Long-range Neural Atom Learning for Molecular Graphs

Xuan Li[1*] **Zhanke Zhou**[1*] **Jiangchao Yao**[2,3] **Yu Rong**[4] **Lu Zhang**[1] **Bo Han**[1]

[1]Hong Kong Baptist University [2]CMIC, Shanghai Jiao Tong University
[3]Shanghai AI Laboratory [4]Tencent AI Lab

{csxuanli, cszkzhou, ericluzhang, bhanml}@comp.hkbu.edu.hk
sunarker@sjtu.edu.cn yu.rong@hotmail.com

Abstract

Graph Neural Networks (GNNs) have been widely adopted for drug discovery with molecular graphs. Nevertheless, current GNNs are mainly good at leveraging short-range interactions (SRI) but struggle to capture long-range interactions (LRI), both of which are crucial for determining molecular properties. To tackle this issue, we propose a method that implicitly projects all original atoms into a few *Neural Atoms*, which abstracts the collective information of atomic groups within a molecule. Specifically, we explicitly exchange the information among neural atoms and project them back to the atoms' representations as an enhancement. With this mechanism, neural atoms establish the communication channels among distant nodes, effectively reducing the interaction scope of arbitrary node pairs into a single hop. To provide an inspection of our method from a physical perspective, we reveal its connection with the traditional LRI calculation method, Ewald Summation. We conduct extensive experiments on three long-range graph benchmarks, covering both graph-level and link-level tasks on molecular graphs. We empirically justify that our method can be equipped with an arbitrary GNN and help to capture LRI.

1 Introduction

Graph Neural Networks (GNNs) show promising ability of modeling complex and irregular interactions [39; 30; 44]. In particular, GNNs attract growing interest in accelerating drug discovery due to the accurate prediction of the molecular properties [25; 15; 35; 21; 45].

The basic elements of a molecule are different types of atoms, while the interactions among atoms can be generally categorized into short-range interactions (SRI) and long-range interactions (LRI). Wherein, SRI, such as covalent bonds or ionic bonds, acts over relatively *small distances* that are typically within the range of a few atomic diameters. By contrast, LRI, such as hydrogen bonds and coulombic interactions, operates over *much larger distances* compared to the typical atomic diameter. Although LRI is typically with weaker strength than SRI, it plays a significant role in determining the physical and chemical properties of molecules [36; 20].

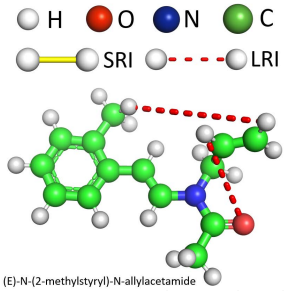


Figure 1: An exemplar molecular with the long-range interactions (dash lines) and short-range interactions (solid lines).

[*]Equal contributions.

NeurIPS 2023 New Frontiers in Graph Learning Workshop (NeurIPS GLFrontiers 2023).

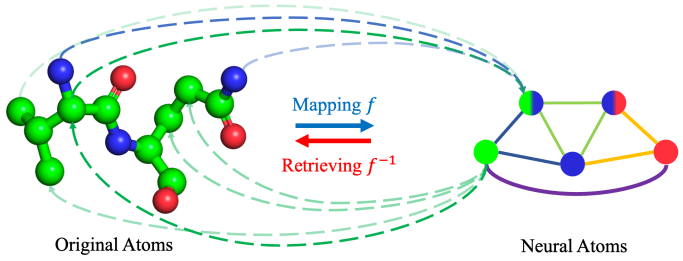


Figure 2: Illustration of *Neural* Atoms. The mapping function f is to project the original atoms to *Neural* Atoms, and the retrieving function f^{-1} aims to inject the information of *Neural* Atoms to original atoms, allowing the GNN model to capture LRI via the interaction between *Neural* Atoms.

In Fig. 1, we give an illustration of SRI and LRI in a molecular graph, which is formed by the covalent bonds with explicit hydrogens. The SRIs, *e.g.*, the covalent bond between carbon (●) and hydrogen (●), are denoted as solid lines (●—●), which involves the electron exchange of atoms to establish the stabilization of the molecular structure. As such, the SRIs are considered to be the explicit edges of the molecular graph. Whereas the LRIs (denoted as dashed lines) are presented as implicit force depending on the distance between atoms in the 3D range and the molecular structure and could involve atoms with multiple hops distance. For example, the hydrogen bond (●, ●) can involve the 5-length path (●, ●, ●, ●, ●, ●) with four middle atoms, whereas the hydrogen bond (●, ●) can involve the 10-length path (●, ● ●, ●, ●, ●, ●, ●, ●, ●, ●) with nine middle atoms.

GNNs can effectively capture the SRI as they aggregate neighboring information in each layer, however, they are intrinsically ineffective in capturing the LRI that is located among distant nodes. Naively stacking more layers to perceive LRI will encounter exponentially increasing neighbor nodes. This induces GNN to compress excessive spurious information of some irrelevant long-range neighbor nodes, which results in the well-known over-squashing problem [1].

A common enhancement of GNNs is Graph Transformer (GT) [18; 19], which uses the self-attention mechanism to process information from neighbors that allows the capture of complex and long-range relationships between nodes. However, with self-attention, a node may attend to a large number of nodes with no direct edge connection, where the excessive information can also bring difficulty in capturing meaningful LRI that are usually sparse, and it could even involve irrelevant information during the message aggregation and update process. In practice, only marginal improvements can GTs achieve compared with GNNs [34], while the self-attention on the entire molecular makes GTs much more computationally expensive than GNNs.

In this work, we aim to design an effective and efficient mechanism to enhance GNNs for capturing long-range interactions. We notice that in areas of computational chemistry and molecular modeling, the pseudo atoms can be employed to group atoms into a more manageable and computationally efficient representation. Note these pseudo atoms are not real atoms with physical properties but are introduced for simplification and convenience in calculations. One can approximate the effects of LRIs by introducing pseudo atoms strategically, making simulations of complex molecular systems more tractable while still retaining the essential characteristics of these interactions. However, pseudo atoms cannot be directly applied here, as they are typically manually designed with expert knowledge.

To close this gap, we propose *Neural Atoms*: *Learning to implicitly project all the original atoms into a few neural atoms* that abstract the collective information of atomic groups in a molecule. Based on this, we explicitly exchange the information among neural atoms and project them back to the atoms' representations as an enhancement. Under this mechanism, the neural atoms establish the communication channels among distant nodes, which can reduce the interaction scope of arbitrary node pairs into a single hop. Note that the proposed method is architecture-agnostic and computation-efficient, which can be equipped with an arbitrary GNN and help to capture LRI. Empirically, we evaluate our method on three long-range graph benckmarks [34][2] with both link-level and graph-level tasks. We achieve up to 27.32% improvement in the Peptides-struct dataset with different kinds of commonly-used GNNs. Our main contributions are summarized as follows.

[2]As we focus on the 2D intra-molecule interaction in this study, our Neural Atom does not consider to handle the 3D coordinate information for the present. Correspondingly, the employed benchmark does not provide the 3D coordinate information. We leave the extension of our Neural Atoms on the 3D scenario in the future.

- We formalize the concept of neural atoms, and instantiate it with dual self-attention mechanisms to learn the atoms-neural atoms projection and model the interactions among neural atoms (Sec. 3).
- We conduct extensive experiments on long-range molecular datasets for property prediction and structure reconstruction, and empirically justify that our method can boost various GNNs (Sec. 4).
- We provide an in-depth understanding of our method from a physical perspective and reveal its intrinsic connection with a commonly-used LRI calculation method, Ewald Summation (Sec. 5).

2 Preliminaries

Notation. An undirected graph is denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the node set and $\mathcal{E}$ is the edge set. For each node $v \in \mathcal{V}$, its D-dimension node feature is denoted as $x_v \in \mathbb{R}^D$. Besides, $\boldsymbol{h}_v^{(\ell)}$ denote the node representation of node v in ℓ-th layer for a L-layer GNNs, where $0 \leq \ell \leq L$ and $\boldsymbol{h}_v^{(0)} = x_v$.

Graph Neural Networks. Given a graph $\mathcal{G}$, the neighbor information aggregate function $f^{(\ell)}$ and the node representation update function $\phi^{(\ell)}$, we can formulate the ℓ-th layer operation of GNNs as

$$\boldsymbol{h}_v^{(\ell)} = \phi^{(\ell)}\Big(\boldsymbol{h}_v^{(\ell-1)}, f^{(\ell)}(\boldsymbol{h}_v^{(\ell-1)})\Big),$$

where $\mathcal{N}(v) = \{u \in \mathcal{V} | (u, v) \in \mathcal{E}\}$ are the neighbor nodes of node v. Different GNNs vary from the design of the function $f^{(\ell)}$ and $\phi^{(\ell)}$. For example, GCN [16] define its $f^{(\ell)}$ as

$$f^{(\ell)}(\boldsymbol{h}_v^{(\ell-1)}) = \Sigma_{u \in \mathcal{N}(v) \cup \{v\}} 1/\sqrt{\hat{d}_u \hat{d}_v} \mathbf{W}^{(\ell)} \boldsymbol{h}_u^{(\ell-1)},$$

and use `ReLu` function as $\phi^{(\ell)}$, where $\mathbf{W}$ is the learnable parameter for filtering the graph signal. Likewise, GIN [40] obtain neighbor information by $f^{(\ell)}(\boldsymbol{h}_v^{(\ell-1)}) = \sum_{u \in \mathcal{N}(v) \cup \{v\}} \boldsymbol{h}_u^{(\ell-1)}$, and update node representation via a feed-forward network. In addition, the node representations in the last GNN layer would then be transformed by a feed-forward network for specific downstream tasks. For example, for graph-level classification tasks, the whole representations $\boldsymbol{H} \in \mathbb{R}^{N \times d}$ would be transformed to the logits $\hat{\boldsymbol{Y}} \in \mathbb{R}^C$ with C classes that usually through the graph pooling operation.

