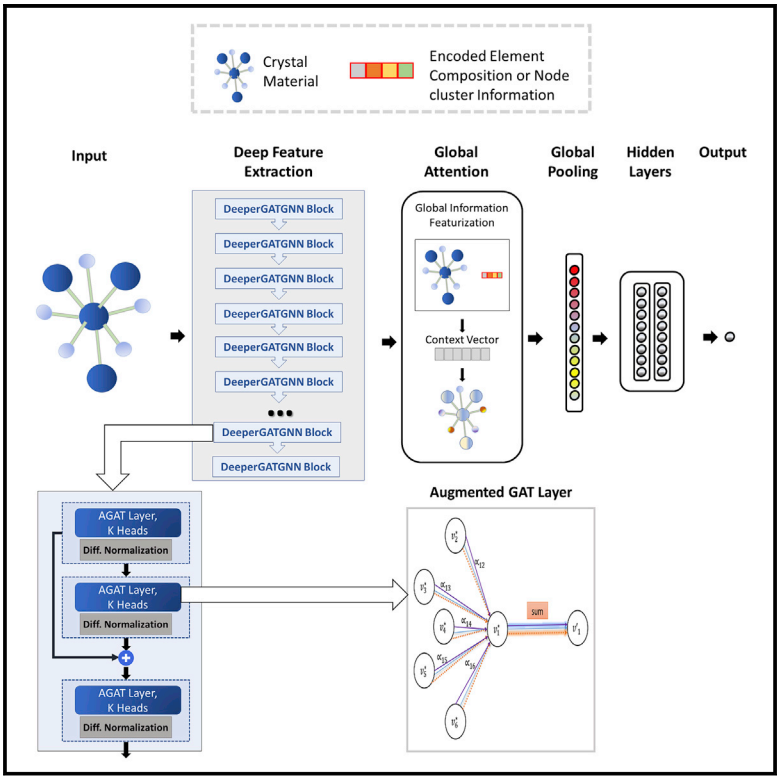
Patterns

Scalable deeper graph neural networks for high-performance materials property prediction

Graphical abstract



Highlights

- Develop scalable graph neural networks with more than 30 layers without overfitting
- State-of-the-art performance for materials property prediction
- Network architecture design to address the over-smoothing issue of GNNs
- Systematic benchmark study over existing GNNs for material property prediction

Authors

Sadman Sadeed Omee, Steph-Yves Louis, Nihang Fu, ..., Rongzhi Dong, Qinyang Li, Jianjun Hu

Correspondence

jianjunh@cse.sc.edu

In brief

We develop scalable graph neural network (GNN) models for high-performance structure-based materials property prediction, by combining a global attention mechanism with differentiable group normalization and residual connections to address the over-smoothing issue. Our model achieves state-of-the-art performance over five out of six datasets. Compared with existing GNNs, our DeeperGATGNN model scales up to more than 30 layers without significant performance degradation while all other GNNs lack this scalability, which can build large GNN models for any application domain.

Omee et al., 2022, Patterns *3*, 100491
May 13, 2022
https://doi.org/10.1016/j.patter.2022.100491

Patterns

CellPress
OPEN ACCESS

Article

Scalable deeper graph neural networks for high-performance materials property prediction

Sadman Sadeed Omee,[1,2] Steph-Yves Louis,[1,2] Nihang Fu,[1] Lai Wei,[1] Sourin Dey,[1] Rongzhi Dong,[1] Qinyang Li,[1] and Jianjun Hu[1,3,*]

[1]Department of Computer Science and Engineering, University of South Carolina, Columbia, SC 29201, USA
[2]These authors contributed equally
[3]Lead contact
*Correspondence: jianjunh@cse.sc.edu
https://doi.org/10.1016/j.patter.2022.100491

THE BIGGER PICTURE Modern deep-learning-based generative models have made it possible to computationally design millions of hypothetical materials. However, fast and accurate materials property prediction models are needed to conduct large-scale screening of these candidates for new materials discovery. Graph neural networks (GNNs) have emerged as the most competitive models for materials property prediction, with their performance and scalability, however, still being constrained by the over-smoothing issue. We present DeeperGATGNN, a global attention-based GNN with differentiable group normalization and residual connection to achieve not only state-of-the-art performance for five out of six datasets but also high scalability. Our technique allows us to build very deep GNNs without significant performance degradation as other GNNs do. Our models can be generally used to build scalable GNNs for any application domain, especially where large deep learning models are needed.

Development/Pre-production: Data science output has been rolled out/validated across multiple domains/problems

SUMMARY

Machine-learning-based materials property prediction models have emerged as a promising approach for new materials discovery, among which the graph neural networks (GNNs) have shown the best performance due to their capability to learn high-level features from crystal structures. However, existing GNN models suffer from their lack of scalability, high hyperparameter tuning complexity, and constrained performance due to over-smoothing. We propose a scalable global graph attention neural network model DeeperGATGNN with differentiable group normalization (DGN) and skip connections for high-performance materials property prediction. Our systematic benchmark studies show that our model achieves the state-of-the-art prediction results on five out of six datasets, outperforming five existing GNN models by up to 10%. Our model is also the most scalable one in terms of graph convolution layers, which allows us to train very deep networks (e.g., >30 layers) without significant performance degradation. Our implementation is available at https://github.com/usccolumbia/deeperGATGNN.

