scientific reports

OPEN

A universal framework for accurate and efficient geometric deep learning of molecular systems

Shuo Zhang[1,2], Yang Liu[1] & Lei Xie[1,2,3]✉

Molecular sciences address a wide range of problems involving molecules of different types and sizes and their complexes. Recently, geometric deep learning, especially Graph Neural Networks, has shown promising performance in molecular science applications. However, most existing works often impose targeted inductive biases to a specific molecular system, and are inefficient when applied to macromolecules or large-scale tasks, thereby limiting their applications to many real-world problems. To address these challenges, we present PAMNet, a universal framework for accurately and efficiently learning the representations of three-dimensional (3D) molecules of varying sizes and types in any molecular system. Inspired by molecular mechanics, PAMNet induces a physics-informed bias to explicitly model local and non-local interactions and their combined effects. As a result, PAMNet can reduce expensive operations, making it time and memory efficient. In extensive benchmark studies, PAMNet outperforms state-of-the-art baselines regarding both accuracy and efficiency in three diverse learning tasks: small molecule properties, RNA 3D structures, and protein-ligand binding affinities. Our results highlight the potential for PAMNet in a broad range of molecular science applications.

The wide variety of molecular types and sizes poses numerous challenges in the computational modeling of molecular systems for drug discovery, structural biology, quantum chemistry, and others[1]. To address these challenges, recent advances in geometric deep learning (GDL) approaches have become increasingly important[2,3]. Especially, Graph Neural Networks (GNNs) have demonstrated superior performance among various GDL approaches[4–6]. GNNs treat each molecule as a graph and perform message passing scheme on it[7]. By representing atoms or groups of atoms like functional groups as nodes, and chemical bonds or any pairwise interactions as edges, molecular graphs can naturally encode the structural information in molecules. In addition to this, GNNs can incorporate symmetry and achieve invariance or equivariance to transformations such as rotations, translations, and reflections[8], which further contributes to their effectiveness in molecular science applications. To enhance their ability to capture molecular structures and increase the expressive power of their models, previous GNNs have utilized auxiliary information such as chemical properties[9–12], atomic pairwise distances in Euclidean space[7,13,14], angular information[15–18], etc.

In spite of the success of GNNs, their application in molecular sciences is still in its early stages. One reason for this is that current GNNs often use targeted inductive bias for modeling a specific type of molecular system, and cannot be directly transferred to other contexts although all molecule structures and their interactions follow the same law of physics. For example, GNNs designed for modeling proteins may include operations that are specific to the structural characteristics of amino acids[19,20], which are not relevant for other types of molecules. Additionally, GNNs that incorporate comprehensive geometric information can be computationally expensive, making them difficult to scale to tasks involving a large number of molecules (e.g., high-throughput compound screening) or macromolecules (e.g., proteins and RNAs). For instance, incorporating angular information can significantly improve the performance of GNNs[15–18], but also increases the complexity of the model, requiring at least $O(Nk^2)$ messages to be computed where N and k denote the number of nodes and the average degree in a graph.

To tackle the limitations mentioned above, we propose a universal GNN framework, <u>P</u>hysics-<u>A</u>ware <u>M</u>ultiplex Graph Neural <u>Net</u>work (PAMNet), for the accurate and efficient representation learning of 3D molecules ranging from small molecules to macromolecules in any molecular system. PAMNet induces a physics-informed bias inspired by molecular mechanics[21], which separately models local and non-local interactions in molecules based

[1]Department of Computer Science, Hunter College, The City University of New York, New York 10065, USA. [2]Helen and Robert Appel Alzheimer's Disease Research Institute, Feil Family Brain and Mind Research Institute, Weill Cornell Medicine, Cornell University, New York 10065, USA. [3]Ph.D. Program in Computer Science, The Graduate Center, The City University of New York, New York 10016, USA. ✉email: lei.xie@hunter.cuny.edu

on different geometric information. To achieve this, we represent each molecule as a two-layer multiplex graph, where one plex only contains local interactions, and the other plex contains additional non-local interactions. PAMNet takes the multiplex graphs as input and uses different operations to incorporate the geometric information for each type of interaction. This flexibility allows PAMNet to achieve efficiency by avoiding the use of computationally expensive operations on non-local interactions, which consist of the majority of interactions in a molecule. Additionally, a fusion module in PAMNet allows the contribution of each type of interaction to be learned and fused for the final feature or prediction. To preserve symmetry, PAMNet utilizes E(3)-invariant representations and operations when predicting scalar properties, and is extended to predict E(3)-equivariant vectorial properties by considering the geometric vectors in molecular structures that arise from quantum mechanics.

To demonstrate the effectiveness of PAMNet, we conduct a comprehensive set of experiments on a variety of tasks involving different molecular systems, including small molecules, RNAs, and protein-ligand complexes. These tasks include predicting small molecule properties, RNA 3D structures, and protein-ligand binding affinities. We compare PAMNet to state-of-the-art baselines in each task and the results show that PAMNet outperforms the baselines in terms of both accuracy and efficiency across all three tasks. Given the diversity of the tasks and the types of molecules involved, the superior performance of PAMNet shows its versatility to be applied in various real-world scenarios.

Overview of PAMNet

Multiplex graph representation

Given any 3D molecule or molecular system, we define a multiplex graph representation as the input of our PAMNet model based on the original 3D structure (Fig. 1a). The construction of multiplex graphs is inspired by molecular mechanics[21], in which the molecular energy E is separately modeled based on local and non-local interactions (Fig. 1c). In detail, the local terms $E_{\text{bond}} + E_{\text{angle}} + E_{\text{dihedral}}$ model local, covalent interactions including E_{bond} that depends on bond lengths, E_{angle} on bond angles, and E_{dihedral} on dihedral angles. The non-local terms $E_{\text{vdW}} + E_{\text{electro}}$ model non-local, non-covalent interactions including van der Waals and electrostatic interactions which depend on interatomic distances. Motivated by this, we also decouple the modeling of these two types of interactions in PAMNet. For local interactions, we can define them either using chemical bonds or by finding the neighbors of each node within a relatively small cutoff distance, depending on the given task. For global interactions that contain both local and non-local ones, we define them by finding the neighbors of each node within a relatively large cutoff distance. For each type of interaction, we use a layer to represent all atoms as nodes and the interactions as edges. The resulting layers that share the same group of atoms form a two-layer multiplex graph $G = \{G_{global}, G_{local}\}$ which represents the original 3D molecular structure (Fig. 1a).

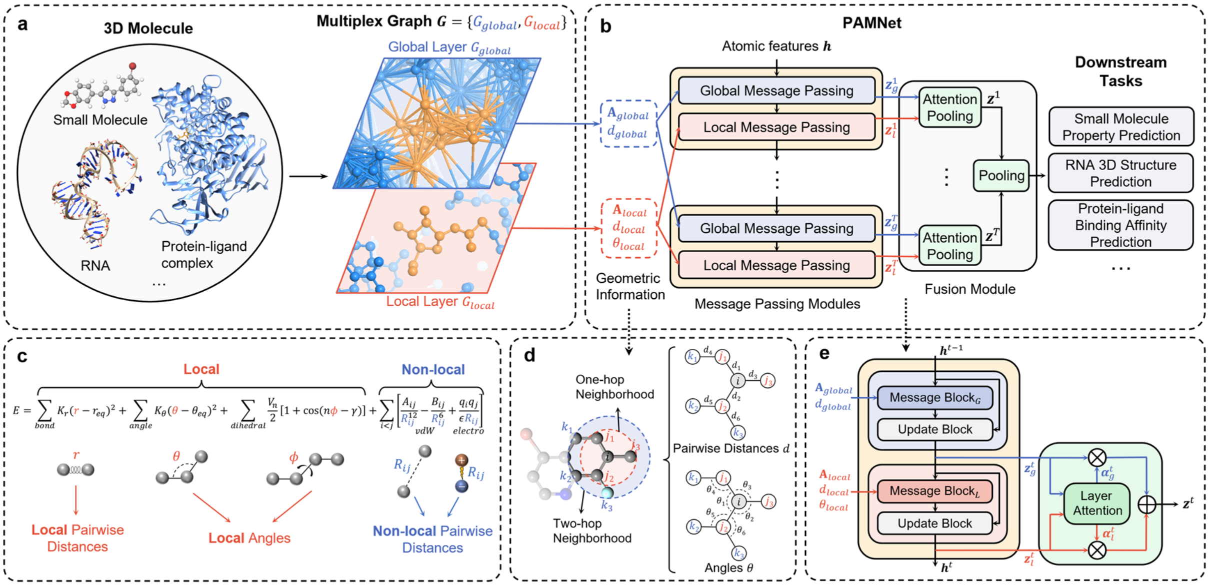

Figure 1. Overview of PAMNet. (**a**), Based on the 3D structure of any molecule or molecular system, a two-layer multiplex graph $G = \{G_{global}, G_{local}\}$ is constructed to separate the modeling of global and local interactions. (**b**), PAMNet takes G as input and learns node-level or graph-level representation for downstream tasks. PAMNet contains stacked message passing modules that update the node embeddings $\boldsymbol{z}$ in G, and a fusion module that learns to combine the updated embeddings. In each message passing module, two message passing schemes are designed to encode the different geometric information in G's two layers. In the fusion module, a two-step pooling process is proposed. (**c**), Calculation of molecular energy E in molecular mechanics. (**d**), An example of the geometric information in G. By considering the one-hop neighbors $\{j\}$ and two-hop neighbors $\{k\}$ of atom i, we can define the pairwise distances d and the related angles θ. (**e**), Detailed architecture of the message passing module and the attention pooling in PAMNet.