Multi-head Attention Mechanism. Consider a molecular graph with N nodes, the input of the attention mechanism f_{Att} consists of three components, termed as query $\boldsymbol{Q} \in \mathbb{R}^{k \times d_k}$, key $\boldsymbol{K} \in \mathbb{R}^{N \times d_k}$, and value $\boldsymbol{V} \in \mathbb{R}^{N \times d_v}$, where d_k and d_v are dimensions. As such, the $f_{\text{Att}} = \sigma(\boldsymbol{Q}\boldsymbol{K}^\top)V$ is calculated via the dot product of the query and the key, with σ denoting the activation function. To extend the attention to the multi-head case, we further utilize the linear projection for $\boldsymbol{Q}, \boldsymbol{K}$, and $\boldsymbol{V}$ to yield M representation subspace. The multi-head attention mechanism allows us to learn different attentions between query and key, thus enhancing the model expressivity. Namely, with M attention heads,

$$\text{MultiHead}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \boldsymbol{W}^O \|_{m=1}^M \boldsymbol{O}_m, \;\; \text{s.t. } \boldsymbol{O}_m = f_{\text{Att}}(\boldsymbol{Q}\boldsymbol{W}_m^Q, \boldsymbol{K}\boldsymbol{W}_m^K, \boldsymbol{V}\boldsymbol{W}_m^V),$$

where $\boldsymbol{W}^O \in \mathbb{R}^{Md_h \times d_{out}}$ is the linear projection matrix for learning the subspace representation of the contracted head outputs to output dimension d_{out}. Besides, $\boldsymbol{W}_m^Q \in \mathbb{R}^{d_k \times d_k}$, $\boldsymbol{W}_m^K \in \mathbb{R}^{d_k \times d_k}$, and $\boldsymbol{W}_m^V \in \mathbb{R}^{d_v \times d_v}$ are the projection matrixes in the m-th head for query, key, and value, respectively.

Graph Hierarchical Learning. The GNNs can only propagate information through edges formed by the SRI. To capture LRI, the model would inevitably aggregate information on the numerous intermediate atoms from the source node to the target node. The overwhelming information could suppress the information from distant target nodes, thus degenerating the ability of the model to capture LRI. An intuitive idea is to group the intermediate atoms and abstract their information into a single node. This could reduce the information propagating length of LRI by decreasing the potential intermediate atoms, which helps enhance the distant atoms' information strength. One straightforward implementation is appending a supernode [10] to graph with connection with all the atoms to extract global information to improve the model's performance without 3D position information available. The supernode technique has been proven to increase the expressiveness and reduce under-reaching issues [14], and the ability to approximate the Graph Transformer [5]. However, the supernode differs from the neural atom regarding the grouping strategy and the information-exchanging mechanism. A detailed discussion can be found in Appendix E and F. The grouping operation is achieved by graph pooling for abstracting the node representation while preserving the local structure information [42; 23; 8; 3]. Such an operator allows the model to obtain multi-scale graph-level representation, which implicitly enhances the model to capture LRI. However, graph

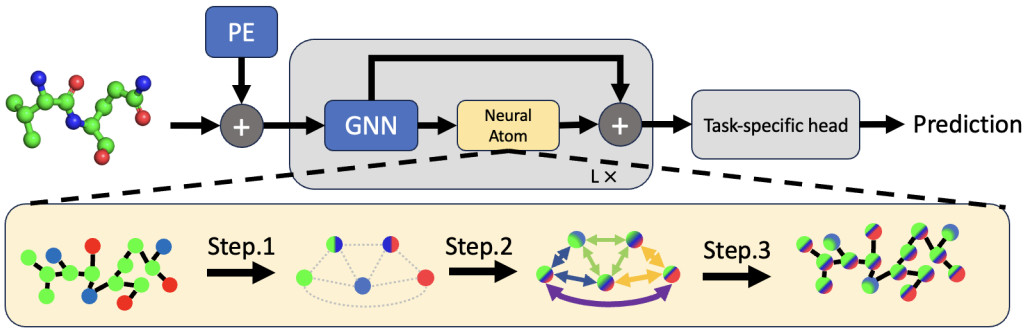


Figure 3: The proposed Neural Atom framework aims to obtain graph representation for different downstream tasks. The Neural Atom can enhance arbitrary by injecting LRI information via the interaction of neural atoms. We demonstrate the information exchange by the mixture of colors.

pooling is designed to obtain global representation without considering cooperating with the LRI and SRI information, which makes the GNN unable to propagate and utilize the LRI information.

Graph Laplacian Position Encoding. The position encoding can benefit graph learning by instilling distinguishable information to the node features, such as local structure and Laplacian eigenvectors [19]. Standard GNNs are known to be bounded by the 1-Weisfeiler-Leman test (1-WL), meaning they fail to distinguish non-isomorphic graphs with 1-hop message passing. Instilling graph position encoding allows GNNs to be more expressive than the 1-WL test, as each node is equipped with the distinguishable information [18; 19].

Ewald Summation. The Ewald Summation [31] is a calculation algorithm for molecular dynamics simulation, which calculates the energy of the interaction by processing the atoms' position and their charges via mathematical models. The Ewald Summation decomposes the actual interaction into the short-range and the long-range part, where the former decays vastly in the real space, and the latter would only exhibit fast decay in the frequency space. The short-range part can be calculated by employing a summation with the distance cut-off. Whereas the long-range part, transformed by Fourier transformation, can be calculated via a low-frequency cut-off summation.

3 Method

In this section, we first introduce the concept of neural atoms and how it can benefit GNNs to capture LRI (Sec. 3.1). Next, we elaborate on the attention-based approach to realize neural atoms (Sec. 3.2). Finally, we connect it with the Ewald Summation from the graph representation learning (Sec. 3.3).

3.1 Formalizing the neural atoms on Molecular Graphs

As aforementioned, we proposed the neural atoms inspired by the pseudo-atoms of molecular dynamics simulations, which abstract prefixed atom groups into individual representations. By mapping the atoms into neural atoms and allowing a fully connected structure with neural atoms, one can reduce the LRI into direct neighbor interaction. The formalization of neural atoms is as follows.

Definition 3.1. *Neural atoms encompass a collection of virtual and adaptable atoms that symbolize a cluster of atoms within a designated molecular graph. The process entails the acquisition of knowledge that enables the transformation of conventional atoms into neural atoms. This transformation can be technically executed through methodologies such as graph clustering or graph coarsening.*

Advantages. Different neural atoms are designed to gather data from distinct local regions, with the intent of preserving local information to the greatest extent possible. This approach serves to alleviate the challenges associated with learning intricate graph structures. Consequently, interactions among atoms that are widely separated are transformed into interactions among neural atoms, effectively narrowing the interaction range from any arbitrary pair of atoms to a single step, as illustrated in Fig. 2. Employing a layer-wise collaboration with GNNs, information from distant atoms can be effectively exchanged through the interactions among neural atoms. In what follows we elaborate on the implementation of neural atoms via the attention mechanism and the collaboration with GNNs.

3.2 Injecting long-range information into message passing by neural atoms

The overall inference pipeline is illustrated in Fig. 3. Specifically, in ℓ-th layer, we obtain the atom representations $\boldsymbol{H}_{\text{GNN}}^{(\ell)}$ with neighborhood information via the ℓ-th GNN layer. Then, we apply neural atoms to enhance the representations as $(\boldsymbol{H}_{\text{GNN}}^{(\ell)}, \boldsymbol{H}_{\text{NA}}^{(\ell)}) \rightarrow \boldsymbol{H}^{(\ell)}$, which includes three steps as follows.

Step1. Project atom representations $\boldsymbol{H}_{\text{GNN}}^{(\ell)}$ to neural atom representations $\boldsymbol{H}_{\text{NA}}^{(\ell)}$. To achieve this projection, we commence by initializing a learnable neural atom weights $\boldsymbol{Q}_{\text{NA}}^{(\ell)} \in \mathbb{R}^{K \times d_k}$, where K ($\ll N$) specifies the number of neural atoms as a hyperparameter and d_k is the embedding dimension. By employing the multi-head attention ($\mathrm{MultiHead}$), we can learn the grouping function for obtaining neural atoms representing the intermediate atoms. Wherein, the atom representations $\boldsymbol{H}_{\text{GNN}}^{(\ell)} \in \mathbb{R}^{N \times d}$ are mapped to the key $\boldsymbol{K} \in \mathbb{R}^{N \times d_h}$ and value $\boldsymbol{V} \in \mathbb{R}^{N \times d_h}$ by linear projection weights $\boldsymbol{W}_K \in \mathbb{R}^{d_h \times d_h}$ and $\boldsymbol{W}_V \in \mathbb{R}^{d_h \times d_h}$, respectively. The allocation matrix $\hat{\boldsymbol{A}}_m$ for m-th head is obtained as $\hat{\boldsymbol{A}}_m = \sigma(\boldsymbol{Q}_{\text{NA}}^{(\ell)} \boldsymbol{K}^\top) \in \mathbb{R}^{K \times N}$, and $\oplus$ denotes operation for combining representations, *e.g.*, sum or a feed forward network. The representations $\boldsymbol{H}_{\text{NA}}^{(\ell)}$ of neural atoms are obtained by:

$$\boldsymbol{H}_{\text{NA}}^{(\ell)} = \mathrm{LayerNorm}\left(\boldsymbol{Q}_{\text{NA}}^{(\ell)} \oplus \mathrm{MultiHead}(\boldsymbol{Q}_{\text{NA}}^{(\ell)}, \boldsymbol{H}_{\text{GNN}}^{(\ell)}, \boldsymbol{H}_{\text{GNN}}^{(\ell)})\right), \tag{1}$$

where the $\mathrm{LayerNorm}$ represents the operation of Layer Normalization [2].

