Molecular Contrastive Learning with Chemical Element Knowledge Graph

Yin Fang[1,2,3*], Qiang Zhang[1,2,3*], Haihong Yang[1,2,3], Xiang Zhuang[1,2,3], Shumin Deng[1,3], Wen Zhang[4], Ming Qin[1,2,3], Zhuo Chen[1,2,3], Xiaohui Fan[5,6,7], Huajun Chen[1,2,3†]

[1] College of Computer Science and Technology, Zhejiang University
[2] ZJU-Hangzhou Global Scientific and Technological Innovation Center, Zhejiang University
[3] AZFT Joint Lab for Knowledge Engine [4] School of Software Technology, Zhejiang University
[5] Pharmaceutical Informatics Institute, College of Pharmaceutical Sciences, Zhejiang University
[6] Innovation Center in Zhejiang University, State Key Laboratory of Component-Based Chinese Medicine
[7] Westlake Laboratory of Life Sciences and Biomedicine
{fangyin, qiang.zhang.cs, haihong825, zhuangxiang, 231sm, wenzhang2015,
qinandming, zhuo.chen, fanxh, huajunsir}@zju.edu.cn

Abstract

Molecular representation learning contributes to multiple downstream tasks such as molecular property prediction and drug design. To properly represent molecules, graph contrastive learning is a promising paradigm as it utilizes self-supervision signals and has no requirements for human annotations. However, prior works fail to incorporate fundamental domain knowledge into graph semantics and thus ignore the correlations between atoms that have common attributes but are not directly connected by bonds. To address these issues, we construct a Chemical Element Knowledge Graph (KG) to summarize microscopic associations between elements and propose a novel **K**nowledge-enhanced **C**ontrastive **L**earning (**KCL**) framework for molecular representation learning. KCL framework consists of three modules. The first module, *knowledge-guided graph augmentation*, augments the original molecular graph based on the Chemical Element KG. The second module, *knowledge-aware graph representation*, extracts molecular representations with a common graph encoder for the original molecular graph and a Knowledge-aware Message Passing Neural Network (KMPNN) to encode complex information in the augmented molecular graph. The final module is a *contrastive objective*, where we maximize agreement between these two views of molecular graphs. Extensive experiments demonstrated that KCL obtained superior performances against state-of-the-art baselines on eight molecular datasets. Visualization experiments properly interpret what KCL has learned from atoms and attributes in the augmented molecular graphs. Our codes and data are available at https://github.com/ZJU-Fangyin/KCL.

1 Introduction

Accurately predicting the properties of molecules lies at the core of fundamental tasks in the chemical and pharmaceutical communities. In light of deep learning, several supervised models have been investigated to learn molecular representations through predicting molecular properties (Gilmer et al. 2017; Yang et al. 2019; Song et al. 2020). While effective, these methods face the challenges of limited labeled data, as laboratory experiments are expensive and time-consuming to annotate data. Moreover, due to the enormous diversity of chemical molecules, these works could barely generalize to unseen cases (Hu et al. 2020; Rong et al. 2020), which greatly hinders practical applicability.

One line of works to alleviate these issues is to design pretext tasks to learn node or graph representations without labels. Several attempts have been made to investigate different strategies for such tasks, including masked attribute prediction (Hu et al. 2020), graph-level motif prediction (Rong et al. 2020), and graph context prediction (Liu et al. 2019). The other line follows a contrastive learning framework from the computer vision domain (Wu et al. 2018b; Chen et al. 2020), which aims to construct similar and dissimilar view pairs via graph augmentations, including node dropping, edge perturbation, subgraph extraction, and attribute masking (You et al. 2020). Due to the smaller amount of parameters and simpler predefined tasks, we adopt contrastive learning in our work.

However, unlike images, contrastive learning on graphs has its unique challenges. First, the structural information and semantics of the graphs vary significantly across domains, which makes it difficult to design a universal augmentation scheme for graphs. Especially for molecular graphs, removing or adding a chemical bond or a functional group will drastically change their identities and properties (You et al. 2020). More importantly, existing graph contrastive learning models mainly focus on graphs structures, without considering fundamental domain knowledge into graph semantics. Another neglected defect is that they model the atoms in molecular graphs as individuals that can only interact when there exists an edge (i.e., a chemical bond), failing to consider the correlations between atoms (e.g., commonalities between atoms of the same attributes).

To overcome these challenges, we enrich the molecular graph contrastive learning by incorporating domain knowledge. Since chemical domain knowledge is crucial prior, we hypothesize that the attributes of elements (atom is an instance of element) can affect molecular properties. To ob-

*These authors contributed equally.
†Corresponding Author.

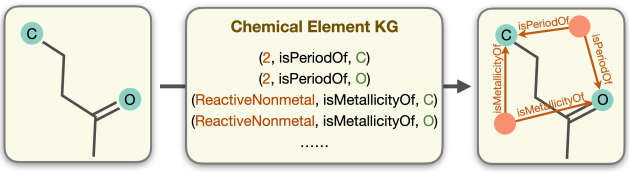

Figure 1: Chemical Element KG builds associations between atoms that are not directly connected by bonds but related in fundamental chemical attributes, as denoted by red arrows.

tain the domain knowledge and build microscopic correlations between atoms, we first construct a Chemical Element Knowledge Graph (KG) based on Periodic Table of Elements [1]. The Chemical Element KG describes the relations between elements (denoted in green in Figure 1) and their basic chemical attributes (e.g., periodicity and metallicity, denoted in red in Figure 1). Then we augment the original molecular graph with the guidance of Chemical Element KG, as shown in Figure 1, which helps to establish the associations between atoms that have common attributes but are not directly connected by bonds. In this way, the augmented molecular graph contains not only structural topologies but also the fundamental domain knowledge of elements.

On top of that, we propose a novel **K**nowledge-enhanced **C**ontrastive **L**earning (KCL) framework to improve the molecular representation with three modules. (1) The *knowledge-guided graph augmentation* module leverages Chemical Element KG to guide the graph augmentation process. While preserving the topology structure, the augmented molecular graph also builds associations that cannot be observed explicitly. (2) The *knowledge-aware graph representation* module learns molecular representations. We adopt a commonly used graph encoder for the original molecular graphs while designing a Knowledge-aware Message Passing Neural Network (KMPNN) encoder to provide heterogeneous attentive message passing for different types of knowledge in the augmented molecular graph. (3) The *contrastive objective* module trains the encoders to maximize the agreement between positives and the discrepancy between hard negatives. To the best of our knowledge, it is the first work to construct KG based on fundamental knowledge of chemical elements and guide molecular contrastive learning. Our contributions can be summarized as follows:

- We construct a Chemical Element KG, which describes the relations between elements and their chemical attributes. It can assist various molecular learning tasks beyond the ones in this paper.
- We develop a new contrastive learning framework (KCL) with three modules: knowledge-guided graph augmentation, knowledge-aware graph representation, and contrastive objective.
- We evaluate KCL on eight various molecular datasets under both fine-tune and linear protocols and demonstrate its superiority over the state-of-the-art methods.

[1] https://ptable.com

2 Related Works

Molecular Representation Learning In light of deep learning, Duvenaud et al. first applied convolutional networks to map molecules into neural fingerprints. Subsequent works fed SMILES (a line notation for describing the structure of chemical species using short ASCII strings) into recurrent networks-based models to produce molecular representations (Jastrzebski et al. 2016; Xu et al. 2017). To utilize topology information in the molecular graph, MPNN (Gilmer et al. 2017) and its variants DMPNN (Yang et al. 2019), CMPNN (Song et al. 2020), CoMPT (Chen et al. 2021) leverage the node and edge attributes during message passing. However, all the above-mentioned works are supervised models, require expensive annotations, and could barely generalize to unseen molecules, which greatly hinders the feasibility in practice.

Self-Supervised Learning on Graphs Self-supervised learning addresses such a limitation by pre-training molecular graphs. Liu et al. exploited the idea of N-gram in NLP and conducted vertices embedding by predicting the vertices attributes. Hu et al. designed two pre-training tasks, i.e., predicting neighborhood context and node attributes, to learn meaningful node representations, then using graph-level multi-task pre-training to refine graph representations. Alternatively, GROVER (Rong et al. 2020) incorporated a Transformer-style architecture and learned node embeddings by predicting contextual properties and motif information. Other works (Shang et al. 2019; Sun, Lin, and Zhu 2020; Yasunaga and Liang 2020) utilized similar strategies for either node or graph level pre-training.

