JCTC Journal of Chemical Theory and Computation

pubs.acs.org/JCTC Article

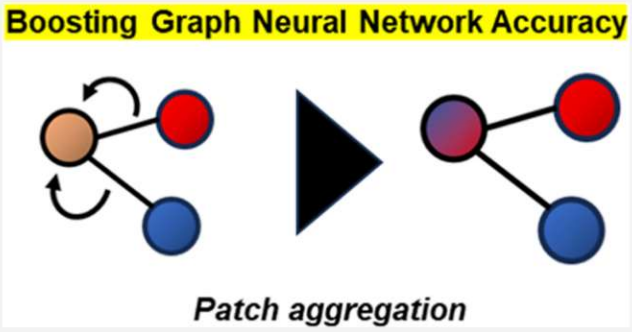
Graph Neural Network-Based Molecular Property Prediction with Patch Aggregation

Teng Jiek See, Daokun Zhang,* Mario Boley,* and David K. Chalmers*

Cite This: J. Chem. Theory Comput. 2024, 20, 8886−8896 Read Online

ACCESS | Metrics & More | Article Recommendations | Supporting Information

ABSTRACT: Graph neural networks (GNNs) have emerged as powerful tools for quantum chemical property prediction, leveraging the inherent graph structure of molecular systems. GNNs depend on an edge-to-node aggregation mechanism for combining edge representations into node representations. Unfortunately, existing learnable edge-to-node aggregation methods substantially increase the number of parameters and, thus, the computational cost relative to simple sum aggregation. Worse, as we report here, they often fail to improve predictive accuracy. We therefore propose a novel learnable edge-to-node aggregation mechanism that aims to improve the accuracy and parameter efficiency of GNNs in predicting molecular properties. The new mechanism, called "patch aggregation", is inspired by the Multi-Head Attention and Mixture of Experts machine learning techniques. We have incorporated the patch aggregation method into the specialized, state-of-the-art GNN models SchNet, DimeNet++, SphereNet, TensorNet, and VisNet and show that patch aggregation consistently outperforms existing learnable and nonlearnable aggregation techniques (sum, multilayer perceptron, softmax, and set transformer aggregation) in the prediction of molecular properties such as QM9 thermodynamic properties and MD17 molecular dynamics trajectory energies and forces. We also find that patch aggregation not only improves prediction accuracy but also is parameter-efficient, making it an attractive option for practical applications for which computational resources are limited. Further, we show that Patch aggregation can be applied across different GNN models. Overall, Patch aggregation is a powerful edge-to-node aggregation mechanism that improves the accuracy of molecular property predictions by GNNs.



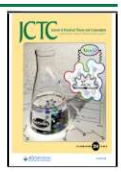
INTRODUCTION

Graph neural networks (GNNs) are a type of neural network that is specifically designed to work with graph-structured data.[1] They are particularly useful for tasks such as node classification,[2] link prediction,[3] and graph regression.[4] GNNs operate by passing information between nodes in a graph structure, allowing them to capture relationships and dependencies between different entities in the graph,[5] which makes them well suited for tasks where the input data can be represented as a graph such as social networks,[6] recommendation systems,[7] and particularly for chemical systems, where their ability to predict chemical properties is leading to a wide range of applications.[8] Recent exciting applications of GNNs in chemistry include the development of high-quality molecular mechanics force fields,[9] prediction of metabolic clearance rates for small molecules,[10] and modeling phase properties of binary and ternary solvent mixtures.[11]

GNNs have become the established state of the art in predicting quantum chemical properties, giving comparable accuracy to direct calculation using density functional theory (DFT)[12] or coupled cluster (CC) methods[13] but with enormously greater computational speed.[14] Current high-performing GNNs for quantum chemical property prediction, such as SchNet,[15] DimeNet++,[16] SphereNet,[17] TensorNet,[18] and VisNet[19] are successfully predicting binding energies[20] and electronic properties of molecules and solids.[4] However, despite this promise, these GNNs face some challenges such as difficulty in predicting energies during an AFIR-based reaction path search in complex systems[21] and instability in molecular dynamics simulations.[22] Therefore, there is considerable interest in further improving the GNN performance.

GNNs are well suited for molecular property prediction because molecules can be well represented as graph-structured data. GNNs model molecules as collections of nodes, representing atoms, and edges, representing bonds or other electronic interactions. A key factor in the accuracy of GNNs is the edge-to-node aggregation step (Figure 1), which transfers features from connected edges to update the center node

Received: June 19, 2024
Revised: September 2, 2024
Accepted: September 19, 2024
Published: October 2, 2024



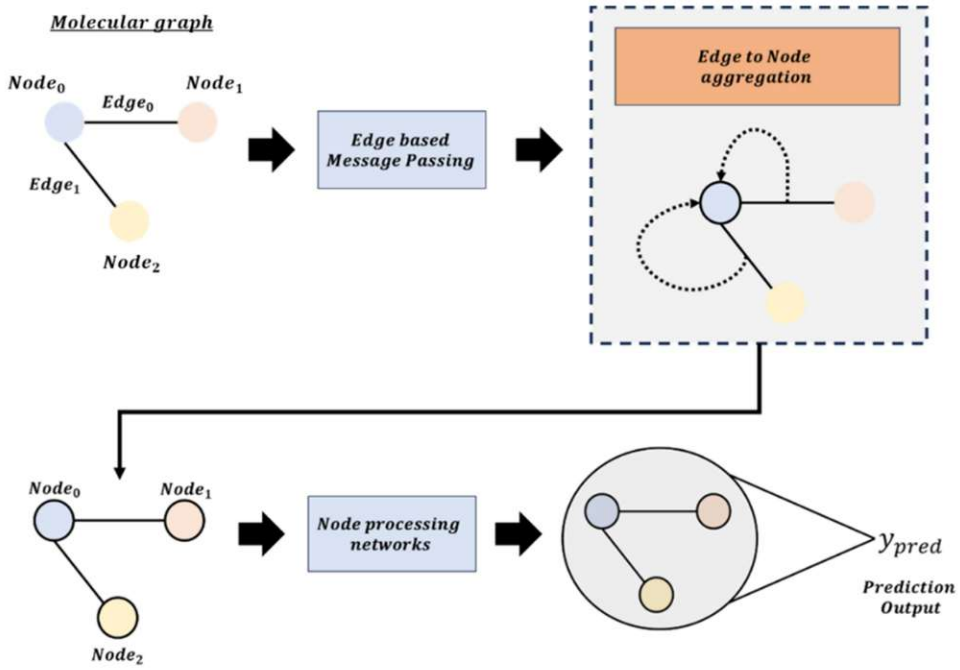


Figure 1. GNNs extract the edges from the molecular graph and pass them through an edge-based message passing mechanism. Edges are aggregated into nodes through edge-to-node aggregation mechanism. These nodes are then processed through node processing networks to generate the prediction output.

properties. Sum aggregation[23] is a simple and frequently used aggregation method where the features of all neighboring nodes of a target node are summed together to produce a single aggregated feature vector; however, this approach does not consider the relative importance or directionality of edge interactions.[24] Recent advancements in GNN propose learnable aggregation techniques such as Multi-Layer Perceptron (MLP),[25] Gated Recurrent Neural Network,[26] Softmax,[27] and Set Transformer aggregation.[25] The aggregation methods selectively aggregate features based on relevance to improve prediction accuracy for QM9 on basic Graph Neural Network architectures (e.g., Graph Convolutional Network,[28] Graph Attention Networks,[29] Graph Isomorphism Network,[24] and Principal Neighborhood Aggregation[30]) as demonstrated by Buturez et al.[25] However, the accuracy improvement obtainable using learnable aggregation methods on specialized GNNs (e.g., SchNet, DimeNet++, SphereNet, TensorNet, and VisNet) is not known.

In this article, we investigate the use of learnable aggregation in GNNs for the prediction of quantum chemical molecular properties. From this work, we propose a novel, parameter-efficient, learnable aggregation mechanism that enhances the accuracy of GNNs, which we denote as Patch aggregation. The Patch aggregation mechanism draws inspiration from two existing techniques: Mixture of Experts (MoE)[31] and Multi-Head Attention (MHA)[32] and operates by dividing the edge vector into smaller segments called patches, similar to the concept of "heads" in Multi-Head Attention,[29] where the attention mechanism operates on different parts of each node feature vector. Each patch in Patch aggregation captures a distinct perspective or aspect of the edge vector, akin to the experts in Mixture of Expert.[33] By employing Patch aggregation, the model can leverage diverse viewpoints of the edge vector, leading to a richer understanding of the task at hand. We conduct a comparative analysis of Patch aggregation with existing learnable aggregation techniques. The comparative analysis is applied to the state-of-the-art GNN models SchNet, DimeNet++, SphereNet, TensorNet, and VisNet. Importantly, we find that the accuracy improvement reported by Butarez et al.[25] using learnable aggregation mechanisms (Softmax, MLP, and Set Transformer) on basic Graph Neural Network backbones (e.g., Graph Convolutional Network, Graph Attention Networks, Graph Isomorphism Network, and Principal Neighborhood Aggregation) does not apply to these specialized GNN models. We show that our Patch aggregation improves accuracy compared to standard sum operations and current leading learnable aggregation methods in predicting chemical thermodynamic properties, molecular energies, and forces from molecular dynamics trajectories.

