A Hypergraph Convolutional Neural Network for Molecular Properties Prediction using Functional Group

Fangying Chen
KAIST
gyjin32@kaist.ac.kr

Junyoung Park
KAIST
junyoungpark@kaist.ac.kr

Jinkyoo Park
KAIST
jinkyoo.park@kaist.ac.kr

Abstract

We propose a Molecular Hypergraph Convolutional Network (MolHGCN)[1] that predicts the molecular properties of a molecule using the atom and functional group information as inputs. Molecules can contain many types of functional groups, which will affect the properties the molecules. For example, the toxicity of a molecule is associated with toxicophores, such as nitroaromatic groups and thiourea. Conventional graph-based methods that consider the pair-wise interactions between nodes are inefficient in expressing the complex relationship between multiple nodes in a graph flexibly, and applying multi-hops may result in oversmoothing and overfitting problems. Hence, we propose MolHGCN to capture the substructural difference between molecules using the atom and functional group information. MolHGCN constructs a hypergraph representation of a molecule using functional group information from the input SMILES strings, extracts hidden representation using a two-stage message passing process (atom and functional group message passing), and predicts the properties of the molecules using the extracted hidden representation. We evaluate the performance of our model using Tox21, ClinTox, SIDER, BBBP, BACE, ESOL, FreeSolv and Lipophilicity datasets. We show that our model is able to outperform other baseline methods for most of the datasets. We particularly show that incorporating functional group information along with atom information results in better separability in the latent space, thus increasing the prediction accuracy of the molecule property prediction.

1 Introduction

Toxicological screening is vital for the development of new drugs, the evaluation of the therapeutic potential of existing molecules, and the assessment of pharmacological activity and toxicity potential of new molecules on human. Traditionally, toxicity studies of molecules relied on animal testings due to the biological and psychological similarities between humans and animals [6]. However, it not only raises ethical issues, but can also invalidate the pharmaceuticals studies and provide inadequate bases for predicting clinical outcomes on humans [1]. The U.S. Food and Drug Administration (FDA) also estimated that it takes about eight-and-a-half years to test and study a new drug before its approval to the general public, which includes early laboratory and animal testing [2]. Hence, this has provided the impetus to search for alternatives to replace or reduce the use of animal testing.

In recent years, machine learning methods have been adopted to assess the effects that chemical substances have on humans and the environment as it is less time-consuming, cheaper and more ethical since it does not rely on animal testing. Graph-based methods, such as graph convolutional networks (GCN), graph attention networks (GAT) and their variants, have been actively used for such

[1]The code can be found in https://github.com/fychen32/MolHGCN.

Preprint. Under review.

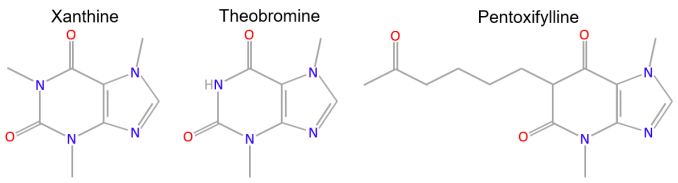


Figure 1: Molecules of similar structures but different properties: xanthine is found in caffeine and temporarily prevents or reduces drowsiness, theobromine is found in cacao and has mood improving effect, and pentoxifylline is a drug used to treat muscle pain in people with peripheral artery disease.

tasks. Due to their ability to represent molecules as graphs, they are preferred in molecule property prediction tasks as the molecular structure is inextricably linked to the molecular properties of a molecule. These methods take the graph representation and node features as inputs and consider the pair-wise relations between two nodes, where messages are propagated from the neighboring nodes within the graph. This, however, is inefficient in expressing the complex relationship between the structure and the properties of the molecule. Figure 1 shows molecules of similar structures but with some different functional groups. From Figure 1, although all three molecules have similar structures, the difference in the number of methyl, amine and ketone groups has resulted in the different effects that the compounds have on the human body [5]. Although xanthine and theobromine are made up of the same functional groups, the subtle difference between the two molecules have resulted in the different properties that they each have. While the pair-wise relationships represented by the graph-based methods can capture the difference between xanthine and theobromine, they are insufficient in capturing the difference between molecules of similar structures that extends beyond two molecules. Using multiple hops to increase the interactive scope of the atoms and structure of a molecule may result in oversmoothing and overfitting problems [8]. Hence, a graph-based method that is able to (1) model the complex relationship between multiple nodes in a graph flexibly, (2) capture the differences between molecules of similar structure and, at the same time, (3) capture the pair-wise relationship between nodes in a graph might be a better option for molecule property prediction tasks.

In this study, we propose a molecular hypergraph convolutional network (MolHGCN) to capture the pair-wise and substructural difference between molecules using the atom and functional group information. Hypergraphs are graphs in which each edge (hyperedge) can connect an arbitrary number of nodes. We choose to represent the functional groups as the hyperedges as it forms the basis of toxicity assessment and can be used to distinguish similar molecules from each other. Hence, this makes functional groups a favorable choice for hyperedges to express the relationship between the substructures and the properties of molecules. In addition to the functional group information, we also give our model the pair-wise atom information so as to capture the subtle difference between molecules that are made up of the same functional groups. Hence, using the functional group information, we are able to model the complex relationship between multiple nodes in a graph flexibly using hypergraphs (1), capture the difference between molecules of similar structure using functional groups (2), and capture the subtle difference between molecules of the same functional groups using the pair-wise atom information (3). Our model consists of:

- **Hypergraph construction from input SMILES strings.** We construct the molecular graph representation of the input simplified molecular-input line-entry system (SMILES) string, and use the information of each node and its respective neighboring nodes and edges to construct the hypergraph representation of each molecule. The initial hyperedge features are initialized using the features of the nodes that are in the hyperedge. We then encode the node and edge features of the graphs and the hyperedge features of the hypergraphs.
- **Hypergraph message passing neural network (`HyperMPNN`).** `HyperMPNN` includes an atom graph convolution (AtomGC) and a functional group GC (FuncGC) unit. We perform message passing on the atom representation in AtomGC, and use the updated atom features as inputs for FuncGC together with the encoded initial functional group features. The updated atom and functional group representations are then used to predict the properties of the molecules.
- **Predictor.** We feed the updated atom features and the updated functional group features into a readout component and then into a regressor to produce the final predictions.