Contrastive Learning on Graphs Contrastive learning is a widely-used self-supervised learning algorithm. Its main idea is to make representations of positive pairs that agree with each other and negatives disagree as much as possible (You et al. 2020). One key component is to generate informative and diverse views from each data instance. Previous graph augmentations generated views by randomly shuffling node features (Velickovic et al. 2019; Hassani and Ahmadi 2020), removing edges or masking nodes (You et al. 2020). However, these perturbations may hurt the domain knowledge inside graphs, especially for chemical compounds. MoCL (Sun et al. 2021) proposed a substructure substitution and incorporated two-level knowledge to learn richer representations, CKGNN (Fang et al. 2021) selected positive pairs via fingerprint similarity. But they ignore the fundamental domain knowledge.

3 Methodology

3.1 Problem Formulation

A molecule can be represented as a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $|\mathcal{V}|$ denotes a set of n atoms (nodes) and $|\mathcal{E}|$ denotes a set of m bonds (edges). Each edge is bidirectional. Let $\mathcal{N}_v$ denote the set of node v's neighbors. We use x_v to represent the initial features of node v, and e_{uv} as the initial features of edge (u, v). Let $\boldsymbol{h}(v)$ be the node hidden state and $\boldsymbol{h}(e_{uv})$ for the edge hidden state. In the setting of self-supervised graph representation learning, our goal is to learn

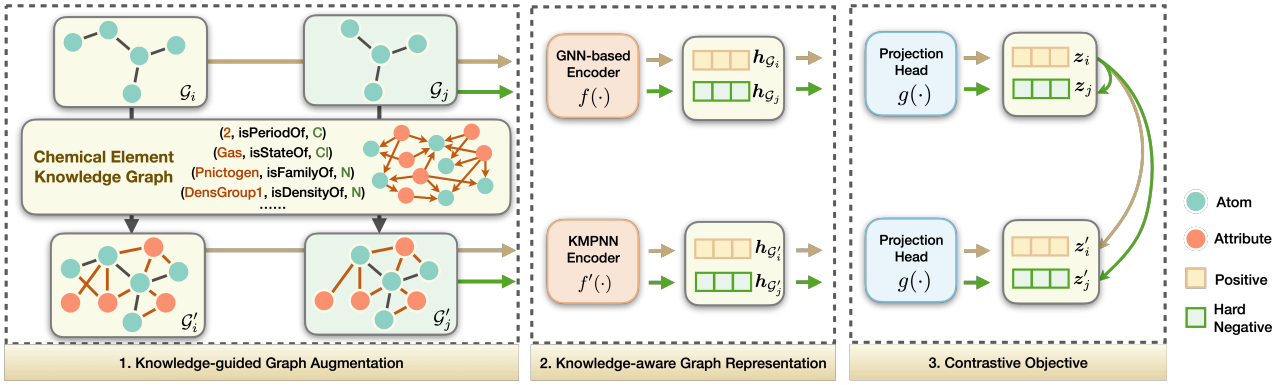

Figure 2: An illustrative example for KCL. We ignore edge directions in four molecular graphs due to space limitation (the direction of an edge between an attribute and an atom is from the former to the latter, while an edge between atoms is bidirectional). Module 1: Knowledge-guided graph augmentation converts the original molecular graph $\mathcal{G}$ into the augmented molecular graph $\mathcal{G}'$ based on Chemical Element KG. Module 2: Knowledge-aware graph representation captures representations from two graph views separately. Module 3: Contrastive objective trains the encoders and the projection head to maximize agreement between positives and disagreement between hard negatives (e.g., $\mathcal{G}_j$ act as the hard negative of $\mathcal{G}_i$) via a contrastive loss.

graph encoders $f : \mathcal{G} \mapsto \mathbb{R}^d$ which maps an input graph to a vector representation without the presence of any labels. The learned encoders and representations can then be used for downstream tasks.

3.2 Overview

Figure 2 shows the overview of our work. We propose a contrastive learning framework called KCL with three modules: (1) Knowledge-guided graph augmentation transforms any given molecule graph $\mathcal{G}$ into an augmented molecular graph $\mathcal{G}'$ with the guidance of Chemical Element KG. (2) Knowledge-aware graph representation aims to extract representations from $\mathcal{G}$ and $\mathcal{G}'$ respectively. (3) Contrastive objective aims to project representations to the space where contrastive loss is applied and train the encoders to maximize the agreement between positive pairs and the discrepancy between hard negatives.

3.3 Knowledge-guided Graph Augmentation

Chemical Element KG Construction. The prerequisite of our work is to collect the fundamental chemical domain knowledge. Previous attempts (Delmas et al. 2021; Lin et al. 2020) built KGs from the public chemical database and scientific literature to extract associations between chemicals and diseases or drug pairs, but none of them considered the fundamental information in chemical elements. In contrast, we crawl all the chemical elements and their attributes from the Periodic Table of Elements. Each element contains more than 15 attributes, including metallicity, periodicity, state, weight, electronegativity, electron affinity, melting point, boiling point, ionization, radius, hardness, modulus, density, conductivity, heat, and abundance.

After that, the extracted triples in the form of (Gas, isStateOf, Cl) are constructed in KG, indicating that there are specified relations between elements and attributes. However, since each element has some different continuous attributes, it is difficult for KG to model their connections. To overcome this difficulty, we histogramize the continuous attributes and convert them into discrete labels (e.g., DensityGroup1, RadiusGroup2). The statistics of Chemical Element KG are summarized in Table 1.

	Chemical Element KG
Elements	118
Attributes	107
Entities	225
Relation Types	17
KG Triples	1643

Table 1: The statistics of Chemical Element KG.

Graph Augmentation. Since most existing augmentation approaches (e.g., node dropping and edge perturbation) violate the chemical semantic inside molecules and ignore the influence of fundamental knowledge on graph semantics, we address these issues by proposing a knowledge-guided graph augmentation module with the guidance of Chemical Element KG. Specifically, as shown in Figure 2, we extract 1-hop neighbor attributes (nodes in red) of atoms (nodes in green) in a molecule from Chemical Element KG and add the triples as edges (edges in red). For example, we add a node "Gas" and an edge from "Gas" to "Cl" to the original molecular graph based on the triple (Gas, isStateOf, Cl). Note that the direction of each edge between the attribute and the atom is from the former to the latter, while the edges between atoms are bidirectional. Then we obtain an augmented molecular graph, in which the original molecular structure is preserved, and neighborhood topologies for atom-related attributes are introduced.

While preserving the topology structure, the augmented molecular graph $\mathcal{G}'$ also considers the fundamental domain knowledge within elements, as well as the microscopic associations between atoms that have common attributes but are not directly connected by bonds. The augmented molecular graph thus contains richer and more complex information, and is treated as a positive sample in contrastive learning.

3.4 Knowledge-aware Graph Representation

Knowledge Feature Initialization. Different from the random initialization of atoms and bonds, in order to obtain the initial features of attributes and relations in the augmented molecular graph, we adopt the commonly used KG embedding method, RotateE (Sun et al. 2019), to train Chemical Element KG. In this way, the initial features can capture the structural information of the triples. The necessity of this step is proved in subsequent experiments. More details are in A.1 of Appendix.

KMPNN Encoder. Although various architectures can be adopted, since the augmented molecular graphs are complex irregular-structured data that combines two types of information (i.e., the structure knowledge implied in molecular bonds and domain knowledge extracted from Chemical Element KG), we design a KMPNN encoder as $f'(\cdot)$ to learn their graph-level representations. The key idea behind KMPNN is that we provide two types of message passing for different types of neighbors, and assign them different attention according to their importance.

Algorithm 1 describes the KMPNN encoding process. The input of the encoder is the augmented molecular graph $\mathcal{G}' = \{\mathcal{V}, \mathcal{E}\}$, including initial features of all nodes x_v, $\forall v \in \mathcal{V}$, and features of all edges e_{uv}, $\forall (u,v) \in \mathcal{E}$. K rounds of message passing are then applied to all nodes. We enable heterogeneous message passing with two MSG functions, where $\mathrm{MSG}_1(\cdot)$ is applied to neighbors representing atoms, and $\mathrm{MSG}_0(\cdot)$ is applied to attributes in the neighborhood. The indicator function $\mathbf{1}_{[\cdot]}$ is used to index the selection of these functions, $\mathbf{1}_{[u=a]} = 1$ if u represents an atom else 0. In this way, the nodes with the same type of knowledge share parameters during message passing.