INTRODUCTION

Out of the almost infinite chemical design space, the largest inorganic crystal materials database (ICSD)[2] contains only 250,000 crystal materials up to July 13, 2021. To push the boundary of existing materials properties, modern artificial intelligence (AI) and machine learning (ML) techniques are now laying the ground for discovering novel materials for ultra-long-life batteries for cell phones and electric vehicles, highly efficient solar panels, room temperature superconductors, etc.[3–9] One of the most promising approaches for exploring the vast materials design space is the deep learning (DL) based generative design paradigm. In this approach, existing materials are fed to a neural network based deep generative model, which learns the atomic assembling rules to form stable crystal structures and uses these rules to generate chemically valid hypothetical structures[4,7,9] or compositions.[8] While these candidate materials can be generated quickly in millions, a fast and accurate materials property

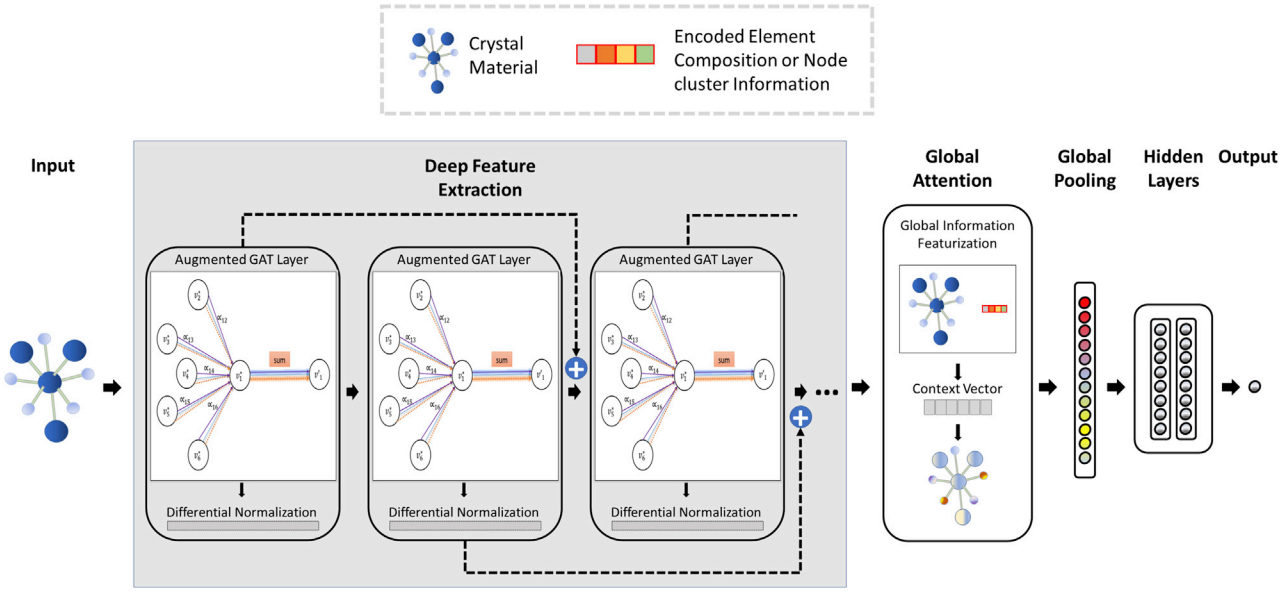


Figure 1. An overview of the DeeperGATGNN architecture

An initial graph-encoded material (on the left in blue) is used as input. Then several AGAT layers (with 64 neurons), followed by DGN are applied. A skip connection is added from the output of the l-th AGAT layer to the output of the $(l+1)$-th AGAT layer (after applying DGN). This completes the deep node feature extraction process. Next, a global attention layer is applied where the node feature vectors are concatenated with the composition encoded vector. After feeding to two fully connected layers, this yields a context vector containing the weight relating to each node's location. The context vector is multiplied with the node feature vectors and then a global pooling of the node feature vectors is applied. The node features are then passed through one to two hidden layers before finally producing the output property via another fully connected layer.

prediction model is needed to screen the most promising ones for further slow property characterization either by first principle density functional theory (DFT) or molecular dynamic (MD) calculations or via experiments. After all these, exotic materials in uncharted design space may be found. In the overall pipeline, fast and accurate prediction models are critically needed for diverse materials properties, such as formation energy, band gap, surface adsorption energy, ion and thermal conductivity, etc.

Indeed, ML models for materials properties have emerged as one of the most promising approaches for materials discovery due to their high prediction accuracy and speed compared with the first-principles calculations.[10] Both composition and structure-based ML models can successfully predict materials properties, whose performance, however, is strongly dependent on the ML algorithm selection, the features, and the available datasets' quality and amount. Among these two types of screening models, the composition-based ML models[11,12] have the advantages of speed and capability to screen large-scale hypothetical compositions generated by generative DL models.[8] However, almost all materials properties are strongly dependent on the materials structures, so the structure-based materials prediction models tend to have much higher prediction accuracy,[13–15] which can be used to screen known materials structure repositories such as the ICSD[2] or the Materials Project Database,[16] or hypothetical crystal materials with structures created by modern generative DL models.[9,17] Structural information of crystal materials can be represented using several methods,[18] including structure graph, Coulomb matrix,[19] Voronoi tessellation,[10] diffraction fingerprint, or voxel grids.[20] However, due to the scarcity of structure data and property labels, it remains an unsolved problem to achieve highly accurate materials property predictions from structures.