We evaluate the effectiveness of MolHGCN on several datasets that are used for molecule property classification and regression tasks and show that MolHGCN is able to outperform other baseline methods for most datasets. We also show that incorporating functional group information along with atom information results in better separability in the latent space, thus increasing the prediction accuracy of the molecule property prediction.

2 Related work

Molecular properties prediction tasks have been attempted using various graph-based methods. Some methods attempted to improve the representability of the features while some methods attempted to utilize substructures from the graphs. In this section, we provide an overview of such methods that will be used for baseline comparisons with MolHGCN.

Improving representations. [3] proposed the message-passing neural network (MPNN) that consists of a message-passing phase and a readout phase. In MPNN, the atom information is transferred from its neighborhood along the graph, which are then updated to obtain an updated atom information. [12] proposed the Directed MPNN (DMPNN) that uses messages based on directed edges (chemical bonds) instead of nodes (atoms) so as to account for the information carried by the bonds and to avoid unnecessary loops in the message passing step as observed in MPNN. [9] proposed the Communicative MPNN (CMPNN) that further improves the molecular graph representation as compared to MPNN and DMPNN by updating the atom and bond representation interactively and enhancing the message generation step using a message booster so as to strengthen the messages between the atoms and bonds.

Although these methods have shown their potential in molecular properties predictions, they mainly focus on obtaining better representations of the pair-wise relationship between two atoms and do not account for other atoms that might give better representations. Instead of using directed bonds like DMPNN and CMPNN, MolHGCN uses undirected bonds so as to account for both atoms that are bonded together. This gives the bond information of the atoms from both ends.

Extracting substructures. [4] proposed the Adaptive GCN (AGCN) so as to extract residual substructures that are not defined by the bonds in the molecules by learning a residual graph adjacency matrix, which constructs the latent structural relations that are unspecified by the graph adjacency matrix, through a learnable distance function. [10] proposed the graph attribute aggregation network (GAAN). GAAN uses the graph attribute convolution operation to classify atoms and bond based on their intrinsic features and updates their features using a convolutional operation, and uses the progressive margin folding (PMF) operation that folds the marginal atoms inwards to increase the sensitivity of local molecular structures. In the PMF operation, a hypergraph is constructed to represent the folding results. [11] proposed the MoleculeKit that uses both the SMILES sequences and graph representation of the molecules as inputs, where the SMILES strings are fed into sequence-based models (a pretrained BERT and subsequence kernel) and the graph representations are fed into graph-based models (MPNN and Weisfeiler-lehman subtree). In MoleculeKit, MPNN is able to aggregate the atom, bond, subgraph and graph information.

Although these methods have tried to *learn* useful substructures from graphs, it is hard to determine if the extracted substructures can help to distinguish between molecules of similar structures or can accurately capture the nodes of an important substructure that determines the properties of the molecules. Also, in GAAN, constructing a hypergraph representation using PMF from marginal nodes is inefficient when applied to straight-chain molecules (i.e. most atoms are connected one after another) as many layers of PMF have to be applied to capture the entire molecule. Instead of learning substructures from the molecular graphs like the above methods, MolHGCN uses *domain knowledge* to extract functional group information so as to obtain a definite substructure information. Unlike the MPNN in MoleculeKit, MolHGCN constructs an edge between the substructures such that the hypergraph becomes a complete graph. Also, instead of using multiple layers, MolHGCN only uses one `HyperMPNN` unit.

3 Methodology

This section highlights the methodology of the proposed MolHGCN. Figure 2 shows the overall architecture of MolHGCN. MolHGCN consists of three main modules: a hypergraph construction

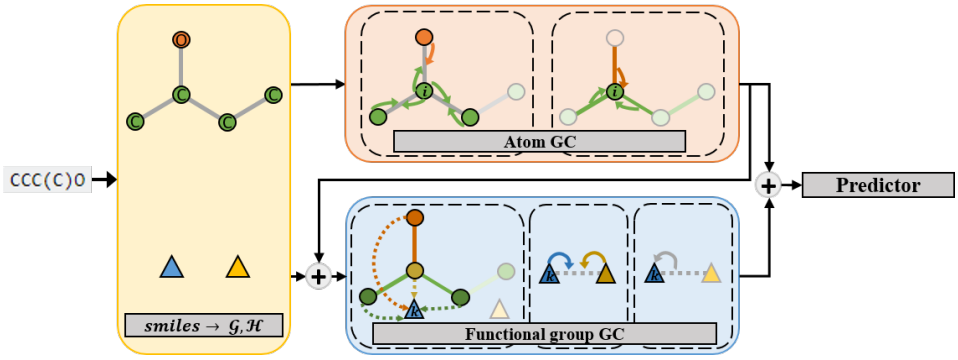


Figure 2: MolHGCN: hypergraph construction (yellow), `HyperMPNN` unit (red/blue), and predictor.

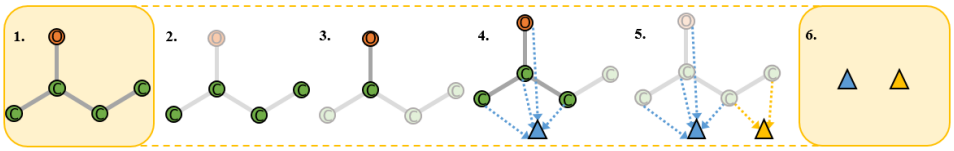


Figure 3: Hypergraph construction: (1) Graph representation. (2) Set C as central atom. (3) Central C atom is bonded to one O and two C atoms. (4) Add the central and adjacent atoms into the hyperedge. (5) Add the remaining atoms that do not belong in any defined groups into their respective hyperedges. (6) Extracted hypergraph.

unit, a `HyperMPNN` unit, and a predictor. The hypergraph construction unit identifies functional groups that are present in the molecules and put them into their respective hyperedges. The `HyperMPNN` unit then updates the node and hyperedge features. The resulting updated features will then be fed into the predictor to produce the final predictions.