Apart from the above, we extend message passing by self-attention. We compute attention coefficients and normalize them with the softmax function to make coefficients easily comparable across different nodes. Following (Velickovic et al. 2018), the coefficients can be expressed as:

$$\alpha_{uv} = \frac{\exp\left(\mathrm{LeakyReLU}\left(\boldsymbol{a}^T\left[\boldsymbol{W}\boldsymbol{h}_u || \boldsymbol{W}\boldsymbol{h}_v\right]\right)\right)}{\sum_{k\in\mathcal{N}_u}\exp\left(\mathrm{LeakyReLU}\left(\boldsymbol{a}^T\left[\boldsymbol{W}\boldsymbol{h}_u || \boldsymbol{W}\boldsymbol{h}_k\right]\right)\right)}, \quad (1)$$

where $\cdot^T$ represents transposition and $||$ is the concatenation operation. The attention mechanism is implemented as a single-layer feedforward neural network, parametrized by a weight vector $\boldsymbol{a}$ and followed by a LeakyRELU activation.

Once obtained, the normalized attention coefficients are used to compute a linear combination of the features corresponding to them:

$$\mathrm{MSG}_0 = \alpha_{uv}\boldsymbol{W}_0\boldsymbol{h}^{k-1}(e_{uv}) \cdot \boldsymbol{h}^{k-1}(u), \quad (2)$$

where $\boldsymbol{W}_0$ denotes the weight matrix operating on incoming relations. This attentive message passing function allows for assigning different attention values to neighbor nodes, based on the intuition that different attributes have different importance to the atom.

Since the messages delivered by different neighbor atoms to the central atom also have various importance, atoms in the neighborhood follow a common process with different parameters:

$$\mathrm{MSG}_1 = \beta_{uv}\boldsymbol{W}_1\boldsymbol{h}^{k-1}(e_{uv}) \cdot \boldsymbol{h}^{k-1}(u), \quad (3)$$

Algorithm 1: KMPNN encoding algorithm.

Input: The augmented molecular graph $\mathcal{G}' = \{\mathcal{V}, \mathcal{E}\}$; message function $\mathrm{MSG}(\cdot)$; aggregate function AGG; update function U.

Output: Graph embedding $\boldsymbol{h}_{\mathcal{G}'}$.

$\boldsymbol{h}^0(v) \leftarrow x_v, \forall v \in \mathcal{V}$; $\boldsymbol{h}^0(e_{uv}) \leftarrow e_{uv}, \forall (u,v) \in \mathcal{E}$
for $k = 1, \ldots, K$ **do**
  for $v \in \mathcal{V}$ **do**
    $\boldsymbol{m}^k(v) \leftarrow \mathrm{AGG}(\{\mathrm{MSG}_{\mathbf{1}_{[u=a]}}(\boldsymbol{h}^{k-1}(e_{uv}), \boldsymbol{h}^{k-1}(u)), \forall u \in \mathcal{N}(v)\})$
    $\boldsymbol{h}^k(v) \leftarrow \mathrm{U}(\boldsymbol{h}^{k-1}(v), \boldsymbol{m}^k(v))$
  end for
end for
$\boldsymbol{h}_{\mathcal{G}'} \leftarrow \mathrm{READOUT}(\{\boldsymbol{h}^K(v), \forall v \in \mathcal{V}\})$

where β_{uv} is the attention coefficients between atoms, $\boldsymbol{W}_1$ is the weight matrix of incoming bonds.

In the message diffusion module, we collect the messages from their neighboring edges in message aggregation,

$$\boldsymbol{m}^k(v) = \sum_{k\in\mathcal{N}_u} \mathrm{MSG}(\boldsymbol{h}^{k-1}(e_{uv}), \boldsymbol{h}^{k-1}(u)), \quad (4)$$

and apply GRU as the update function.

$$\boldsymbol{h}^k(v) = \mathrm{GRU}(\boldsymbol{h}^{k-1}(v), \boldsymbol{m}^k(v)), \quad (5)$$

where GRU is the Gated Recurrent Unit introduced in (Cho et al. 2014). After K steps' iteration, a readout operator is applied to get a graph-level representation for the molecule:

$$\boldsymbol{h}_{\mathcal{G}'} = \mathrm{Set2set}(\boldsymbol{h}^K(v)), \quad (6)$$

where set2set (Vinyals, Bengio, and Kudlur 2016) is specifically designed to operate on sets and have more expressive power than simply summing the final node states.

GNN-based Encoder. There is no constraint of network architecture for $f(\cdot)$. We opt for simplicity and adopt the commonly used GCN (Kipf and Welling 2017) to obtain $\boldsymbol{h}_{\mathcal{G}} = f(\mathcal{G})$, which is the output after weighted sum and max pooling readout.

3.5 Contrastive Objective

Projection Head. A non-linear transformation $g(\cdot)$ named projection head maps both the original and augmented representations to another latent space where the contrastive loss is calculated, as advocated in (Chen et al. 2020). In KCL, a two-layer perceptron (MLP) is applied to obtain $\boldsymbol{z} = g(\boldsymbol{h}_{\mathcal{G}})$ and $\boldsymbol{z'} = g(\boldsymbol{h}_{\mathcal{G}'})$. Note that after pre-training is completed, we throw the projection head away and only use the encoders for downstream tasks.

Negative Mining. Instead of randomly choose graphs other than the anchor instance as negatives (You et al. 2020; Sun et al. 2021), we consider an additional hard negative mining scheme by treating molecules similar to the anchor instance as negatives. Specifically, we represent each molecule by its Morgan Fingerprints (Rogers and Hahn 2010), which perceive the presence of specific circular substructures around each atom in a molecule and encode it in a

fixed length binary vector. Then we calculate the molecular similarity through their Tanimoto coefficient (Bajusz, Rácz, and Héberger 2015):

$$s(\boldsymbol{e}_1, \boldsymbol{e}_2) = \frac{N_{12}}{N_1 + N_2 - N_{12}}, \tag{7}$$

where $\boldsymbol{e}_1, \boldsymbol{e}_2$ denotes the fingerprints, N_1, N_2 denotes the number of 1s in $\boldsymbol{e}_1, \boldsymbol{e}_2$ respectively, and N_{12} denotes the number of 1s in the intersection of $\boldsymbol{e}_1, \boldsymbol{e}_2$. In order to ensure all molecules have negative samples, instead of setting a fixed threshold, we sorted samples by similarity and selected a batch of most similar molecules as the negative samples.

Contrastive Loss. We augmented a minibatch of N similar molecular graphs with knowledge-guided graph augmentation, resulting in a total of $2N$ graphs. Following (You et al. 2020; Chen et al. 2020), given a positive pair, we treat the other $2(N-1)$ graphs within the same minibatch as hard negative samples. We utilize NT-Xent as our objective function like in (Hjelm et al. 2019; Chen et al. 2020; You et al. 2020; Carse, Carey, and McKenna 2021). The training objective for $(\mathcal{G}_i, \mathcal{G}_i')$ is defined as

$$\ell_i = -\log \frac{e^{\mathrm{sim}\left(\boldsymbol{z}_{\mathrm{i}}, \boldsymbol{z}_{\mathrm{i}}'\right)/\tau}}{\sum_{j=1}^{N}\left(e^{\mathrm{sim}\left(\boldsymbol{z}_{\mathrm{i}}, \boldsymbol{z}_{\mathrm{j}}'\right)/\tau} + e^{\mathrm{sim}\left(\boldsymbol{z}_{\mathrm{i}}', \boldsymbol{z}_{\mathrm{j}}\right)/\tau}\right)}, \tag{8}$$

where τ denotes the temperature parameter and $\mathrm{sim}(\boldsymbol{z}_1, \boldsymbol{z}_2)$ is the cosine similarity $\frac{\boldsymbol{z}_1^\top \boldsymbol{z}_2}{\|\boldsymbol{z}_1\| \cdot \|\boldsymbol{z}_2\|}$. The final loss is computed across all positive pairs in the minibatch.