Step2. Exchange information among neural atoms $\boldsymbol{H}_{\text{NA}}^{(\ell)} \rightarrow \tilde{\boldsymbol{H}}_{\text{NA}}^{(\ell)}$. Then, we explicitly exchange the information among neural atoms to capture the long-range interactions in this molecular graph. We further employ the self-attention mechanism for efficient information exchange, namely,

$$\tilde{\boldsymbol{H}}_{\text{NA}}^{(\ell)} = \mathrm{LayerNorm}\left(\boldsymbol{H}_{\text{NA}}^{(\ell)} \oplus \mathrm{MultiHead}(\boldsymbol{H}_{\text{NA}}^{(\ell)}, \boldsymbol{H}_{\text{NA}}^{(\ell)}, \boldsymbol{H}_{\text{NA}}^{(\ell)})\right). \tag{2}$$

Step3. Project neural atoms back and enhance the atoms' representation $(\boldsymbol{H}_{\text{GNN}}^{(\ell)}, \boldsymbol{H}_{\text{NA}}^{(\ell)}) \rightarrow \boldsymbol{H}^{(\ell)}$. So far, the obtained $\tilde{\boldsymbol{H}}_{\text{NA}}^{(\ell)}$ contains information from different atom groups. To cooperate the $\tilde{\boldsymbol{H}}_{\text{NA}}^{(\ell)}$ with the original molecular atom, we aggregate the allocation matrix $\hat{\boldsymbol{A}}_m$ of different heads and project the neural atoms into the atom space with size N. Here, we perform the matrix reduction operations (*e.g.*, mean or summation) to aggregate the multi-head $\hat{\boldsymbol{A}}_m$ and get $\tilde{\boldsymbol{A}}_{\text{NA}}^{(\ell)}$, which allows the model to learn different allocation weights. The final atom representations $\boldsymbol{H}^{(\ell)}$ are obtained by enhancing the atom representations $\boldsymbol{H}_{\text{GNN}}^{(\ell)}$ with the neural atoms' representations $\tilde{\boldsymbol{A}}_{\text{NA}}^{(\ell)} \tilde{\boldsymbol{H}}_{\text{NA}}^{(\ell)}$, *i.e.*,

$$\boldsymbol{H}^{(\ell)} = \boldsymbol{H}_{\text{GNN}}^{(\ell)} \oplus \tilde{\boldsymbol{A}}_{\text{NA}}^{(\ell)} \tilde{\boldsymbol{H}}_{\text{NA}}^{(\ell)}, \;\; \text{s.t.} \;\; \tilde{\boldsymbol{A}}_{\text{NA}}^{(\ell)} = \mathrm{Aggregate}\left(\{\hat{\boldsymbol{A}}_m\}_{m=1}^{M}\right)^\top \in \mathbb{R}^{N \times K}. \tag{3}$$

The overall procedure. We summarize the forward pipeline in Algorithm 1. In brief, given the atom representations of a molecular graph $\boldsymbol{H}_{\text{GNN}}^{(\ell-1)}$ in $(\ell-1)$-th layer, we first get the updated atoms' representations $\boldsymbol{H}_{\text{GNN}}^{(\ell)}$ by the GNN in ℓ-th layer to capture the SRI. Then, the atoms $\boldsymbol{H}_{\text{GNN}}^{(\ell)}$ are projected to neural atoms $\tilde{\boldsymbol{A}}_{\text{NA}}^{(\ell)} \tilde{\boldsymbol{H}}_{\text{NA}}^{(\ell)}$ to capture the LRI with the three above steps. Finally, the enhanced representations $\boldsymbol{H}^{(\ell)}$ are obtained by mixing both atoms' and neural atoms' representations.[3]

Remark 3.1. *We map the atoms in the molecular graph to K neural atoms, which are far less than the size of the atoms of the original molecular graph. Such a mechanism allows us to filter out potentially irrelevant information since the attention mechanism aggregates the information according to the similarity between the embedding of neural atoms and the atoms within the original graph. In addition, rather than processing attention on the fully connected graph with huge time consumption, neural atoms can reduce the computation burden. Specifically, instead of directly modeling the atom interaction in a fully connected graph, we map the potential interaction into the space constructed by neural atoms, which is more sparse compared to the original graph.*

[3] One can scale neural atoms to a large molecular graph by employing a linear Transformer-like Performer [7] or BigBrid [43]. One can also prune the connection between neural atoms and atoms within the molecular graph based on the attention score to reduce the computation.

Algorithm 1 Message propagation with neural atoms.

Require: Molecular graph $\mathcal{G}$, atoms feature X, and GNN model f.

Initialize $\boldsymbol{H}^{(0)} \leftarrow X$
for $\ell = 1 \dots L$ **do**
  $\boldsymbol{H}^{(\ell)}_{\text{GNN}} \leftarrow f^{(\ell)}(\boldsymbol{H}^{(\ell-1)}, \mathcal{G})$
  $\boldsymbol{H}^{(\ell)}_{\text{NA}} \leftarrow \text{LayerNorm}\left(\boldsymbol{Q}^{(\ell)}_{\text{NA}} \oplus \text{MultiHead}(\boldsymbol{Q}^{(\ell)}_{\text{NA}}, \boldsymbol{H}^{(\ell)}_{\text{GNN}}, \boldsymbol{H}^{(\ell)}_{\text{GNN}})\right)$
  $\tilde{\boldsymbol{H}}^{(\ell)}_{\text{NA}} \leftarrow \text{LayerNorm}\left(\boldsymbol{H}^{(\ell)}_{\text{NA}} \oplus \text{MultiHead}(\boldsymbol{H}^{(\ell)}_{\text{NA}}, \boldsymbol{H}^{(\ell)}_{\text{NA}}, \boldsymbol{H}^{(\ell)}_{\text{NA}})\right)$
  $\tilde{\boldsymbol{A}}^{(\ell)}_{\text{NA}} \leftarrow \text{Aggregate}\left(\{\hat{\boldsymbol{A}}_m\}_{m=1}^{M}\right)^{\top}$
  $\boldsymbol{H}^{(\ell)} \leftarrow \boldsymbol{H}^{(\ell)}_{\text{GNN}} \oplus \tilde{\boldsymbol{A}}^{(\ell)}_{\text{NA}} \tilde{\boldsymbol{H}}^{(\ell)}_{\text{NA}}$,
end for
return $\boldsymbol{H}^{(L)}$.

3.3 Connection to the Ewald Summation

Here, we utilize the aforementioned Ewald sum matrix [9] to show and understand the interaction among particles. Each element in the matrix represents the corresponding Ewald summation, which offers a more rapidly converging series that ensure accurate results even for distant interactions. Specifically, the Ewald summation decomposes the interatomic interaction into short-range and long-range parts. Wherein, the short-range interaction (denoted as $x^{(r)}_{ij}$) can be straightforwardly summed by imposing a distance cutoff; while the long-range component (termed as $x^{(\ell)}_{ij}$) exhibits a rapid decay in its Fourier transform although diminishing slowly with distance, *i.e.*,

$$x_{ij} = \underbrace{Z_i Z_j \sum_{\mathbf{L}} \frac{\operatorname{erfc}\left(a \left\|\mathbf{r}_i - \mathbf{r}_j + \mathbf{L}\right\|_2\right)}{\left\|\mathbf{r}_i - \mathbf{r}_j + \mathbf{L}\right\|_2}}_{\text{SRI: } x^{(r)}_{ij}} + \underbrace{\frac{Z_i Z_j}{\pi V} \sum_{\mathbf{G}} \frac{e^{-\|\mathbf{G}\|_2^2/(2a)^2}}{\|\mathbf{G}\|_2^2} \cos\left(\mathbf{G} \cdot \left(\mathbf{r}_i - \mathbf{r}_j\right)\right)}_{\text{LRI: } x^{(\ell)}_{ij}} + x^{(s)}_{ij}$$

The above equation gives the interatomic interaction strength, *i.e.*, the non-diagonal elements in the Ewald sum matrix. The Z_i and $\mathbf{r}_i$ are the atomic number and position of the $i-$th atom, and V is the unit cell volume. Besides, L denotes the lattice vectors within the distance cutoff, G is the non-zero reciprocal lattice vectors, and a is the hyperparameter that controls the summation converge speed.

The strength of SRI and LRI are determined by multiplying the atomic numbers, which indicate the number of positive charges. The difference lies in the manner of calculating the distance between atoms. The $x^{(r)}_{ij}$, corresponding to the strength of the interaction in the real space, is calculated by the error function erfc and the Euclidian distance $|\mathbf{r}_i - \mathbf{r}_j + \mathbf{L}|_2$ between atoms. The $x^{(\ell)}_{ij}$ describe the interaction strength in the reciprocal space, which is calculated by the distance given by the dot product of G and the interatomic distance $(\mathbf{r}_i - \mathbf{r}_j)$. Whereas the self-energy $x^{(s)}_{ij}$ is a constant term for the interaction from the positive atomic cores, which is irrelevant to the interatomic distance.