Currently, there are two major categories of ML approaches for structure-based materials property prediction based on their descriptors or features used: (1) the heuristic feature-based models,[21–23] of which the features are designed based on existing physicochemical knowledge; (2) learned feature-based models, of which the descriptors are learned by DL algorithms.[13,14,24] While the heuristic feature-based ML models have demonstrated some successes in a variety of applications, such as formation energy prediction[21] and ion conductivity screening,[25] large-scale benchmark studies have shown that the representation learning based deep graph neural network (GNN) models have achieved much better performance in materials property prediction, which highlights the importance of developing more advanced DL models for materials property prediction.[1,26]

Since 2018, a variety of GNNs have been proposed to improve the prediction performance, such as SchNet,[24] CGCNN,[13] MEGNet,[14] MPNN,[27] iCGCNN,[28] and GATGNN.[29] Each of these architectures has utilized the graph representation as input along with slightly different additional information, convolution operators, and neural network architectures.[13,14,24] However, a recent large-scale benchmark study over five different datasets of varying sizes[1] has shown that, while the performance of the existing GNN models is in general much better than those of non-GNN approaches, the best four GNN models' performances tend to be saturated without significant difference between one another. For example, for the bulk crystal formation energy prediction problem, the mean absolute error (MAE) range is 0.046 eV/atom

Table 1. Details of the six benchmark datasets used in this work

Dataset	Unit	Source	No. of elements	No. of samples
Bulk Materials Formation Energy	eV/atom	Materials Project[16]	87	36,839
Alloy Surface Adsorption Energy	eV	CatHub[38]	42	37,334
Pt-cluster Total Energy	eV	the literature[39]	1	19,801
2D Materials Work Function	eV	C2DB[40]	60	3,814
MOF Band Gap	eV	QMOF[26]	78	18,321
Bulk Materials Band Gap	eV	Materials Project[16]	87	36,837

(MPNN) and 0.05 eV/atom (SchNet). Also, there is no dominant winner among these four GNN models. After close investigation, we find that the optimized architectures of these models have only one to nine graph convolution (GC) layers, which is in sharp contrast to other fields of ML applications, such as computer vision and natural language processing, etc. In computer vision, ResNet[30] and DenseNet[31] with up to 1,000 layers have been trained. In natural language modeling, the smallest GPT-3 model (125M parameters) has 12 attention layers and the largest GPT-3 model (175B parameters) uses 96 attention layers.[32] Considering that, in the materials property prediction problem, the number of element types and their sophisticated interactions are both much more complex than the pixels and their neighboring patterns, we believe very deep networks are needed to achieve significantly better results than the current state-of-the-art (SOTA) results as reported in the most recent benchmark study.[1]

Recently, Jha et al.[33] applied the residual skip-connection idea to vector input-based materials property prediction. Their experiments showed that, when the dataset size is more than 15,000, their individual residual networks architecture outperforms both the plain multi-layer perceptron networks and the stacked residual network. For composition datasets with only 234,299 samples, their 48-layer IRNet beat their 17-layer IRNet, which outperformed all other ML and plain neural network models. However, no studies have been shown on GNNs, which can better capture how the structural features affect their properties. Another study by Yang et al.[34] applied a deep convolution network with residual skip connections for crystal plasticity prediction with good performance. However, their models are still limited to vector representations instead of graph representations. To our knowledge, there is no study on whether deeper GNNs can significantly push the frontier or SOTA performance in materials property prediction.

In this work, we propose DeeperGATGNN, a very deep graph attention neural network model for large-scale materials property prediction using differentiable group normalization (DGN) and skip-connections. Our architecture allows training very deep in terms of GC layers. Our model also has a useful characteristic—no tedious and expensive hyperparameter tuning is needed, only a sufficient number of GC layers needs to be set. We also apply our scaling strategy (DGN and skip-connections) to the other four GNNs and achieve significant performance improvements on some datasets for most of the models. We call these models DeeperCGCNN, DeeperMEGNet, DeeperMPNN, and DeeperSchNet. Our contribution in this paper can be summarized as follows:

- We address the main challenge and bottleneck of GNN for materials property prediction and propose a novel global attention-based GNN architecture that uses an efficient technique (DGN + skip-connections) for increasing the networks' depth to overcome the barrier. The simplicity of our model with almost hassle-free hyperparameter tuning and high scalability makes it ideal for large-scale materials property prediction.
- We evaluate our DeeperGATGNN model on six public benchmark datasets and show that our model achieves SOTA results on five out of six datasets with significant performance improvements, with MAE errors reduction up to 10% over previous SOTA results.
- Empirically, we show that our model is the most scalable one among all existing GNN models for materials property prediction despite having a very low number of training parameters compared with those of other models.
- We demonstrate that our deeper GNN enabling strategy for materials property prediction can also be applied to the other four GNNs to achieve improved performance on several benchmark datasets.