3.1 Hypergraph construction

We represent the molecules as conventional pair-wise graphs and hypergraphs. The conventional pair-wise graphs are defined as $\mathcal{G} = \{\mathbb{V}, \mathbb{E}\}$, where $\mathbb{V}$ is a set of nodes (atoms) $v_i \in \mathbb{V}$, and $\mathbb{E}$ is a set of edges (bonds) $e_{ij} \in \mathbb{E}$ if a bond between v_i and v_j exists. The features of v_i and e_{ij} are defined as x_i and x_{ij} respectively. The hypergraph is defined as $\mathcal{H} = \{\mathcal{H}_k | k = 1, ..., n_K\}$, where $\mathcal{H}_k$ is a set of hyperedges (functional groups). The features of $\mathcal{H}_k$ are defined as x_k.

When constructing $\mathcal{H}$ from $\mathcal{G}$, we consider atoms in cyclic and open-chain (non-cyclic) groups separately. The minimal collection of cycles in the molecular graphs are extracted as the cyclic groups. For the non-cyclic groups, the vicinity of the functional group, which is defined as a central atom and the atoms attached to it, is considered when extracting the hyperedge representation. The main atoms that are used are carbon (C), nitrogen (N), oxygen (O), phosphorus (P) and sulfur (S), and the main bond types that are used are the single, double and triple bonds. The extraction process of the non-cyclic groups can be described as follows:

1. Set an atom type to be the central atom (C, N, O, P or S).
2. Set the atom (C, N, O, P or S) and bond (single, double or triple) type of the adjacent atoms for a specific functional group type.
3. Add the central atom and adjacent atoms into $\mathcal{H}_k$.
4. If the adjacent atom is not C and/or not single-bonded to the central atom, add their neighbors into $\mathcal{H}_k$.

Hence, the central atom, adjacent atoms and, if the conditions are met, the neighbors of the adjacent atoms are added into $\mathcal{H}_k$. Different combinations of the central atoms, and adjacent atoms and bonds are used to match each functional group. The remaining atoms that do not belong to any of the specified functional groups are put into the same hyperedge if they are connected by an edge. Figures 3 and 4 show the construction procedure for the alcohol group and examples of the extracted

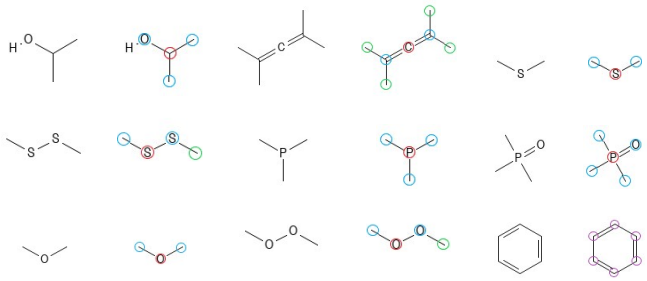


Figure 4: Examples of extracted functional groups: red circled atoms represent the central atoms, blue circled atoms represent the adjacent atoms of the central atoms, green circled atoms represents the neighbors of the adjacent atoms, and purple circled atoms represent that atoms in cyclic groups.

functional groups respectively. The algorithm used to extract the functional group from a given molecular graph and the list of functional groups used in this paper are given in Appendix A.1.

3.2 Hypergraph message passing neural network (HyperMPNN)

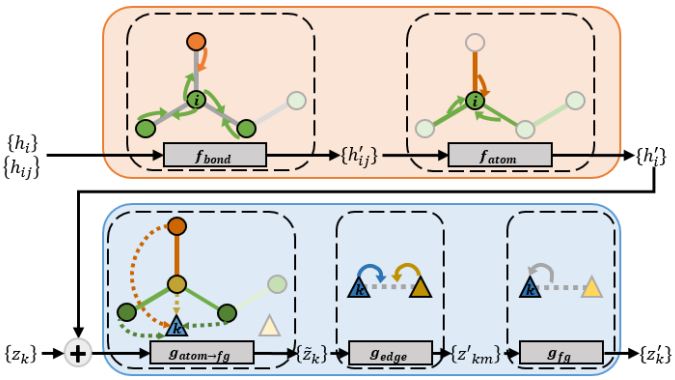


Figure 5: HyperMPNN: AtomGC (red) and FuncGC (blue)

The HyperMPNN unit is designed to integrate the atom and functional group information so as to extract crucial characteristic properties of a molecule that helps to identify and classify the molecules. It consists of an atom and a functional group GC. Before $\mathcal{G}$ and $\mathcal{H}$ are fed into the HyperMPNN unit, $\{x_i \,|\, v_i \in \mathbb{V}\}$, $\{x_{ij} \,|e_{ij} \in \mathbb{E}\}$, and $\{x_k \,|\mathcal{H}_k \in \mathcal{H}\}$ are encoded via feed-forward networks f_{enc} to produce their encoded features, $\{h_i \,|\, v_i \in \mathbb{V}\}$, $\{h_{ij} \,|e_{ij} \in \mathbb{E}\}$, $\{z_k \,|\mathcal{H}_k \in \mathcal{H}\}$, respectively. $\{h_i\}$ and $\{h_{ij}\}$ are then fed into the atom GC and $\{z_k\}$ is fed into the functional group GC. The overall updating procedure of the HyperMPNN unit can be defined as:

$$\{h'_i\}, \{h'_{ij}\}, \{z'_k\} = \text{HyperMPNN}(\{h_i\}, \{h_{ij}\}, \{z_k\}) \tag{1}$$

where $\{h'_i\}$, $\{h'_{ij}\}$ and $\{z'_k\}$ are a set of updated node, edge, hyperedge features respectively. Figure 5 illustrates the details of the HyperMPNN unit.