4 Experiments

In this section, we conduct extensive experiments to examine the proposed method by answering the following questions:

Q1: How does KCL perform compared with state-of-the-art methods for molecular property prediction?

Q2: Does the knowledge-guided graph augmentation in Module 1 learns better representations than general augmentations?

Q3: How do knowledge feature initialization and graph encoders in Module 2 affect KCL?

Q4: How useful are the self-supervised contrastive learning and hard negative strategy in Module 3?

Q5: How can we interpret KCL(KMPNN) from a domain-specific perspective?

4.1 Experimental Setup

Pre-training Data Collection. We collect 250K unlabeled molecules sampled from the ZINC15 datasets (Sterling and Irwin 2015) to pre-train KCL.

Fine-tuning Tasks and Datasets. We use 8 benchmark datasets from the MoleculeNet (Wu et al. 2018a) to perform the experiments, which cover a wide range of molecular tasks such as quantum mechanics, physical chemistry, biophysics, and physiology. For each dataset, as suggested by (Wu et al. 2018a), we apply three independent runs on three random-seeded random splitting or scaffold splitting with a ratio for train/validation/test as 8:1:1. Details of datasets and dataset splitting are deferred to Appendix B.1.

Baselines. We adopt three types of baselines:

- *Supervised learning methods*: GCN (Kipf and Welling 2017) and Weave (Kearnes et al. 2016) are two types of graph convolutional methods. MPNN (Gilmer et al. 2017) and its variants DMPNN (Yang et al. 2019), CMPNN (Song et al. 2020), CoMPT (Chen et al. 2021) consider the edge features and strengthen the message interactions between bonds and atoms during message passing.
- *Pre-trained methods*: N-GRAM (Liu et al. 2019) conducts node embeddings by predicting the node attributes. Hu et al. (Hu et al. 2020) and GROVER (Rong et al. 2020) are pre-trained models incorporating both node-level and graph-level pretext tasks.
- *Graph contrastive learning baselines*: InfoGraph (Sun et al. 2020) maximizes the mutual information between nodes and graphs. MICRO-Graph (Subramonian 2021) is a motif-based contrastive method. GraphCL (You et al. 2020) constructs contrastive views of graph data via hand-picking ad-hoc augmentations. JOAO (You et al. 2021) automates the augmentation selection. MoCL (Sun et al. 2021) utilizes domain knowledge at two levels to assist representation learning.

Evaluation Protocol. The evaluation process follows two steps. We first pre-train the model and then evaluate the learned model on downstream tasks under two protocols.

- *Fine-tune protocol*: To achieve the full potential of our model, given graph embeddings output by the KCL encoder, we use an additional MLP to predict the property of the molecule. Fine-tune parameters in the encoders and the MLP.
- *Linear Protocol*: For comparison of our model and contrastive learning baselines, we fix the graph embeddings from the pre-trained model, and train a linear classifier.

Implementation details. We use the Adam optimizer with an initial learning rate of 0.0001 and batch size of 256. For pre-training models, the running epoch is fixed to 20. The temperature τ is set as 0.1. For downstream tasks, we use early stopping on the validation set. We apply the random search to obtain the best hyper-parameters based on the validation set. Our model is implemented with PyTorch (Paszke et al. 2019) and Deep Graph Library (Wang et al. 2019). We develop all codes on a Ubuntu Server with 4 GPUs (NVIDIA GeForce 1080Ti). More experimental details are available in Appendix C and D.

4.2 Performance Comparison (Q1 & Q2)

Performance under Fine-tune Protocol. We first examine whether the proposed KCL performs better than SOTA methods. Table 2 displays the complete results of supervised learning baselines and pre-trained methods, where the underlined cells indicate the previous SOTAs, and the cells with bold show the best result achieved by KCL. The Tox21, SIDER, and ClinTox are all multiple-task learning tasks, including totally 42 classification tasks. We also implemented two versions of our KCL model, the original molec-

Task	Classification (ROC-AUC)						Regression (RMSE)	
Dataset	BBBP	Tox21	ToxCast	SIDER	ClinTox	BACE	ESOL	FreeSolv
#Molecules	2039	7831	8575	1427	1478	1513	1128	642
#Tasks	1	12	617	27	2	1	1	1
GCN (Kipf and Welling 2017)	0.877	0.772	0.650	0.638	0.807	0.854	1.068	2.900
Weave (Kearnes et al. 2016)	0.837	0.741	0.678	0.621	0.823	0.791	1.158	2.398
MPNN (Gilmer et al. 2017)	0.913	0.808	0.691	0.641	0.879	0.815	1.167	2.185
DMPNN (Yang et al. 2019)	0.919	0.826	0.718	0.632	0.897	0.852	0.980	2.177
CMPNN (Song et al. 2020)	0.927	0.806	0.738	0.636	0.902	0.869	0.798	0.956
CoMPT (Chen et al. 2021)	0.938	0.809	0.740	0.634	0.934	0.871	0.774	1.855
N-GRAM (Liu et al. 2019)	0.912	0.769	-	0.632	0.870	0.876	1.100	2.512
Hu et al. (Hu et al. 2020)	0.915	0.811	0.714	0.614	0.762	0.851	-	-
GROVER (Rong et al. 2020)	0.940	0.831	0.737	0.658	0.944	0.894	0.831	1.544
KCL(GCN)	0.956	0.856	**0.757**	0.666	0.945	**0.934**	**0.582**	0.854
KCL(KMPNN)	**0.961**	**0.859**	0.740	**0.671**	**0.958**	0.924	0.732	**0.795**

Table 2: The property prediction performance (lower is better for regression) of KCL under the fine-tune protocol, compared with supervised learning (first group) and pre-training methods (second group) baselines on 8 datasets.

Dataset	BBBP	Tox21	ToxCast	SIDER	ClinTox	BACE
Node	0.843	0.728	0.633	0.577	0.635	0.746
Edge	0.833	0.715	0.619	0.605	0.630	0.657
Subgraph	0.815	0.727	0.625	0.583	0.603	0.629
Attribute	0.826	0.726	0.623	0.621	0.671	0.796
InfoGraph	0.611	0.615	0.562	0.502	0.458	0.594
MICRO	0.830	0.718	0.595	0.573	0.735	0.708
GraphCL	0.697	0.739	0.624	0.605	0.760	0.755
JOAO	0.714	0.750	0.632	0.605	0.813	0.773
MoCL	0.905	0.768	0.653	0.628	0.750	0.845
KCL(G)	**0.929**	0.821	0.696	0.620	**0.909**	**0.902**
KCL(K)	0.927	**0.825**	**0.709**	**0.659**	0.898	0.860

Table 3: The performance of KCL under the linear protocol on 6 datasets, compared with contrastive learning baselines. The metric is ROC-AUC.

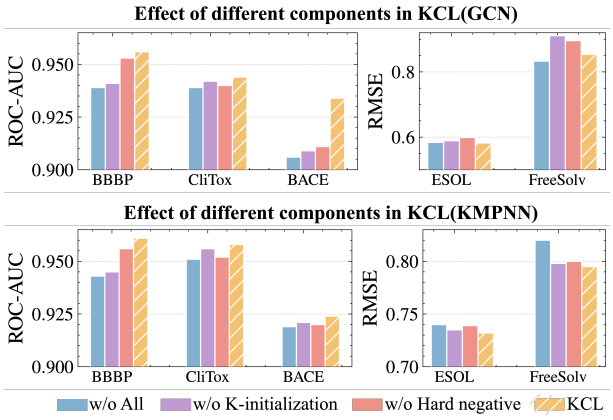

Figure 3: Performance of KCL with different settings under the fine-tune protocol (lower is better for regression).

ular graph with GCN encoder and the augmented molecular graph with KMPNN as the encoder.

Table 2 offers the following observations: (1) KCL consistently achieves the best performance on all datasets with large margins. The overall relative improvement is 7.1% on all datasets (2.6% on classification tasks and 20.4% on regression tasks)[2]. This notable performance improvement suggests the effectiveness of KCL for molecular property prediction tasks. (2) In the small dataset FreeSolv with only 642 labeled molecules, KCL gains a 16.8% improvement over SOTA baselines. This confirms the strength of KCL since it can significantly help with tasks with very limited label information.