RESULTS

Architecture description

DeeperGATGNN is based on our previously proposed GATGNN[29] model. In GATGNN, we developed a GNN that uses two graph soft-attention variants to learn inorganic molecules properties.[35–37] The first type of soft-attention consists of additive multi-head attention (four or eight) applied to the one-hop neighbors of each atom. These attention layers are named augmented graph attention (AGAT) layers because we augment the node feature vectors with the features from their connecting edge. These AGAT layers are only used to extract the locally dependent features between neighboring atoms. Afterward, upon extracting the local features, GATGNN then uses a unique soft-attention at the end to transform neighborhood-dependent information to a global context (concerning all other atoms in the crystal). The local soft-attention α_{ij} between a node i and a neighbor j can be represented as the following equation:

$$\alpha_{ij} = \frac{\exp(a_{ij})}{\sum_{k \in N_i} \exp(a_{ik})}. \quad \text{(Equation 1)}$$

In Equation 1, N_i represents the neighborhood of node i and a_{ij} is the parameterized weight coefficient between nodes i and j, which represents the importance of node j to node i. The global attention: g_i, which is applied right before the global pooling, calculates each node's overall importance. It can be described as the following equation:

$$g_i = \frac{(\mathbf{x}_i \parallel \mathbf{E}) \cdot \mathbf{W}}{\sum_{x_c \in \mathbf{x}} (\mathbf{x}_c \parallel \mathbf{E}) \cdot \mathbf{W}}. \quad \text{(Equation 2)}$$

In Equation 2, $\mathbf{x} \in \mathbb{R}^F$ represents a learned embedding, $\mathbf{E}$ is a compositional vector of the crystal, $\mathbf{W} \in \mathbb{R}^{1 \times (F + |E|)}$ is a

Table 2. Five-fold cross-validation performance comparison of DeeperGATGNN (with 10, 15, 20, 25, and 30 GC layers) versus other SOTA GNN models for different materials property prediction problems

ML models	GC layers	Bulk crystals (Eform)	Alloy surfaces	MOFs	2D materials	Pt clusters	Band gap (bulk)
SchNet	misc.	0.05	0.063	0.228[a]	0.214	0.151[a]	0.2817
MPNN	misc.	0.046	0.058[a]	0.245	0.204[a]	0.182	0.2649
CGCNN	misc.	0.049	0.060	0.233	0.208	0.205	0.2598[a]
MEGNet	misc.	0.048	0.069	0.253	0.224	0.180	0.2659
GATGNN	5	0.0454[a]	0.0806	0.2422	0.2075	0.2825	0.2734
DeeperGATGNN	10	0.0302	0.0502	0.2238	0.1916	**0.1298**	0.2624
DeeperGATGNN	15	**0.0296**	0.0416	0.2178	0.2139	0.1363	0.2559
DeeperGATGNN	20	0.0297	**0.0409**	**0.2158**	0.1775	0.1321	0.2550
DeeperGATGNN	25	0.0306	0.0411	0.2169	**0.1718**	0.1413	**0.2457**
DeeperGATGNN	30	0.0304	0.0427	0.2178	0.1730	0.1522	0.2459
% of improvements over previous best		34.97	29.55	5.34	15.76	14.03	5.42
% of improvements over GATGNN		34.97	49.32	10.07	17.16	54.04	10.13

[a]The previous SOTA results.

parameterized matrix, and x_c is the learned embedding of any atom c within the crystal. By using the combination of three to five of these local soft-attentions and one global soft-attention, GATGNN was able to match SOTA of inorganic materials properties prediction for most of the properties (except formation energy) at the time and also provide interpretable results in terms of each atom's contribution. Nevertheless, the over-smoothing issue—a general challenge that prevents using more than a few layers in GNNs—also affects GATGNN. GATGNN model's performance begins to considerably decrease by adding eight or more GC layers. With the expectation that a deeper model should be able to extract even more of these inter-atomic-dependent features, we aim to overcome this over-smoothing limitation so that our model can more effectively extract the physics-dependent features of crystals. So, we devised DeeperGATGNN (later we show that it can go very deep in terms of GC layers), which uses additive skip-connections between these attention layers that extract the local features and further improve the learning by including DGN layers.

The overall DeeperGATGNN architecture is shown in Figure 1. It consists of several AGAT layers, followed by DGN operators and skip connections between each of these layers, which are followed by a global attention layer and a global pooling layer. Finally, a few fully connected hidden layers are added before one fully connected layer to produce the predicted output.

Dataset description

We evaluate our model's performance and other baselines using six datasets, including five benchmark datasets used in a previous evaluation study.[1] The first five datasets are all for formation/surface energy predictions of nanocluster materials (Pt-cluster), alloy surface, bulk materials, 2D materials, and MOF materials. In addition, we include a band gap dataset for the same bulk materials set. The datasets' details are shown in Table 1. Total samples and total elements range from 3,814 to 37,334 and 1 to 87, respectively, reflecting the datasets' diversity and the challenge in predicting corresponding materials properties.