Remark 3.2. *Recall in Eqn. (3), the final atom representations $\boldsymbol{H}^{(\ell)}$ are obtained by combining both the atoms' representations $\boldsymbol{H}^{(\ell)}_{GNN}$ and the neural atoms' representations $\tilde{\boldsymbol{A}}^{(\ell)}_{NA}\tilde{\boldsymbol{H}}^{(\ell)}_{NA}$. In association with the Ewald summation, The $\boldsymbol{H}^{(\ell)}_{GNN}$ contains the information of the SRI term $x^{(r)}_{ij}$ and self-energy term $x^{(s)}_{ij}$ of the Ewald summation, while the enhanced $\tilde{\boldsymbol{A}}^{(\ell)}_{NA}\tilde{\boldsymbol{H}}^{(\ell)}_{NA}$ is to approximate the LRI term $x^{(\ell)}_{ij}$.*

4 Experiments

In this section, we empirically evaluate the proposed method on real-world molecular graph datasets for both graph-level and link-level tasks. All the datasets require modeling LRI for accurate prediction on downstream tasks. We aim to provide answers to two following questions. **Q1**: How effective are the proposed methods on real-world molecular datasets with common GNNs? **Q2**: How does the grouping strategy of neural atoms affect the performance of different GNNs for capturing the LRI?

Setup. We implement the neural atom with GNNs in the GraphGPS framework [19], which provide various choices of positional and structural encodings. All the experiments are run on an NVIDIA RTX

Table 1: Test performance on three LRGB datasets. Shown is the mean ± s.d. of 4 runs.

Model	Peptides-func	Peptides-struct	PCQM-Contact
	AP ↑	MAE ↓	MRR ↑
Transformer+LapPE	0.6326 ± 0.0126	0.2529 ± 0.0016	0.3174 ± 0.0020
SAN+LapPE	0.6384 ± 0.0121	0.2683 ± 0.0043	0.3350 ± 0.0003
GraphGPS	0.6535 ± 0.0041	0.2500 ± 0.0005	0.3337 ± 0.0006
GCN	0.5930 ± 0.0023	0.3496 ± 0.0013	0.2329 ± 0.0009
+ Neural Atoms	**0.6220 ± 0.0046**	**0.2606 ± 0.0027**	**0.2534 ± 0.0200**
GINE	0.5498 ± 0.0079	0.3547 ± 0.0045	0.3180 ± 0.0027
+ Neural Atoms	**0.6154 ± 0.0157**	**0.2553 ± 0.0005**	**0.3126 ± 0.0021**
GCNII	0.5543 ± 0.0078	0.3471 ± 0.0010	0.3161 ± 0.0004
+ Neural Atoms	**0.5996 ± 0.0033**	**0.2563 ± 0.0020**	**0.3049 ± 0.0006**
GatedGCN	0.5864 ± 0.0077	0.3420 ± 0.0013	0.3218 ± 0.0011
+ Neural Atoms	**0.6562 ± 0.0075**	**0.2585 ± 0.0017**	**0.3258 ± 0.0003**
GatedGCN+RWSE	0.6069 ± 0.0035	0.3357 ± 0.0006	0.3242 ± 0.0008
+ Neural Atoms	**0.6591 ± 0.0050**	**0.2568 ± 0.0005**	**0.3262 ± 0.0010**

3090 GPU with AMD Ryzen 3960X as the CPU. We employ the molecular datasets (Peptides-Func, Petides-Struct, PCQM-Contact) that exhibit LRI from Long Range Graph Benchmarks (LRGB) [34]. Detailed experiment settings are shown in the appendix B.

Baseline. We employ the GCN [16], GINE [13], GCNII [6] and GatedGCN [4] and GatedGCN augmented augmented with Random Walk Structure Encoding (RWSE) as the baseline GNNs. To capture the LRI, one could adopt the fully connected graph to obtain interatomic interaction for distant atom pairs explicitly. Here, we adopt the Graph Transformers for comparison. Specifically, we introduce the fully connected Trasnforemr [32] with Laplacian Positon Encodings [33], SAN [18] and the recent GraphGPS [19], which utilize GNNs and Transformer to capture short-range and long-range information respectively. We cooperate our neural atom with different GNN models to evaluate its performance, denoted as + *Neural Atom*.

4.1 Quantitative Results

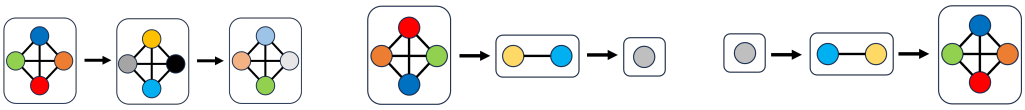
Shown as Tab. 1, all the GNNs achieve significant improvement with the assistance of neural atoms. The GNNs gain improvement from 8.17% to 12.55% on the peptides-func dataset. The GNNs also receive significant improvement on the peptides-func dataset, at most 27.32% for the GINE model. The improvement for different GNNs on various datasets empirically proves that neural atoms can enhance the common GNNs to capture LRI on molecular graphs. Especially for the GatedGCN, which exceeds the Transformer with LapPE on the peptides-func dataset and shows competitive results for the other counterparts on both the peptides-func and PCQM-Contact datasets.

We also notice that more powerful GNNs, especially with edge learning or edge attention filtering mechanisms, could lead to better performance in the scenario with LRI. Compared to other GNN models, the significant improvement of GatedGCN could be the incoming information filtering via the gate mechanism, which allows the neural atoms to obtain representation with rich SRI information.

4.2 The varying choice of K

As previously noted, our methodology involves the categorization of atoms inside the initial molecule into K neural atoms. When determining the appropriate hyperparameter K, we take into account the fluctuating quantity of atoms present in the molecules, aligning it with the average number of atoms found within the dataset. We present an analysis of the impact of different strategies of the K. The fixed technique, denoted as *fixed K*, involves setting the value of K as a proportion relative to the average number of atoms in the dataset, as seen in Fig. 4(a). The technique involves setting the value of K as a proportion to the average number of atoms on the dataset at the initial layer. Subsequently, each layer is determined as a proportion to the previous value of k, as depicted in Fig. 4(b). The *incremental* strategy can be considered as the antithesis of the *decremental* approach, wherein the value of K is sequentially increased incrementally, as depicted in Fig. 4(c). We assess various approaches for determining the count of neural atoms on peptides-struct, as depicted in Tab. 2.

The *fixed* configuration enables the model to establish a stable interaction space, which is generated by a predetermined amount of neural atoms in each layer. This arrangement can be advantageous for most GNNs. While employing the *decremental* approach, the model is capable of adopting an



(a) Fixed (b) Decremental (c) Incremental

Figure 4: Neural Atom grouping strategies.

Table 2: Test performance for different grouping strategies on **Peptides-struct**.

Model	Fixed	Incremental	Decremental
GCN	**0.2582 ± 0.0011**	0.3239 ± 0.0014	0.2606± 0.0003
GINE	**0.2559 ± 0.0001**	0.2795 ± 0.0012	0.2578 ± 0.0017
GCNII	0.2579 ± 0.0025	0.4084 ± 0.0025	**0.2563 ± 0.0020**
GatedGCN	0.2592 ± 0.0017	**0.2568 ± 0.0009**	**0.2569 ± 0.0007**
GatedGCN+RWSE	**0.2521 ± 0.0014**	0.2600 ± 0.0012	0.2568± 0.0005

Table 3: Test performance for different proportions (#neural atoms / #atoms) on **Peptides-func**.

Model	proportion = 0.1	proportion = 0.5	proportion = 0.9
GCN	0.5859 ± 0.0073	0.5903 ± 0.0054	**0.6220 ± 0.0046**
GINE	0.6128 ± 0.0060	0.6147 ± 0.0121	**0.6154 ± 0.0157**
GCNII	0.5862 ± 0.0066	0.5909 ± 0.0099	**0.5996 ± 0.0033**
GatedGCN	0.6533 ± 0.0030	**0.6562 ± 0.0044**	**0.6562 ± 0.0075**
GatedGCN+RWSE	0.6550 ± 0.0032	0.6565 ± 0.0074	**0.6591 ± 0.0050**

Table 4: Running time comparison and relative improvements for GNNs with *Neural* Atom and Graph Transformers. Average epoch time (average of 5 epochs, including validation performance evaluation) is shown for each model and dataset combination. The GTs are compared to the average time and performance of the three shown GNNs. Full results can be found in the Appendix. C.

	GINE	**GCNII**	**GatedGCN**	**Transformer+LapPE**	**SAN+LapPE**
Peptides-func	(+2.2s, 11.90%↑)	(+2.2s, 11.90%↑)	(+2.8s, 11.90%↑)	(+3.6s, 12.26%↑)	(+57.2s, 13.29%↑)
Peptides-struct	(+1.6s, 27.32%↓)	(+2.8s, 26.16%↓)	(+2.3s, 24.88%↓)	(+3.5s, 27.30%↓)	(+54.8s, 22.88%↓)

interaction space characterized by an inverse pyramid structure, hence facilitating the acquisition of information from a local to a global perspective. The use of *incremental* methods may not be appropriate for the LRI scenario due to the potential disruption of the local structure. This disruption can occur when the atom representation is mapped to an increasingly complicated interaction space.