METHODS

Patch Aggregation. Patch aggregation is a learnable edge-to-node aggregation method that draws inspiration from the Mixture of Experts and multihead attention machine learning techniques. Mixture of Experts (MoE)[34] uses multiple expert neural networks to handle different aspects of input data that are weighted by coefficients produced by a gating network (Figure 2A). Each expert specializes in specific aspects of the task, and collectively, the experts make predictions or decisions based on input data. In contrast, Multi-Head Attention (MHA)[35] processes input vectors, dividing them into small vectors for N attention heads to calculate attention weights, capturing different aspects of the data (Figure 2B). Each attention head learns different relationships within the data, and their results are combined to create a comprehensive representation of each input vector, integrating diverse perspectives.

Our Patch aggregation method is designed such that it emphasizes different aspects of the original edge representation. Our Patch aggregation mechanism uses features inspired from both MoE and MHA. Similar to MHA, the input edge

https://doi.org/10.1021/acs.jctc.4c00798
J. Chem. Theory Comput. 2024, 20, 8886−8896

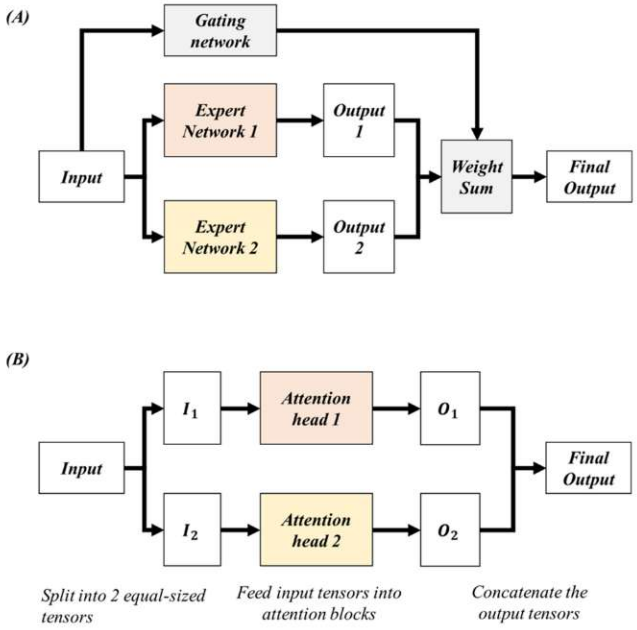


Figure 2. (A) Mixture of Experts with two experts and a gating network. Mixture of Experts (MoE) combines multiple neural networks (experts) to predict or decide. A gating network decides how much weight should be assigned to each expert based on input, producing gating coefficients for a total weighted sum output. Experts specialize in different aspects, learning to make accurate predictions within their expertise. (B) Multi-Head Attention with two attention heads. Multi-Head Attention (MHA) processes input vectors by splitting them into smaller vectors and feeding them into multiple attention heads. Each head calculates attention weights between input vectors to encode information from different perspectives. The parameters in each head allow for learning of diverse data relationships. The resulting attention weights from all heads are combined to create a single representation for each input vector, capturing specific aspects of the data.

representation is divided into several small patches of equal size that are used to generate weight tensors (referred to as patches). Each patch provides specialized weighting by manifesting different parts of the edge representation that corresponds to disparate aspect information on the edge information akin to the MoE expert specialization process.[34] These patches are used to perform patch-wise weighting on the edge representation. The patch-wise weighting process aims to highlight the relative significance of different parts of the edge representation feature vector, which correspond to various semantic aspects of the atom−atom interaction (Figure 3). After patch-wise weighting, the weighted edge representations are aggregated into node representation.

As shown in Figure 3 (step 1), we start by extracting the edge representations of the GNN models. Generally, for each edge e, two input representation variants are provided to the GNN models, denoted as $\boldsymbol{e}_1 \in \mathbb{R}^D$ and $\boldsymbol{e}_2 \in \mathbb{R}^D$, where D is the dimension of the input edge representations. In DimeNet++ and SphereNet, the second variant $\boldsymbol{e}_2$ is obtained by element-wise multiplication $\circ$ of $\boldsymbol{e}_1$ with radial basis function $\boldsymbol{d}_{\text{rbf}} \in \mathbb{R}^D$ as $\boldsymbol{e}_2 = \boldsymbol{e}_1 \circ \boldsymbol{d}_{\text{rbf}}$.

In the case of SchNet, TensorNet, and VisNet, which lack a distinct second variant of the edge representation, we assigned $\boldsymbol{e}_2 = \boldsymbol{e}_1$.

We segment out P patches from $\boldsymbol{e}_1$ by evenly splitting it into P vectors: $\boldsymbol{e}_1 = [\boldsymbol{e}_{1,1}, \boldsymbol{e}_{1,2}, \cdots, \boldsymbol{e}_{1,P}]^T \in \mathbb{R}^{P\times(D/P)}$ as illustrated in Figure 3 (step 2), where $\boldsymbol{e}_{1,i} \in \mathbb{R}^{D/P}$ is the ith patch of the edge representation. Similar to the attention heads in MHA,[32,35,36] segmenting naturally forces each patch to contain a subset of the total edge information. The patch is then used to create a specialized patch weight tensor. As illustrated in Figure 3 (steps 3 and 4), patch weight tensors $\boldsymbol{g}_1$ are created by feeding the edge patches $\boldsymbol{e}_1 \in \mathbb{R}^{P\times(D/P)}$ into a patch-wise feed-forward network f_{all} and restricting $\boldsymbol{e}_1$ values to the range $[0,1]$ using f_{restrict} (f_{restrict} is an operation that clamps values into the range $[0, 1]$)

$$\boldsymbol{g}_1 = f_{\text{restrict}}(f_{\text{all}}(\boldsymbol{e}_1))\text{s. t. } \boldsymbol{g}_1 \in [0, 1]^{P\times D} \tag{1}$$

where $\boldsymbol{g}_1 = [\boldsymbol{g}_{1,1}, \boldsymbol{g}_{1,2}\cdots\boldsymbol{g}_{1,P}]^T$ and $\boldsymbol{g}_{1,i} = f_{\text{restrict}}(f_{\text{all}}(\boldsymbol{e}_{1,i})) \in [0,1]^D$ is the weight tensor produced by the ith patch $\boldsymbol{e}_{1,i}$.

Similarly, we split the edge variant $\boldsymbol{e}_2 \in \mathbb{R}^D$ into P patches: $\boldsymbol{e}_2 = [\boldsymbol{e}_{2,1}, \boldsymbol{e}_{2,2}, \cdots, \boldsymbol{e}_{2,P}]^T \in \mathbb{R}^{P\times(D/P)}$, where $\boldsymbol{e}_{2,i} \in \mathbb{R}^{D/P}$ is the ith patch of the edge representation.

As illustrated in Figure 3 (step 5), we then transform the edge patches $\boldsymbol{e}_2$ into a weighted variant $\boldsymbol{e}_t \in \mathbb{R}^{P\times(D/P)}$, by multiplying $\boldsymbol{e}_2$ with $\boldsymbol{g}_1$ in a patch-wise manner

$$\boldsymbol{e}_{t,i} = \boldsymbol{e}_{2,i} * \boldsymbol{g}_{1,i} \tag{2}$$

where $\boldsymbol{e}_{t,i} \in \mathbb{R}^{D/P}$ is the ith patch of $\boldsymbol{e}_t$ and the multiplication operator $*$: $\mathbb{R}^{D/P} \times \mathbb{R}^D \to \mathbb{R}^{D/P}$ is defined as

$$[\boldsymbol{a} * \boldsymbol{b}] = \sum_{j=1}^{P} \boldsymbol{a} \circ \boldsymbol{b}_{[N*(j-1):N*j]} \tag{3}$$

where $\circ$ denotes the Hadamard product, $N = D/P$ refers to the patch size, and $\boldsymbol{b}_{[N*(j-1):N*j]}$ refers to the subvector from index $N*(j-1)$ to index $N * j$ (exclusive) of the vector $\boldsymbol{b}$.

Analogous to the experts in Mixture of Experts (MoE),[31,33,34,37] each patch weight tensor in $\boldsymbol{g}_1$ allocates a different weight to each input edge patch of $\boldsymbol{e}_2$, where the patch weight tensors are learnable and specialized. The resulting edge patch set $\boldsymbol{e}_t \in \mathbb{R}^{P\times(D/P)}$ is then reshaped to the corresponding edge representation $\boldsymbol{e}_t \in \mathbb{R}^D$ by sequentially concatenating all patches.