Model comparison

To fairly and objectively evaluate and compare our model's performance with other SOTA models, it is critical to ensure all models are trained on the same datasets with appropriate training and optimal hyperparameters and using the same cross-validation method. An excellent benchmark study by Fung et al.[1] implemented seven different prediction models (including four GNN models) and a dummy baseline model in the same code base and evaluated their performance on the same set of five datasets using large-scale computationally expensive hyperparameter tuning (limited to 200 epochs due to the computational burden) to identify eight optimal parameters for all models on different datasets. To compare, we re-implemented our model in their code framework and use a single parameter setting (with 10, 15, 20, 25, and 30 GC layers) for all datasets (see Note S1). All the models are trained with 2,500 epochs. The results are shown in Table 2, in which the results for SchNet, MPNN, CGCNN, and MEGNet are all from the previous benchmark study.[1]

We found from the benchmark study[1] that, for different datasets, there are different winner models with different hyperparameter settings except that MEGNet does not win on any of the six datasets. Our previous GATGNN achieved better results on the bulk crystal formation energy prediction problem than the other four models. However, our DeeperGATGNN with 20 GC layers beat all the previous best results. We further tried 25 and 30 GC layers, which led to further improvements for the 2D Materials dataset and the band gap problem. The last row of Table 2 summarizes the performance improvement percentages ranging from 5.34% (for band gap prediction) to a significant 34.97% (for bulk crystal formation energy prediction), all achieved with a single hyperparameter setting (except GC layers) across six different datasets. Compared with our

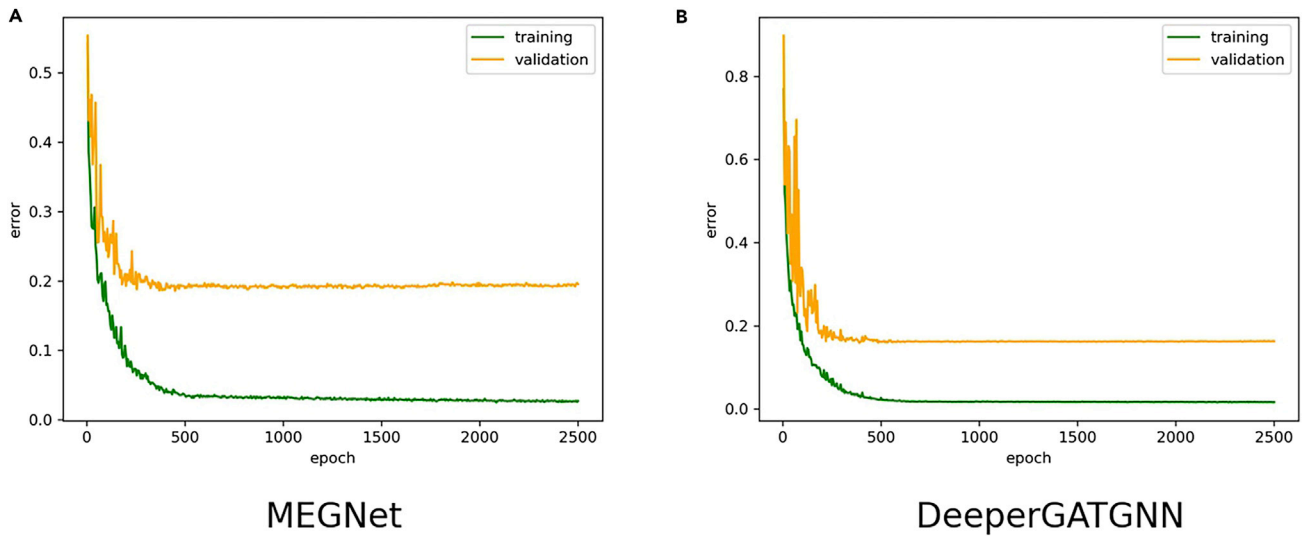


Figure 2. Training/validation error plots of MEGNet and DeeperGATGNN

(A–B) Training/validation errors of (A) MEGNet and (B) DeeperGATGNN on the 2D Materials dataset. All models become stagnant after 500 epochs. So our DeeperGATGNN models trained with 2,500 epochs should have similar performance when trained with 500 epochs.

previous GATGNN model, by addressing the over-smoothing issue using DGN and skip connections, our DeeperGATGNN models achieve from 10.07% to 54.04% reduction in the MAE prediction errors across the six materials prediction problems.

Next, we investigated if the number of epochs had made the difference in performance. So, we plotted the training/validation errors over the training process for all the six GNN models; two of them are shown in Figure 2 and others are shown in Figure S2. We noticed that, with the current hyperparameter setting (especially the learning rate scheduling), almost all models stagnate at around 500 rather than the 200 epochs that were used in the previous benchmark study.[1] The new results of all the models with 500 epochs and our model's best results are summarized in Table 3.

We found that all models' optimal performances have been significantly improved by increasing the training epochs to 500 (see Note S3 for more details). However, our DeeperGATGNN model achieves the best results over five out of six datasets, with performance improvements from 0.05% to 10.29%. Since training such large-scale GNNs is very computationally intensive (some models take 2–7 days to train), finding optimal hyperparameters with large epochs is infeasible. In this case, our DeeperGATGNN's easy hyperparameter setting is a very attractive feature combined with its outstanding SOTA performance.

We further checked whether increasing the number of GC layers can improve existing GNN models' performance. We found that it does not help (see Figure 7). Increasing the GC layers moderately deteriorates SchNet's and GATGNN's performance and also of CGCNN's and MEGNet's performance to a lower degree.