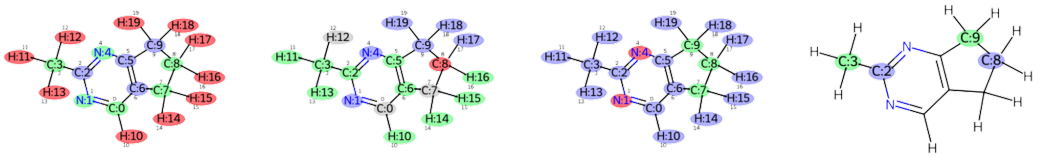
An experiment is conducted to investigate the effects of different proportions, as presented in Tab. 3. Reduced grouping proportions are associated with heightened levels of assertive grouping techniques, while conversely, higher proportions tend to be linked to more moderate approaches. It has been observed that employing a more moderate grouping approach yields superior outcomes. The grouping procedure has the potential to coarsen the molecular graph, hence potentially improving the preservation of the local structure, specifically the SRI.

In order to showcase the effectiveness of our suggested neural atom, we have conducted calculations to determine the duration of each training epoch. The results are presented in Tab. 4. Although our method requires slightly more time, it exhibits more computational efficiency compared to the Transformer approach, particularly in the case of the SAN with LapPE. We also provide a comparison on larger graphs in Tab. 10 to further demonstrate the running time gaps between the Transformer-based method and our proposed Nerual Atoms.

5 Understanding

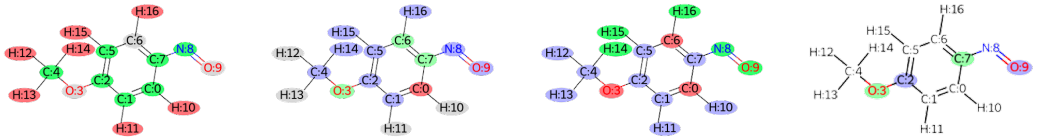
In order to demonstrate the impact of neural atoms on the process of grouping atoms and establishing high-level connections between them, as well as its capability to form meaningful groups of atoms, we adopt the Mutagenicity dataset [24] as a case study. This dataset offers explicit labels for atom groups within the molecular graph category, specifically accounting for the presence of $-NO$ and $-NH_2$ groups, which are indicative of the *mutagen* properties of the respective molecules.

To represent the potential interatomic interactions, we utilize the Ewald sum matrix to visualize. The Ewald sum matrix comprises elements that represent the Ewald Summation for distinct interatomic interactions, as denoted by the respective row and column indices within the matrix. In this study, we utilize a three-layer Graph Isomorphism Network (GIN) model [40], with a predetermined number of



Layer-0 Layer-1 Layer-2 Ewald Sum

Figure 9: Neural Atom grouping pattern at each layer for non-mutagen molecular



Layer-0 Layer-1 Layer-2 Ewald Sum

Figure 14: Neural Atom grouping pattern at each layer for mutagen molecular ($-NO$)

neural atoms. Specifically, our model consists of four neural atoms in each layer. The atom allocation matrix is visualized through the assignment of a distinct color to each atom within the original chemical graph based on the index of the highest attention weight within the matrix. The allocation pattern for the neural atoms at each layer, as well as the interatomic interactions suggested by the Ewald sum matrix, are visualized in Fig. 5 to 7. The observed grouping pattern is consistent with the interatomic interaction as suggested by the Ewald sum matrix, with the application of thresholding for the purpose of enhancing visual clarity. Atoms displaying the same color are indicative of their possession of a high degree of interaction potential. As illustrated in Fig. 9, the atoms located within the range of (C:3-C:9) and (C:2-C:8) are assigned to separate neural atoms. This enables the model to depict their interaction by sharing information between these neural atoms.

Furthermore, we employ visual representation to depict the molecular structure of the mutagen compound containing the $-NO$ group, as illustrated in Fig. 14. In the primary layer, hydrogen atoms (H) and oxygen atoms (O) are segregated into distinct neural atoms, regardless of the multi-hop distance between them. This enables the model to get the atom representation corresponding to each element. Within the remaining layers, the constituents of $-NO$ (N:8 and O:9) are organized into a neural atom, enabling the model to capture their representation comprehensively. This facilitates the prediction of the molecular graph. It is worth noting that the observed grouping pattern is consistent with the interatomic interaction, as shown by the Ewlad sum matrix. The atoms denoted as (C:2-O:9) and (C:7-O:3) are assigned to distinct neural atoms. The model is capable of accurately representing the LRI even when there are multiple hops and intermediate atoms between the interacting entities. The presented visualization showcases the efficacy of our proposed methodology in facilitating the connection between remote atoms. Additionally, the atom groups identified in this study have the potential to impact the accurate prediction of molecular graph features significantly. We provide more visualization understanding in the Appendix C.

6 Discussion and Conclusion

Limitation. This work mainly focuses on the molecular graph without 3D coordinate information, which could benefit the incorporation of LRI, as it depends on the interatomic distance in 3D space. In addition, our work pays close attention to the intra-molecular interaction rather than the inter-molecular interaction, where more valuable information might lie. For example, protein docking also involves the formation of the hydrogen bond between two biochemical components [38].

Extension. One intuitive approach is extending the *Neural* Atom to leverage the atomic coordinate information better to capture the LRI. The *Neural* Atom can also model the inter-molecular interaction by extending to the 3D scenario. Another possible direction is to instill expert knowledge in the grouping strategy to improve the interpretation ability and discriminability of the grouped atoms.

Conclusion. In this study, we aim to enhance GNNs to capture long-range interactions better. We achieve this by transforming original atoms into neural atoms, facilitating information exchange, and then projecting the improved information back to atomic representations. This novel approach reduces interaction distances between nodes to a single hop. Extensive experiments on three long-range graph benchmarks validate our method's ability to enhance any GNN to capture long-range interactions.

Statements

Ethic Statement. This paper does not raise any ethical concerns. This study does not involve human subjects, practices to data set releases, potentially harmful insights, methodologies and applications, potential conflicts of interest and sponsorship, discrimination/bias/fairness concerns, privacy and security issues, legal compliance, and research integrity issues.

Reproducibility Statement. The experimental setups for training and evaluation, as well as the hyperparameters, are described in detail in Section 4 and Appendix B, and the experiments are all conducted using public datasets.

References

[1] U. Alon and E. Yahav. On the bottleneck of graph neural networks and its practical implications. In ICLR, 2020.

[2] J. Ba, J. Kiros, and G. Hinton. Layer normalization. arXiv:1607.06450, 2016.

[3] J. Baek, M. Kang, and S. Hwang. Accurate learning of graph representations with graph multiset pooling. In ICLR, 2021.

[4] X. Bresson and T. Laurent. Residual gated graph convnets. arXiv:1711.07553, 2017.

[5] C. Cai, T. Hy, R. Yu, and Y. Wang. On the connection between mpnn and graph transformer. ICML, 2023.

[6] M. Chen, Z. Wei, Z. Huang, B. Ding, and Y. Li. Simple and deep graph convolutional networks. In ICML, 2020.

[7] K. Choromanski, V. Likhosherstov, D Dohan, X. Song, A. Gane, T. Sarlos, P. Hawkins, J. Davis, A. Mohiuddin, L. Kaiser, D. Belanger, L. Colwell, and A. Weller. Rethinking attention with performers. In ICLR, 2021.

[8] R. Ekagra, S. Soumya, and T. Partha. ASAP: Adaptive structure aware pooling for learning hierarchical graph representations. In AAAI, 2020.

[9] F. Faber, A. Lindmaa, O A. Von L., and R. Armiento. Crystal structure representations for machine learning models of formation energies. IJQC, 115(16):1094–1101, 2015.

[10] J. Gilmer, S. Schoenholz, P. Riley, O. Vinyals, and G. Dahl. Neural message passing for quantum chemistry. In ICML, 2017.

[11] M. Gromiha and S. Selvaraj. Importance of long-range interactions in protein folding. Biophysical chemistry, 77(1):49–68, 1999.

[12] W. Hu, M. Fey, M. Zitnik, Y. Dong, H. Ren, B. Liu, M. Catasta, and J. Leskovec. Open graph benchmark: Datasets for machine learning on graphs. NeurIPS, 2020.

[13] W. Hu, B. Liu, J. Gomes, M. Zitnik, P. Liang, V. Pande, and J. Leskovec. Strategies for pre-training graph neural networks. In ICLR, 2020.

[14] E. Hwang, V. Thost, S. Dasgupta, and T. Ma. An analysis of virtual nodes in graph neural networks for link prediction (extended abstract). In LoG, 2022.

[15] V. Ioannidis, D. Zheng, and G. Karypis. Few-shot link prediction via graph neural networks for covid-19 drug-repurposing. arXiv:2007.10261, 2020.

[16] T. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In ICLR, 2016.

[17] A. Kosmala, J. Gasteiger, N. Gao, and S. Günnemann. Ewald-based long-range message passing for molecular graphs. In ICML, 2023.

[18] D. Kreuzer, D. Beaini, W. Hamilton, V. Létourneau, and P. Tossou. Rethinking graph transformers with spectral attention. In NeurIPS, 2021.

[19] R. Ladislav, M. Galkin, D. Prakash, L. Tuan, W. Guy, and B. Dominique. Recipe for a general, powerful, scalable graph transformer. In NeurIPS, 2022.

[20] D. Leeson and J. Young. Molecular property design: does everyone get it?, 2015.

[21] Z. Li, M. Jiang, S. Wang, and S. Zhang. Deep learning methods for molecular representation and property prediction. Drug Discovery Today, page 103373, 2022.

[22] S. Liu, H. Wang, W. Liu, J. Lasenby, H. Guo, and J. Tang. Pre-training molecular graph representation with 3d geometry. In ICLR, 2022.