Finally, as illustrated in Figure 3 (step 6), for each node v, its representation $\boldsymbol{v} \in \mathbb{R}^D$ is obtained by summing up the representations of all edges connecting it

$$\boldsymbol{v} = \sum_{e\in\mathcal{N}(v)} \boldsymbol{e}_t \tag{4}$$

where $\mathcal{N}(v)$ is the set of edges connecting node v.

In summary, the Patch Aggregation method captures unique semantic information from various segments of the edge representation feature vector by applying specialized patch-wise weighting. This approach emphasizes the relative importance of different parts of the edge representation, resulting in processed edge feature vectors that are enriched with semantic information. Ultimately, these enhanced edge features are aggregated into a node representation, making them well suited for Graph Neural Network applications.

Data Sets. We used the QM9[38] and MD17[39] data sets to assess the aggregation techniques. The QM9 data set consists of 12 molecular properties, namely: dipole moment (Mu), isotropic polarizability (α), highest occupied molecular orbital energy (HOMO), lowest unoccupied molecular orbital energy (LUMO), energy gap between HOMO and LUMO (Gap),

https://doi.org/10.1021/acs.jctc.4c00798
J. Chem. Theory Comput. 2024, 20, 8886−8896

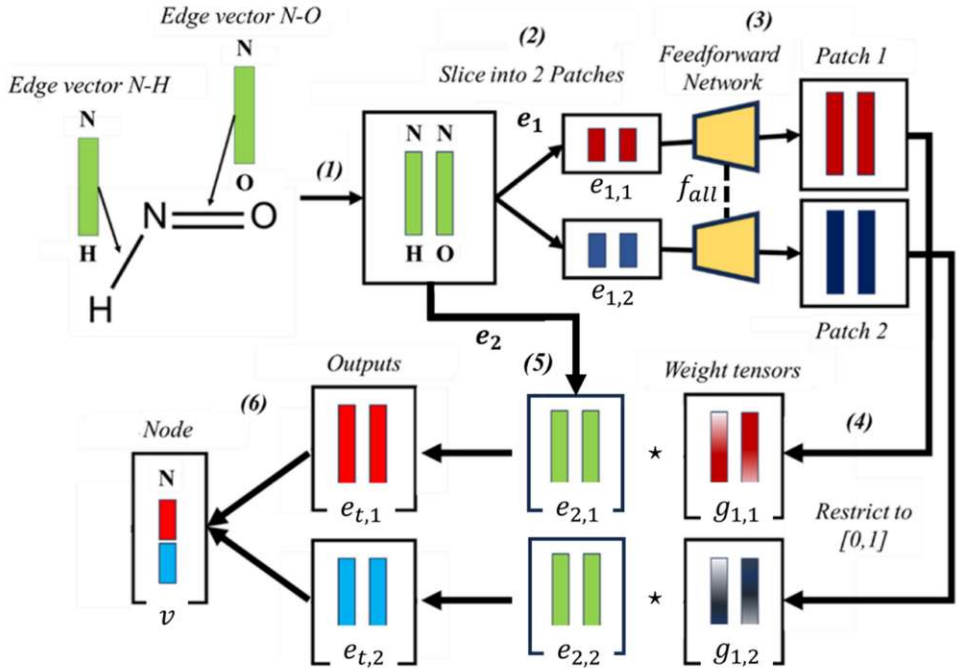


Figure 3. Patch aggregation method using the molecule azanone (HNO) as an example. (1) Atomic interactions within a cutoff distance with respect to the main atom (nitrogen) are encoded into edge representations. (2) The edge representations are sliced into two smaller specialized patches via reshaping. (3) The small patches are expanded to match the original size using a feed-forward neural network (f_{all}). (4) The expanded patches are transformed into weight tensors by restricting their values to the range [0, 1]. (5) The original edge representations are patch-wise multiplied by the weight tensors. (6) Edge representations are aggregated into node representation to represent the nitrogen atom.

Table 1. Number of Molecule Samples in Training, Validation, and Test Sets of QM9 and MD17[a]

	number of molecular conformations/samples		
data set	training	validation	test
QM9	110 000	10 000	23 885
MD17	1000	1000	507 983 (benzene)
			13 770 (uracil)
			206 250 (naphthalene)
			91 762 (aspirin)
			200 231 (salicylic acid)
			873 237 (malonaldehyde)
			435 092 (ethanol)
			442 790 (toluene)

[a]The QM9 data set comprises a significantly larger number of molecules with 110 000 for training, 10 000 for validation, and 23885 for testing. In contrast, the MD17 data set is much smaller with 1000 conformers each for training and validation. The MD17 test set size varies across different molecules ranging from 13 770 conformations for uracil to 873 237 for malonaldehyde.

electronic spatial extent ($\langle R^2 \rangle$), zero-point vibrational energy (ZPVE), internal energy at 0 K (U_0), internal energy at 298.15 K (U), enthalpy at 298.15 K (H), Gibbs free energy at 298.15 K (G), and heat capacity at 298.15 K (C_v). The values have been calculated for approximately 134,000 stable small organic molecules (Table 1). Each molecule is composed of up to nine heavy atoms (C, N, O, and F).

The MD17 data set consists of energy and forces calculated from the molecular trajectories of eight small organic molecules. Each molecule has 15 770 to 875 237 conformers (Table 1). The QM9 data set was divided into three parts: a training set containing 110 000 molecules, a validation set with 10,000 molecules, and a test set comprising the remaining molecules. Similarly, the MD17 data set was split into three sections: a training set composed of 1000 conformers, a validation set of 1000 conformers, and a test set containing remaining conformers.

Model Training. The GNN models (SchNet, DimeNet++, SphereNet, TensorNet, and VisNet) were obtained from Liu et al.,[17] Simeon et al.,[18] and Wang et al.[19] The learnable aggregation mechanisms are available in the PyTorch Geometric library.[40] We implemented our Patch aggregation using PyTorch.[41] We swapped the default sum aggregation in the GNN models with the learnable aggregation mechanisms. The models were trained by minimizing the mean absolute error loss using the Adam optimizer[42] with the specific parameters: *learning rate* = 5×10^{-4}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\varepsilon = 10^{-8}$. Additionally, depending on the training data set, a linear learning rate warm-up[43] was used for either 3000 (QM9) or 1000 steps (MD17), followed by a systematic reduction in the learning rate through cosine annealing decay. The training processes spanned 150 epochs with a batch size of 32 for QM9 and 2000 epochs with a batch size of 8 for MD17. In MD17 training, we incorporated a hybrid energy-force error loss function with the

https://doi.org/10.1021/acs.jctc.4c00798
J. Chem. Theory Comput. 2024, 20, 8886−8896

Table 2. Evaluation of Learnable Edge-to-Node Aggregation Methods: Sum Aggregation (Sum), Multi-Layer-Perceptron Aggregation (MLP), Softmax Aggregation, Set Transformer Aggregation, and Patch Aggregation. Five GNN models were evaluated: SchNet, DimeNet++, SphereNet, TensorNet, and VisNet. Comparison uses the QM9 data set[a]