Message passing modules

To update the node embeddings in the multiplex graph G, we design two message passing modules that incorporate geometric information: *Global Message Passing* and *Local Message Passing* for updating the node embeddings in G_{global} and G_{local}, respectively (Fig. 1b). These message passing modules are inspired by physical principles from molecular mechanics (Fig. 1c): When modeling the molecular energy E, the terms for local interactions require geometric information including interatomic distances (bond lengths) and angles (bond angles and dihedral angles), while the terms for non-local interactions only require interatomic distances as geometric information. The message passing modules in PAMNet also use geometric information in this way when modeling these interactions (Fig. 1b,e). Specifically, we capture the pairwise distances and angles contained within up to two-hop neighborhoods (Fig. 1d). The *Local Message Passing* requires the related adjacency matrix $\mathbf{A}_{local}$, pairwise distances d_{local} and angles θ_{local}, while the *Global Message Passing* only needs the related adjacency matrix $\mathbf{A}_{global}$ and pairwise distances d_{global}. Each message passing module then learns the node embeddings $\boldsymbol{z}_g$ or $\boldsymbol{z}_l$ in G_{global} and G_{local}, respectively.

For the operations in our message passing modules, they can preserve different symmetries: E(3)-invariance and E(3)-equivariance, which contain essential inductive bias incorporated by GNNs when dealing with graphs with geometric information[8]. E(3)-invariance is preserved when predicting E(3)-invariant scalar quantities like energies, which remain unchanged when the original molecular structure undergoes any E(3) transformation including rotation, translation, and reflection. To preserve E(3)-invariance, the input node embeddings $\boldsymbol{h}$ and geometric features are all E(3)-invariant. To update these features, PAMNet utilizes operations that can preserve the invariance. In contrast, E(3)-equivariance is preserved when predicting E(3)-equivariant vectorial quantities like dipole moment, which will change according to the same transformation applied to the original molecular structure through E(3) transformation. To preserve E(3)-equivariance, an extra associated geometric vector $\boldsymbol{v} \in \mathbb{R}^3$ is defined for each node. These geometric vectors are updated by operations inspired by quantum mechanics[22], allowing for the learning of E(3)-equivariant vectorial representations. More details about the explanations of E(3)-invariance, E(3)-equivariance, and our operations can be found in Methods.

Fusion module

After updating the node embeddings $\boldsymbol{z}_g$ or $\boldsymbol{z}_l$ of the two layers in the multiplex graph G, we design a fusion module with a two-step pooling process to combine $\boldsymbol{z}_g$ and $\boldsymbol{z}_l$ for downstream tasks (Fig. 1b). In the first step, we design an attention pooling module based on attention mechanism[23] for each hidden layer t in PAMNet. Since G_{global} and G_{local} contains the same set of nodes $\{N\}$, we apply the attention mechanism to each node $n \in \{N\}$ to learn the attention weights (α_g^t and α_l^t) between the node embeddings of n in G_{global} and G_{local}, which are $\boldsymbol{z}_g^t$ and $\boldsymbol{z}_l^t$. Then the attention weights are treated as the importance of $\boldsymbol{z}_g^t$ and $\boldsymbol{z}_l^t$ to compute the combined node embedding $\boldsymbol{z}^t$ in each hidden layer t based on a weighted summation (Fig. 1e). In the second step, the $\boldsymbol{z}^t$ of all hidden layers are summed together to compute the node embeddings of the original input. If a graph embedding is desired, we compute it using an average or a summation of the node embeddings.

Results and discussion

In this section, we will demonstrate the performance of our proposed PAMNet regarding two aspects: accuracy and efficiency. Accuracy denotes how well the model performs measured by the metrics corresponding to a given task. Efficiency denotes the memory consumed and the inference time spent by the model.

Performance of PAMNet regarding accuracy

Small molecule property prediction

To evaluate the accuracy of PAMNet in learning representations of small 3D molecules, we choose QM9, which is a widely used benchmark for the prediction of 12 molecular properties of around 130k small organic molecules with up to 9 non-hydrogen atoms[24]. Mean absolute error (MAE) and mean standardized MAE (std. MAE)[15] are used for quantitative evaluation of the target properties. Besides evaluating the original PAMNet which captures geometric information within two-hop neighborhoods of each node, we also develop a "simple" PAMNet, called PAMNet-s, that utilizes only the geometric information within one-hop neighborhoods. The PAMNet models are compared with several state-of-the-art models including SchNet[13], PhysNet[14], MGCN[25], PaiNN[26], DimeNet++[16], and SphereNet[27]. More details of the experiments can be found in Methods and Supplementary Information.

We compare the performance of PAMNet with those of the baseline models mentioned above on QM9, as shown in Table 1. PAMNet achieves 4 best and 6 second-best results among all 12 properties, while PAMNet-s achieves 3 second-best results. When evaluating the overall performance using the std. MAE across all properties, PAMNet and PAMNet-s rank 1 and 2 among all models with 10% and 5% better std. MAE than the third-best model (SphereNet), respectively. From the results, we can observe that the models incorporating only atomic pairwise distance d as geometric information like SchNet, PhysNet, and MGCN generally perform worse than those models incorporating more geometric information like PaiNN, DimeNet++, SphereNet, and our PAMNet. Besides, PAMNet-s which captures geometric information only within one-hop neighborhoods performs worse than PAMNet which considers two-hop neighborhoods. These show the importance of capturing rich geometric information when representing 3D small molecules. The superior performance of PAMNet models demonstrates the power of our separate modeling of different interactions in molecules and the effectiveness of the message passing modules designed.

When predicting dipole moment μ as a scalar value, which is originally an E(3)-equivariant vectorial property $\boldsymbol{\mu}$, PAMNet preserves the E(3)-equivariance to directly predict $\boldsymbol{\mu}$ first and then takes the magnitude of $\boldsymbol{\mu}$ as the final prediction. As a result, PAMNet and PAMNet-s all get lower MAE (10.8 mD and 11.3 mD) than the previous best result (12 mD) achieved by PaiNN, which is a GNN with equivariant operations for predicting vectorial

Property	Unit	SchNet	PhysNet	MGCN	PaiNN	DimeNet++	SphereNet	PAMNet-s	PAMNet
μ	mD	21	52.9	56	12	29.7	24.5	*11.3*	**10.8**
α	a_0^3	0.124	0.0615	**0.030**	0.045	*0.0435*	0.0449	0.0466	0.0447
ϵ_{HOMO}	meV	47	32.9	42.1	27.6	24.6	**22.8**	*23.9*	**22.8**
ϵ_{LUMO}	meV	39	24.7	57.4	20.4	19.5	**18.9**	20.0	*19.2*
$\Delta\epsilon$	meV	74	42.5	64.2	45.7	32.6	*31.1*	32.4	**31.0**
$\langle R^2 \rangle$	a_0^2	0.158	0.765	0.11	**0.066**	0.331	0.268	0.094	*0.093*
ZPVE	meV	1.616	1.39	**1.12**	1.28	1.21	**1.12**	1.24	*1.17*
U_0	meV	12	8.15	12.9	**5.85**	6.32	6.26	6.05	*5.90*
U	meV	12	8.34	14.4	**5.83**	6.28	6.36	6.08	*5.92*
H	meV	12	8.42	16.2	**5.98**	6.53	6.33	6.19	*6.04*
G	meV	13	9.40	14.6	7.35	7.56	7.78	*7.34*	**7.14**
c_v	$\frac{\mathrm{cal}}{\mathrm{molK}}$	0.034	0.0280	0.038	0.024	*0.0230*	**0.0215**	0.0234	0.0231
std. MAE	%	1.78	1.37	1.89	1.01	0.98	0.91	*0.87*	**0.83**

Table 1. Performance comparison on QM9. The best results are marked in bold and the second-best results with Italics.

properties. Note that the remaining baselines all directly predict dipole moment as a scalar property by preserving invariance. We also examine that by preserving invariance in PAMNet and directly predicting dipole moment as a scalar property, the MAE (24.0 mD) is much higher than the equivariant version. These results demonstrate that preserving equivariance is more helpful than preserving invariance for predicting dipole moments.