[23] Y. Ma, S. Wang, C. Aggarwal, and J. Tang. Graph convolutional networks with eigenpooling. In KDD, 2019.

[24] C. Morris, N. Kriege, F. Bause, K. Kersting, P. Mutzel, and M. Neumann. Tudataset: A collection of benchmark datasets for learning with graphs. In ICML Workshop on Graph Representation Learning and Beyond, 2020.

[25] M. Rycker, B. Baragaña, L. Duce, and I. Gilbert. Challenges and recent progress in drug discovery for tropical diseases. Nature, 559(7715):498–506, 2018.

[26] K. Schütt, P. Kindermans, Huziel E. Sauceda F., S. Chmiela, A. Tkatchenko, and K. Müller. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. NeurIPS, 30, 2017.

[27] O. Shchur, M. Mumme, A. Bojchevski, and S. Günnemann. Pitfalls of graph neural network evaluation. Relational Representation Learning Workshop (R2L 2018), NeurIPS, 2018.

[28] A. Stuke, C. Kunkel, D. Golze, M. Todorović, J. Margraf, K. Reuter, P. Rinke, and H. Oberhofer. Atomic structures and orbital energies of 61,489 crystal-forming organic molecules. Scientific data, 7(1):58, 2020.

[29] H. Stärk, D. Beaini, G. Corso, P. Tossou, C. Dallago, S. Günnemann, and P. Liò. 3d infomax improves gnns for molecular property prediction. arXiv:2110.04126, 2021.

[30] J. Thomas, A. Moallemy-Oureh, S. Beddar-Wiesing, and C. Holzhüter. Graph neural networks designed for different graph types: A survey. Transactions on Machine Learning Research, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=h4BYtZ79uy.

[31] A. Toukmaji and J. Board Jr. Ewald summation techniques in perspective: a survey. Computer physics communications, 95(2-3):73–92, 1996.

[32] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In NeurIPS, 2017.

[33] D. Vijay, L. Anh, L. Thomas, B. Yoshua, and B. Xavier. Graph neural networks with learnable structural and positional representations. In ICLR, 2022.

[34] D. Vijay, R. Ladislav, G. Mikhail, P. Ali, W. Guy, L. Anh, and B. Dominique. Long range graph benchmark. In NeurIPS Datasets and Benchmarks Track, 2022.

[35] O. Wieder, S. Kohlbacher, M. Kuenemann, A. Garon, P. Ducrot, T. Seidel, and T. Langer. A compact review of molecular property prediction with graph neural networks. Drug Discovery Today: Technologies, 37:1–12, 2020.

[36] W. Winterbach, P. Mieghem, M. Reinders, H. Wang, and D. Ridder. Topology of molecular interaction networks. BMC systems biology, 7:1–15, 2013.

[37] F. Wu, D. Radev, and S. Li. Molformer: Motif-based transformer on 3d heterogeneous molecular graphs. In AAAI, 2023.

[38] M. Wu, D. Dai, and H. Yan. Prl-dock: Protein-ligand docking based on hydrogen bond matching and probabilistic relaxation labeling. Proteins: Structure, Function, and Bioinformatics, 80(9): 2137–2153, 2012.

[39] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. Yu. A comprehensive survey on graph neural networks. IEEE transactions on neural networks and learning systems, 32(1):4–24, 2020.

[40] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. How powerful are graph neural networks? In ICLR, 2018.

[41] C. Ying, T. Cai, S. Luo, S. Zheng, G. Ke, D. He, Y. Shen, and T. Liu. Do transformers really perform badly for graph representation? In NeurIPS, 2021.

[42] Z. Ying, J. You, C. Morris, X. Ren, W. Hamilton, and J. Leskovec. Hierarchical graph representation learning with differentiable pooling. In NeurIPS, 2018.

[43] M. Zaheer, G. Guruganesh, K. Dubey, J. Ainslie, C. Alberti, S. Ontanon, P. Pham, A. Ravula, Q. Wang, and L. Amr A. Yang. Big bird: Transformers for longer sequences. NeurIPS, 2020.

[44] Y. Zhang, Y. Hu, N. Han, A. Yang, X. Liu, and H. Cai. A survey of drug-target interaction and affinity prediction methods via graph neural networks. Computers in Biology and Medicine, page 107136, 2023.

[45] Z. Zhang, L. Chen, F. Zhong, D. Wang, J. Jiang, S. Zhang, H. Jiang, M. Zheng, and X. Li. Graph neural network approaches for drug-target interactions. Current Opinion in Structural Biology, 73:102327, 2022.

Appendix

Table of Contents

A Long Range Interaction examples

The long-range interaction affects the surface area of the molecule or contributes to the formation of hydrogen bonds, which in turn affects the properties, such as melting point, water affinity, viscosity, of the molecule [41; 22; 29; 11]. Understanding and quantifying these long-range interactions is crucial in chemistry, physics, and materials science, as they influence the behavior and properties of molecules, materials, and biological systems. Here, we show different types of LRI and their properties.

- **Van der Waals Force** is a weak attractive interaction that occurs between all atoms and molecules. These forces arise due to temporary fluctuations in electron distribution, creating temporary dipoles. Van der Waals forces include London dispersion forces (arising from instantaneous dipoles), dipole-dipole interactions, and induced dipole-induced dipole interactions. These forces can act over relatively long distances and are responsible for the condensation of gases into liquids.
- **Hydrogen Bond** is a special type of dipole-dipole interaction that occurs when hydrogen is bonded to a highly electronegative atom (such as oxygen, nitrogen, or fluorine) and is attracted to another electronegative atom in a nearby molecule. Hydrogen bonds are relatively strong compared to other long-range interactions and play a crucial role in the structure and properties of water, DNA, and proteins.

  Electrostatic Interaction, also known as Coulombic interactions, occur between charged particles. While ionic bonds are a type of strong electrostatic interaction, long-range electrostatic interactions can also occur between charged ions or polar molecules that are not directly bonded to each other. These interactions can be both attractive and repulsive, depending on the charges involved.

  Dipole-Dipole Interaction occurs between molecules that have permanent dipoles, meaning they have a separation of positive and negative charges within the molecule. These interactions are stronger than van der Waals forces and can act over longer distances.

  Ion-Dipole Interaction occurs when an ion interacts with the dipole moment of a polar molecule. These interactions are important in solutions where ions are dissolved in a polar solvent.

Dispersion Interaction, also known as London dispersion forces, is a component of van der Waals forces. They arise from temporary fluctuations in electron distribution and can act between all molecules, even non-polar ones. These forces can be relatively weak but can accumulate to have a significant impact on molecular interactions.

Magnetic Interaction is a long-range magnetic interaction that can occur between magnetic moments associated with atoms or ions. These interactions are responsible for the behavior of ferromagnetic and antiferromagnetic materials.

B Reproduction details

B.1 Hyperparameters

We performed our experiment on four seeds and reported the mean with standard deviation as the final result. We summarize the common hyperparameters that are shared across different models on the introduced datasets, along with the model-specific hyperparameters, shown as Tab. 5 to Tab. 8. Our code is added in the anonymous link: https://anonymous.4open.science/r/Neural-Atom-CF5C/.

Table 5: Common hyperparameters for datasets from Long Range Graph Benchmark.

Hyperparameter	**PCQM-Contact**	**Peptides-func**	**Peptides-struct**
Dropout	0	0.12	0.2
Allocation matrix Grouping	mean	mean	mean
Positional Encoding	LapPE-10	LapPE-10	LapPE-10
PE dim	16	16	20
PE encoder	DeepSet	DeepSet	DeepSet
Batch size	256	128	128
Learning Rate	0.001	0.0003	0.0003
# Epochs	200	200	200

Table 6: Model-specific hyperparameters for PCQM-Contact

Hyperparameter	**# GNN Layers**	**Hidden dim**	**# Heads**	**proportion**	**# Neural Atoms**
GCN	5	300	1	0.9	27
GCNII	5	100	2	0.8	24
GINE	5	100	1	0.95	28
GatedGCN	8	72	1	0.5	15

Table 7: Model-specific hyperparameters for Peptides-func

Hyperparameter	**# GNN Layers**	**Hidden dim**	**# Heads**	**proportion**	**# Neural Atoms**
GCN	5	155	1	0.15	22
GCNII	5	88	1	0.2	27
GINE	5	88	2	0.9	135
GatedGCN	5	88	1	0.5	75

Table 8: Model-specific hyperparameters for Peptides-struct

Hyperparameter	**# GNN Layers**	**Hidden dim**	**# Heads**	**proportion**	**# Neural Atoms**
GCN	5	155	1	0.15	22
GCNII	5	88	1	0.2	30
GINE	5	88	2	0.9	135
GatedGCN	5	88	1	0.5	135

B.2 Dataset details

The statistical information of the datasets is shown as Tab. 9. Note that all three datasets consist of multiple graphs and are evaluated under the inductive setting, which means that the evaluating

Table 9: Dataset statistical information. All the datasets consist of molecular graphs that exhibit LRI.

Dataset	Total Graphs	Total Nodes	Avg Nodes	Mean Deg.	Total Edges	Avg Edges	Avg Short.Path.	Avg Diameter
pcqm-contact	529,434	15,955,687	30.14	2.03	32,341,644	61.09	4.63±0.63	9.86±1.79
pepfunc	15,535	2,344,859	150.94	2.04	4,773,974	307.30	20.89±9.79	56.99±28.72
pepstruct	15,535	2,344,859	150.94	2.04	4,773,974	307.30	20.89±9.79	56.99±28.72

Table 10: Dataset statistical information for larger graph.