	tasks	Mu	α	HOMO	LUMO	Gap	$\langle R^2 \rangle$
GNN	AGG/units	D	a_0^3	meV	meV	meV	a_0^2
SchNet	Sum	0.0334	0.0721	35.7	29.4	46.6	0.226
	Patch (Ours)	**0.0291 [−12.9%]**[e]	**0.0636 [−11.8%]**[f]	**33.6 [−5.9%]**[f]	**27.6 [−6.1%]**[h]	**44.3 [−4.9%]**[f]	**0.149 [−34.1%]**[f]
	Patch (Ours) (Scaled)[m]	**0.0299 [−10.5%]**[f]	**0.0657 [−8.9%]**[f]	**34.8 [−2.5%]**[f]	**27.7 [−5.8%]**[f]	**44.6 [−4.3%]**[f]	**0.158 [−30.1%]**[f]
	Softmax	0.0319 [−4.5%]	0.0777 [+7.8%]	38.5 [+7.8%]	30.2 [+2.7%]	57.1 [+22.5%]	0.205 [−9.3%]
	Set Transformer[d]	0.0314 [−5.9%]	0.0737 [+2.2%]	36.8 [+3.1%]	29.4 [−0.0%]	54.9 [+17.8%]	0.224 [−0.9%]
	Set Transformer (Scaled)[m]	0.0351 [+5.1%]	0.0829 [+14.9%]	41.2 [+15.4%]	31.6 [+7.5%]	58.5 [+25.5%]	0.246 [+8.8%]
	MLP[c]	0.0945 [+182.9%]	0.1458 [+102.0%]	77.8 [+118.0%]	57.7 [+96.3%]	109.6 [+135.2%]	0.418 [+84.9%]
	MLP (Scaled)[m]	0.1124 [+236.5%]	0.1762 [+144.4%]	94.6 [+164.9%]	71.5 [+149.4%]	116.2 [+80.9%]	0.409 [+71.6%]
DimeNet++	Sum[b]	0.0267	0.0424	**23.9**	18.3	**30.9**	0.317
	Patch (Ours)	**0.0262 [−1.9%]**[h]	**0.0418 [−1.4%]**[h]	**23.9 [0.0%]**[h]	**18.2 [−0.5%]**[f]	**30.9 [−0.0%]**[f]	**0.299 [−5.7%]**[f]
	Softmax	0.0301 [+12.7%]	0.0461 [+8.7%]	28.1 [+17.6%]	18.8 [+2.7%]	45.4 [+46.9%]	0.407 [+28.4%]
	Set Transformer[d]	0.0288 [+7.9%]	0.0506 [+19.3%]	27.9 [+16.7%]	19.2 [+4.9%]	45.9 [+48.5%]	0.456 [+43.8%]
	MLP[c]	0.0316 [+18.4%]	0.0479 [+13.0%]	29.9 [+25.1%]	20.8 [+13.7%]	49.1 [+58.9%]	0.563 [+77.6%]
SphereNet	Sum[b]	0.0319	0.0499	**27.4**	21.4	35.5	**0.386**
	Patch (Ours)	**0.0302 [−5.3%]**[g]	**0.0495 [−0.8%]**[g]	27.6 [+0.7%][g]	**21.1 [−1.4%]**[g]	**35.2 [−0.8%]**[g]	0.390 [+1.0%][g]
TensorNet	Sum[b]	**0.0128**	**0.0419**	22.4	17.7	38.6	27.7
	Patch (Ours)	0.0131 [+2.3%][f,j,k]	**0.0419 [−0.0%]**[f,i]	**22.2 [−0.9%]**[f,i]	**17.2 [−2.8%]**[f,i]	**38.4 [−0.5%]**[f,i]	**26.8 [−3.2%]**[f,i]
VisNet	Sum[b]	0.0151	**0.0430**	**24.3**	19.2	43.2	**0.235**
	Patch (Ours)	**0.0148 [−1.9%]**[f,l]	0.0432 [+0.5%][f,l]	24.4 [+0.4%][f,l]	**18.8 [−2.1%]**[f,l]	**42.5 [−1.6%]**[f,l]	0.237 [+0.9%][f,l]

	tasks	ZPVE	U_0	U	H	G	C_v
GNN	AGG/Units	meV	meV	meV	meV	meV	$\frac{\text{cal}}{\text{mol K}}$
SchNet	Sum[b]	1.553	14.16	15.68	15.55	13.74	0.0311
	Patch (Ours)	**1.460 [−5.9%]**[g]	**11.99 [−15.3%]**[e]	**11.88 [−24.2%]**[g]	**12.19 [−21.6%]**[g]	**12.29 [−10.6%]**[f]	**0.0270 [−13.2%]**[f]
	Patch (Ours) (Scaled)[m]	**1.54 [−0.6%]**[f]	**12.53 [−11.5%]**[f]	**13.06 [−16.7%]**[f]	**13.42 [−13.7%]**[f]	**11.94 [−13.1%]**[f]	**0.0280 [−9.7%]**[f]
	Softmax	1.711 [+10.2%]	15.43 [+8.9%]	17.75 [+13.2%]	15.55 [−0.0%]	15.50 [+12.8%]	0.0291 [−6.4%]
	Set Transformer[d]	1.721 [+10.8%]	15.29 [+7.9%]	15.17 [−3.3%]	14.78 [−4.9%]	13.67 [−0.5%]	0.0292 [−6.1%]
	Set Transformer (Scaled)[d,m]	1.77 [+14.2%]	15.37 [+8.5%]	17.05 [+8.7%]	15.29 [−1.7%]	17.16 [+24.9%]	0.031 [−0.0%]
	MLP[c]	2.432 [+56.6%]	30.83 [+118.0%]	29.46 [+87.9%]	30.75 [+97.7%]	29.98 [+118.2%]	0.0530 [+70.4%]
	MLP (Scaled)[c,m]	2.66 [+138.3%]	33.74 [+114.9%]	33.69 [+114.9%]	33.59 [+116.0%]	33.29 [+142.3%]	0.060 [+93.5%]
DimeNet++	Sum[b]	1.244	6.22	6.11	6.14	6.98	0.0232
	Patch (Ours)	**1.221 [−1.8%]**[f]	**5.94 [−4.5%]**[f]	**5.87 [−3.9%]**[h]	**6.12 [−0.3%]**[f]	**6.87 [−1.6%]**[f]	**0.0232 [−0.0%]**[f]
	Softmax	1.472 [+18.3%]	9.63 [+54.8%]	23.20 [+280.0%]	7.91 [+28.8%]	10.30 [+47.6%]	0.0240 [+3.4%]
	Set Transformer[d]	1.314 [+5.6%]	9.21 [+48.1%]	10.10 [+65.3%]	9.97 [+62.4%]	9.67 [+38.5%]	0.0241 [+3.9%]
	MLP[c]	1.263 [+1.5%]	8.43 [+35.5%]	8.49 [+39.0%]	7.67 [+24.9%]	7.88 [+12.9%]	0.0263 [+13.4%]
SphereNet	Sum[b]	1.391	**7.33**	7.79	7.45	8.73	**0.0263**
	Patch (Ours)	**1.372 [−1.4%]**[g]	7.76 [+5.9%][g]	**7.59 [−2.6%]**[g]	**7.43 [−0.3%]**[g]	**8.56 [−1.9%]**[g]	**0.0263 [−0.0%]**[g]
TensorNet	Sum[b]	1.148	5.47	5.44	**5.37**	6.58	**0.0224**
	Patch (Ours)	**1.141 [−0.6%]**[f,k]	**5.41 [−1.1%]**[f,i]	**5.32 [−2.2%]**[f,i]	5.44 [+1.3%][f,i]	**6.50 [−1.2%]**[f,i]	**0.0226 [+0.9%]**[f,j]
VisNet	Sum[b]	1.183	**6.81**	8.85	**6.86**	7.60	0.0233
	Patch (Ours)	**1.179 [−0.3%]**[f,l]	6.91 [+1.5%][f,l]	**7.07 [−20.1%]**[f,l]	6.96 [+1.5%][f,l]	**7.58 [−0.3%]**[f,l]	**0.0232 [−0.4%]**[f,l]

[a]Patch aggregation is the only method that reduces the mean absolute error (MAE) over the simple Sum method. When different aggregation mechanisms in SchNet are scaled to match the same number of parameters as those in SchNet (denoted as Scaled), Patch aggregation also outperforms other aggregation mechanisms. Values in brackets are the percentage reduction in mean absolute error when using a learnable aggregation mechanism with respect to using sum aggregation mechanism (ΔE_{mae}). [b]Reproduced in this work. [c]1 layer of MLP. [d]1 encoder and 1 decoder layers. [e]1 patch. [f]2 patches. [g]4 patches. [h]8 patches. [i]I_j aggregated[18] via patch aggregation. [j]A_j aggregated[18] via patch aggregation. [k]S_j aggregated[18] via patch aggregation. [l]Scalar aggregation[19] via patch aggregation. [m]Same number of parameters as the original SchNet (Sum aggregation).

same hyperparameters as Klicpera et al.[16] to train the GNN models on both energy and force labels. We used two metrics for evaluating the model performance. First, we used mean absolute error (MAE or E_{mae})

$$E_{\text{mae}} = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i| \tag{5}$$

https://doi.org/10.1021/acs.jctc.4c00798
J. Chem. Theory Comput. 2024, 20, 8886−8896

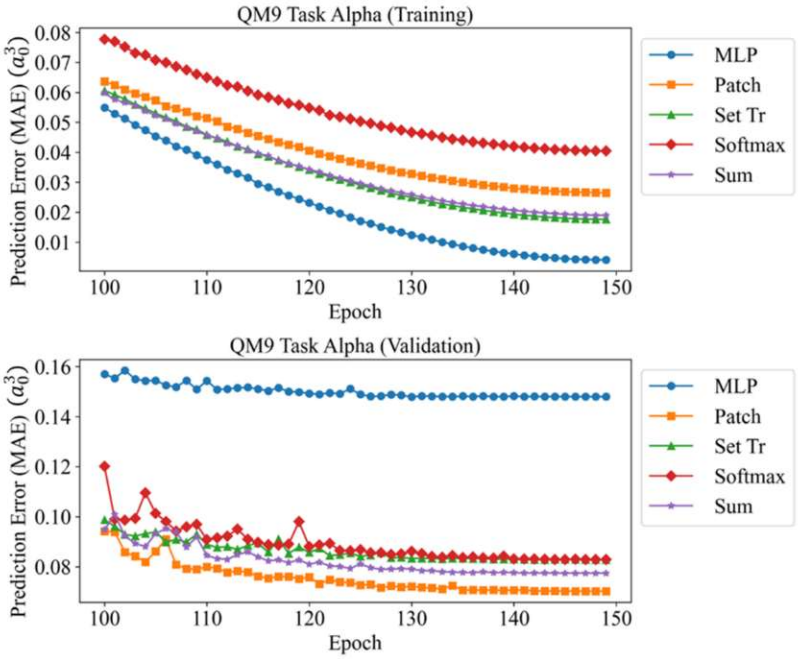


Figure 4. Training and validation loss curves for the edge-to-node aggregation mechanisms (Sum aggregation—Sum, Softmax aggregation—Softmax, Multi-Layer Perceptron—MLP, Set Transformer—Set Tr, and Patch aggregation—Patch (2 Patches)) for SchNet on QM9 α task. Patch aggregation improved the validation accuracy while sacrificing training accuracy to enable better generalization across unseen data. The training and validation curves for other QM9 tasks are available in the Supporting Information (Figure S1).