Performance under Linear Protocol. We next study whether the knowledge-guided graph augmentation in Module 1 helps learn better molecular representations. Table 3 shows the comparison results of different augmentation (node dropping, edge perturbation, subgraph extraction and attribute masking) and contrastive learning methods. To be consistent with prior works and make the comparisons fair, we use the linear protocol, which is exactly what baselines have done, to evaluate the performance on classification datasets. Results on regression tasks are deferred to Appendix E.1.

Both versions of KCL produce better results compared to alternative graph augmentation methods (the first group in Table 3). This verifies our assumption that knowledge-guided graph augmentation does not violate the biological semantic in molecules and thus works better than other augmentations. Moreover, KCL gains a 7.0% improvement over the previous best contrastive learning methods (the second group), which confirms that better representations of molecular graphs could be obtained by incorporating fundamental chemical domain knowledge and capturing microscopic associations between atoms.

4.3 Ablation Study (Q3 & Q4)

We then conducted ablation studies to investigate components in Module 1 and 2 that influence the performance of the proposed KCL framework.

[2]We use relative improvement to provide the unified descriptions.

Task	Classification	Regression
GCN(No contrast)	0.766	1.984
KMPNN(No contrast)	0.806	1.531
KCL(GIN)	0.849	0.718
KCL(GAT)	0.850	0.724
KCL(GCN)	0.852	0.718
KCL(RGCN)	0.831	1.008
KCL(MPNN)	0.833	0.927
KCL(KMPNN)	0.852	0.765

Table 4: Ablation results under the fine-tune protocol. Each value represents the average result of the task, and the underline marks the best in the group.

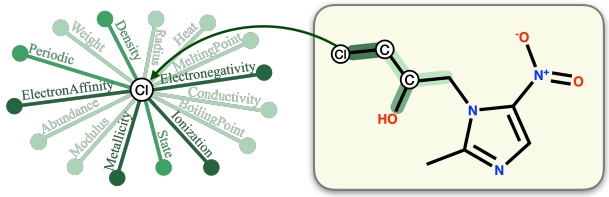

Figure 4: An attention visualization example of different types of neighbors (attributes and atoms) in the BBBP dataset. The attention weights assigned for bonds connected to the two C atoms are visualized on the right. The darker the color, the higher the attention.

As shown in Figure 3, KCL with knowledge feature initialization and hard negative mining scheme (bar in yellow) shows the best performance among all architectures. Models with random initialization and random negative sampling denoted by "w/o ALL" almost always perform the worst. Excluding any of these two components can easily result in a decrease in performance. This illustrates that both knowledge feature initialization and hard negative mining strategy are necessary for KCL, because the former captures the structural triple information, while the latter guides the encoders to generate more discriminative representations.

Since our graph encoders are pluggable, we replaced both GCN, KMPNN with other architectures to explore the impact of graph encoders. The results in Table 4 demonstrate that applying different encoders (e.g., GIN (Xu et al. 2019), GAT (Velickovic et al. 2018)) on original molecular graphs has no significant impact on performance. In addition, we ignore the different types of nodes and edges in augmented graphs and replace KMPNN with previous heterogeneous graph neural network (RGCN (Schlichtkrull et al. 2018)) and general message passing framework (MPNN (Gilmer et al. 2017)). The comparisons reveal that KMPNN has better expressive power by providing heterogeneous attentive message passing for different types of knowledge on the augmented molecular graphs. The specific values are deferred to Appendix E.2 and E.3.

To investigate the contribution of the self-supervision strategy, we compare the performances between KCL with and without contrastive learning under the fine-tune protocol (the counterpart under linear protocol is deferred to Appendix E.4). We report the comparison results in Table 4. The self-supervised contrastive learning leads to a performance boost with an average increase of 8.5% on classification and 56.9% on regression over the model without contrastive learning. This confirms that contrastive learning can learn better representations by narrowing the distance between the structural view and the knowledgeable view in the latent space, and enhance the prediction performance of downstream tasks.

4.4 Chemical Interpretability Analysis (Q5)

Finally, we explore the interpretability of our model by visualizing the attention of each edge in a molecule. Specifically, we extracted and normalized the atom's attention weights to their neighbors from the last layer of KCL(KMPNN).

Figure 4 illustrates an example in the BBBP dataset (Martins et al. 2012). BBBP involves records of whether a compound carries the permeability property of penetrating the blood-brain barrier. As shown in the left part of the figure, atoms tend to assign more attention to their electron affinity, electronegativity, metallicity, and ionization. These attributes are closely related to atoms' ability to lose electrons. The strength of the atom's ability to gain or lose electrons will largely affect the polarity of the molecule, thereby affecting its permeability. In addition, more lively atomic neighbors are easier to be noticed, as illustrated on the right side of Figure 4. The element Cl has relatively higher electronegativity, so it has a stronger ability to obtain electrons. Also, the hydroxyl group promotes hydrophilicity and thus is assigned higher attention. Another interesting observation is that fine-grained attributes (e.g., weight, radius) receive less attention than coarse-grained attributes (e.g., electron affinity, electronegativity, metallicity, and ionization). It is because coarse-grained attributes are more abstract and informative than fine-grained attributes, and therefore contain richer domain knowledge. This is in line with hierarchical machine learning where coarse-grained features at higher levels can be seen as a summary of fine-grained features in terms of target prediction. More examples and discussions on other datasets are in Appendix E.5.

5 Conclusion and Future Work

This paper aims to incorporate fundamental domain knowledge into molecular graph representation learning. We construct Element KG to build microscopic connections between elements, and propose to utilize knowledge in the KCL framework to enhance molecular graph contrastive learning. We demonstrate the effectiveness of KCL under both fine-tune and linear protocols, and experiments show that KCL excels previous methods with better interpretation and representation capability.

In the future, we intend to extend our work in several aspects. First, we would introduce different granularity of domain knowledge to enrich Chemical Element KG. Also, we will improve the current KG with more description logics defined in OWL2, such as more object properties and axioms. Third, we will open-source Chemical Element KG, continue to improve its quality and expand its scale.

Acknowledgement

We want to express gratitude to the anonymous reviewers for their hard work and kind comments. This work is funded by NSFCU19B2027/91846204, national key research program 2018YFB1402800.

References

2017. Tox21 challenge. *https://tripod.nih.gov/tox21/challenge/*.

Bajusz, D.; Rácz, A.; and Héberger, K. 2015. Why is Tanimoto index an appropriate choice for fingerprint-based similarity calculations? *J. Cheminformatics*, 7: 20:1–20:13.

Bemis, G. W.; and Murcko, M. A. 1996. The properties of known drugs. 1. Molecular frameworks. *Journal of medicinal chemistry*, 39(15): 2887–2893.

Carse, J.; Carey, F. A.; and McKenna, S. J. 2021. Unsupervised Representation Learning From Pathology Images With Multi-Directional Contrastive Predictive Coding. In *ISBI*, 1254–1258. IEEE.

Chen, J.; Zheng, S.; Song, Y.; Rao, J.; and Yang, Y. 2021. Learning Attributed Graph Representations with Communicative Message Passing Transformer. *CoRR*, abs/2107.08773.

Chen, T.; Kornblith, S.; Norouzi, M.; and Hinton, G. E. 2020. A Simple Framework for Contrastive Learning of Visual Representations. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, 1597–1607. PMLR.

Cho, K.; van Merrienboer, B.; Bahdanau, D.; and Bengio, Y. 2014. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. In *SSST@EMNLP*, 103–111. Association for Computational Linguistics.

Delaney, J. S. 2004. ESOL: estimating aqueous solubility directly from molecular structure. *Journal of chemical information and computer sciences*, 44(3): 1000–1005.

Delmas, M.; Filangi, O.; Paulhe, N.; Vinson, F.; Duperier, C.; Garrier, W.; Saunier, P.-E.; Pitarch, Y.; Jourdan, F.; Giacomoni, F.; et al. 2021. Building a Knowledge Graph from public databases and scientific literature to extract associations between chemicals and diseases. *bioRxiv*.

Duvenaud, D.; Maclaurin, D.; Aguilera-Iparraguirre, J.; Gómez-Bombarelli, R.; Hirzel, T.; Aspuru-Guzik, A.; and Adams, R. P. 2015. Convolutional Networks on Graphs for Learning Molecular Fingerprints. In *NIPS*, 2224–2232.