RNA 3D structure prediction

Besides small molecules, we further apply PAMNet to predict RNA 3D structures for evaluating the accuracy of PAMNet in learning representations of 3D macromolecules. Following the previous works[28–30], we refer the prediction to be the task of identifying accurate structural models of RNA from less accurate ones: Given a group of candidate 3D structural models generated based on an RNA sequence, a desired model that serves as a scoring function needs to distinguish accurate structural models among all candidates. We use the same datasets as those used in[30], which include a dataset for training and a benchmark for evaluation. The training dataset contains 18 relatively older and smaller RNA molecules experimentally determined[31]. Each RNA is used to generate 1000 structural models via the Rosetta FARFAR2 sampling method[29]. The benchmark for evaluation contains relatively newer and larger RNAs, which are the first 21 RNAs in the RNA-Puzzles structure prediction challenge[32]. Each RNA is used to generate at least 1500 structural models using FARFAR2, where 1% of the models are near-native (i.e., within a 2 Å RMSD of the experimentally determined native structure). In practice, each scoring function predicts the root mean square deviation (RMSD) from the unknown true structure for each structural model. A lower RMSD would suggest a more accurate structural model predicted. We compare PAMNet with four state-of-the-art baselines: ARES[30], Rosetta (2020 version)[29], RASP[33], and 3dRNAscore[28]. Among the baselines, only ARES is a deep learning-based method, and is a GNN using equivariant operations. More details of the experiments are introduced in Methods and Supplementary Information.

On the RNA-Puzzles benchmark for evaluation, PAMNet significantly outperforms all other four scoring functions as shown in Fig. 2. When comparing the best-scoring structural model of each RNA (Fig. 2a), the probability of the model to be near-native ($<$ 2 Å RMSD from the native structure) is 90% when using PAMNet, compared with 62, 43, 33, and 5% for ARES, Rosetta, RASP, and 3dRNAscore, respectively. As for the 10 best-scoring structural models of each RNA (Fig. 2b), the probability of the models to include at least one near-native model is 90% when using PAMNet, compared with 81, 48, 48, and 33% for ARES, Rosetta, RASP, and 3dRNAscore, respectively. When comparing the rank of the best-scoring near-native structural model of each RNA (Fig. 2c), the geometric mean of the ranks across all RNAs is 1.7 for PAMNet, compared with 3.6, 73.0, 26.4, and 127.7 for ARES, Rosetta, RASP, and 3dRNAscore, respectively. The lower mean rank of PAMNet indicates that less effort is needed to go down the ranked list of PAMNet to include one near-native structural model. A more detailed analysis of the near-native ranking task can be found in Supplementary Figure S1.

Protein-ligand binding affinity prediction

In this experiment, we evaluate the accuracy of PAMNet in representing the complexes that contain both small molecules and macromolecules. We use PDBbind, which is a well-known public database of experimentally measured binding affinities for protein-ligand complexes[34]. The goal is to predict the binding affinity of each complex based on its 3D structure. We use the PDBbind v2016 dataset and preprocess each original complex to a structure that contains around 300 nonhydrogen atoms on average with only the ligand and the protein residues within 6 Å around it. To comprehensively evaluate the performance, we use Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Pearson's correlation coefficient (R) and the standard deviation (SD) in regression following[18]. PAMNet is compared with various comparative methods including machine learning-based methods (LR, SVR, and RF-Score[35]), CNN-based methods (Pafnucy[36] and OnionNet[37]), and GNN-based methods (GraphDTA[38], SGCN[39], GNN-DTI[40], D-MPNN[12], MAT[41], DimeNet[15], CMPNN[42], and SIGN[18]). More details of the experiments are provided in Methods and Supplementary Information.

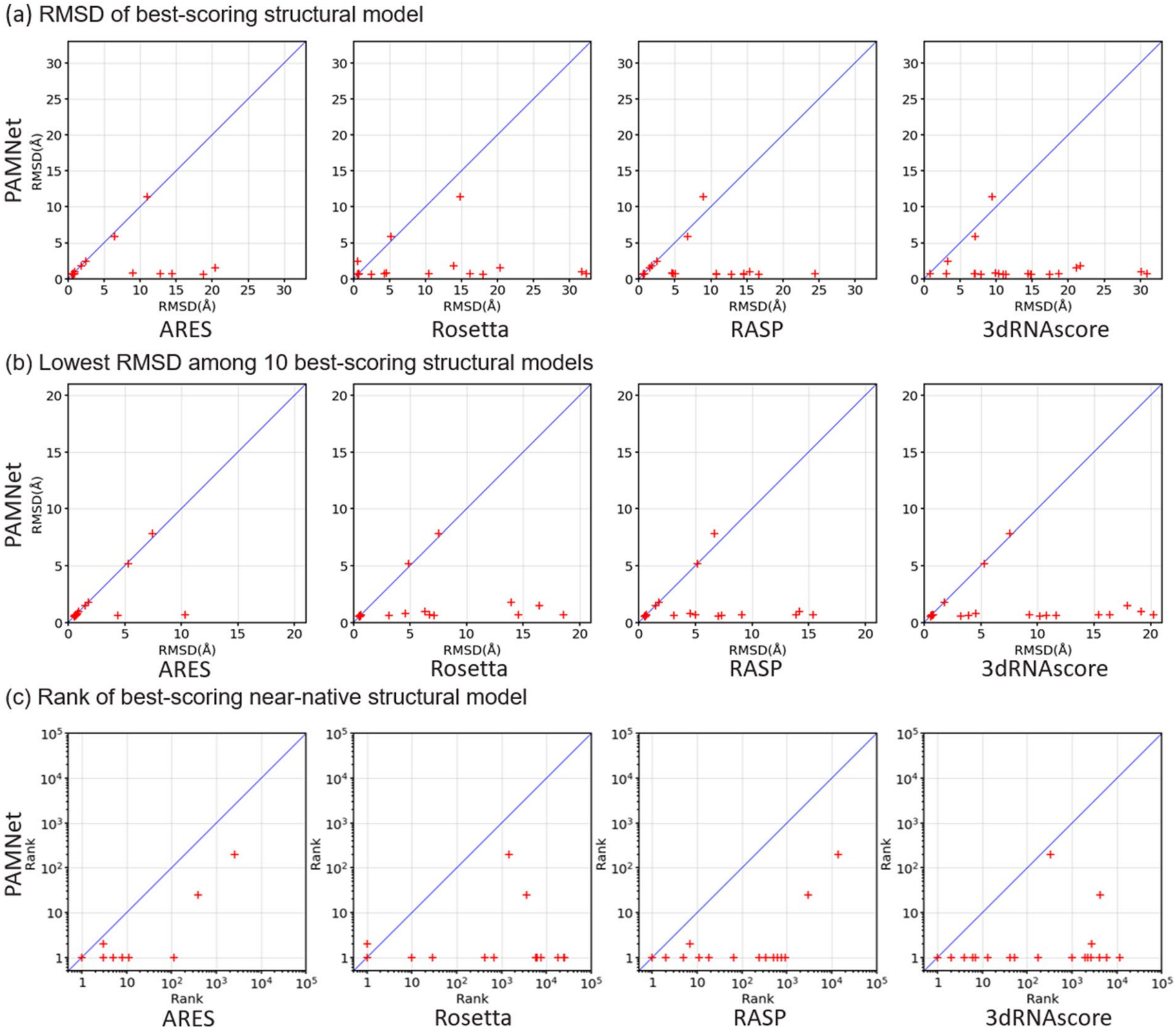

Figure 2. Performance comparison on RNA-Puzzles. Given a group of candidate structural models for each RNA, we rank the models using PAMNet and the other four leading scoring functions for comparison. Each cross in the figures corresponds to one RNA. (**a**) The best-scoring structural model of each RNA predicted by the scoring functions is compared. PAMNet in general identifies more accurate models (with lower RMSDs from the native structure) than those decided by the other scoring functions. (**b**) Comparison of the 10 best-scoring structural models. The identifications of PAMNet contain accurate models more frequently than those from other scoring functions. (**c**) The rank of the best-scoring near-native structural model for each RNA is used for comparison. PAMNet usually performs better than the other scoring functions by having a lower rank.

We list the results of all models and compare their performance in Table 2 and Supplementary Table S1. PAMNet achieves the best performance regarding all 4 evaluation metrics in our experiment. When compared with the second-best model, SIGN, our PAMNet performs significantly better with p-value < 0.05. These results clearly demonstrate the accuracy of our model when learning representations of 3D macromolecule complexes.