Table 3. Number of Parameters in GNN Models and the Edge-to-Node Aggregation Mechanisms[a]

number of parameters in GNN model	SchNet	480 385
	DimeNet++	1 887 110
	SphereNet	1 898 566
	TensorNet	1 329 905
	VisNet	1 706 550
number of parameters in aggregation mechanism	Sum	0
	MLP	2 458 240
	Softmax	0
	Set Transformer	1 321 600
	Patch	81 920, 40 960, 20 480, 10 240

[a]SchNet has the lowest number of parameters compared to DimeNet++, SphereNet, TensorNet, and VisNet. Patch aggregation uses substantially fewer parameters than MLP and Set Transformer aggregation.

where y_i is the truth property value of the ith molecule/conformer, $\hat{y}_i$ is the corresponding prediction from the model, and n is the number of samples.

Second, we used percentage reduction in mean absolute error when using a learnable aggregation mechanism compared to using sum aggregation mechanism (ΔE_{mae}),

$$\Delta E_{\text{mae}} = \frac{E_{\text{mae,agg}} - E_{\text{mae,sum}}}{E_{\text{mae,sum}}} \times 100\% \quad (6)$$

where $E_{\text{mae,agg}}$ is the MAE of a learnable aggregation method and $E_{\text{mae,sum}}$ is the MAE of sum aggregation method.

The code that we developed and used can be found in github.com/tengjieksee/Patch-aggregation-Graph-Neural-Network.

RESULTS

Comparison of Patch Aggregation with Other Learnable Aggregation Methods. We benchmarked patch aggregation against three advanced, learnable aggregation techniques: Multi-Layer Perceptron (MLP) aggregation, Softmax aggregation, and Set Transformer aggregation. The aggregation methods were incorporated into the SchNet and DimeNet++ architectures. Patch aggregation was also evaluated within the SphereNet, TensorNet, and VisNet architectures. Results are compared to the baseline sum aggregation in each case. Using the SchNet model, Patch aggregation boosted the accuracy in all 12 QM9 property prediction tasks (Table 2). MLP aggregation gave the poorest performance, failing to improve on sum aggregation for any method (Table 2). Softmax aggregation improved over sum in three tasks (Mu, $\langle R^2 \rangle$, C_v) (Table 2). Set Transformer aggregation improved over sum in six tasks (Mu, $\langle R^2 \rangle$, U, H, G, C_v) with less overall accuracy degradation compared to Softmax and MLP aggregations (Table 2). Using the DimeNet++ model, Patch aggregation enhanced the accuracy of most tasks. In contrast, MLP aggregation, Softmax aggregation, and Set Transformer aggregation failed to improve the prediction accuracy. During training, Patch aggregation enhanced validation accuracy while reducing training accuracy compared to sum aggregation,

https://doi.org/10.1021/acs.jctc.4c00798
J. Chem. Theory Comput. 2024, 20, 8886−8896

thereby promoting better generalization to unseen data (Figure 4).

To determine if the accuracy improvement with Patch aggregation was simply due to an increase in the number of parameters, we tested SchNet with different aggregation methods while keeping the number of parameters constant. Patch aggregation consistently improved SchNet prediction accuracy across all QM9 tasks under the same number of parameters (see Table 2). In contrast, other aggregation methods mostly degraded the accuracy of SchNet. The accuracy enhancement achieved through Patch aggregation is not solely attributable to the increase in the number of parameters.

Patch aggregation achieved the highest accuracy improvement while also using significantly fewer parameters compared to MLP and Set Transformer aggregation methods (Table 3). Specifically, Patch aggregation increased the total number of model parameters (including the parameters in the GNN model) by around 0.5 to 17.1% (Figure 5). In contrast, MLP and

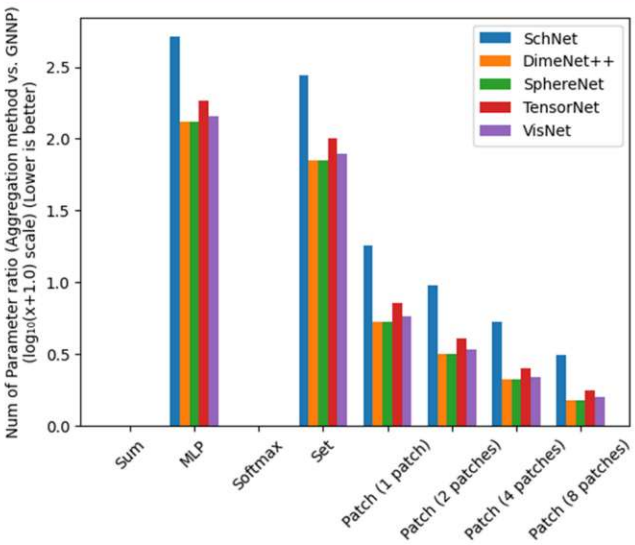


Figure 5. Ratio between the edge-to-node aggregation component parameters and the GNN backbone parameters (in log-cale). Patch aggregation contributes substantially less parameters toward GNN models compared to MLP and Set Transformer aggregation. Sum and Softmax aggregation methods do not add additional parameters toward GNN models. Further details on the ratio between the number of parameters of each aggregation method and GNN model can be found in the Supporting Information (Table S1).

Set Transformer aggregation added around 130 to 511.7% to the total model parameters (Figure 5). In summary, Patch aggregation emerged as the most effective method, consistently enhancing accuracy across all prediction tasks to a greater extent than other learnable aggregation methods while also maintaining parameter efficiency.

Applicability of Patch Aggregation across GNN Models. We evaluated the Patch method in five GNN models, SchNet, DimeNet++, SphereNet, TensorNet, and VisNet, under the same learning schedule and batch size (Table 4). We found that Patch aggregation reduces the MAE of SchNet (−13.8% on average) in all 12 QM9 prediction tasks. Moreover, it also decreases the MAE of DimeNet++ (−1.7%), SphereNet (−0.6%), TensorNet (−0.7%), and VisNet (−1.8%) in most tasks. However, the mean absolute error reduction was notably greater for SchNet than for DimeNet++, SphereNet, TensorNet, and VisNet. It is noteworthy that, while the mean absolute error reduction was relatively lower for DimeNet++, SphereNet, TensorNet, and VisNet, Patch aggregation was the only aggregation method that consistently reduced the MAE.

Number of Patches and Its Effect on Prediction Accuracy. We investigated how the number of patches affected the prediction accuracy using SchNet. The number of patches was varied between 1 and 8 while the learning rate schedules and batch sizes were held constant (Figure 6). Patch aggregation performed better than sum aggregation across all QM9 prediction tasks regardless of the number of patches used. The mean performance, evaluated both as the mean reduction in mean average error (MAE) (Table 5) and as the number of winning tasks, was maximized with two patches. A single patch also gave significant improvements, and 8 patches gave smaller improvements. Interestingly, the nature of the Patch aggregation method means that fewer parameters are required as the number of patches are increased (Figure 6). Therefore, the parameter efficiency of the Patch aggregation mechanism increases as the number of patches is increased.

Molecular Dynamics Trajectory Energies and Forces. We evaluated the ability of Patch aggregation to predict energies and forces from molecular trajectory data, comparing the performance of Sum and Patch aggregation within the DimeNet ++ GNN model using the MD17 data set (Table 6). This data set consists of structures sampled from molecular dynamics trajectories of 8 molecules. We used a training set comprising 1000 conformers, a validation set of equal size, and a test set of the remaining molecules (15 770 to 875 237 conformers). We found that Patch aggregation gave superior accuracy over Sum aggregation, especially in predicting molecular forces, achieving a lower mean absolute error across diverse molecules. In the energy prediction, Patch aggregation outperformed Sum aggregation in all but one case. Overall, Patch aggregation gave a significant reduction in MAE of 18.8% (ΔE_{mae}) for energy prediction and 42.4% (ΔE_{mae}) for force prediction compared to Sum aggregation (Table 6).