As most existing message-passing-based GNN models suffer from the over-smoothing issue, we checked if DGN and skip connections can help improve other GNN models so that they can also benefit from deeper GC layers. To verify this, we increased the GC layers to 10 for all the models as evaluated in the benchmark study, replaced their original weight normalization method with DGN and added the skip connections, and trained the models using their optimal hyperparameters except that we used 500 epochs. Then we calculated the performance changes after these modifications. The results are shown in Figure 3. For the SchNet model, the DeeperSchNet achieves a 19% reduction in MAE error for the Bulk Materials Formation Energy dataset, while getting worse results for all other datasets. For the Pt clusters, its performance dropped by 29.5%. For the MPNN model, all enhanced models achieve much worse results, ranging from −14.2% to −47.7%, demonstrating its lack of scalability. For CGCNN, surprisingly for all datasets except the Pt clusters, its results became worse. However, for the Pt clusters, the DeeperCGCNN reduces the MAE error of CGCNN by almost 43% from 0.30 to 0.17 eV. MEGNet and GATGNN are the only two models that get performance boosting from adding DGN and skip connections.

To further explain why the existing GNN models benefit little from the DGN and skip-connections in most cases, we plotted the number of parameters for these models and their deeper versions as shown in Figure 4. We found that, for MPNN and DeeperMPNN, the parameter number increases very rapidly, reaching more than 8 million when the number of GC layers reaches 8. With limited training samples, it causes some serious problems. The second most parameter-rich models are MEGNet and DeeperMEGNet, both have almost 6 million parameters when the number of GC layers approximate 30, which leads to their collapsed performance (see Figure 7). The CGCNN and SchNet models along with their variants are more parameter parsimonious, but adding more layers does not improve the results except for some special datasets. Our GATGNN and DeeperGATGNN are the most parsimonious models. Even with 50 GC layers, the number of parameters is under 1.8 million compared with MEGNet's and DeeperMEGNet's more than 9 million parameters, which

Table 3. DeeperGATGNN's performance comparison with corrected benchmark results of other models (GCL denotes the number of GC layers of the corresponding optimal models)

ML models	Bulk crystals	GCL	Alloy surfaces	GCL	MOFs	GCL	2D materials	GCL	Pt clusters	GCL
SchNet	0.0556	1	0.0470	5	0.2312	9	0.2240	4	0.1726	9
MPNN	0.0349	5	0.0488	5	0.2070	4	0.1893	2	0.1384[a]	3
CGCNN	0.0349	7	0.0424[a]	8	0.2141	6	0.2001	6	0.3024	1
MEGNet	0.0329[a]	5	0.0469	8	0.1968[a]	7	0.1719[a]	4	0.2877	8
GATGNN	0.0475	5	0.0727	5	0.2265	5	0.1869	5	0.1748	5
Ours	**0.0296**	15	**0.0409**	20	0.2158	20	**0.1718**	25	**0.1298**	10
% improve	**10.29**		**3.61**		−9.66		**0.05**		**6.21**	

[a]The best results of the other models.

may partially explain why DeeperGATGNN can improve the base model significantly.

We also plotted the scatterplots of the predicted surface energies versus the true values for the Alloy Surface dataset (see Figure S1), over which the DeeperGATGNN achieved the best performance, with an MAE score of 0.041 eV. We can see that the plot in Figure S1F has the smallest deviation from the diagonal lines with a more narrow distribution of the points.

Parameter study

We conducted different parameter studies of our DeeperGATGNN model using 10 GC layers and 500 epochs. The Pt-cluster dataset is used here for this purpose. We calculated the results using 5-fold cross-validation as is done in the previous subsection.

First, we experimented with how the dropout rate affects the prediction performance. The best result we achieved on this dataset in the SOTA performance study is 0.1298 eV (MAE) using 10 GC layers and no dropout. Several experiments with varying dropout rates are run and the results are shown in Figure 5A. It is observed that increasing the dropout rate degrades our model's performance. Even with a large number of GC layers, our model is very scalable and does not need any dropout. Our model with 35 GC layers achieves ever better results without dropout (see Figure 5D). Also, in the next subsection, we show that, except for our model, all the other models perform very poorly with a certain number of GC layers, but our model performs fine even with 30 GC layers, all without using any dropout.

Second, we evaluated how changing the learning rate affects our model performance. We can see from Figure 5B that the best result is achieved with the learning rate of 0.005, which we use as the default learning rate for our architecture. We would like to mention that we used a learning rate scheduler library for each of the experiments as done in the benchmark study.[1]

Third, we ran experiments with different batch sizes for our model. We increased the batch size from 100 (default batch size for our model) up to 500 with intervals of 100. It is observed from Figure 5C that a larger batch size usually leads to worse performance than smaller ones[41,42] and our model performs best with the default batch size.

Next, we investigated the prediction performance by changing our model's activation function. By default, our DeeperGATGNN model uses the Softplus activation function. We compared the Softplus activation function result (MAE: 0.1298 eV) with those of ReLU and Leaky-ReLU activation functions. The MAE values obtained for ReLU and Leaky-ReLU were 0.1514 and 0.1511 eV, respectively. We can see that the Softplus activation function beats the other activation functions by a large margin.