Atom graph convolution (AtomGC). AtomGC is designed to capture the subtle difference between molecules that are made up of the same functional groups as discussed in Section 1. It involves updating the edge features using the features of the edges and nodes that it connects, and updating the node features using the updated edge features. The edge update step is given as:

$$h'_{ij} = f_{\text{bond}}(h_i, h_j, h_{ij}) \tag{2}$$

where $f_{\text{bond}}(\cdot)$ is the edge MLP. It is noteworthy that, for the target tasks, the edge information is essential as the chemical bonds contains crucial information about the molecular properties. In the

node update step, the h'_{ij} is aggregated to produce the h'_i as follows:

$$\alpha_{ij} = f_{\text{attn}}(h_i, h_j, h_{ij}) \tag{3}$$

$$h'_i = f_{\text{atom}}\Big(h_i, \sum_{j=\mathcal{N}(i)} \alpha_{ij} h'_{ij}\Big) \tag{4}$$

where α_{ij} is the attention coefficient of e_{ij}, $f_{\text{attn}}(\cdot)$ is the attention multi-layer perceptron (MLP) whose output activation is the sigmoid activation function, and $f_{\text{atom}}(\cdot)$ is the node MLP and $\mathcal{N}(i)$ is a set of incoming edges of v_i. Here, unlike many attention modules that normalizes the attention scores so that the summantion of the scores becomes 1.0, we normalize each attention score to be between 0.0 and 1.0 similar to [7]. We empirically confirmed that this selection results in better prediction performance than the conventional attention modules.

Functional group graph convolution (FuncGC). FuncGC is designed to capture the difference between molecules of similar structures. Although the same functional groups can be present in many molecules, the effects that they have on the molecular properties may differ depending on their neighboring functional groups. To account for such differences, we utilize the updated node feature that contains local information from the molecular graphs when generating the localized functional group features. We start the functional group GC by updating z_k using h'_i as follows:

$$\tilde{z}_k = g_{\text{atom}\to\text{fg}}\Big(z_k, \sum_{i\in\mathcal{H}_k} h'_i\Big) \tag{5}$$

where $\tilde{z}_k$ is the localized feature, where we propagate localized information from AtomGC to FuncGC, and $g_{\text{atom}\to\text{fg}}(\cdot)$ is the localizing MLP. Unlike $\mathcal{G}$, $\mathcal{H}$ has no naturally defined edges. Hence, we learn the edges among the hyperedges as follows:

$$z'_{km} = g_{\text{edge}}(\tilde{z}_k, \tilde{z}_m) \tag{6}$$

where z'_{km} is the learnt edge feature between $\mathcal{H}_k$ and $\mathcal{H}_m$, and g_{edge} is the edge MLP. z'_{km} thus captures the interaction between $\mathcal{H}_k$ and $\mathcal{H}_m$. Lastly, we perform the hyperedge update with z'_{km} as follows:

$$\beta_{km} = g_{\text{attn}}(\tilde{z}_k, \tilde{z}_m) \tag{7}$$

$$z'_k = g_{\text{fg}}\Big(z_k, \sum_{m\in\mathcal{H}} \beta_{km} z'_{km}\Big) \tag{8}$$

where β_{km} is the attention coefficient between the k^{th} and the m^{th} hyperedge, $g_{\text{attn}}(\cdot)$ is the attention MLP whose output activation is sigmoid activation function as in the atom-level attention function, and $g_{\text{fg}}(\cdot)$ is the hyperedge update function.

3.3 Predictor

After we have obtained the updated node and hyperedge features from AtomGC and FuncGC respsectively, we feed them into the predictor to produce the final predictions. In the predictor, h'_i and z'_k are used to produce the final predictions y. They are first fed into a readout function $\rho(\cdot)$, and then concatenated before being fed into a regressor $f_{\text{reg}}(\cdot)$ to produce y as follows:

$$h' = \rho(\{h'_i | i \in \mathbb{V}\}) \tag{9}$$

$$z' = \rho(\{z'_k | k \in \mathcal{H}\}) \tag{10}$$

$$y = f_{\text{reg}}(h', z') \tag{11}$$

4 Experiments

This section highlights the performance of MolHGCN as compared to other baselines, and the ablation study that analyzes the effectiveness of atom and functional group incorporation on the performance of MolHGCN. We also compare between the usage of domain-knowledge (MolHCGN) and data-driven methods, which learn the hyperedges from the data, when extracting substructures from molecules.

Table 1: MolHGCN performance in graph classification and regression tasks (↑ means that higher result is better and ↓ means that lower result is better. − means that the result is not available.)

	Metric	AUROC					RMSE		
	Dataset	Tox21 (↑)	ClinTox (↑)	SIDER (↑)	BBBP (↑)	BACE (↑)	ESOL (↓)	FreeSolv (↓)	Lipophilicity (↓)
REPR	● MPNN (atom only) [12]	0.845	0.896	0.644	0.908	0.864	0.719	1.243	0.625
	⋆ MPNN [12]	0.844	0.881	0.641	0.910	0.850	0.702	1.242	0.645
	× DMPNN [12]	0.845	0.894	0.646	0.913	0.878	0.665	1.167	0.596
	▲ CMPNN-MLP [9]	0.856	0.933	0.666	**0.963**	0.887	**0.233**	0.819	0.580
SUB	● AGCN [4]	0.802	0.868	0.592	–	–	–	–	–
	⋆ GAAN [10]	0.839	0.888	0.658	–	–	0.294	1.057	0.605
MULT	● MoleculeKit [11]	**0.868**	0.962	0.718	–	–	0.513	0.99	**0.515**
	● MolHGCN	0.749	**0.974**	**0.728**	0.891	**0.890**	0.489	**0.810**	0.588