DISCUSSION

In this work, we have developed Patch aggregation, a novel learnable edge-to-node aggregation mechanism that enhances the performance of graphene network (GNN) in predicting quantum molecular properties. The Patch aggregation method is inspired by the Multi-Head Attention and Mixture of Experts and utilizes patches as a weighting mechanism for aggregating relevant edge vectors into node vectors. Patch aggregation enhances edge representations by dividing them into smaller patches, with each patch emphasizing different semantic aspects similar to MoE and MHA. These patches apply specialized weighting to highlight key features before aggregating them into a node representation.

We evaluated how the number of patches impacts prediction accuracy using SchNet and found that an optimal number of patches exists, beyond which increasing the number of patches diminishes the accuracy improvement (Figure 6). Notably, we also found that increasing the number of patches decreases the total number of parameters required for the Patch aggregation mechanism, which we attribute to partitioning of the input vector into smaller information subsets. This inherently reduces the dimensionality of the input space for the feed-forward network, allowing it to operate on lower-dimensional representations of the data. The feed-forward network can still achieve good accuracy improvement with fewer parameters, as it

https://doi.org/10.1021/acs.jctc.4c00798
J. Chem. Theory Comput. 2024, 20, 8886−8896

Table 4. Assessment of Average MAE Reduction across 12 QM9 Tasks for Learnable Edge-to-Node Aggregation Techniques Includes Multilayer Perceptron (MLP) Aggregation, Softmax Aggregation, Set Transformer Aggregation, and Patch Aggregation Compared to Sum Aggregation[a]

metrics	GNN model	sum	MLP	Softmax	Set Transformer	Patch (ours)
% reduction in MAE (ΔE_{mae}) (lower is better)	SchNet		+105.7%	+5.5%	+1.7%	**−13.8%**
	DimeNet++		+27.4%	+45.9%	+30.5%	**−1.7%**
	SphereNet					**−0.6%**
	TensorNet					**−0.7%**
	VisNet					**−1.8%**
winning tasks	SchNet	0/12	0/12	0/12	0/12	**12/12**
	DimeNet++	0/12	1.5/12	0/12	0/12	**10.5/12**
	SphereNet	3.5/12				**8.5/12**
	TensorNet	3.5/12				**8.5/12**
	VisNet	5/12				**7/12**

[a]Five GNN models (SchNet, DimeNet++, SphereNet, TensorNet, and VisNet) are assessed using the QM9 data set. Notably, Patch aggregation emerges as the only method enhancing the average prediction accuracy compared to the basic Sum method.

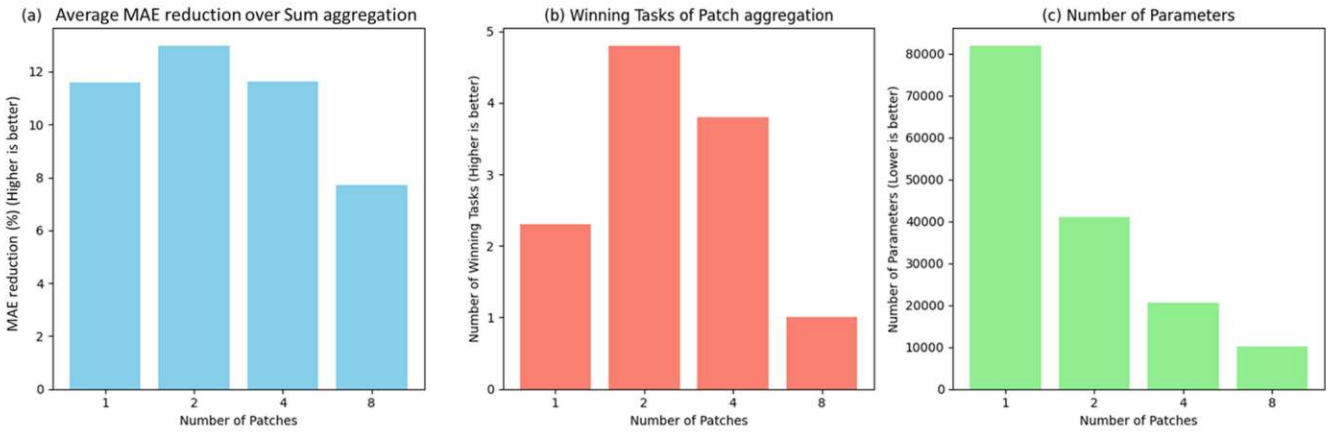


Figure 6. Effect of varying number of patches aggregation on accuracy improvement assessed using SchNet with the QM9 data set. (a) MAE reduction with respect to Sum aggregation. Accuracy is greatest with 2 patches and declines as more patches are used. (b) The number of winning QM9 tasks is greatest with 2 patches. (c) The number of additional model parameters decreases sharply as the number of patches is increased.

Table 5. Comparison of the Mean Absolute Error in Predictions between the Patch Aggregation Method and the Simple Sum Aggregation Method Using SchNet, Assessed with the QM9 Data Set[a]

		mean absolute error (lower is better)				
target	unit	sum	Patch aggregation (1 patch) (ours)	Patch aggregation (2 patches) (ours)	Patch aggregation (4 patches) (ours)	Patch aggregation (8 patches) (ours)
Mu	D	0.0334	**0.0291 [−12.9%]**	0.0293 [−12.3%]	0.0303 [−9.3%]	0.0304 [−8.9%]
α	a_0^3	0.0721	0.0671 [−6.9%]	**0.0636 [−11.8%]**	0.0654 [−9.3%]	0.0687 [−4.7%]
HOMO	meV	35.7	33.7 [−5.6%]	**33.6 [−5.9%]**	34.2 [−4.2%]	35.8 [+0.3%]
LUMO	meV	29.4	27.9 [−5.1%]	28.2 [−4.1%]	28.3 [−3.7%]	**27.6 [−6.1%]**
Gap	meV	46.6	44.5 [−4.5%]	**44.3 [−4.9%]**	45.5 [−2.4%]	46.2 [−0.8%]
$\langle R^2 \rangle$	a_0^2	0.226	0.167 [−26.1%]	**0.149 [−34.1%]**	0.175 [−22.6%]	0.188 [−16.8%]
ZPVE	meV	1.55	1.47 [−5.2%]	1.49 [−3.9%]	**1.46 [−5.8%]**	1.59 [+2.6%]
U_0	meV	14.16	**11.99 [−15.3%]**	12.11 [−14.5%]	12.33 [−12.9%]	12.72 [−10.2%]
U	meV	15.68	12.51 [−20.2%]	12.26 [−21.8%]	**11.88 [−24.2%]**	13.33 [−14.9%]
H	meV	15.55	12.35 [−20.6%]	12.55 [−19.3%]	**12.19 [−21.6%]**	12.39 [−20.3%]
G	meV	13.74	13.22 [−3.8%]	**12.29 [−10.6%]**	**12.29 [−10.6%]**	13.36 [−2.8%]
C_v	$\frac{cal}{mol\ K}$	0.031	**0.027 [−12.9%]**	**0.027 [−12.9%]**	**0.027 [−12.9%]**	0.028 [−9.7%]

[a]The number of patches spans from 1 to 8. Employing 2 patches gives the greatest improvement mean accuracy improvement, while utilizing 8 patches uses the fewest parameters. Patch aggregation consistently outperforms Sum aggregation, regardless of the number of patches employed. Values in brackets are the percentage reduction in mean absolute error when using a learnable aggregation mechanism with respect to using sum aggregation mechanism (ΔE_{mae}).

focuses on learning more localized and specialized patterns within each patch, resulting in high parameter efficiency. However, further increasing the number of patches inevitably leads to a reduction in the overall accuracy improvement. Overall, the optimal number of patches is a balance between the number of parameters and the accuracy improvement.

https://doi.org/10.1021/acs.jctc.4c00798
J. Chem. Theory Comput. 2024, 20, 8886−8896

Table 6. Improvement in Accuracy Using Patch Aggregation Compared to Sum Aggregation for Molecular Energies and Forces Prediction for the MD17 Data Set. Methods are compared using 4 patches with DimeNet++[a]

tasks	mean absolute error (lower is better)			
	energy (kcal/mol)		force (kcal/mol/Å)	
	Sum	Patch	Sum	Patch
aspirin	0.211	**0.187** [**−11.4%**]	0.477	**0.390** [**−18.2%**]
benzene	0.358	**0.354** [**−1.1%**]	0.206	**0.187** [**−9.2%**]
ethanol	**0.058**	0.059 [+1.7%]	0.174	**0.149** [**−14.4%**]
malonaldehyde	0.128	**0.083** [**−35.2%**]	1.418	**0.241** [**−83.0%**]
naphthalene	0.184	**0.120** [**−34.8%**]	0.806	**0.207** [**−74.3%**]
salicylic acid	0.299	**0.145** [**−51.5%**]	1.101	**0.557** [**−49.4%**]
toluene	0.099	**0.097** [**−2.0%**]	0.182	**0.124** [**−31.9%**]
uracil	0.129	**0.108** [**−16.3%**]	0.482	**0.199** [**−58.7%**]
MAE reduction with respect to Sum aggregation (ΔE_{mae})		**−18.8%**		**−42.4%**
number of winning tasks	1/8	7/8	0/8	8/8

[a]Patch aggregation improves energy calculations for 7 of 8 molecules and is substantially more accurate in force prediction for all 8 molecules. Values in brackets are the percentage reduction in mean absolute error when using a learnable aggregation mechanism with respect to using Sum aggregation mechanism (ΔE_{mae}).