We then checked how our DeeperGATGNN model's performance might be improved by increasing the number of GC layers while keeping other hyperparameters as defaults (no dropout, batch size 100, learning rate 0.005, and Softplus activation function). Five-fold cross-validation was used for performance evaluation in this experiment. The GC layers' impact is shown in Figure 5D. We found that, when the number of GC layers is less than 30, the best result is achieved using 10 GC layers (MAE: 0.1298 eV). Our model's performance starts to degrade with increasing number of GC layers until it reaches 35 GC layers when it achieves a new SOTA result (MAE: 0.1280 eV) for this dataset. We further examined up to 50 GC layers and found no better result than the result by DeeperGATGNN with 35 GC layers. One important point to note is that, even with 50 GC layers, our model's performance has not degraded too much: the results with 30 and 50 GC layers are very close. So, if the SOTA result can be achieved by our model with 35 GC layers, there is a possibility that we might achieve even better results if we keep going deeper in terms of GC layers (>50 GC layers) on this dataset with more training samples. But, due to computational burden, we did not go any deeper.

We also examined our model with different cutoff radius values (on all the datasets) where we got the best results for the default 8 Å radius. The results are shown in Table S1.

Finally, we conducted experiments to evaluate the effect of changing the training set size. DL architectures' success largely depends on the amount of training data and our model is no different. Figure 6 shows that our model's accuracy continues to improve with increasing number of training set samples. We believe that, if more samples are added to this dataset, our model can achieve even better performance than the current SOTA result.

Scalability comparison

We investigated our DeeperGATGNN's scalability and that of other models (both shallow and deeper versions). We used the Bulk Materials Formation Energy dataset for this purpose. All the experiments are again conducted using 500 epochs and 5-fold cross-validation. We trained each of the models for 4, 6, 8, 10, 15, 20, 25, and 30 GC layers and examined their scalability. We limited our experiments to 30 GC layers because we showed

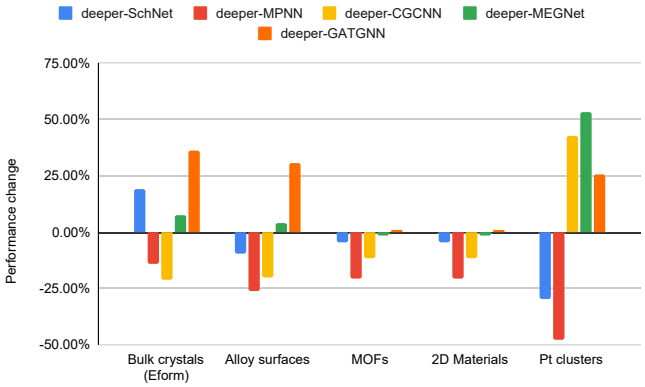


Figure 3. Performance change of all GNN models after modification
Performance change of each GNN model after implementing DGN and skip-connection and then adding more GC layers. GATGNN shows the highest performance improvement after the modification (DeeperGATGNN).

that 30 GC layers are enough to conclude that our model is the most scalable one among all. We excluded MPNN from this experiment due to its exceptionally large parameter number: it has approximately 4.39, 10.9, and 32.6 million trainable parameters for 4, 10, and 30 GC layers, respectively, which are much higher than any other model on average. So the memory it costs and the time it takes to finish training are too huge to be conducted for this experiment.

First, we examined how the change of parameter number affects the prediction performance with increasing numbers of GC layers for existing graph networks as shown in Figure 7. We can spot that all the existing GNNs deteriorate after adding a certain number of GC layers, i.e., the MAE becomes too large so that it can no longer be used for accurate property prediction. For example, SchNet deteriorates with 20 or more GC layers. The MAE value of SchNet increases from 0.0516 eV/atom (15 GC layers) to 0.1893 eV/atom (20 GC layers) where the number of parameters increases from 1,102,401 (15 GC layers) to 1,455,901 (20 GC layers).

CGCNN becomes unscalable with 30 GC layers. The MAE value and the number of trainable parameters increase from 0.0351 eV/atom and 1,337,101 for 25 GC layers to 2.2776 eV/atom and 1,590,101, respectively for 30 GC layers. The same deterioration occurs for MEGNet as well, which also deteriorates with 30 GC layers, and the MAE value and the number of parameters increase from 0.0302 eV/atom and 4,857,201 for 25 GC layers to 0.2237 eV/atom and 5,819,201, respectively, for 30 GC layers. But the change of MAE value is not as drastic as that of CGCNN, although MEGNet has a much higher number of parameters than that of CGCNN. For better visualization, we limit the y axis of Figure 7 to 1.25 eV/atom, which is why the MAE value plot gets trimmed for CGCNN for 30 GC layers.

Our earlier GATGNN model also deteriorates with 30 GC layers. It also has a drastic performance change from 25 to 30 GC layers, such as with CGCNN. The MAE and the number of parameters increase from 0.0575 eV/atom and 863,370 for 25 GC layers to 2.7937 eV/atom and 1,030,770, respectively, for 30 GC layers. We can see that both GATGNN and CGCNN have a much smaller number of trainable parameters compared with that of MEGNet and they are also very similar in terms of scalability.

As we saw that none of the existing GNNs can scale up to 30 or more GC layers, next we examined these models' deeper versions. DeeperSchNet again deteriorates with 20 GC layers, which means that the modification did not improve its scalability. The MAE value and the number of parameters change from 0.0421 eV/atom and 1,144,401 (for 15 GC layers) to 0.9656 eV/atom and 1,511,901, respectively (for 20 GC layers). Although the result for DeeperSchNet with 15 GC layers is better than that of the original SchNet, surprisingly the change of MAE for 20 GC layers is much more drastic in DeeperSchNet than that of SchNet.