Dataset	Total Graphs	Total Nodes	Total Edges	Task Type	Task Metric
ogbn-arXiv [12]	1	169,343	1,166,243	Node multi-class classification	Accuracy
Amazon Product-Computer [27]	1	13,752	491,7222	Node multi-class classification	Accuracy

Table 11: Wall-clock run times. Average epoch time (average of 5 epochs, including validation performance evaluation) is shown for each model and dataset combination.

avg. time / epoch	Peptides-func	Peptides-struct	PCQM-Contact
GCN	2.6s	2.5s	56.9s
+ Neural Atom	5.5s	4.9s	65.1s
GINE	2.6s	2.6s	56.7s
+ Neural Atom	4.8s	4.2s	66.8s
GCNII	2.5s	2.3s	56.9s
+ Neural Atom	4.7s	5.1s	59.4s
GatedGCN	3.3s	3.2s	56.5s
+ Neural Atom	6.1s	5.5s	61.6s
GatedGCN+RWSE	3.4s	4.1s	59.4s
+ Neural Atom	6.4s	5.2s	65.0s
Transformer+LapPE	6.4s	6.2s	59.2s
SAN+LapPE	60s	57.5s	205s
GraphGPS	6.5s	6.5s	61.5s

portion of the dataset differs from the training counterparts. For the PCQM-Contact dataset, the task is to predict whether the distant node pairs (with more than 5 hops away in a molecular graph) would be in contact with each other in the 3D space, *i.e.*, forming hydrogen bonds, which is the inductive link prediction task. The metric for the performance of the model is measured by the Mean Reciprocal Rank (MRR). Both Peptides-func and Peptides-struct are constructed from the same source but serve different purposes. The Peptides-func is a multi-label graph classification dataset for evaluating the model's ability to capture molecular properties. We adopt the unweighted mean Average Precision (AP) as the metric. The Peptides-struct is a multi-label graph regression dataset based on the 3D structure of the peptides and uses Mean Absolute Error (MAE) as the metric.

C Running time compairson

We show the running time for different models in Tab. 11, training with the hyperparameters given as Tab. 6 to Tab. 8.

we provide a running time comparison on the two graph datasets to demonstrate the gap between the Transformer-based method and our proposed Nerual Atoms. The statistical information of the two applied datasets is shown in Tab. 10, and running times are listed in Tab. **??**.

D Comparison with Ewald-based GNN in OE62

We directly adopt the neural atoms to the 3D scenario without 3D coordinate information and provide the comparison with Ewald-based GNN [17] (denoted as +Ewald Block) on the OE62 [28] dataset. We follow the same setting from Ewald-based GNN and use SchNet [26] as the backbone GNN. We only take **10** Nerual Atoms (denoted as +Neural Atoms), and *half* the *hidden channels*, *num filters*, and *num interactions* according to the setting of the Ewald-based GNN. Shown as Tab. 12,

Table 12: Energy MAE and MSE comparison on OE62

	Energy MAE ↓	Energy MSE ↓	Using 3D information
Baseline (SchNet)	0.1351	0.0658	✓
+ Ewald Block	**0.0811**	**0.0301**	✓
+ Neural Atoms	**0.0834**	**0.0309**	×

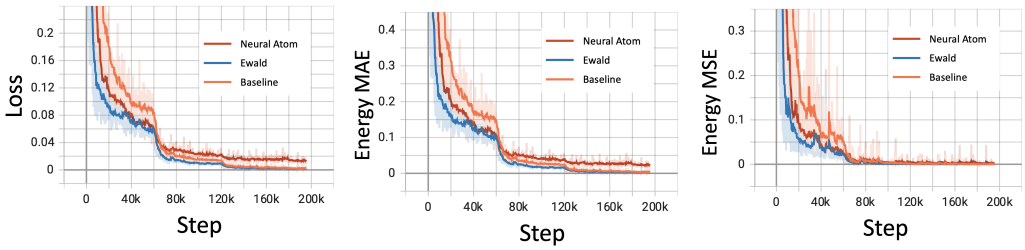


Figure 15: Training curves visualizations (1) the training loss curve, (2) the training MAE curve, (3) the training MSE curve.

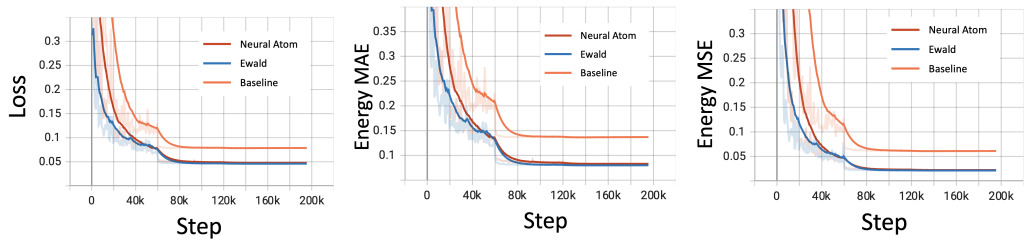


Figure 16: Validation curves visualizations (1) the validation loss curve, (2) the validation MAE curve, (3) the validation MSE curve.

our method achieves competitive performance even with the absence of 3D coordinate information, showing the generalization and effectiveness of neural atoms.

D.1 The training/validation curves

We provide the training curves for the methods in Tab. 12, shown as Fig. 15 and 16 with smoothing ratio as 0.9.

E Comparison with super/virtual node

Both the pipeline of supernode or virtual node and neural atoms for obtaining global graph information can be described as two steps.

- Information aggregating: The information of atoms within the molecular graph is aggregated into either supernode or multiple neural atoms with pair-wise connection.
- Interaction among node/atoms: The second step shows differences in interaction among node/atoms. Supernode commonly exists alone, which means there is only one super node and thus lacks the ability to interact with others like neural atoms.
- Backward Projection: The final step shows differences in terms of the interaction among nodes/atoms. As supernode commonly exists alone, it thereby lacks the ability to interact with others like neural atoms.

We provide a detailed comparison in Tab. 13.

Table 13: Comparison with super/virtual node

	Super/Virtual Node	**Neural Atoms**
#Atoms or #Nodes	1 (In most cases, there is only one single supernode for aggregating the global graph information, as increasing the number of which might not lead to performance improvement.)	K (Our neural atoms can be defined as the proportion of the average number of atoms of the molecular graph dataset, which is significantly smaller than the original number of atoms. The more neural atoms, the better the performance.)
Information aggregating	The global pooling method, e.g., global sum/mean/max pooling, treats all the information from nodes the same, thus lacking diversity among different nodes in the graph.	The multi-head Attention mechanism allows aggregating information with different weights according to the similarity between specific neural atoms and the original atoms, which allows diversity among different atoms.
Interaction among supernode/neural atoms	None. (Since supernode usually exists alone, it thus lacks the ability to interact with others.)	A fully connected graph with an attention mechanism to bridge the neural atoms for information exchange. This allows the information located in a different part of the graph, even with a large hop distance, to share information based on their embedding similarities.
Backward projection	Direct element-wise adding. The information of the supernode is directly added to the representation of each node, which fuses the information from the single supernode, which might lead to the similarity among different atoms.	Weighted combination. Our proposed neural atoms can fuse the information within according to the similarity score between neural atoms and the atoms in the original molecular graph. Such a mechanism allows the model to obtain diversity representation for further purposes.

F Further discussion

Molformer [37] combines molecular motifs and 3D geometry information by heterogeneous self-attention to create expressive molecular representations. The paper uses a virtual atom as a starting point to extract the graph representation for downstream graph-level tasks. The paper proposes attentive farthest point sampling for sampling atoms in 3D space not only according to their coordinates but also their attention score. However, the virtual atom they utilize does not participate in the message aggregation nor graph-level representation extraction, as they claim to "locate a virtual node in 3D space and build connections to existing vertices." As such, the potential long-range interaction they capture might be due to the atom pair-wise heterogeneous self-attention, which differs from our method, where the long-range interaction is captured by both the attention mechanism (step.1 in Fig. 3) and the interaction among the neural atoms (step.2 in Fig. 3).

[10] firstly introduce the concept of Message Passing Neural Networks (MPNN) to develop a unified framework for predicting molecular properties. The paper introduces the "virtual node" as an argument for global information extraction. The virtual node, connected to all other nodes within the graph, acts as the global communication channel, enhancing the model's ability to capture long-range interactions and dependencies in molecular graphs. The authors experimented with a "master node" connected to all other nodes, serving as a global feature aggregator. This approach showed promise in improving the model's performance, especially in scenarios where spatial information, e.g., 3D coordination, is limited or absent.

[14] study the benefit of introducing single or multiple virtual nodes for link prediction tasks. Virtual node, traditionally thought to serve as aggregated representations of the entire graph, is

connected to subsets of graph nodes based on either randomness or clustering mechanism. Such methodology significantly increases the expressiveness and reduces under-reaching issues in MPNN. The study reveals that virtual nodes, when strategically integrated, can provide stable performance improvements across various MPNN architectures and are particularly beneficial in dense graph scenarios. Their virtual node differs from our Neural Atom regarding the grouping strategy and the information-exchanging mechanism.

[5] investigates the relationship between MPNN and Graph Transformers (GT) via the bridge of the virtual node. It demonstrates that MPNN augmented with virtual nodes can approximate the self-attention layer of GT. Under certain circumstances, the paper provides a construction for MPNN + VN with $O(1)$ width and $O(n)$ depth to approximate the self-attention layer in GTs. The paper provides valuable insight into understanding the theoretical capabilities of MPNN with virtual nodes in approximating GT. Compared to our neural atoms, we do not focus on establishing a theoretical connection between MPNN and GT. Instead, we are interested in leveraging the attention mechanism's ability to construct an interaction subspace constructed by the neural atoms. As such, the subspace acts as a communication channel to reduce interaction distances between nodes to a single hop.