In general, we find that the models with explicitly encoded 3D geometric information like DimeNet, SIGN, and our PAMNet outperform the other models without the information directly encoded. An exception is that DimeNet cannot beat CMPNN. This might be because DimeNet is domain-specific and is originally designed for small molecules rather than macromolecule complexes. In contrast, our proposed PAMNet is more flexible to learn representations for various types of molecular systems. The superior performance of PAMNet for predicting binding affinity relies on the separate modeling of local and non-local interactions. For protein-ligand complexes, the local interactions mainly capture the interactions inside the protein and the ligand, while the non-local interactions can capture the interactions between protein and ligand. Thus PAMNet is able to effectively handle diverse interactions and achieve accurate results.

Performance of PAMNet regarding efficiency

To evaluate the efficiency of PAMNet, we compare it to the best-performed baselines in each task regarding memory consumption and inference time and summarize the results in Table 3. Theoretically, DimeNet++, SphereNet,

Model		**RMSE ↓**	**MAE ↓**	**SD ↓**	**R ↑**
ML-based	LR	1.675 (0.000)	1.358 (0.000)	1.612 (0.000)	0.671 (0.000)
	SVR	1.555 (0.000)	1.264 (0.000)	1.493 (0.000)	0.727 (0.000)
	RF-Score	1.446 (0.008)	1.161 (0.007)	1.335 (0.010)	0.789 (0.003)
CNN-based	Pafnucy	1.585 (0.013)	1.284 (0.021)	1.563 (0.022)	0.695 (0.011)
	OnionNet	1.407 (0.034)	1.078 (0.028)	1.391 (0.038)	0.768 (0.014)
GNN-based	GraphDTA	1.562 (0.022)	1.191 (0.016)	1.558 (0.018)	0.697 (0.008)
	SGCN	1.583 (0.033)	1.250 (0.036)	1.582 (0.320)	0.686 (0.015)
	GNN-DTI	1.492 (0.025)	1.192 (0.032)	1.471 (0.051)	0.736 (0.021)
	D-MPNN	1.493 (0.016)	1.188 (0.009)	1.489 (0.014)	0.729 (0.006)
	MAT	1.457 (0.037)	1.154 (0.037)	1.445 (0.033)	0.747 (0.013)
	DimeNet	1.453 (0.027)	1.138 (0.026)	1.434 (0.023)	0.752 (0.010)
	CMPNN	1.408 (0.028)	1.117 (0.031)	1.399 (0.025)	0.765 (0.009)
	SIGN	*1.316* (0.031)	*1.027* (0.025)	*1.312* (0.035)	*0.797* (0.012)
Ours	**PAMNet**	**1.263 (0.017)**	**0.987 (0.013)**	**1.261 (0.015)**	**0.815 (0.005)**

Table 2. Performance comparison on PDBbind. We report the averaged results together with standard deviations. For the evaluation metrics, ↓ denotes the lower the better, while ↑ denotes the higher the better. The best results are marked in bold and the second-best results with italics.

Dataset	**Model**	**Memory (GB)**	**Inference time (s)**
QM9	DimeNet++	21.1	11.3
	SphereNet	22.7	11.1
	PAMNet-s	**6.0**	**7.3**
	PAMNet	6.2	11.0
RNA-Puzzles	ARES	13.5	2.1
	PAMNet	**7.8**	**0.6**
PDBbind	SIGN	19.7	12.0
	PAMNet	**13.1**	**1.8**

Table 3. Results of efficiency evaluation. We compare PAMNet with the best-performed baselines in each of the three tasks regarding memory consumption and inference time. The most efficient results are marked in bold.

and SIGN all require $O(Nk^2)$ messages in message passing, while our PAMNet requires $O(N(k_g + {k_l}^2))$ messages instead, where N is the number of nodes, k is the average degree in a graph, k_g and k_l denotes the average degree in G_g and G_l in the corresponding multiplex graph G. When $k_g \sim k$ and $k_l \ll k_g$, PAMNet is much more efficient regarding the number of messages involved. A more detailed analysis of computational complexity is included in Methods. Based on the results in Table 3 empirically, we find PAMNet models all require less memory consumption and inference time than the best-performed baselines in all three tasks, which matches our theoretical analysis. We also compare the memory consumption when using a different largest cutoff distance d of the related models in Fig. 3. From the results, we observe that the memory consumed by DimeNet and SIGN increases much faster than PAMNet when d increases. When fixing $d = 5$ Å as an example, PAMNet requires 80% and 71% less memory than DimeNet and SIGN, respectively. Thus PAMNet is much more memory-efficient and is able to capture longer-range interactions than these baselines with restricted resources. The efficiency of PAMNet models comes from the separate modeling of local and non-local interactions in 3D molecular structures. By doing so, when modeling the non-local interactions, which make up the majority of all interactions, we utilize a relatively efficient message passing scheme that only encodes pairwise distances d as the geometric information. Thus when compared with the models that require more comprehensive geometric information when modeling all interactions, PAMNet significantly reduces the computationally expensive operations. More information about the details of experimental settings is included in Methods.

All components in PAMNet contribute to the performance

To figure out whether all of the components in PAMNet, including the fusion module and the message passing modules, contribute to the performance of PAMNet, we conduct an ablation study by designing PAMNet variants. Without the attention pooling, we use the averaged results from the message passing modules in each hidden layer to build a variant. We also remove either the Local Message Passing or the Global Message Passing for investigation. The performances of all PAMNet variants are evaluated on the three benchmarks. Specifically, the std. MAE across all properties on QM9, the geometric mean of the ranks across all RNAs on RNA-Puzzles, and the four metrics used in the experiment on PDBbind are computed for comparison. The results in Fig. 4 show

Attention weight	QM9											RNA-puzzles	PDBbind
	μ	α	ϵ_{HOMO}	ϵ_{LUMO}	$\langle R^2 \rangle$	ZPVE	U_0	U	H	G	c_v		
$\overline{\alpha_l}$	**0.64**	**0.53**	0.50	0.50	0.29	**0.54**	**0.60**	**0.60**	**0.60**	**0.57**	**0.58**	0.22	0.34
$\overline{\alpha_g}$	0.36	0.47	0.50	0.50	**0.71**	0.46	0.40	0.40	0.40	0.43	0.42	**0.78**	**0.66**

Table 4. Comparison of the average attention weights $\overline{\alpha_l}$ and $\overline{\alpha_g}$ for local and global interactions in attention pooling. The higher attention weight for each target is marked in bold.

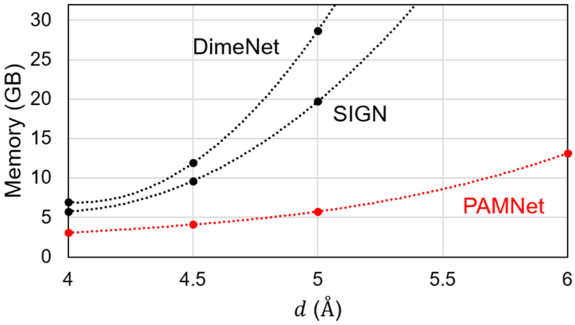

Figure 3. Memory consumption vs. the largest cutoff distance d on PDBbind. We compare PAMNet with the GNN baselines that also explicitly incorporate the 3D molecular geometric information like pairwise distances and angles.

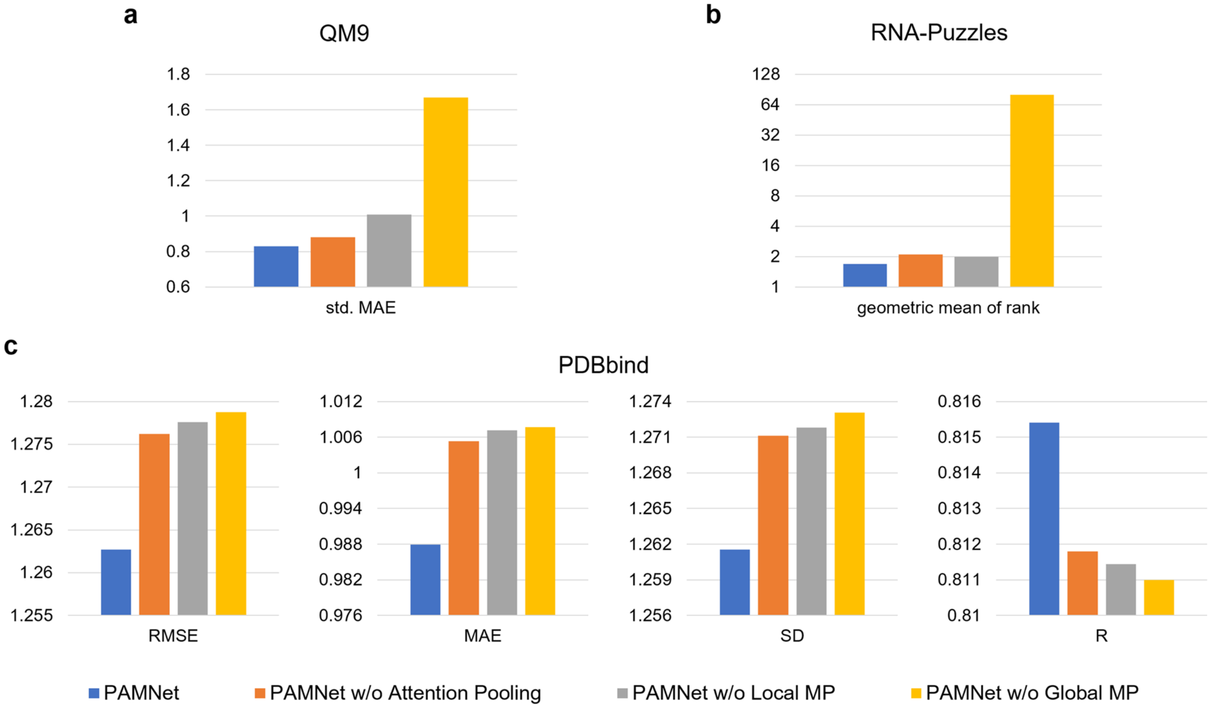

Figure 4. Ablation study of PAMNet. We compare the variants with the original PAMNet and report the differences.

that all variants decrease the performance of PAMNet in the evaluations, which clearly validates the contributions of all those components. Detailed results of the properties on QM9 can be found in Supplementary Table S2.