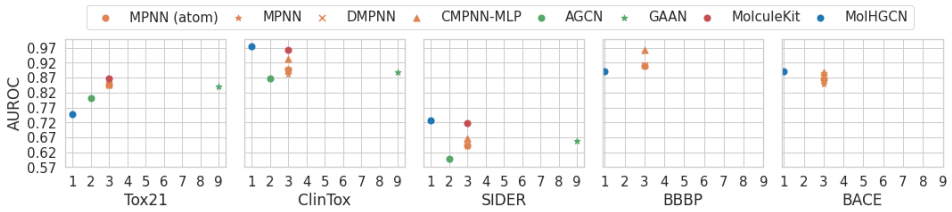


Figure 6: Number of GCs vs classification performances

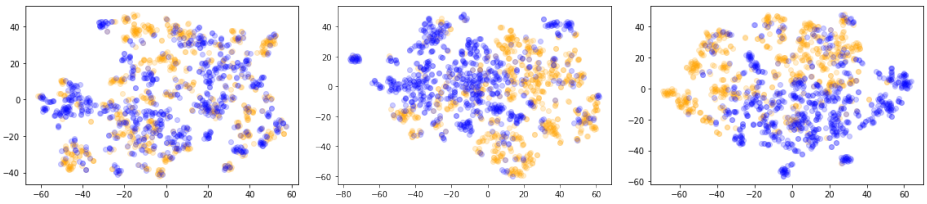
4.1 Benchmark results

We evaluate the performance of MolHGCN with baselines that improve the representability of the input features (*REPR*), extract substructural information (*SUB*), and utilize multiple molecule representations (*MULT*). This is done so as to analyze the usage of atoms, directed and undirected bonds information, and further enhanced representations (*REPR*), compare the effectiveness of the different types of substructure that are designed or learnt with those that are extracted from MolHGCN (*SUB*), and compare the effectiveness of the usage of multiple input representations and models (*MULT*). The benchmark datasets for the performance evaluation includes Tox21, ClinTox, SIDER, BBBP, BACE, ESOL, FreeSolv and Lipophilicity. We use the standard featurization strategy for x_i and x_{ij}. The hyperedge features x_k is initialized as either the average nodes features in $\mathcal{H}_k$ or zero vectors. We also make the inclusion of cyclic groups an option for the model inputs. We randomly split the datasets into 80:10:10 ratio as the training, validation and test sets. Further details of data preparation and model training can be found in Appendix A.2.

Table 1 shows the overall results of MolHGCN on graph classification and regression tasks. From Table 1, we can see that MolHGCN has outperformed the other baselines in four out of eight of the datasets (ClinTox, SIDER, BACE and FreeSolv). From the results of *REPR*, we can see that the usage of atoms, directed and undirected bond information does not have a significant impact on the performance. Instead, increasing the interactions between the atoms and bonds in CMPNN-MLP gives better results, especially for ClinTox, BBBP and ESOL. MolHGCN is comparable with CMPNN-MLP even though we did not employ any special methods to enhance the atom and bond interactions or the message generation. From the results of *SUB*, we can see that MolHGCN outperforms the other baselines for most of the datasets. This shows that giving the model more chemically meaningful substructural information is more beneficial. It can be seen that MolHGCN did not do as well as *REPR* models for Tox21. This may be because the extraction of substructural information from the graphs is not beneficial for Tox21 as none of the baselines in *SUB* did better than the worse-performing baseline in *REPR* (MPNN). Notably, MolHCN outperforms *MULT*, which further utilizes the SMILES strings and the pretrained BERT, and Weisfeiler-Lehman kernel that is known as one of the winning graph kernel method. Although applying multiple GCs allows the models to aggregate information from the higher-order neighborhood, MolHGCN outperformed the other baselines that uses multiple GC layer with only 1 `HyperMPNN` unit as shown in Figure 6. This shows the efficacy of employing the functional group representation in conducting the benchmark tasks.

Table 2: Ablation study on the effects of functional groups (↑ means that higher result is better and ↓ means that lower result is better. We run the experiments for 5 times with the different random seeds.)

Dataset	AtomGC	FuncGC	F.G. in $\mathcal{H}$	Tox21 (↑)	ClinTox (↑)	SIDER (↑)	BBBP (↑)	BACE (↑)	ESOL (↓)	FreeSolv (↓)	Lipophilicity (↓)
MolHGCN -AtomGC	✔	✗	✗	0.742 (±0.002)	0.960 (±0.003)	0.715 (±0.007)	0.865 (±0.040)	0.869 (0.006)	0.561 (±0.025)	0.869 (±0.189)	0.618 (±0.040)
MolHGCN -FuncGC	✗	✔	✗	0.731 (±0.007)	0.934 (±0.007)	0.702 (±0.008)	0.846 (±0.010)	0.857 (0.033)	0.677 (±0.028)	1.357 (±0.169)	0.699 (±0.018)
MolHGCN -NoFG	✔	✔	✗	**0.779** (±0.004)	0.952 (±0.014)	0.713 (±0.006)	0.864 (±0.003)	0.870 (0.022)	0.577 (±0.051)	1.255 (±0.106)	**0.545** (±0.013)
MolHGCN	✔	✔	✔	0.749 (±0.004)	**0.974** (±0.014)	**0.728** (±0.006)	**0.891** (±0.003)	**0.890** (0.022)	**0.489** (±0.051)	**0.810** (±0.106)	0.588 (±0.013)



(a) AtomGC (b) FuncGC (c) MolHGCN

Figure 7: Latent space visualization of BACE using t-SNE

4.2 Ablation studies

In this section, we show that the algorithmic choice of MolHGCN that uses atoms and functional group information together is beneficial for molecule properties predictions. We also show that using domain knowledge to construct the hypredges is more favorable than that of data-driven approaches.

4.2.1 Atom and functional group utilization

We evaluate the effectiveness of the atom and functional group incorporation by comparing MolHGCN, MolHGCN-AtomGC, MolHGCN-FuncGC and MolHGCN-`NoFG`. In MolHGCN-AtomGC, only AtomGC is utilized to extract the features and then $f_{\text{reg}}(h')$ to predict the labels. Similarly MolHGCN-FuncGC only utilizes FuncGC, where $\tilde{z}_k = z_k$ and $f_{reg}(z')$ is used to predict the labels. In MolHGCN-`NoFG`, each hyperedge is constructed with two nodes that are bonded together (no functional group information). $\mathcal{H}$ in MolHGCN-`NoFG` can be viewed as an edge-centric reformulation of $\mathcal{G}$. MolHGCN-`NoFG` is designed to provide a fair comparison to study the effect of constructing the hyperedges as functional groups to the performances of the models using the same model architecture. We use the same datasets and experimental settings as in Tables A.5 and A.6 in Appendix A.2 for all models, except x_k in MolHGCN-FuncGC where x_k is the average node features in $\mathcal{H}_k$.

Table 2 shows the results of the MolHGCN and its variants. From Table 2, it can be seen that the atom and functional group information complement each other well as MolHGCN outperforms the other models in general. This can also be proven as MolHGCN-AtomGC/FuncGC did not outperform MolHGCN. For Tox21 and Lipophilicity, MolHGCN-`NoFG` outperforms MolHGCN. Hence, this shows that increasing the connectivity between all atoms in each molecule to enhance the representability of the features may be more important than that of substructure extraction for Tox21 and Lipophilicity. This is also observed in *REPR*, where *REPR* models outperformed MolHGCN for Tox21 and Lipophilicity.