G Performance comparison for neural atoms and virtual nodes

In this section, we show the difference between Nerual Atoms and virtual nodes, w.r.t. the performance by increasing their number. Specifically, we borrow the setting of Tab.3 in our draft by aligning the number of neural atoms and virtual nodes and the backbone GNN they used. We employ the "VirtualNode" data transform from the PyG framework and set all VNs connected for a fair comparison.

Table 14: Performance for virtual nodes (VNs) and neural atoms (NAs) in Peptide-Func, evaluated by AP (the higher, the better).

Model	**Method**	**#VNs /#NAs** $= 5$	**#VNs /#NAs** $= 15$	**#VNs /#NAs** $= 75$	**#VNs /#NAs** $= 135$
GCN	VNs	0.5566	0.5543	0.5568	0.5588
	NAs	**0.5962**	**0.5859**	**0.5903**	**0.6220**
GINE	VNs	0.5437	0.5500	0.5426	0.5426
	NAs	**0.6107**	**0.6128**	**0.6147**	**0.6154**
GCNII	VNs	0.5086	0.5106	0.5077	0.5083
	NAs	**0.6061**	**0.5862**	**0.5909**	**0.5996**
GatedGCN	VNs	0.5810	0.5868	0.5761	0.5810
	NAs	**0.6660**	**0.6533**	**0.6562**	**0.6562**

Table 15: Performance for virtual nodes (VNs) and neural atoms (NAs) in Peptide-Struct, evaluated by MAE (the lower, the better).

Model	**Method**	**#VNs /#NAs** $= 5$	**#VNs /#NAs** $= 15$	**#VNs /#NAs** $= 75$	**#VNs /#NAs** $= 135$
GCN	VNs	0.3499	0.3492	0.3504	0.3492
	NAs	**0.2635**	**0.2581**	**0.2575**	**0.2582**
GINE	VNs	0.3665	0.3614	0.3653	0.3687
	NAs	**0.2624**	**0.2565**	**0.2580**	**0.2598**
GCNII	VNs	0.3686	0.3644	0.3648	0.3632
	NAs	**0.2670**	**0.2577**	**0.2551**	**0.2606**
GatedGCN	VNs	0.3425	0.3398	0.3409	0.3374
	NAs	**0.2596**	**0.2553**	**0.2467**	**0.2473**

As can be seen from Tab. 14 and Tab. 15, neural atoms achieve consistently and significantly better performance than the virtual nodes approach, regardless of their number. In both datasets, a larger

number of neural atoms could lead to a better performance. In contrast, virtual nodes achieve almost identical performance with the increase of their number, even with the pair-wise connections among them.

We speculate that such a phenomenon is because multiple virtual nodes might not learn representative subgraph patterns, which is crucial for the model to learn long-range interaction. The poor performance of multiple virtual nodes might be caused by the overwhelming aggregated information from all atoms within the graph, which leads to over-squashing and decreases the quality of the virtual node embeddings. This aligns with the description in the "Information aggregating" in Tab. 13. In addition, the virtual nodes are simply connected to all atoms within the molecular graph without considering their discrepancy. This could encourage the similarity among atom embeddings, which leads to poor performance. This aligns with the description in the "Backward projection" in Tab. 13.

Thus, we could claim that adopting and increasing the number of virtual nodes could not bring a noticeable improvement; by contrast, neural atoms could alleviate these issues and achieve better performance.

H Neural Atoms Assignment and Interaction Visualization

We visualize the interaction strength for the Mutagenicity dataset. Specifically, we extract the attention weight for both the allocation matrix $\hat{A}$ (upper figure), the attention weights for different neural atoms, which we denote as interaction matrix (lower left figure), and the original molecular graph with atom indices (lower right figure).

Shown as Fig. 17 to 21, the neural atoms aggregate information from different atoms within the molecular graph with varying attention strength. Such attention patterns allow the neural atoms to ensure the diversity of aggregated information by assigning different weights to different atom information. The interactions among neural atoms are established via the neural atom numbered one, which shows weak attention strength for all atoms in the original graph. Such an attention pattern indicates that the model learns to leverage the communication channel, i.e., the Neural Atom numbered one, to bridge the atoms with potential long-range interactions.

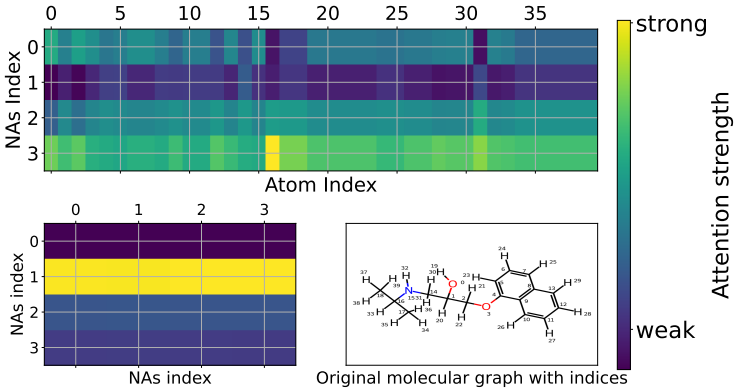


Figure 17: Mutagenicity test set index-18

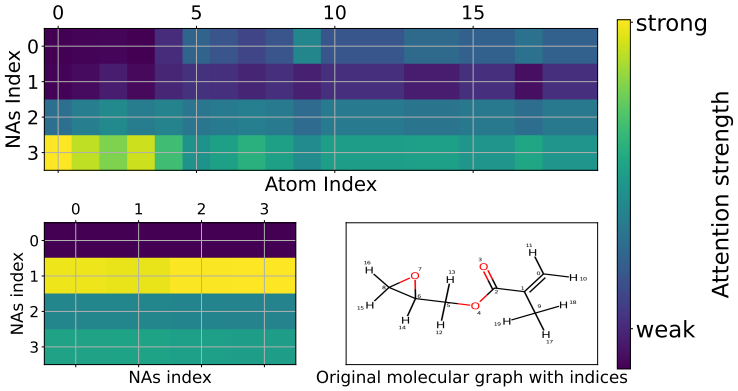


Figure 18: Mutagenicity test set index-26

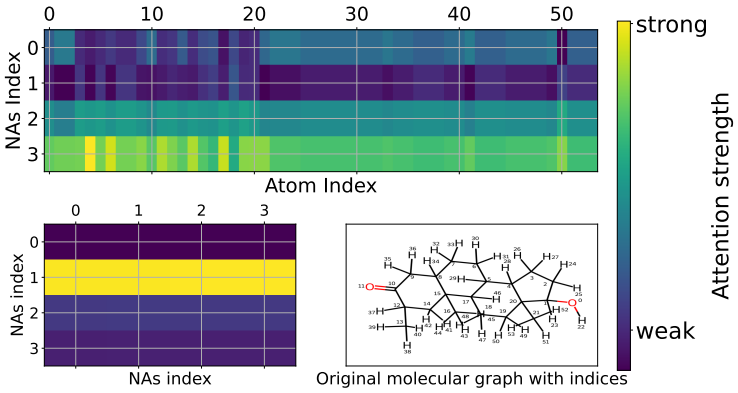


Figure 19: Mutagenicity test set index-29

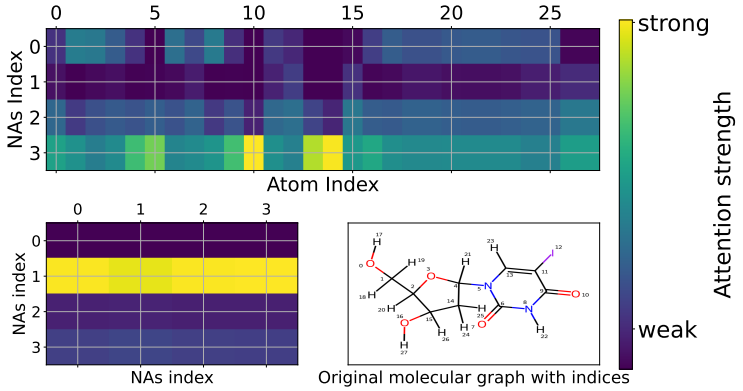


Figure 20: Mutagenicity test set index-36

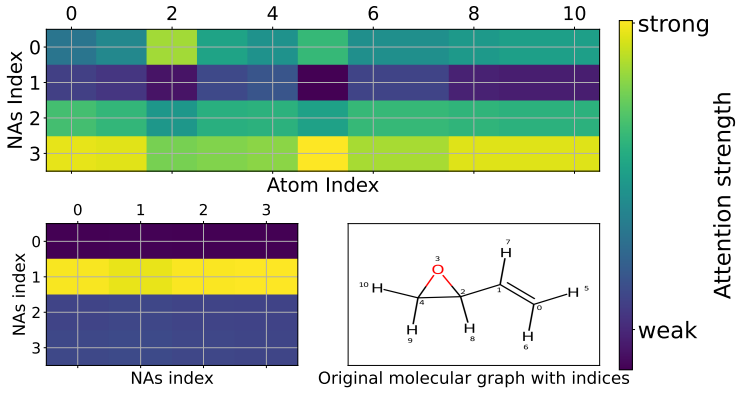


Figure 21: Mutagenicity test set index-211