Analysis of the contribution of local and global interactions

A salient property of PAMNet is the incorporation of the attention mechanism in the fusion module, which takes the importance of node embeddings in G_{local} and G_{global} of G into consideration in learning combined node embeddings. Recall that for each node n in the set of nodes $\{N\}$ in G, the attention pooling in the fusion module learns the attention weights α_l and α_g between n's node embedding $\boldsymbol{z}_l$ in G_{local} and n's node embedding $\boldsymbol{z}_g$ in G_{global}. α_l and α_g serve as the importance of $\boldsymbol{z}_l$ and $\boldsymbol{z}_g$ when computing the combined node embedding $\boldsymbol{z}$. To better understand the contribution of $\boldsymbol{z}_l$ and $\boldsymbol{z}_g$, we conduct a detailed analysis of the learned attention weights α_l and α_g in the three tasks we experimented with. Since the node embeddings are directly related to the involved interactions, such analysis can also reveal the contribution of local and global interactions on the predictions in different tasks. In each task, we take an average of all α_l or α_g to be the overall importance of the corresponding group of interactions. Then we compare the computed average attention weights $\overline{\alpha_l}$ and $\overline{\alpha_g}$ and list the results in Table 4. A higher attention weight in each task indicates a stronger contribution of the corresponding interactions on solving the task.

For the targets being predicted in QM9, we find that all of them have $\overline{\alpha_l} \geq \overline{\alpha_g}$ except the electronic spatial extent $\langle R^2 \rangle$, indicating a stronger contribution of the local interactions, which are defined by chemical bonds in this task. This may be because QM9 contains small molecules with only up to 9 non-hydrogen atoms, local interactions can capture a considerable portion of all atomic interactions. However, when predicting electronic spatial extent $\langle R^2 \rangle$, we notice that $\overline{\alpha_l} < \overline{\alpha_g}$, which suggests that $\langle R^2 \rangle$ is mainly affected by the global interactions that are the pairwise interactions within 10 Å in this case. This is not surprising since $\langle R^2 \rangle$ is the electric field area affected by the ions in the molecule, and is directly related to the diameter or radius of the molecule. Besides, previous study[43] has demonstrated that graph properties like diameter and radius cannot be computed by message passing-based GNNs that rely entirely on local information, and additional global information is needed. Thus it is expected that global interactions have a stronger contribution than local interactions on predicting electronic spatial extent.

For the RNA 3D structure prediction on RNA-Puzzles and the protein-ligand binding affinity prediction on PDBbind, we find $\overline{\alpha_l} < \overline{\alpha_g}$ in both cases, which indicates that global interactions play a more important role than local interactions. It is because the goals of these two tasks highly rely on global interactions, which are necessary for representing the global structure of RNA when predicting RNA 3D structure, and are crucial for capturing the relationships between protein and ligand when predicting binding affinity.

Conclusion

In this work, we tackle the limitations of previous GNNs regarding their limited applicability and inefficiency for representation learning of molecular systems with 3D structures and propose a universal framework, PAMNet, to accurately and efficiently learn the representations of 3D molecules in any molecular system. PAMNet explicitly models local and non-local interaction as well as their combined effects inspired by molecular mechanics. The resulting framework incorporates rich geometric information like distances and angles when modeling local interactions, and avoids using expensive operations on modeling non-local interactions. Besides, PAMNet learns the contribution of different interactions to combine the updated node embeddings for the final output. When designing the aforementioned operations in PAMNet, we preserve E(3)-invariance for scalar output and preserve E(3)-equivariance for vectorial output to enable more applicable cases. In our experiments, we evaluate the performance of PAMNet with state-of-the-art baselines on various tasks involving different molecular systems, including small molecules, RNAs, and protein-ligand complexes. In each task, PAMNet outperforms the corresponding baselines in terms of both accuracy and efficiency. These results clearly demonstrate the generalization power of PAMNet even though non-local interactions in molecules are modeled with only pairwise distances as geometric information.

An under-investigated aspect of our proposed PAMNet is that PAMNet preserves E(3)-invariance in operations when predicting scalar properties while requiring additional representations and operations to preserve E(3)-equivariance for vectorial properties. Considering that various equivariant GNNs have been proposed for predicting either scalar or vectorial properties solely by preserving equivariance, it would be worth extending the idea in PAMNet to equivariant GNNs with a potential to further improve both accuracy and efficiency. Another interesting direction is that although we only experiment PAMNet on single-task learning, PAMNet is promising to be used in multi-task learning across diverse tasks that involve molecules of varying sizes and types to gain better generalization. Besides using PAMNet for predicting physiochemical properties of molecules, PAMNet can be used as a universal building block for the representation learning of molecular systems in various molecular science problems. Another promising application of PAMNet is self-supervised learning for molecular systems with few labeled data (e.g., RNA structures). For example, we can use the features in one graph layer to learn properties in another graph layer by utilizing the multiplex nature of PAMNet.

Methods

Details of PAMNet

In this section, we will describe PAMNet in detail, including the involved features, embeddings, and operations.

Input features

The input features of PAMNet include atomic features and geometric information as shown in Fig. 1b. For atomic features, we use only atomic numbers Z for the tasks on QM9 and RNA-Puzzles following[13–16, 30], and use 18 chemical features like atomic numbers, hybridization, aromaticity, partial charge, etc., for the task on PDBbind following[18, 36]. The atomic numbers Z are represented by randomly initialized, trainable embeddings according to[13–16]. For geometric information, we capture the needed pairwise distances and angles in the multiplex molecular graph G as shown in Fig. 1d. The features (d, θ) for the distances and angles are computed with the basis functions in[15] to reduce correlations. For the prediction of vectorial properties, we use the atomic position $\boldsymbol{r}$ to be the initial associated geometric vector $\boldsymbol{v}$ of each atom.

Message embeddings

In the message passing scheme[7], the update of node embeddings $\boldsymbol{h}$ relies on the passing of the related messages $\boldsymbol{m}$ between nodes. In PAMNet, we define the input message embeddings $\boldsymbol{m}$ of message passing schemes with the following way:

$$\boldsymbol{m}_{ji} = \mathrm{MLP}_{\mathrm{m}}([\boldsymbol{h}_{\mathrm{j}} | \boldsymbol{h}_{\mathrm{i}} | \boldsymbol{e}_{\mathrm{ji}}]), \tag{1}$$

where $i, j \in G_{global}$ or G_{local} are connected nodes that can define a message embedding, MLP denotes the multi-layer perceptron, | denotes the concatenation operation. The edge embedding $\boldsymbol{e}_{ji}$ encodes the corresponding pairwise distance d between node i, j.

Global message passing

As depicted in Fig. 1e, the Global Message Passing in each hidden layer of PAMNet, which consists of a message block and an update block, updates the node embeddings $\boldsymbol{h}$ in G_{global} by using the related adjacency matrix $\mathbf{A}_{global}$ and pairwise distances d_{global}. The message block is defined as below to perform the message passing operation:

$$\boldsymbol{h}_i^t = \boldsymbol{h}_i^{t-1} + \sum\nolimits_{j \in \mathcal{N}(i)} \boldsymbol{m}_{ji}^{t-1} \odot \phi_d(\boldsymbol{e}_{ji}), \tag{2}$$

where $i, j \in G_{global}$, ϕ_d is a learnable function, $\boldsymbol{e}_{ji}$ is the embedding of pairwise distance d between node i, j, and $\odot$ denotes the element-wise production. After the message block, an update block is used to compute the node embeddings $\boldsymbol{h}$ for the next layer as well as the output $\boldsymbol{z}$ for this layer. We define the update block using a stack of three residual blocks, where each residual block consists of a two-layer MLP and a skip connection across the MLP. There is also a skip connection between the input of the message block and the output of the first residual block. After the residual blocks, the updated node embeddings $\boldsymbol{h}$ are passed to the next layer. For the output $\boldsymbol{z}$ of this layer to be combined in the fusion module, we further use a three-layer MLP to get $\boldsymbol{z}$ with desired dimension size.