Fang, Y.; Yang, H.; Zhuang, X.; Shao, X.; Fan, X.; and Chen, H. 2021. Knowledge-aware Contrastive Molecular Graph Learning. *CoRR*, abs/2103.13047.

Gayvert, K. M.; Madhukar, N. S.; and Elemento, O. 2016. A data-driven approach to predicting successes and failures of clinical trials. *Cell chemical biology*, 23(10): 1294–1301.

Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; and Dahl, G. E. 2017. Neural Message Passing for Quantum Chemistry. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, 1263–1272. PMLR.

Hassani, K.; and Ahmadi, A. H. K. 2020. Contrastive Multi-View Representation Learning on Graphs. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, 4116–4126. PMLR.

Hjelm, R. D.; Fedorov, A.; Lavoie-Marchildon, S.; Grewal, K.; Bachman, P.; Trischler, A.; and Bengio, Y. 2019. Learning deep representations by mutual information estimation and maximization. In *ICLR*. OpenReview.net.

Hu, W.; Liu, B.; Gomes, J.; Zitnik, M.; Liang, P.; Pande, V. S.; and Leskovec, J. 2020. Strategies for Pre-training Graph Neural Networks. In *ICLR*. OpenReview.net.

Jastrzebski, S.; Lesniak, D.; Czarnecki, W. M.; and . 2016. Learning to SMILE(S). *CoRR*, abs/1602.06289.

Kearnes, S. M.; McCloskey, K.; Berndl, M.; Pande, V. S.; and Riley, P. 2016. Molecular graph convolutions: moving beyond fingerprints. *J. Comput. Aided Mol. Des.*, 30(8): 595–608.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR (Poster)*. OpenReview.net.

Kuhn, M.; Letunic, I.; Jensen, L. J.; and Bork, P. 2016. The SIDER database of drugs and side effects. *Nucleic Acids Res.*, 44(Database-Issue): 1075–1079.

Lin, X.; Quan, Z.; Wang, Z.; Ma, T.; and Zeng, X. 2020. KGNN: Knowledge Graph Neural Network for Drug-Drug Interaction Prediction. In *IJCAI*, 2739–2745. ijcai.org.

Liu, S.; Demirel, M. F.; Liang, Y.; and . 2019. N-Gram Graph: Simple Unsupervised Representation for Graphs, with Applications to Molecules. In *NeurIPS*, 8464–8476.

Martins, I. F.; Teixeira, A. L.; Pinheiro, L.; and Falcão, A. O. 2012. A Bayesian Approach to *in Silico* Blood-Brain Barrier Penetration Modeling. *J. Chem. Inf. Model.*, 52(6): 1686–1697.

Mobley, D. L.; and Guthrie, J. P. 2014. FreeSolv: a database of experimental and calculated hydration free energies, with input files. *Journal of computer-aided molecular design*, 28(7): 711–720.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Kopf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; and Chintala, S. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Wallach, H.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 32*, 8024–8035. Curran Associates, Inc.

Richard, A. M.; Judson, R. S.; Houck, K. A.; Grulke, C. M.; Volarath, P.; Thillainadarajah, I.; Yang, C.; Rathman, J.; Martin, M. T.; Wambaugh, J. F.; et al. 2016. ToxCast chemical landscape: paving the road to 21st century toxicology. *Chemical research in toxicology*, 29(8): 1225–1251.

Rogers, D.; and Hahn, M. 2010. Extended-Connectivity Fingerprints. *J. Chem. Inf. Model.*, 50(5): 742–754.

Rong, Y.; Bian, Y.; Xu, T.; Xie, W.; Wei, Y.; Huang, W.; and Huang, J. 2020. Self-Supervised Graph Transformer on Large-Scale Molecular Data. In *NeurIPS*.

Schlichtkrull, M. S.; Kipf, T. N.; Bloem, P.; van den Berg, R.; Titov, I.; and Welling, M. 2018. Modeling Relational

Data with Graph Convolutional Networks. In *ESWC*, volume 10843 of *Lecture Notes in Computer Science*, 593–607. Springer.

Shang, J.; Ma, T.; Xiao, C.; and Sun, J. 2019. Pre-training of Graph Augmented Transformers for Medication Recommendation. In *IJCAI*, 5953–5959. ijcai.org.

Song, Y.; Zheng, S.; Niu, Z.; Fu, Z.; Lu, Y.; and Yang, Y. 2020. Communicative Representation Learning on Attributed Molecular Graphs. In *IJCAI*, 2831–2838. ijcai.org.

Sterling, T.; and Irwin, J. J. 2015. ZINC 15 - Ligand Discovery for Everyone. *J. Chem. Inf. Model.*, 55(11): 2324–2337.

Subramanian, G.; Ramsundar, B.; Pande, V.; and Denny, R. A. 2016. Computational modeling of β-secretase 1 (BACE-1) inhibitors using ligand based approaches. *Journal of chemical information and modeling*, 56(10): 1936–1949.

Subramonian, A. 2021. MOTIF-Driven Contrastive Learning of Graph Representations. In *AAAI*, 15980–15981. AAAI Press.

Sun, F.; Hoffmann, J.; Verma, V.; and Tang, J. 2020. InfoGraph: Unsupervised and Semi-supervised Graph-Level Representation Learning via Mutual Information Maximization. In *ICLR*. OpenReview.net.

Sun, K.; Lin, Z.; and Zhu, Z. 2020. Multi-Stage Self-Supervised Learning for Graph Convolutional Networks on Graphs with Few Labeled Nodes. In *AAAI*, 5892–5899. AAAI Press.

Sun, M.; Xing, J.; Wang, H.; Chen, B.; and Zhou, J. 2021. MoCL: Contrastive Learning on Molecular Graphs with Multi-level Domain Knowledge. *CoRR*, abs/2106.04509.

Sun, Z.; Deng, Z.; Nie, J.; and Tang, J. 2019. RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. In *ICLR (Poster)*. OpenReview.net.

Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. In *ICLR (Poster)*. OpenReview.net.

Velickovic, P.; Fedus, W.; Hamilton, W. L.; Liò, P.; Bengio, Y.; and Hjelm, R. D. 2019. Deep Graph Infomax. In *ICLR (Poster)*. OpenReview.net.

Vinyals, O.; Bengio, S.; and Kudlur, M. 2016. Order Matters: Sequence to sequence for sets. In *ICLR (Poster)*.

Wang, M.; Zheng, D.; Ye, Z.; Gan, Q.; Li, M.; Song, X.; Zhou, J.; Ma, C.; Yu, L.; Gai, Y.; Xiao, T.; He, T.; Karypis, G.; Li, J.; and Zhang, Z. 2019. Deep Graph Library: A Graph-Centric, Highly-Performant Package for Graph Neural Networks. *arXiv preprint arXiv:1909.01315*.

Wu, Z.; Ramsundar, B.; Feinberg, E. N.; Gomes, J.; Geniesse, C.; Pappu, A. S.; Leswing, K.; and Pande, V. 2018a. MoleculeNet: a benchmark for molecular machine learning. *Chemical science*, 9(2): 513–530.

Wu, Z.; Xiong, Y.; Yu, S. X.; and Lin, D. 2018b. Unsupervised Feature Learning via Non-Parametric Instance Discrimination. In *CVPR*, 3733–3742. IEEE Computer Society.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2019. How Powerful are Graph Neural Networks? In *ICLR*. OpenReview.net.

Xu, Z.; Wang, S.; Zhu, F.; and Huang, J. 2017. Seq2seq Fingerprint: An Unsupervised Deep Molecular Embedding for Drug Discovery. In *BCB*, 285–294. ACM.

Yang, K.; Swanson, K.; Jin, W.; Coley, C. W.; Eiden, P.; Gao, H.; Guzman-Perez, A.; Hopper, T.; Kelley, B.; Mathea, M.; Palmer, A.; Settels, V.; Jaakkola, T. S.; Jensen, K. F.; and Barzilay, R. 2019. Analyzing Learned Molecular Representations for Property Prediction. *J. Chem. Inf. Model.*, 59(8): 3370–3388.

Yasunaga, M.; and Liang, P. 2020. Graph-based, Self-Supervised Program Repair from Diagnostic Feedback. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, 10799–10808. PMLR.

You, Y.; Chen, T.; Shen, Y.; and Wang, Z. 2021. Graph Contrastive Learning Automated. In *ICML*, volume 139 of *Proceedings of Machine Learning Research*, 12121–12132. PMLR.