We compared Patch and Sum aggregation with other learnable aggregation methods MLP, Softmax, and Set Transformer across the GNN models SchNet, DimeNet++, SphereNet, TensorNet, and VisNet (Table 2). Patch aggregation consistently enhanced the performance of these GNN models across the range of QM9 thermodynamic property prediction tasks. Notably, Patch aggregation improved the accuracy of SchNet across all thermodynamic prediction tasks and improved DimeNet++, SphereNet, TensorNet, and VisNet in the large majority of cases. In contrast, other learnable aggregation methods failed to improve, and often degraded the accuracy, of the GNNs, which we attribute to inherent limitations within these methods. For instance, MLP aggregation lacks permutation invariance (i.e., changing the atom input order changes the results), which leads to poor generalization and accuracy.[25] Set Transformer and Softmax aggregation, which employ attention mechanisms, are susceptible to oversmoothing issues[44] that compromise molecular representation learning and accuracy. Importantly, Patch aggregation does not suffer from these limitations. Further, Patch aggregation consistently enhanced SchNet prediction accuracy across all QM9 tasks without requiring an increase in the number of parameters (Table 2). In contrast, other aggregation methods generally diminished SchNet prediction accuracy. Patch aggregation requires only a small number of additional parameters, but the improvement in accuracy is not solely due to the increase in parameters. Patch aggregation is therefore a versatile and general mechanism for enhancing prediction accuracy in diverse scenarios.

A standout advantage of Patch aggregation is that it is parameter-light. MLP aggregation, which uses a large neural network, and Set Transformer aggregation, which uses an attention mechanism and a large feed-forward network, are parameter-heavy and increase the number of parameters relative to the base GNN model by 70 to 512% (Table S1). In contrast, Patch aggregation increases the number of parameters by 0.5 to 17% depending on the number of patches used.

We evaluated Patch aggregation using a second data set of energies and forces from MD simulations (MD17). We took the DimeNet++ model and compared Patch aggregation using 2 patches to Sum aggregation. We found that Patch aggregation consistently achieved lower MAE in predicting molecular energy and forces across 8 organic molecules (Table 6). Despite a limited training of only 1000 molecular conformations, DimeNet++ with Patch aggregation effectively encoded the complex atomic interactions, giving more accurate energy and greatly improved force predictions.

CONCLUSIONS

Graph Neural Networks (GNNs) have become important tools in the quantum chemical property prediction field. They used the graph structure of molecular systems to learn useful representations of molecules that capture their structural and chemical properties. The effectiveness of these models heavily relies on the edge-to-node aggregation mechanism. However, the current state-of-the-art learnable edge-to-node aggregation methods such as multilayer perceptron and set transformers aggregation increase the number of model parameters by around 50% to 500% relative to the GNN models themselves. Addressing these problems, we introduced Patch aggregation, a low parameter, learnable edge-to-node aggregation mechanism that is inspired by the Multi-Head Attention and Mixture of Experts. Patch aggregation segments each edge vector into multiple patches, allowing different patches to receive specialized subsets of information. This ensures that each patch benefits from specialized weighting mechanisms to enable better edge-to-node aggregation.

Using the specialized GNN models SchNet, DimeNet++, SphereNet, TensorNet, and VisNet, we have shown that Patch aggregation consistently outperforms existing state-of-the-art learnable aggregation mechanisms in chemical thermodynamic and molecular dynamics trajectory prediction tasks. Using an optimal number of patches not only enhances prediction accuracy but also improves parameter efficiency, making it more suitable for resource-constrained applications. Patch aggregation is an effective and general method for improving the molecular property prediction with GNN models.

ASSOCIATED CONTENT

Supporting Information

The Supporting Information is available free of charge at https://pubs.acs.org/doi/10.1021/acs.jctc.4c00798.

Detailed ratio comparing the number of parameters in edge-to-node aggregation mechanisms relative to GNN

https://doi.org/10.1021/acs.jctc.4c00798
J. Chem. Theory Comput. 2024, 20, 8886−8896

models (Table S1) and training and validation loss curves (Figure S1) (PDF)

AUTHOR INFORMATION

Corresponding Authors

Daokun Zhang − *School of Computer Science, University of Nottingham Ningbo China, Ningbo 315100, China*; Email: Daokun.Zhang@nottingham.edu.cn

Mario Boley − *Department of Data Science and AI, Faculty of Information Technology, Monash University, Clayton Campus, VIC 3800, Australia*; orcid.org/0000-0002-0704-4968; Email: mario.boley@monash.edu

David K. Chalmers − *Medicinal Chemistry, Monash Institute of Pharmaceutical Sciences, Monash University, Parkville, VIC 3068, Australia*; orcid.org/0000-0003-2366-569X; Email: david.chalmers@monash.edu

Author

Teng Jiek See − *Medicinal Chemistry, Monash Institute of Pharmaceutical Sciences, Monash University, Parkville, VIC 3068, Australia*

Complete contact information is available at: https://pubs.acs.org/10.1021/acs.jctc.4c00798

Notes

The authors declare no competing financial interest.

REFERENCES

(1) Ying, C.; Cai, T.; Luo, S.; Zheng, S.; Ke, G.; He, D.; Shen, Y.; Liu, T.-Y. Do Transformers Really Perform Bad for Graph Representation? *Neural Inform. Processing Syst.* **2021**, *34*, 28877−28888.

(2) Wu, Q.; Zhao, W.; Li, Z.; Wipf, D. P.; Yan, J. NodeFormer: A Scalable Graph Structure Learning Transformer for Node Classification. 2023, arXiv:2306.08385. arXiv.org e-Print archive https://arxiv.org/abs/2306.08385.

(3) Zhang, M.; Chen, Y. Link Prediction Based on Graph Neural Networks. *Neural Information Processing Systems* 2018, *31*.

(4) 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.* **2018**, *31*, 3564−3572.

(5) Han, J.; Cen, J.; Wu, L.; Li, Z.; Kong, X.; Jiao, R.; Yu, Z.; Xu, T.; Wu, F.; Wang, Z.; et al.. A Survey of Geometric Graph Neural Networks: Data Structures, Models and Applications. 2024, arXiv:2403.00485. arXiv.org e-Print archive https://arxiv.org/abs/2403.00485.

(6) Guo, Z.; Yu, K.; Jolfaei, A.; Li, G.; Ding, F.; Beheshti, A. Mixed Graph Neural Network-Based Fake News Detection for Sustainable Vehicular Social Networks. *IEEE Transactions on Intelligent Transportation Systems* **2023**, *24*, 15486−15498.

(7) Sun, J.; Gao, L.; Shen, X.; Liu, S.; Liang, R.; Du, S.; Liu, S. Separated Graph Neural Networks for Recommendation Systems. *IEEE Transactions on Industrial Informatics* **2023**, *19*, 382−393.

(8) Wieder, O.; Kohlbacher, S.; Kuenemann, M.; Garon, A.; Ducrot, P.; Seidel, T.; Langer, T. A compact review of molecular property prediction with graph neural networks. *Drug Discov Today Technol.* **2020**, *37*, 1−12. Fung, V.; Zhang, J. X.; Juarez, E.; Sumpter, B. G. Benchmarking graph neural networks for materials chemistry. *Npj Comput. Mater.* **2021**, *7* (1), 84. Reiser, P.; Neubert, M.; Eberhard, A.; Torresi, L.; Zhou, C.; Shao, C.; Metni, H.; van Hoesel, C.; Schopmans, H.; Sommer, T.; et al. Graph neural networks for materials science and chemistry. *Commun. Mater.* **2022**, *3*, 93 DOI: 10.1038/s43246+022+00315+6.

(9) Thurlemann, M.; Boselt, L.; Riniker, S. Regularized by Physics: Graph Neural Network Parametrized Potentials for the Description of Intermolecular Interactions. *J. Chem. Theory Comput.* **2023**, *19*, 562−579, DOI: 10.1021/acs.jctc.2c00661.

(10) Zhu, S.; Nguyen, B. H.; Xia, Y. C.; Frost, K.; Xie, S. F.; Viswanathan, V.; Smith, J. A. Improved environmental chemistry property prediction of molecules with graph machine learning. *Green Chem.* **2023**, *25* (17), 6612−6617.

(11) Qin, S. Y.; Jiang, S. L.; Li, J. P.; Balaprakash, P.; Van Lehn, R. C.; Zavala, V. M. Capturing molecular interactions in graph neural networks: a case study in multi-component phase equilibrium. *Digit Discovery* **2023**, *2* (1), 138−151.