To further validate the effects of the functional group information on the prediction performance, we compare the latent space visualization of MolHGCN-AtomGC/FuncGC and MolHGCN. Figure 7 visualizes the latent space of BACE. The dots with different colors represents the different class labels. From Figure 7, we can see a clearer separation between the two classes for MolHGCN as compared to MolHGCN-AtomGC/FuncGC. This further proves that the atom and functional group information complement each other well. Comparing Figures 7a and 7b, we can see that using MolHGCN-FuncGC has better separability than MolHGCN-AtomGC even though MolHGCN-AtomGC has outperformed MolHGCN-FuncGC as in Table 2. Hence, we can conclude that, although functional groups are more able to express the complex relationship between substructures and the molecular properties well, the addition of pair-wise atom information further increases the separability between molecules of different properties.

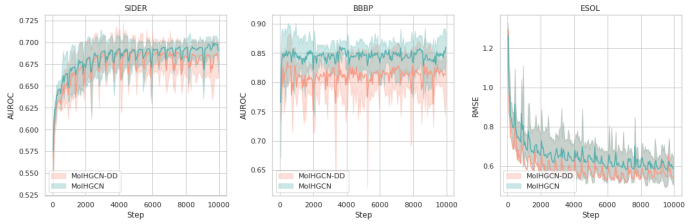


Figure 8: Domain knowledge vs data-driven: the lines represents the average test performances and the shadows represents the standard deviation. We repeat the runs five times.

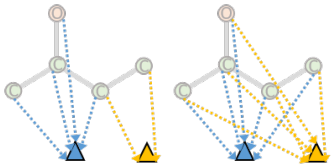


Figure 9: $\mathcal{H}$ construction of MolHGN (left) and MolHGCN-DD (right)

4.2.2 Comparison with data-driven method

To verify the advantage of the provision of accurate and definite functional group information when constructing hyperedges using domain knowledge (MolHGCN), we extend MolHGCN such that it learns to construct *soft* hyperedges via an attention mechanism. The attention variant of MolHGCN (MolHGCN-DD) constructs $\mathcal{H}_k$ using all nodes in $\mathcal{G}$ while setting the number of hyperedges to be the same as that in MolHGCN. Figure 9 shows the hyperedge construction comparison of MolHGCN and MolHGCN-DD. We use the the same experimental setting for both models as given in Appendix A.2.

Figure 8 visualizes the test AUROC and RMSE over the training steps for SIDER, BBBP and ESOL. We provide the performance curves of these datasets only as the other datasets show similar trends and asymptotic behaviors. As shown in Figure 8, MolHGCN converges faster and has better asymptotic performance than that of MolHCGN-DD. Hence, this shows that using domain knowledge to construct hypergraphs in MolHGCN is more beneficial than that of data-driven hypergraph construction. This may be because, with a more accurate and definite functional group information extracted using domain knowledge, the model is able to differentiate and classify molecules of different properties more accurately. The performance trend of the other datasets can be found in Figure A.1 in Appendix A.3.

5 Conclusion

We propose molecule hypergraph convolutional network (MolHGCN) to integrate the atom and functional group information so as to extract crucial characteristic properties of a molecule. We construct a hypergraph representation of the molecules using functional groups, and update the node and hyperedge features using the HyperMPNN unit. We evaluate the performance of our model with several baseline methods. We show that our model is able to outperform other baseline methods for most of the datasets. In our ablation study, we evaluate the effectiveness of the functional groups information by showing the better performance of the proposed model and comparing the latent representations of the models that utilize the conventional graph convolution and hypergraph convolutions. We also verify that extracting the substructural information using domain knowledge is more beneficial than that of data-driven approaches. We acknowledge that the effectiveness of MolHGCN is limited by the current chemical knowledge of molecules. To this end, we aim to learn the functional group extraction procedure alongside with the domain knowledge to make use of MolHGCN to discover new molecules. Such research direction can also be used to study the reactivity of compounds as it is crucial in directing and controlling organic reactions.

References

[1] A. Akhtar. The flaws and human harms of animal experimentation. *Cambridge quarterly of healthcare ethics : CQ : the international journal of healthcare ethics committees*, 24:407–19, 09 2015. doi: 10.1017/S0963180115000079.

[2] U. Food and D. Administration. The beginnings: Laboratory and animal studies, 2015. URL https://www.fda.gov/drugs/information-consumers-and-patients-drugs/beginnings-laboratory-and-animal-studies#:~:text=The%20FDA%20estimates%20that%2C%20on,clinical%20trials%20using%20human%20subjects.

[3] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. Neural message passing for quantum chemistry. *CoRR*, abs/1704.01212, 2017. URL http://arxiv.org/abs/1704.01212.

[4] R. Li, S. Wang, F. Zhu, and J. Huang. Adaptive graph convolutional neural networks. *CoRR*, abs/1801.03226, 2018. URL http://arxiv.org/abs/1801.03226.

[5] S. P and P. S. A short review on the effect of functional group in methylxanthine (caffeine) class of drugs. *Biochemistry and Pharmacology: Open Access*, 07, 01 2018. doi: 10.4172/2167-0501.1000257.

[6] S. Parasuraman. Toxicological screening. *Journal of Pharmacology and Pharmacotherapeutics*, 2:74–79, 04 2011. doi: 10.4103/0976-500X.81895.

[7] J. Park and J. Park. Physics-induced graph neural network: An application to wind-farm power estimation. *Energy*, 187:115883, 08 2019. doi: 10.1016/j.energy.2019.115883.

[8] Y. Rong, W. Huang, T. Xu, and J. Huang. Dropedge: Towards deep graph convolutional networks on node classification, 2020.

[9] Y. Song, S. Zheng, Z. Niu, Z.-h. Fu, Y. Lu, and Y. Yang. Communicative representation learning on attributed molecular graphs. In C. Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 2831–2838. International Joint Conferences on Artificial Intelligence Organization, 7 2020. doi: 10.24963/ijcai.2020/392. URL https://doi.org/10.24963/ijcai.2020/392. Main track.