You, Y.; Chen, T.; Sui, Y.; Chen, T.; Wang, Z.; and Shen, Y. 2020. Graph Contrastive Learning with Augmentations. In *NeurIPS*.

Appendix

A Details of KCL

A.1 Feature Initialization

The feature initialization contains two parts: 1) Atom / bond feature initialization. We randomly initialize different vectors for each type of atoms and bonds, with dimensions 128 and 64 respectively. We try a variety of different combinations of dimensions, and find that the impact on performance was not significant. Therefore, here we select the dimensions when the performance is optimal. 2) Attribute / relation feature initialization. We use RotatE (Sun et al. 2019) which defines each relation as a rotation in the complex vector space to train Chemical Element KG. Its score function is formulated as follows:

$$f(\mathbf{h}, \mathbf{r}, \mathbf{t}) = \|\mathbf{h} \circ \mathbf{r} - \mathbf{t}\|, \tag{9}$$

where $\mathbf{h}, \mathbf{r}, \mathbf{t}$ denote the embedding of head, relation and tail respectively, and $\circ$ is the Hadmard product. The dimensions of attributes and relations are also 128 and 64.

A.2 Architecture of KMPNN

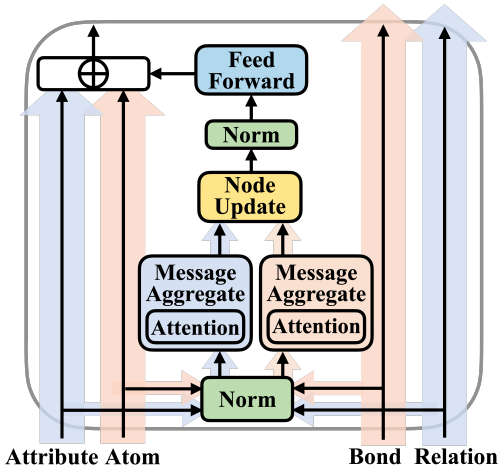

Figure 5: Architecture of KMPNN.

Figure 5 illustrates the architecture of KMPNN. We applied heterogeneous message passing (arrows in different colors) to neighbors representing atoms and attributes in the augmented molecular graph, respectively. In this way, the node information corresponding to the same type of knowledge share parameters during message passing. Moreover, we extend the message passing process by the self-attention mechanism. We use different parameters to compute attention coefficients between atoms and attributes, atoms and atoms. This attentive message passing allows assigning different attention values to neighbor nodes.

A.3 GNN-based Encoder

As for the original molecular graph G, we apply a Graph Neural Network (GNN)-based encoder to extract its graph-level representation. Formally, the message passing operation in iteration k can be formulated as:

$$\boldsymbol{h}^k(v) = \mathrm{U}(\boldsymbol{h}^{k-1}(v), \textsc{Agg}(\{\boldsymbol{h}^{k-1}(u)), \forall u \in \mathcal{N}_v\})), \tag{10}$$

there are several ways of choosing $\textsc{Agg}$, such as mean, sum, and max pooling. Then we convert them to graph embedding with a readout function. Note that no constraints are imposed on the GNN architecture, and experiments have also proved that different architectures have little impact on performance. So here we adopt the most classic GCN (Kipf and Welling 2017) architecture, combined with weighted sum and max pooling to compute the final readout.

B Details about Experimental Setup

B.1 Dataset Description

Table 5 summarizes information of benchmark datasets, including task type, dataset size, split type, and evaluation metrics. We used 6 binary graph classification datasets and 2 binary graph regression datasets. #Tasks means the number of binary prediction tasks in each dataset. The details of each dataset are listed below (Wu et al. 2018a):

Molecular Classification Datasets.

- *BBBP* (Martins et al. 2012) involves records of whether a compound carries the permeability property of penetrating the blood-brain barrier.
- *SIDER* (Kuhn et al. 2016) records marketed drugs along with their adverse drug reactions, also known as the Side Effect Resource.
- *ClinTox* (Gayvert, Madhukar, and Elemento 2016) compares drugs approved through FDA and drugs eliminated due to toxicity during clinical trials.
- *BACE* (Subramanian et al. 2016) is collected for recording compounds that could act as the inhibitors of human β-secretase 1 (BACE-1) in the past few years.
- *Tox21* (tox 2017) is a public database measuring the toxicity of compounds, which has been used in the 2014 Tox21 Data Challenge.
- *ToxCast* (Richard et al. 2016) contains multiple toxicity labels over thousands of compounds by running high-throughput screening tests on thousands of chemicals.

Molecular Regression Datasets.

- *ESOL* is a small dataset documenting the solubility of compounds (Delaney 2004).
- *FreeSolv* (Mobley and Guthrie 2014) is selected from the Free Solvation Database, which contains the hydration free energy of small molecules in water from both experiments and alchemical free energy calculations.

Dataset Splitting. As shown in Table 5, we apply the scaffold splitting (Bemis and Murcko 1996) and random splitting as recommended in (Wu et al. 2018a) for all tasks on all datasets. Scaffold splitting splits the molecules with distinct two-dimensional structural frameworks into different subsets. It is a more challenging but practical setting since the test molecular can be structurally different from the training set. Here we apply these splitting methods to construct the train/validation/test sets.

B.2 Baselines

We adopt three types of baselines. The details of each baseline are listed below:

Type	Category	Dataset	# Tasks	# Compounds	Split	Metric
Classification	Biophysics	BBBP	1	2039	Scaffold	ROC-AUC
	Physiology	SIDER	27	1427	Random	ROC-AUC
		ClinTox	2	1478	Random	ROC-AUC
		BACE	1	1513	Scaffold	ROC-AUC
		Tox21	12	7831	Random	ROC-AUC
		ToxCast	617	8575	Random	ROC-AUC
Regression	Physical chemistry	FreeSolv	1	642	Random	RMSE
		ESOL	1	1128	Random	RMSE

Table 5: Dataset information.

hyper-parameter	Description	Value
epoch	the number of training epochs.	20
τ	the temperature parameter.	0.1
batch_size	the input batch_size.	256
lr	the learning rate.	0.0001
GCN_layers	the number of layers of GCN.	2
GCN_node_hidden	the node hidden size for GCN.	64
KMPNN_step	the step of message passing for KMPNN.	6
KMPNN_node_hidden	the node hidden size for KMPNN.	64
KMPNN_edge_hidden	the edge hidden size for KMPNN.	64
node_out	the node out size for GCN/KMPNN readout.	64
edge_out	the edge out size for GCN/KMPNN readout.	64

Table 6: The pre-train hyper-parameters.

Supervised Learning Methods.

- *GCN* (Kipf and Welling 2017) is a convolutional method that focuses on learning the relationship with the nearest neighbor node.
- *Weave* (Kearnes et al. 2016) transformed feature vectors using pair features with distant atoms in addition to atom features focused only on atoms.
- *MPNN* (Gilmer et al. 2017) utilized features from nodes and edges, and summarize it into a framework.
- *DMPNN* (Yang et al. 2019) treated the molecular graph as an edge-oriented directed structure, avoiding the information redundancy during iterations.
- *CMPNN* (Song et al. 2020) introduced the node-edge interaction module to leverage the node and edge attributes during message passing.
- *CoMPT* (Chen et al. 2021) invoked a communicative message-passing paradigm based on Transformer.

Pre-trained methods.

- *N-GRAM* (Liu et al. 2019) exploited the idea of N-gram in NLP and conducted vertices embedding by predicting the vertices attributes.
- *Hu et al.* (Hu et al. 2020)designed two pre-training tasks, i.e., predicting neighborhood context and node attributes, to learn meaningful node representations, then using graph-level multi-task pre-training to refine graph representations.
- *GROVER* (Rong et al. 2020) incorporated a Transformer-style architecture and learned node embeddings by predicting contextual properties and motif information.

Graph Contrastive Learning Baselines.

- *InfoGraph* (Sun et al. 2020) maximized the mutual information between nodes and graphs.
- *MICRO-Graph* (Subramonian 2021) is a motif-based contrastive method.
- *GraphCL* (You et al. 2020) constructed contrastive views of graph data via hand-picking augmentation.
- *JOAO* (You et al. 2021) automated the augmentation selection in contrastive learning.
- *MoCL* (Sun et al. 2021) utilized domain knowledge at both local- and global-level to assist representation learning.

