}	CrystalNN
Best MAE (eV)			

The more restrictive definitions of motifs increased in performance incrementally. This is likely since the larger motifs would tend to 'smoosh' the atomic information too much through aggregation of neighborhood features and messages.

B Motif Features: Structure Order Parameters & Continuous Symmetry Measures

The geometry of the motifs were incorporated as features composed of a concatenated list of structure order parameters and continuous symmetry measures (CSMs) for a set of common local environments.

Structure order parameters are coordinate system invariant measures of 3 dimensional structure that are designed to be close to one when a given structure is similar to some prototypical arrangement. Note that this isn't in general a true 'distance'-like measure to some shape as a CSM is, however. The list of order parameters included those implemented in pymatgen code [?] and described in [ZHJH17, ZJ20].

A CSM is defined precisely so that it may act as a 'distance' from some prototypical shape to some given structure.

C CHGConv

A specific implementation of a hypergraph convolutional operator in the hypergraph message passing framework is a generalization of CGConv implemented in pytorch geometric and based on CGCNN's convolutional operator defined in eq (5) of the original paper.

$$\begin{aligned}
x_i^{t+1} &= \sum_{b_j} f(x_i^t, b_j, \mathrm{AGG}(\{x_j^t \in b_j\})) \\
&= \mathrm{BN}\Bigg[\sum_{b_j} \sigma\big(W_c \cdot [x_j \oplus b_j \oplus \mathrm{AGG}(\{x_j^t \in b_j\}]\big)\big) \\
&\qquad \cdot S^+(W_f \cdot (x_j \oplus b_j \oplus \mathrm{AGG}(\{x_j^t \in b_j\})))\Bigg]
\end{aligned}$$

In the model utilized in this work, we generally employed use of a learnable set of common aggregation functions for the neighborhood feature aggregation (AGG above), inspired by *ChemGNN* [?].

D Hyperparameters for Testing

For each convolutional structure, testing was done for a model with 3 convolutional layers for 300 epochs. Stochastic gradient descent (SGD) was used as an optimizer through training with an initial learning rate of 0.01, and a multi-step learning rate scheduler dividing this learning rate by a factor of 10 at epochs 150 and 250.

Hidden node features were of dimension 64 through all convolutional layers, and a hidden output layer of dimension 128 was used (similar to CGCNN's architecture). The loss functions utilized were MSE (for regression tasks) and cross entropy (for classification tasks). Accuracy is then reported in MAE for regression tasks and percentage correctly classified for classification tasks. Datasets were split 80% for training and 20% for validation tests.