[10] P. Sun, J. Qu, X. Lyu, H. Ling, and Z. Tang. Graph attribute aggregation network with progressive margin folding, 05 2019.

[11] Z. Wang, M. Liu, Y. Luo, Z. Xu, Y. Xie, L. Wang, L. Cai, and S. Ji. Advanced graph and sequence neural networks for molecular property prediction and drug discovery, 2021.

[12] K. Yang, K. Swanson, W. Jin, C. Coley, P. Eiden, H. Gao, A. Guzman-Perez, T. Hopper, B. Kelley, M. Mathea, A. Palmer, V. Settels, T. Jaakkola, K. Jensen, and R. Barzilay. Analyzing learned molecular representations for property prediction. *Journal of Chemical Information and Modeling*, 59, 07 2019. doi: 10.1021/acs.jcim.9b00237.

A Appendix

A.1 Hypergraph construction algorithm and list

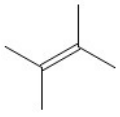
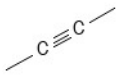
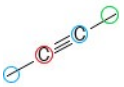
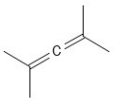
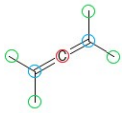
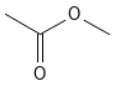
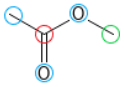
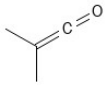
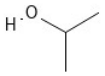
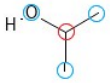
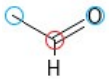
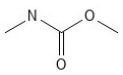
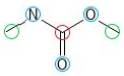
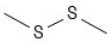
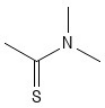
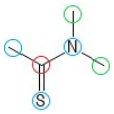
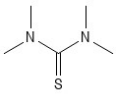
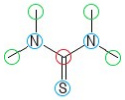
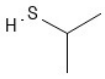
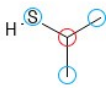
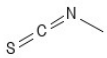
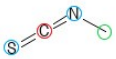
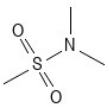
This section provides the hypergraph construction details. The general algorithm used to extract functional group from a given molecular graph is given in Algorithm 1 and the list of functional groups used in this paper is given in Table A.1.

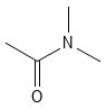
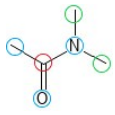
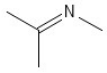
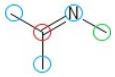
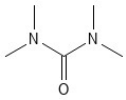
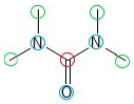
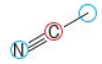
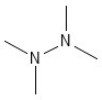
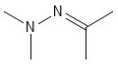
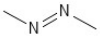
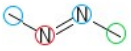
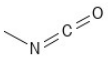
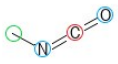
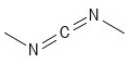
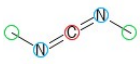
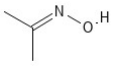
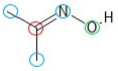
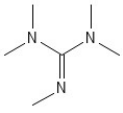
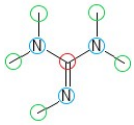
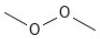
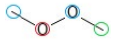
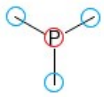
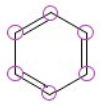
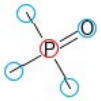
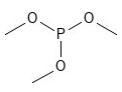
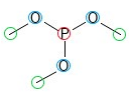
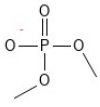
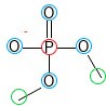
Algorithm 1: General algorithm for hypergraph construction

input : Graph $\mathcal{G} = (\mathbb{V}, \mathbb{E})$
Central atom types = [*C*, *O*, *N*, *P*, *S*]
input :
$\mathcal{H} \leftarrow \{\}$; `// Initialize empty set.`
$k \leftarrow 0$; `// Initialize hyperedge counter.`
for $cycle \in$ *FindCycles*($\mathcal{G}$) **do**
  $\mathcal{H}_k \leftarrow cycle$
  $\mathcal{H} \leftarrow \mathcal{H} \cup \mathcal{H}_k$; `// Append cycle to` $\mathcal{H}$
  $k \leftarrow k + 1$
end
for $v_i \in \mathbb{V} \setminus$ *FindCycles*($\mathcal{G}$) **do**
  if $v_i \in$ *Central atom types* **then**
    $\mathcal{H}_k \leftarrow \{v_i\}$; `// Section 3.1 Step (3)`
    for $v_j \in \mathcal{N}(i) \cup \{v_i\}$ **do**
      $\mathcal{H}_k \leftarrow \mathcal{H}_k \cup \{v_j\}$; `// Section 3.1 Step (3)`
      if $v_j \neq C$ *and/or* $e_{ij} \neq$ *single-bond* **then**
        $\mathcal{H}_k \leftarrow \mathcal{H}_k \cup \mathcal{N}(j)$; `// Section 3.1 Step (4)`
      end
    end
    $\mathcal{H} \leftarrow \mathcal{H} \cup \mathcal{H}_k$
    $k \leftarrow k + 1$
  end
end
return : Extracted hypergraph $\mathcal{H}$

Note that Algorithm 1 is a simplified way to construct the hypergraphs. If one wants to specify the functional groups that are extracted, Step 2 in Section 3.1 should be used. An example is given in Figure 3.

Table A.1: Functional group list and constructed hyperedges: red circled atoms represent the central atoms, blue circled atoms represent the adjacent atoms of the central atom, green circled atoms represents the neighbors of the adjacent atoms that are included in the hyperedges, and purple circled atoms represent the atoms that are in cycles.

Functional group	Structure	Hyperedge	Functional group	Structure	Hyperedge
Alkene			Alkyne		
Allene			Carboxyl		
Ketene			Alcohol		
Ketone			Aldehyde		
Ether			Carbamate		
Thioether			Disulfide		
Sulfone			Thioamide		
Thiourea			Thiol		
Thione			Sulfoxide		
Isothiocynate			Sulfonamide		
Sulfonate			Hydroxylamine		

Functional group	Structure	Hyperedge	Functional group	Structure	Hyperedge
Amide			Imine		
Carbamide			Nitrile		
Hydrazine			Hydrazone		
Azo			Isocynate		
Nitro			Carbodiimide		
Oxime			Nitroso		
Carboximidamide			Peroxide		
Phosphorus groups			Cycles		
					
					
					

A.2 Data and training details

This section provide the data and training details.