C Implementation and Pre-training Details

We use PyTorch (Paszke et al. 2019) and Deep Graph Library (Wang et al. 2019) to implement KCL. Table 6 demonstrates all the hyper-parameters of the pre-training model. We develop all codes on a Ubuntu Server with 4 GPUs (NVIDIA GeForce 1080Ti).

D Downstream Details

For downstream tasks, we use early stopping on the validation set. We try different hyper-parameter combinations via random search to find the best results. Table 7 demonstrates all the hyper-parameters of the fine-tuning model. All fine-tuning tasks are run on a single GPU.

hyper-parameter	Description	Range
batch_size	the input batch_size.	64,128,256
lr	the learning rate.	0.0001∼0.1
hidden_size	the hidden size for predictor.	32,64,128
patience	the patience for early stopping.	20

Table 7: The downstream hyper-parameters.

E Additional Experimental Results

E.1 Regression Results under Linear Protocol

To be consistent with prior contrastive learning works and make the comparisons fair, we use the linear protocol to evaluate the performance only on classification datasets in main content. Table 8 depicts the additional results of the performance on regression tasks. In order to understand it intuitively, we also listed results under the fine-tune protocol.

Protocol	Method	ESOL	FreeSolv
Linear	KCL(GCN)	0.709	1.085
	KCL(KMPNN)	0.867	1.093
Fine-tune	KCL(GCN)	**0.582**	0.854
	KCL(KMPNN)	0.736	**0.795**

Table 8: The performance of KCL under the linear protocol and fine-tune protocol.

E.2 Effect of Different Settings

We investigate components that influence the performance of the proposed KCL. Table 9 and Table 10 report specific values of the fine-tuning results in Figure 3. KCL with knowledge feature initialization and negative sampling scheme shows the best performance among all architectures. Models without all of these two components almost always perform the worst. Excluding any of these two components results in a decrease in performance.

	KCL(GCN)			
Dataset	w/oALL	w/oInit	w/oNS	ALL
BBBP	0.939	0.941	0.953	**0.956**
Tox21	0.846	0.853	0.852	**0.856**
ToxCast	0.750	0.751	0.753	**0.757**
SIDER	0.650	0.663	0.665	**0.666**
CliTox	0.939	0.942	0.940	**0.945**
BACE	0.906	0.909	0.911	**0.934**
ESOL	0.584	0.589	0.599	**0.582**
FreeSolv	**0.833**	0.911	0.896	0.854
Ave(Cls)	0.840	0.843	0.845	**0.852**
Ave(Reg)	**0.709**	0.750	0.748	0.718

Table 9: Ablation results on molecular graphs.

E.3 Effect of Different Encoders

Since our graph encoder module is pluggable, we explore the impact of different encoders. We replace GCN, KMPNN with GIN and GAT, RGCN and MPNN, respectively. Table 11 shows the specific values of performance. The results demonstrate that applying different GNN-based encoders on original molecular graphs has no significant impact on performance. Furthermore, KMPNN has a better expressive power on the augmented molecular graphs than the previous heterogeneous GNN and general message passing framework.

	KCL(KMPNN)			
Dataset	w/oA	w/oInit	w/oNS	ALL
BBBP	0.943	0.945	0.956	**0.961**
Tox21	0.840	0.853	0.856	**0.859**
ToxCast	0.737	0.735	0.739	**0.740**
SIDER	0.650	0.659	0.661	**0.671**
CliTox	0.951	0.956	0.952	**0.958**
BACE	0.919	0.921	0.920	**0.924**
ESOL	0.740	0.735	0.739	**0.732**
FreeSolv	0.820	0.798	0.800	**0.795**
Ave(Cls)	0.840	0.845	0.847	**0.852**
Ave(Reg)	0.780	0.765	0.770	**0.764**

Table 10: Ablation results on the augmented molecular graphs.

E.4 Effect of Contrastive Learning

We investigate the contribution of contrastive learning strategy. Table 12 depicts the additional results of the comparison of KCL(KMPNN) and KMPNN without contrastive learning under fine-tune and linear protocols.

Similar to performance under the fine-tune protocol, contrastive learning leads to a performance boost with an average of 0.7% on classification and 76.5% on regression over the model without contrastive learning. This reinforces our claim that contrastive learning can incorporate domain knowledge into molecular representations and enhance the prediction performance of downstream tasks.

E.5 Chemical Interpretability Analysis

We visualize the attention of each edge in a molecule to explore the interpretability of KMPNN. Figure 6 illustrates another example in the BBBP dataset. Similar to Figure 4, atoms tend to assign more attention to their electron affinity, electronegativity, metalicity, and ionization, which are closely related to atoms' ability to lose electrons. Also, more lively atomic neighbors are easier to be noticed, as shown on the right side of Figure 6.

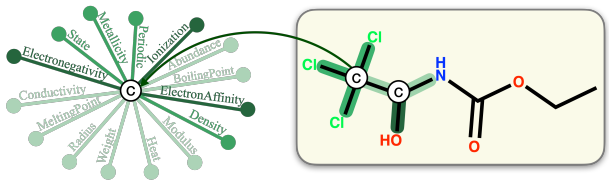

Figure 6: Another example in the BBBP dataset.

Task	Classification						Regression	
Dataset	BBBP	Tox21	ToxCast	SIDER	ClinTox	BACE	ESOL	FreeSolv
KCL(GIN)	0.954	0.854	0.748	0.660	**0.945**	0.932	0.580	0.856
KCL(GAT)	0.956	**0.857**	0.750	0.663	0.942	0.930	0.588	0.860
KCL(GCN)	**0.956**	0.856	**0.757**	**0.666**	**0.945**	**0.934**	**0.582**	**0.854**
KCL(R-GCN)	0.936	0.830	0.735	0.637	0.948	0.898	0.780	1.236
KCL(MPNN)	0.940	0.835	0.738	0.640	0.950	0.895	0.743	1.111
KCL(KMPNN)	**0.961**	**0.859**	**0.740**	**0.671**	**0.958**	**0.924**	**0.732**	**0.795**

Table 11: Results comparison with different graph encoders.

	Fine-tune Protocol			Linear Protocol		
	KCL	KMPNN	Abs.Imp.	KCL	KMPNN	Abs.Imp.
BBBP	**0.961**	0.915	+0.046	**0.927**	0.915	+0.012
Tox21	**0.859**	0.804	+0.055	**0.825**	0.804	+0.021
ToxCast	**0.740**	0.725	+0.015	0.709	**0.725**	-0.016
SIDER	**0.671**	0.645	+0.026	**0.659**	0.645	+0.014
ClinTox	**0.958**	0.892	+0.066	**0.898**	0.892	+0.006
BACE	**0.924**	0.856	+0.068	**0.860**	0.856	+0.004
ESOL	**0.736**	0.895	+0.159	**0.736**	0.895	+0.159
FreeSolv	**0.795**	2.167	+1.372	**0.795**	2.167	+1.372
Ave(Cls)	**0.852**	0.806	+0.046	**0.813**	0.806	+0.007
Ave(Reg)	**0.765**	1.531	+0.766	**0.766**	1.531	+ 0.765

Table 12: Results comparison between KCL(KMPNN) and KMPNN without contrastive learning.

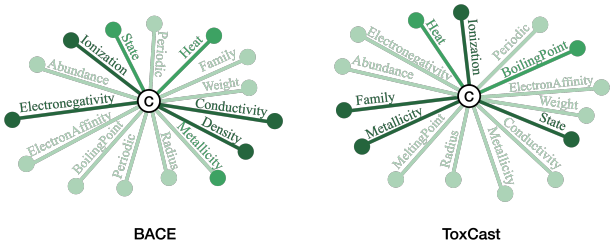

Figure 7: Attention visualization examples of attributes in the BACE and ToxCast datasets.

Figure 7 shows attention visualization examples on BACE and ToxCast datasets. Most atoms have similar characteristics, here we show two examples of them. Consistent with the conclusion obtained on the BBBP dataset, we observe that fine-grained attributes (e.g., weight, radius) receive less attention than coarse-grained attributes (e.g., electronegativity, conductivity, density, Ionization, metallicity, state, family). This confirms our claim that coarse-grained attributes are more abstract and informative than fine-grained attributes, and therefore contain richer domain knowledge.