Local message passing

For the updates of node embeddings $\boldsymbol{h}$ in G_{local}, we incorporate both pairwise distances d_{local} and angles θ_{local} as shown in Fig. 1e. To capture θ_{local}, we consider up to the two-hop neighbors of each node. In Fig. 1d, we show an example of the angles we considered: Some angles are between one-hop edges and two-hop edges (e.g. $\angle ij_1k_1$), while the other angles are between one-hop edges (e.g. $\angle j_1ij_2$). Compared to previous GNNs[15–17] that incorporate only part of these angles, our PAMNet is able to encode the geometric information more comprehensively. In the Local Message Passing, we also use a message block and an update block following the design of the Global Message Passing as shown in Fig. 1e. However, the message block is defined differently as the one in the Global Message Passing to encode additional angular information:

$$\boldsymbol{m}_{ji}^{'t-1} = \boldsymbol{m}_{ji}^{t-1} + \sum_{j' \in \mathcal{N}(i) \setminus \{j\}} \boldsymbol{m}_{j'i}^{t-1} \odot \phi_d(\boldsymbol{e}_{j'i}) \odot \phi_\theta(\boldsymbol{a}_{j'i,ji}) + \sum_{k \in \mathcal{N}(j) \setminus \{i\}} \boldsymbol{m}_{kj}^{t-1} \odot \phi_d(\boldsymbol{e}_{kj}) \odot \phi_\theta(\boldsymbol{a}_{kj,ji}), \tag{3}$$

$$\boldsymbol{h}_i^t = \boldsymbol{h}_i^{t-1} + \sum_{j \in \mathcal{N}(i)} \boldsymbol{m}_{ji}^{'t-1} \odot \phi_d(\boldsymbol{e}_{ji}), \tag{4}$$

where $i, j, k \in G_{local}$, $\boldsymbol{e}_{ji}$ is the embedding of pairwise distance d between node i, j, $\boldsymbol{a}_{kj,ji}$ is the embedding of angle $\theta_{kj,ji} = \angle kji$ defined by node i, j, k, and ϕ_d, ϕ_θ are learnable functions. In Eq. (3), we use two summation terms to separately encode the angles in different hops with the associated pairwise distances to update $\boldsymbol{m}_{ji}$. Then in Eq. (4), the updated message embeddings $\boldsymbol{m}'_{ji}$ are used to perform message passing. After the message block, we use the same update block as the one used in the Global Message Passing for updating the learned node embeddings.

Fusion module

The fusion module consists of two steps of pooling as shown in Fig. 1b. In the first step, attention pooing is utilized to learn the combined embedding $\boldsymbol{z}^t$ based on the output node embeddings $\boldsymbol{z}_g^t$ and $\boldsymbol{z}_l^t$ in each hidden layer t. The detailed architecture of attention pooling is illustrated in Fig. 1e. We first compute the attention weight $\alpha_{p,i}$ on node i that measures the contribution of the results from plex or graph layer $p \in \{g, l\}$ in multiplex graph G:

$$\alpha_{p,i}^t = \frac{\exp\left(\mathrm{LeakyReLU}\left(\mathbf{W}_p^t \boldsymbol{z}_{p,i}^t\right)\right)}{\sum_p \exp\left(\mathrm{LeakyReLU}\left(\mathbf{W}_p^t \boldsymbol{z}_{p,i}^t\right)\right)}, \tag{5}$$

where $\mathbf{W}_p^t \in \mathbb{R}^{1 \times F}$ is a learnable weight matrix different for each hidden layer t and graph layer p, and F is the dimension size of $\boldsymbol{z}_{p,i}^t$. With $\alpha_{p,i}^t$, we can compute the combined node embedding $\boldsymbol{z}_i^t$ of node i using a weighted summation:

$$\boldsymbol{z}_i^t = \sum\nolimits_p \alpha_{p,i}^t \left(\mathbf{W}_p^{'t} \boldsymbol{z}_{p,i}^t\right), \tag{6}$$

where $\mathbf{W}_p^{'t} \in \mathbb{R}^{D \times F}$ is a learnable weight matrix different for each hidden layer t and graph layer p, D is the dimension size of $\boldsymbol{z}_i^t$, and F is the dimension size of $\boldsymbol{z}_{p,i}^t$.

In the second step of the fusion module, we sum the combined node embedding $\boldsymbol{z}$ of all hidden layers to compute the final node embeddings $\boldsymbol{y}$. If a graph-level embedding $\boldsymbol{y}$ is desired, we compute as follows:

$$\boldsymbol{y} = \sum\nolimits_{i=1}^{N} \sum\nolimits_{t=1}^{T} \boldsymbol{z}_i^t. \tag{7}$$

Preservation of E(3)-invariance & E(3)-equivariance

For the operations described above, they preserve the E(3)-invariance of the input atomic features and geometric information and can predict E(3)-invariant scalar properties. To predict E(3)-equivariant vectorial property $\boldsymbol{u}$, we introduce an associated geometric vector $\boldsymbol{v}_i$ for each node i and extend PAMNet to preserve the E(3)-equivariance for learning $\boldsymbol{u}$. In detail, the associated geometric vector $\boldsymbol{v}_i^t$ of node i in hidden layer t is defined as:

$$\boldsymbol{v}_i^t = f_v(\{\boldsymbol{h}^t\}, \{\boldsymbol{r}\}), \tag{8}$$

where $\{\boldsymbol{h}^t\}$ denotes the set of learned node embeddings of all nodes in hidden layer t, $\{\boldsymbol{r}\}$ denotes the set of position vectors of all nodes in 3d coordinate space, and f_v is a function that preserves the E(3)-equivariance of $\boldsymbol{v}_i^t$ with respect to $\{\boldsymbol{r}\}$. Equation (8) is computed after each message passing module in PAMNet.

To predict a final vectorial property $\boldsymbol{u}$, we modify Eqs. (6) and (7) in the fusion module as the following operations:

$$\boldsymbol{u}_i^t = \sum\nolimits_{p} \alpha_{p,i}^t \left(\mathbf{W}'^t_p \boldsymbol{z}_{p,i}^t\right) \boldsymbol{v}_{p,i}^t, \tag{9}$$

$$\boldsymbol{u} = \sum\nolimits_{i=1}^{N} \sum\nolimits_{t=1}^{T} \boldsymbol{u}_i^t, \tag{10}$$

where $\boldsymbol{v}_{p,i}^t$ is the associated geometric vector of node i on graph layer p in hidden layer t, $\boldsymbol{u}_i^t$ is the learned vector of node i in hidden layer t, and $\mathbf{W}'^t_p \in \mathbb{R}^{1\times F}$ is a learnable weight matrix different for each hidden layer t and graph layer p. In Eq. (9), we multiply $\boldsymbol{v}_{p,i}^t$ with the learned scalar node contributions. In Eq. (10), we sum all node-level vectors in all hidden layers to compute the final prediction $\boldsymbol{u}$.

For predicting dipole moment $\boldsymbol{\mu}$, which is an E(3)-equivariant vectorial property that describes the net molecular polarity in electric field, we design f_v in Eq. (8) as motivated by quantum mechanics[44]. The conventional method to compute molecular dipole moment involves approximating electronic charge densities as concentrated at each atomic position, resulting in $\boldsymbol{\mu} = \sum_i \boldsymbol{r}_{c,i} q_i$, where q_i is the partial charge of node i, and $\boldsymbol{r}_{c,i} = \boldsymbol{r}_i - \left(\sum_i \boldsymbol{r}_i\right)/N$ is the relative atomic position of node i. However, this approximation is not accurate enough. Instead, we use a more accurate approximation by adding dipoles onto atomic positions in the distributed multipole analysis (DMA) approach[22]. This results in the dipole moment equation: $\boldsymbol{\mu} = \sum_i (\boldsymbol{r}_{c,i} q_i + \boldsymbol{\mu}_i)$, where $\boldsymbol{\mu}_i$ is the associated partial dipole of node i. The equation can be rewritten as $\boldsymbol{\mu} = \sum_i f_v(\boldsymbol{r}_i) q_i$, where q_i is the scalar atomic contribution that can be modeled by an invariant fashion. By treating $f_v(\boldsymbol{r}_i)$ as $\boldsymbol{v}_i^t$, the equation has a similar format as a combination of Eqs. (9) and (10). We update $\boldsymbol{v}_i^t$ in the following way:

$$\boldsymbol{v}_i^t = \sum\nolimits_{j\in\mathcal{N}(i)} (\boldsymbol{r}_i - \boldsymbol{r}_j)\|\boldsymbol{m}_{ij}^t\|, \tag{11}$$

where $\|\cdot\|$ denotes the L2 norm. Since $\boldsymbol{v}_i^t$ as well as $\boldsymbol{\mu}$ are computed by a linear combination of $\{\boldsymbol{r}\}$, our PAMNet can preserve E(3)-equivariance with respect to $\{\boldsymbol{r}\}$ when performing the prediction.