(12) Smith, J. S.; Isayev, O.; Roitberg, A. E. ANI+1: an extensible neural network potential with DFT accuracy at force field computational cost. *Chem. Sci.* **2017**, *8* (4), 3192−3203.

(13) Smith, J. S.; Nebgen, B. T.; Zubatyuk, R. I.; Lubbers, N.; Devereux, C.; Barros, K.; Tretiak, S.; Isayev, O.; Roitberg, A. E. Approaching coupled cluster accuracy with a general-purpose neural network potential through transfer learning. *Nat. Commun.* **2019**, *10*, 2903 DOI: 10.1038/s41467-019-10827-4.

(14) Kocer, E.; Ko, T. W.; Behler, J. Neural Network Potentials: A Concise Overview of Methods. *Annu. Rev. Phys. Chem.* **2022**, *73*, 163−186, DOI: 10.1146/annurev-physchem-082720-034254.

(15) Schütt, K.; Kindermans, P.-J.; Felix, H. E. S.; Chmiela, S.; Tkatchenko, A.; Müller, K.-R. SchNet: A continuous-filter convolutional neural network for modeling quantum interactions *NIPS*, 30, 2017.

(16) Klicpera, J.; Giri, S.; Margraf, J. T.; Gunnemann, S. Fast and Uncertainty-Aware Directional Message Passing for Non-Equilibrium Molecules. 2020, arXiv:2011.14115. arXiv.org e-Print archive https://arxiv.org/abs/2011.14115.

(17) Liu, Y.; Wang, L.; Liu, M.; Zhang, X.; Oztekin, B.; Ji, S. Spherical Message Passing for 3D Graph Networks. 2021, arXiv:2102.05013. arXiv.org e-Print archive https://arxiv.org/abs/2102.05013.

(18) Simeon, G.; Fabritiis, G. D. TensorNet: Cartesian Tensor Representations for Efficient Learning of Molecular Potentials. 2023, arXiv:2306.06482. arXiv.org e-Print archive https://arxiv.org/abs/2306.06482.

(19) Wang, Y.; Wang, T.; Li, S.; He, X.; Li, M.; Wang, Z.; Zheng, N.; Shao, B.; Liu, T.-Y. Enhancing geometric representations for molecules with equivariant vector-calar interactive message passing. *Nat. Commun.* **2024**, *15*, 313 DOI: 10.1038/s41467-023-43720-2.

(20) Blank, T. B.; Brown, S. D.; Calhoun, A. W.; Doren, D. J. Neural network models of potential energy surfaces. *J. Chem. Phys.* **1995**, *103* (10), 4129−4137.

(21) Staub, R.; Gantzer, P.; Harabuchi, Y.; Maeda, S.; Varnek, A. Challenges for Kinetics Predictions via Neural Network Potentials: A Wilkinson's Catalyst Case. *Molecules* **2023**, *28* (11), 4477.

(22) Raja, S.; Amin, I.; Pedregosa, F.; Krishnapriyan, A. S. Stability-Aware Training of Neural Network Interatomic Potentials with Differentiable Boltzmann Estimators. 2024, arXiv:2402.13984. arXiv.org e-Print archive https://arxiv.org/abs/2402.13984.

(23) Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; Dahl, G. E. Neural Message Passing for Quantum Chemistry. 2017, arXiv:1704.01212. arXiv.org e-Print archive https://arxiv.org/abs/1704.01212.

(24) Xu, K.; Hu, W.; Leskovec, J.; Jegelka, S. How Powerful are Graph Neural Networks? 2018, arXiv:1810.00826. arXiv.org e-Print archive https://arxiv.org/abs/1810.00826.

(25) Buterez, D.; Janet, J. P.; Kiddle, S. J.; Oglic, D.; Lio', P. Graph Neural Networks with Adaptive Readouts. 2022, arXiv:2211.04952. arXiv.org e-Print archive https://arxiv.org/abs/2211.04952.

(26) Chung, J.; Gülçehre, Ç.; Cho, K.; Bengio, Y. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. 2014, arXiv:1412.3555. arXiv.org e-Print archive https://arxiv.org/abs/1412.3555.

(27) Li, G.; Xiong, C.; Thabet, A. K.; Ghanem, B. DeeperGCN: All You Need to Train Deeper GCNs. 2020, arXiv:2006.07739. arXiv.org e-Print archive https://arxiv.org/abs/2006.07739.

(28) Chen, M.; Wei, Z.; Huang, Z.; Ding, B.; Li, Y. Simple and Deep Graph Convolutional Networks. In *International Conference on Machine Learning*, 2020.

(29) Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio', P.; Bengio, Y. Graph Attention Networks. 2017, arXiv:1710.10903. arXiv.org e-Print archive https://arxiv.org/abs/1710.10903.
(30) Corso, G.; Cavalleri, L.; Beaini, D.; Lio', P.; Velickovic, P. Principal Neighbourhood Aggregation for Graph Nets. 2020, arXiv:2004.05718. arXiv.org e-Print archive https://arxiv.org/abs/2004.05718.
(31) Chen, Z.; Deng, Y.; Wu, Y.; Gu, Q.; Li, Y.-F. Towards Understanding the Mixture-of-Experts Layer in Deep Learning. *Neural Information Processing Systems* **2022**, *35*, 23049−23062.
(32) Vaswani, A.; Shazeer, N. M.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; Polosukhin, I. Attention is All you Need.*NIPS* 2017.
(33) Zhou, Y.-Q.; Lei, T.; Liu, H.-C.; Du, N.; Huang, Y.; Zhao, V.; Dai, A. M.; Chen, Z.; Le, Q. V.; Laudon, J. Mixture-of-Experts with Expert Choice Routing. 2022, arXiv:2202.09368. arXiv.org e-Print archive https://arxiv.orgabs/2202.09368.
(34) Shazeer, N. M.; Mirhoseini, A.; Maziarz, K.; Davis, A.; Le, Q. V.; Hinton, G. E.; Dean, J. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer, 2017, arXiv:1701.06538. arXiv.org e-Print archive https://arxiv.org/abs/1701.06538.
(35) Zhang, Y.; Liu, C.; Liu, M.; Liu, T.; Lin, H.; Huang, C.-B.; Ning, L. Attention is all you need: utilizing attention in AI-enabled drug discovery. *Briefings Bioinf.* **2023**, *25*, bbad467 DOI: 10.1093/bib/bbad467.
(36) Frank, J. T.; Unke, O. T.; Müller, K.-R. So3krates - Self-attention for higher-order geometric interactions on arbitrary length-cales. 2022, arXiv:2205.14276. arXiv.org e-Print archive https://arxiv.org/abs/2205.14276. Pan, X.; Ye, T.; Xia, Z.; Song, S.; Huang, G. Slide-Transformer: Hierarchical Vision Transformer with Local Self-Attention. 2023, arXiv:2304.04237. arXiv.org e-Print archive https://arxiv.org/abs/2304.04237.
(37) Riquelme, C.; Puigcerver, J.; Mustafa, B.; Neumann, M.; Jenatton, R.; Pinto, A. S.; Keysers, D.; Houlsby, N. Scaling Vision with Sparse Mixture of Experts. *Adv. Neural Information Processing Systems* **2021**, *34*, 8583−8595.
(38) Ramakrishnan, R.; Dral, P. O.; Rupp, M.; von Lilienfeld, O. A. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data* **2014**, *1* (1), No. 140022.
(39) Chmiela, S.; Tkatchenko, A.; Sauceda, H. E.; Poltavsky, I.; Schütt, K. T.; Müller, K.-R. Machine learning of accurate energy-conserving molecular force fields. *Sci. Adv.* **2017**, *3* (5), No. e1603015.
(40) Fey, M.; Lenssen, J. E. Fast Graph Representation Learning with PyTorch Geometric. 2019, arXiv:1903.02428. arXiv.org e-Print archive https://arxiv.org/abs/1903.02428.
(41) Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al.. PyTorch: An Imperative Style, High-Performance Deep Learning Library, 2019, arXiv:1912.01703. arXiv.org e-Print archive https://arxiv.org/abs/1912.01703.
(42) Kingma, D. P.; Ba, J. Adam: A Method for Stochastic Optimization *CoRR* 2014, arXiv:1412.6980. arXiv.org e-Print archive https://arxiv.org/abs/1412.6980.
(43) Gotmare, A. D.; Keskar, N. S.; Xiong, C.; Socher, R. A Closer Look at Deep Learning Heuristics: Learning rate restarts, Warmup and Distillation. 2018, arXiv:1810.13243. arXiv.org e-Print archive https://arxiv.org/abs/1810.13243.
(44) Wu, X.; Ajorlou, A.; Wu, Z.; Jadbabaie, A. Demystifying Oversmoothing in Attention-Based Graph Neural Networks. 2023, arXiv:2305.16102. arXiv.org e-Print archive https://arxiv.org/abs/2305.16102.

https://doi.org/10.1021/acs.jctc.4c00798
J. Chem. Theory Comput. 2024, 20, 8886−8896