Data details. The dataset information are given in Table A.2. The atom and bond features that are used as the initial node and edge features are given in Tables A.3 and A.4 respectively. We use the BaseAtomFeaturizer and BaseBondFeaturizer of DGL-LifeSci to extract features from the initial atom and bond features. The hypergraphs were constructed using Networkx. The x_k and cyclic group information are given in Table A.5.

Table A.2: Datasets types, number of tasks, performance metric and split type

Dataset	Task	Number of tasks	Metric	Split
Tox21	Classification	12	AUROC	Random
ClinTox	Classification	2	AUROC	Random
SIDER	Classification	27	AUROC	Random
BBBP	Classification	1	AUROC	Random
BACE	Classification	1	AUROC	Random
ESOL	Regression	1	RMSE	Random
FreeSolv	Regression	1	RMSE	Random
Lipophilicity	Regression	1	RMSE	Random

Table A.3: Atom features used to featurize the node features

Atom Features	Number of Features
atom type one hot	43
atomic number	1
atom mass	1
atom degree one hot	11
atom explicit valence one hot	6
atom implicit valence one hot	7
atom total num H one hot	5
atom formal charge one hot	5
atom hybridisation one hot	5
atom num radical electrons one hot	5
atom is aromiatic one hot	2
atom is in ring one hot	2
atom chiral tag one hot	4
atom chirality type one hot	2
atom is chiral center	1

Table A.4: Bond features used to featurize the edge features

Bond Features	Number of Features
bond type one hot	4
bond is in ring	1
bond is conjugated one hot	2

Table A.5: Initial hyperedge features x_k and inclusion of cyclic groups. x_k is initialized as the average node features if TRUE and zero vectors if FALSE. Cyclic groups are included if TRUE and, otherwise, FALSE.

Dataset	x_k	Cycles
Tox21	FALSE	FALSE
ClinTox	FALSE	FALSE
SIDER	TRUE	FALSE
BBBP	TRUE	FALSE
BACE	TRUE	TRUE
ESOL	TRUE	FALSE
FreeSolv	FALSE	FALSE
Lipophilicity	TRUE	TRUE

Training details. We use the AdamP optimizer, whose learning rate is initialized as 0.001 and scheduled by the cosine annealing method. We use the binary cross entropy loss function and give extra weights to the minority class in the loss function based on the ratio of minority to majority class of each task to handle the class imbalance problem in the datasets for the classification tasks. We use the weighted sum and max function as the readout function in the predictor. We also use the batch normalization in the MLP. We use a batch size of 128 and ran the model for 2400 epochs. TITIAN RTX GPU is used to run the experiments. The training details can be found in Table A.6.

Table A.6: Node, edge and hyperedge (NEH) dropout, regressor dropout, MLP neurons and hidden dimensions of each dataset

Dataset	NEH dropout	Regressor dropout	MLP neurons	Hidden dimensions
Tox21	0.2	0.2	–	32
ClinTox	0	0	–	32
SIDER	0	0	–	32
BBBP	0	0.1	–	32
BACE	0	0.1	–	128
ESOL	0	0	$[64, 32]$	32
FreeSolv	0	0	$[128, 64]$	32
Lipophilicity	0	0	$[64, 32]$	32

A.3 Result details

This section provides the details of the results obtained from MolHGCN. The performance trends of Tox21, ClinTox, BACE, FreeSolve and Lipophilicity can be found in Figure A.1.

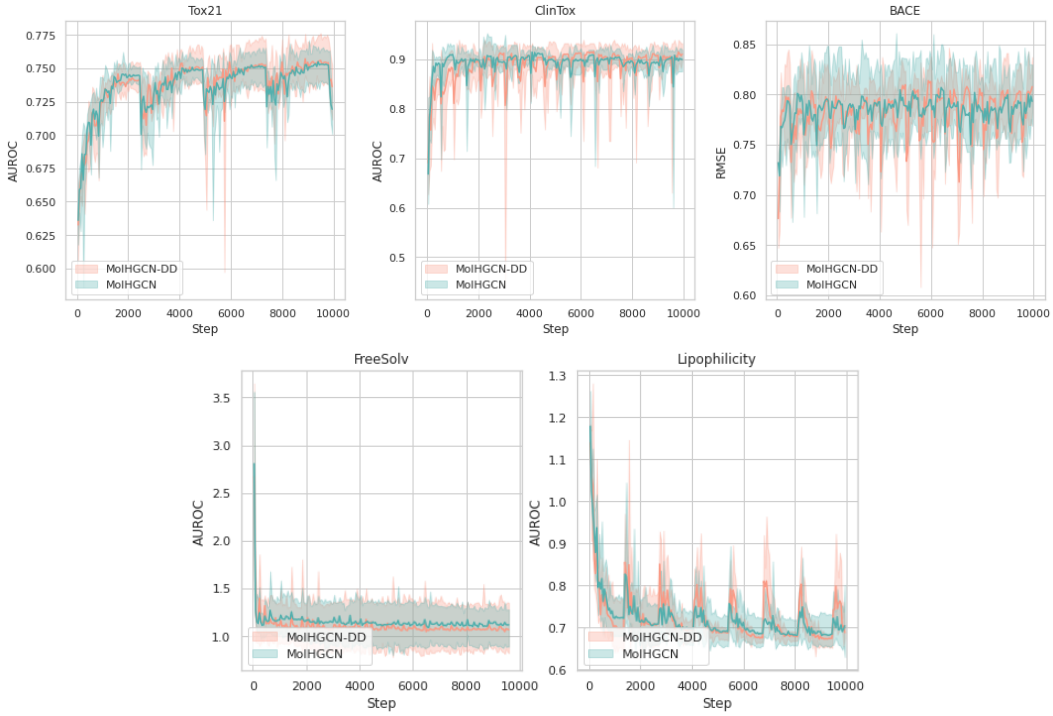


Figure A.1: Domain knowledge vs data-driven: green lines represent MolHGCN and orange lines represent MolHGCN (data-driven)