Computational complexity

We analyze the computational complexity of PAMNet by addressing the number of messages. We denote the cutoff distance when creating the edges as d_g and d_l in G_g and G_l. The average degree is k_g in G_g and is k_l in G_l. In each hidden layer of PAMNet, Global Message Passing needs $O(Nk_g)$ messages because it requires one message for each pairwise distance between the central node and its one-hop neighbor. While Local Message Passing requires one message for each one-hop or two-hop angle around the central node. The number of angles can be estimated as follows: For k edges connected to a node, they can define $(k(k-1))/2$ angles which result in a complexity of $O\left(Nk^2\right)$. The number of one-hop angles and two-hop angles all has such complexity. So that Local Message Passing needs $O\left(2N{k_l}^2\right)$ messages. In total, PAMNet requires the computation of $O\left(Nk_g + 2N{k_l}^2\right)$ messages in each hidden layer, while previous approaches[15–18, 27] require $O\left(N{k_g}^2\right)$ messages. For 3D molecules, we have $k_g \propto {d_g}^3$ and $k_l \propto {d_l}^3$. With proper choices of d_l and d_g, we have $k_l \ll k_g$. In such cases, our model is more efficient than the related GNNs. We here list the comparison of the number of messages needed in our experiments as an example: On QM9 with $d_g = 5$ Å, our model needs 0.5k messages/molecule on average, while DimeNet++ needs 4.3k messages. On PDBBind with $d_l = 2$ Å and $d_g = 6$ Å, our model needs only 12k messages/molecule on average, while DimeNet++ needs 264k messages.

Data collection and processing

QM9

For QM9, we use the source provided by[24]. Following the previous works[15–17], we process QM9 by removing about 3k molecules that fail a geometric consistency check or are difficult to converge[45]. For properties U_0, U, H, and G, only the atomization energies are used by subtracting the atomic reference energies as in[15–17]. For property $\Delta\epsilon$, we follow the same way as the DFT calculation and predict it by calculating $\epsilon_{\mathrm{LUMO}} - \epsilon_{\mathrm{HOMO}}$. For property μ, the final result is the magnitude of the predicted vectorial $\boldsymbol{\mu}$ when using our geometric vector-based approaches with PAMNet. The 3D molecular structures are processed using the RDKit library[46]. Following[15], we randomly use 110000 molecules for training, 10000 for validation and 10831 for testing. In our multiplex molecular graphs, we use chemical bonds as the edges in the local layer, and a cutoff distance (5 or 10 Å) to create the edges in the global layer.

RNA-puzzles
RNA-Puzzles consists of the first 21 RNAs in the RNA-Puzzles structure prediction challenge[32]. Each RNA is used to generate at least 1500 structural models using FARFAR2, where 1% of the models are near native (i.e., within a 2 Å RMSD of the experimentally determined native structure). Following[30], we only use the carbon, nitrogen, and oxygen atoms in RNA structures. When building multiplex graphs for RNA structures, we use cutoff distance $d_l = 2.6$ Å for the local interactions in G_{local} and $d_g = 20$ Å for the global interactions in G_{global}.

PDBBind
For PDBBind, we use PDBbind v2016 following[18, 36]. Besides, we use the same data splitting method according to[18] for a fair comparison. In detail, we use the core subset which contains 290 complexes in PDBbind v2016 for testing. The difference between the refined and core subsets, which includes 3767 complexes, is split with a ratio of 9:1 for training and validation. We use $\log K_i$ as the target property being predicted, which is proportional to the binding free energy. In each complex, we exclude the protein residues that are more than 6 Å from the ligand and remove all hydrogen atoms. The resulting complexes contain around 300 atoms on average. In our multiplex molecular graphs, we use cutoff distance $d_l = 2$ Å in the local layer and $d_g = 6$ Å in the global layer.

Experimental settings

In our message passing operations, we define $\phi_d(\boldsymbol{e}) = \mathbf{W}_e\boldsymbol{e}$ and $\phi_\theta(\boldsymbol{\alpha}) = \mathrm{MLP}_\alpha(\boldsymbol{\alpha})$, where $\mathbf{W}_e$ is a weight matrix, MLP_α is a multi-layer perceptron (MLP). All MLPs used in our model have two layers by taking advantage of the approximation capability of MLP[47]. For all activation functions, we use the self-gated Swish activation function[48]. For the basis functions, we use the same parameters as in[15]. To initialize all learnable parameters, we use the default settings used in PyTorch without assigning specific initializations except the initialization for the input node embeddings on QM9: $\boldsymbol{h}$ are initialized with random values uniformly distributed between $-\sqrt{3}$ and $\sqrt{3}$. In all experiments, we use the Adam optimizer[49] to minimize the loss. In Supplementary Table S3, we list the typical hyperparameters used in our experiments. All of the experiments are done on an NVIDIA Tesla V100 GPU (32 GB).

Small molecule property prediction
In our experiment on QM9, we use the single-target training following[15] by using a separate model for each target instead of training a single shared model for all targets. The models are optimized by minimizing the mean absolute error (MAE) loss. We use a linear learning rate warm-up over 1 epoch and an exponential decay with a ratio 0.1 every 600 epochs. The model parameter values for validation and testing are kept using an exponential moving average with a decay rate of 0.999. To prevent overfitting, we use early stopping on the validation loss. For properties ZPVE, U_0, U, H, and G, we use the cutoff distance in the global layer $d_g = 5$ Å. For the other properties, we use $d_g = 10$ Å. We repeat our runs 3 times for each PAMNet variant following[50].

RNA 3D structure prediction
PAMNet is optimized by minimizing the smooth L1 loss[51] between the predicted value and the ground truth. An early-stopping strategy is adopted to decide the best epoch based on the validation loss.

Protein-ligand binding affinity prediction
We create three weight-sharing, replica networks, one each for predicting the target G of complex, protein pocket, and ligand following[52]. The final target is computed by $\Delta G_{\text{complex}} = G_{\text{complex}} - G_{\text{pocket}} - G_{\text{ligand}}$. The full model is trained by minimizing the mean absolute error (MAE) loss between $\Delta G_{\text{complex}}$ and the true values. The learning rate is dropped by a factor of 0.2 every 50 epochs. Moreover, we perform 5 independent runs according to[18].

Efficiency comparison
In the experiment on investigating the efficiency of PAMNet, we use NVIDIA Tesla V100 GPU (32 GB) for a fair comparison. For small molecule property prediction, we use the related models for predicting property U_0 of QM9 as an example. We use batch size=128 for all models and use the configurations reported in the corresponding papers. For RNA 3D structure prediction, we use PAMNet and ARES to predict the structural models of RNA in puzzle 5 of RNA-Puzzles challenge. The RNA being predicted has 6034 non-hydrogen atoms. The model settings of PAMNet and ARES are the same as those used for reproducing the best results. We use batch size=8 when performing the predictions. For protein-ligand binding affinity prediction, we use the configurations that can reproduce the best results for the related models.

Data availability

The QM9 dataset is available at https://figshare.com/collections/Quantum_chemistry_structures_and_properties_of_134_kilo_molecules/978904. The datasets for RNA 3D structure prediction can be found at https://purl.stanford.edu/bn398fc4306. The PDBbind v2016 dataset is available at http://www.pdbbind.org.cn/ or https://github.com/PaddlePaddle/PaddleHelix/tree/dev/apps/drug_target_interaction/sign.

Code availability

The source code of our model is publicly available on GitHub at the following repository: https://github.com/XieResearchGroup/Physics-aware-Multiplex-GNN.

Received: 23 May 2023; Accepted: 31 October 2023
Published online: 06 November 2023

References

1. Holtje, H.-D., Sippl, W., Rognan, D. & Folkers, G. *Molecular Modeling* Vol. 5 (Wiley-VCH Weinheim, Germany, 2003).
2. Atz, K., Grisoni, F. & Schneider, G. Geometric deep learning on molecular representations. *Nat. Mach. Intell.* 1–10 (2021).
3. Isert, C., Atz, K. & Schneider, G. in *Structure-Based Drug Design with Geometric Deep Learning*. arXiv:2210.11250 (2022).
4. Sun, M. *et al.* Graph convolutional networks for computational drug development and discovery. *Brief. Bioinform.* **21**, 919–935 (2020).
5. Bronstein, M. M., Bruna, J., Cohen, T. & Veličković, P. *Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges*. arXiv:2104.13478 (2021).
6. Reiser, P. *et al.* Graph neural networks for materials science and chemistry. *Commun. Mater.* **3**, 93 (2022).
7. Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O. & Dahl, G. E. Neural message passing for quantum chemistry. In *International Conference on Machine Learning* 1263–1272 (PMLR, 2017).
8. Han, J., Rong, Y., Xu, T. & Huang, W. *Geometrically Equivariant Graph Neural Networks: A Survey*. arXiv:2202.07230 (2022).
9. Duvenaud, D. K. *et al.* Convolutional networks on graphs for learning molecular fingerprints. *Adv. Neural Inform. Process. Syst.* 2224–2232 (2015).
10. Kearnes, S., McCloskey, K., Berndl, M., Pande, V. & Riley, P. Molecular graph convolutions: Moving beyond fingerprints. *J. Comput. Aided Mol. Des.* **30**, 595–608 (2016).
11. Chen, C., Ye, W., Zuo, Y., Zheng, C. & Ong, S. P. Graph networks as a universal machine learning framework for molecules and crystals. *Chem. Mater.* **31**, 3564–3572 (2019).
12. Yang, K. *et al.* Analyzing learned molecular representations for property prediction. *J. Chem. Inf. Model.* **59**, 3370–3388 (2019).
13. Schutt, K. *et al.* Schnetpack: A deep learning toolbox for atomistic systems. *J. Chem. Theory Comput.* **15**, 448–455 (2018).
14. Unke, O. T. & Meuwly, M. Physnet: A neural network for predicting energies, forces, dipole moments, and partial charges. *J. Chem. Theory Comput.* **15**, 3678–3693 (2019).
15. Klicpera, J., Groß, J. & Günnemann, S. Directional message passing for molecular graphs. In *International Conference on Learning Representations* (2020).
16. Klicpera, J., Giri, S., Margraf, J. T. & Günnemann, S. Fast and uncertainty-aware directional message passing for non-equilibrium molecules. In *Machine Learning for Molecules Workshop at NeurIPS 2020* (2020).
17. Shui, Z. & Karypis, G. Heterogeneous molecular graph neural networks for predicting molecule properties. In *2020 IEEE International Conference on Data Mining (ICDM)* (IEEE, 2020).
18. Li, S. *et al.* Structure-aware interactive graph neural networks for the prediction of protein-ligand binding affinity. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining* 975–985 (2021).
19. Jumper, J. *et al.* Highly accurate protein structure prediction with alphafold. *Nature* **596**, 583–589 (2021).
20. Baek, M. *et al.* Accurate prediction of protein structures and interactions using a three-track neural network. *Science* **373**, 871–876 (2021).
21. Schlick, T. *Molecular Modeling and Simulation: An Interdisciplinary Guide* (Springer Science & Business Media, 2010).
22. Stone, A. Distributed multipole analysis, or how to describe a molecular charge distribution. *Chem. Phys. Lett.* **83**, 233–239 (1981).
23. Veličković, P. *et al.* Graph attention networks. In *International Conference on Learning Representations* (2018).
24. Ramakrishnan, R., Dral, P. O., Rupp, M. & Von Lilienfeld, O. A. Quantum chemistry structures and properties of 134 kilo molecules. *Sci. Data* **1**, 1–7 (2014).
25. Lu, C. *et al.* Molecular property prediction: A multilevel quantum interactions modeling perspective. *Proc. AAAI Conf. Artif. Intell.* **33**, 1052–1060 (2019).
26. Schütt, K., Unke, O. & Gastegger, M. Equivariant message passing for the prediction of tensorial properties and molecular spectra. In *International Conference on Machine Learning* 9377–9388 (PMLR, 2021).
27. Liu, Y. *et al.* Spherical message passing for 3D molecular graphs. In *International Conference on Learning Representations (ICLR)* (2022).
28. Wang, J., Zhao, Y., Zhu, C. & Xiao, Y. 3D RNAscore: A distance and torsion angle dependent evaluation function of 3D RNA structures. *Nucleic Acids Res.* **43**, e63–e63 (2015).
29. Watkins, A. M., Rangan, R. & Das, R. Farfar2: Improved de novo Rosetta prediction of complex global RNA folds. *Structure* **28**, 963–976 (2020).
30. Townshend, R. J. *et al.* Geometric deep learning of RNA structure. *Science* **373**, 1047–1051 (2021).
31. Das, R. & Baker, D. Automated de novo prediction of native-like RNA tertiary structures. *Proc. Natl. Acad. Sci.* **104**, 14664–14669 (2007).
32. Miao, Z. *et al.* RNA-puzzles round IV: 3D structure predictions of four ribozymes and two aptamers. *RNA* **26**, 982–995 (2020).
33. Capriotti, E., Norambuena, T., Marti-Renom, M. A. & Melo, F. All-atom knowledge-based potential for RNA structure prediction and assessment. *Bioinformatics* **27**, 1086–1093 (2011).
34. Wang, R., Fang, X., Lu, Y. & Wang, S. The pdbbind database: Collection of binding affinities for protein- ligand complexes with known three-dimensional structures. *J. Med. Chem.* **47**, 2977–2980 (2004).
35. Ballester, P. J. & Mitchell, J. B. A machine learning approach to predicting protein-ligand binding affinity with applications to molecular docking. *Bioinformatics* **26**, 1169–1175 (2010).
36. Stepniewska-Dziubinska, M. M., Zielenkiewicz, P. & Siedlecki, P. Development and evaluation of a deep learning model for protein-ligand binding affinity prediction. *Bioinformatics* **34**, 3666–3674 (2018).
37. Zheng, L., Fan, J. & Mu, Y. Onionnet: A multiple-layer intermolecular-contact-based convolutional neural network for protein-ligand binding affinity prediction. *ACS Omega* **4**, 15956–15965 (2019).
38. Nguyen, T. *et al.* Graphdta: Predicting drug-target binding affinity with graph neural networks. *Bioinformatics* **37**, 1140–1147 (2021).
39. Danel, T. *et al.* Spatial graph convolutional networks. In *Neural Information Processing: 27th International Conference, ICONIP 2020, Bangkok, Thailand, November 18–22, 2020, Proceedings, Part V* 668–675 (Springer, 2020).
40. Lim, J. *et al.* Predicting drug-target interaction using a novel graph neural network with 3D structure-embedded graph representation. *J. Chem. Inf. Model.* **59**, 3981–3988 (2019).
41. Maziarka, Ł. *et al. Molecule Attention Transformer*. arXiv:2002.08264 (2020).
42. Song, Y. *et al.* Communicative representation learning on attributed molecular graphs. In *IJCAI* 2831–2838 (2020).
43. Garg, V., Jegelka, S. & Jaakkola, T. Generalization and representational limits of graph neural networks. In *International Conference on Machine Learning* 3419–3430 (PMLR, 2020).

44. Veit, M., Wilkins, D. M., Yang, Y., DiStasio, R. A. Jr. & Ceriotti, M. Predicting molecular dipole moments by combining atomic partial charges and atomic dipoles. *J. Chem. Phys.* **153**, 024113 (2020).
45. Faber, F. A. *et al.* Prediction errors of molecular machine learning models lower than hybrid dft error. *J. Chem. Theory Comput.* **13**, 5255–5264 (2017).
46. Landrum, G. Rdkit: A software suite for cheminformatics, computational chemistry, and predictive modeling. (2013).
47. Hornik, K., Stinchcombe, M. & White, H. Multilayer feedforward networks are universal approximators. *Neural Netw.* **2**, 359–366 (1989).
48. Ramachandran, P., Zoph, B. & Le, Q. V. *Searching for Activation Functions*. arXiv:1710.05941 (2017).
49. Kingma, D. P. & Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations* (2018).
50. Anderson, B., Hy, T. S. & Kondor, R. Cormorant: Covariant molecular neural networks *Adv. Neural Inform. Process. Syst.* 14537–14546 (2019).
51. Ren, S., He, K., Girshick, R. & Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *Adv. Neural Inform. Process. Syst.* **28** (2015).
52. Gomes, J., Ramsundar, B., Feinberg, E. N. & Pande, V. S. in *Atomic convolutional networks for Predicting Protein-ligand Binding Affinity*. arXiv:1703.10603 (2017).

Acknowledgements

This project has been funded with federal funds from the National Institute of General Medical Sciences of National Institute of Health (R01GM122845) and the National Institute on Aging of the National Institute of Health (R01AG057555).

Author contributions

L.X. and S.Z. conceived and designed the method and the experiments. S.Z. and Y.L. prepared the data. S.Z. implemented the algorithm and performed the experiments. All authors wrote and reviewed the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1038/s41598-023-46382-8.

Correspondence and requests for materials should be addressed to L.X.