Both DeeperCGCNN and DeeperMEGNet perform worse than the original CGCNN and MEGNet in terms of scalability. DeeperCGCNN deteriorates with 20 GC layers, which is quicker than the original CGCNN, which deteriorates with 30 GC layers. The MAE and the number of parameters increase from 0.0425 eV/atom and 873,101 (for 15 GC layers) to 0.7783 eV/atom and 1,140,101, respectively (for 20 GC layers). DeeperMEGNet also cannot scale up to 20 GC layers. The MAE and the number of parameters increase from 0.03 eV/atom and 3,185,201 (for 15 GC layers) to 0.9651 eV/atom and 4,231,201, respectively (for 20 GC layers). We did not do experiments with DeeperMEGNet for 25 and 30 GC layers because it already became unscalable and the number of parameters is already huge for just 20 GC layers. We also want to highlight that, for the CGCNN model, its deeper version DeeperCGCNN can only reduce the MAE for 4 GC layers. Instead for the MEGNet model, our modification implemented in DeeperMEGNet improves its prediction performance for almost all 4, 6, 8, 10, and 15 GC layers before it deteriorates with 20 GC layers.

Now we discuss our DeeperGATGNN model's scalability. We can see that our model achieves the SOTA result with 15 GC layers on this dataset (see Table 2). Our model is the only model that did not deteriorates with even 30 GC layers. Also, the number of trainable parameters of our model is the smallest among other GNNs (except GATGNN) for each specific number of GC layers. The MAE and the number of parameters of our model with 30 GC layers are 0.0304 eV/atom and 1,089,906, respectively. One of the key points to note here is that, although our model can scale up to 30 or more GC layers, the performance did not improve after 15 GC layers (but also did not get worse). A similar observation is also found in the experiments over the Pt-cluster dataset in the previous subsection: our model's performance did not improve after 10 GC layers until reaching 35 GC layers. So as our model shows its capability to scale to 30 or more GC layers, there is a strong possibility that our model might achieve an even better result on this dataset if we go deeper with more training samples. Overall, our experiments with 30 GC layers have shown that our DeeperGATGNN model is the most scalable model of all. We would like to mention that we also tried using a little dropout with 20, 25, and 30 GC layers with our model but it did not improve our results. Since our model performance does not become worse with increasing number of GC layers, it tends to have strong robustness against overfitting.

Ablation study

We perform ablation experiments on DeeperGATGNN to understand the contribution of each component in the prediction

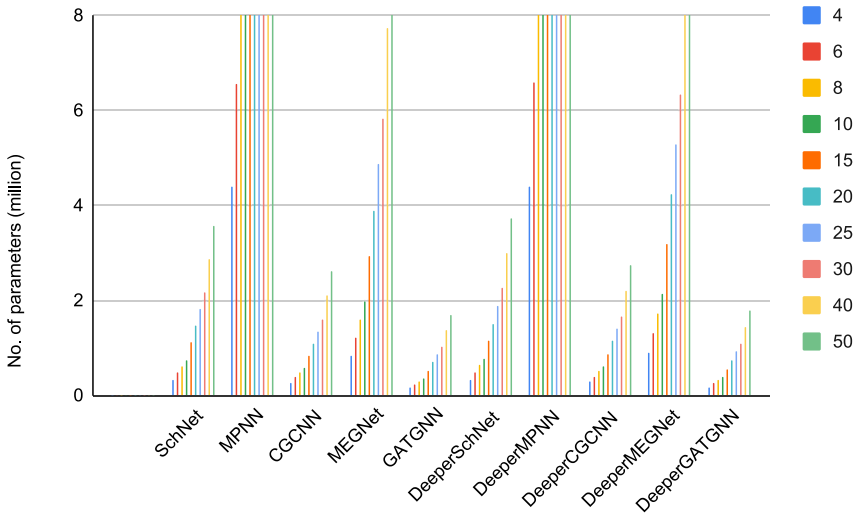


Figure 4. Parameter numbers for GNN models with increasing numbers of GC layers

We limit the y axis to 8 million for better visualization. GATGNN and its improved and deeper version DeeperGATGNN have a much shorter number of parameters compared with other GNN models and their deeper versions.

accuracy. We choose the Bulk Crystal Formation Energy and the Alloy Surface Adsorption Energy datasets for this purpose. The results are presented in Table 4. Overall, skip connection has a larger effect on the results than DGN (13.42% and 14.13% improvement compared with 24.30% and 31.48% improvement for formation energy and adsorption energy prediction, respectively). But using both DGN and skip connection with the shallow GATGNN, we get much better accuracy than using individual components. The accuracy on the bulk crystal dataset and the alloy surface dataset increased up to 26.67% and 37.12%, respectively. Finally, we increased the number of GC layers along with the previous modification. The prediction accuracy increases up to 34.97% and 49.32% for the formation energy and adsorption energy prediction problems, respectively. The over-smoothing problem that resulted from going deeper in terms of GC layers is now greatly alleviated by the use of DGN and skip connection. Moreover, the accuracy is much improved when the architecture is deep, which demonstrates the necessity to use more GC layers (as it can better capture embeddings and make effective use of the higher-order neighbors' attributes).

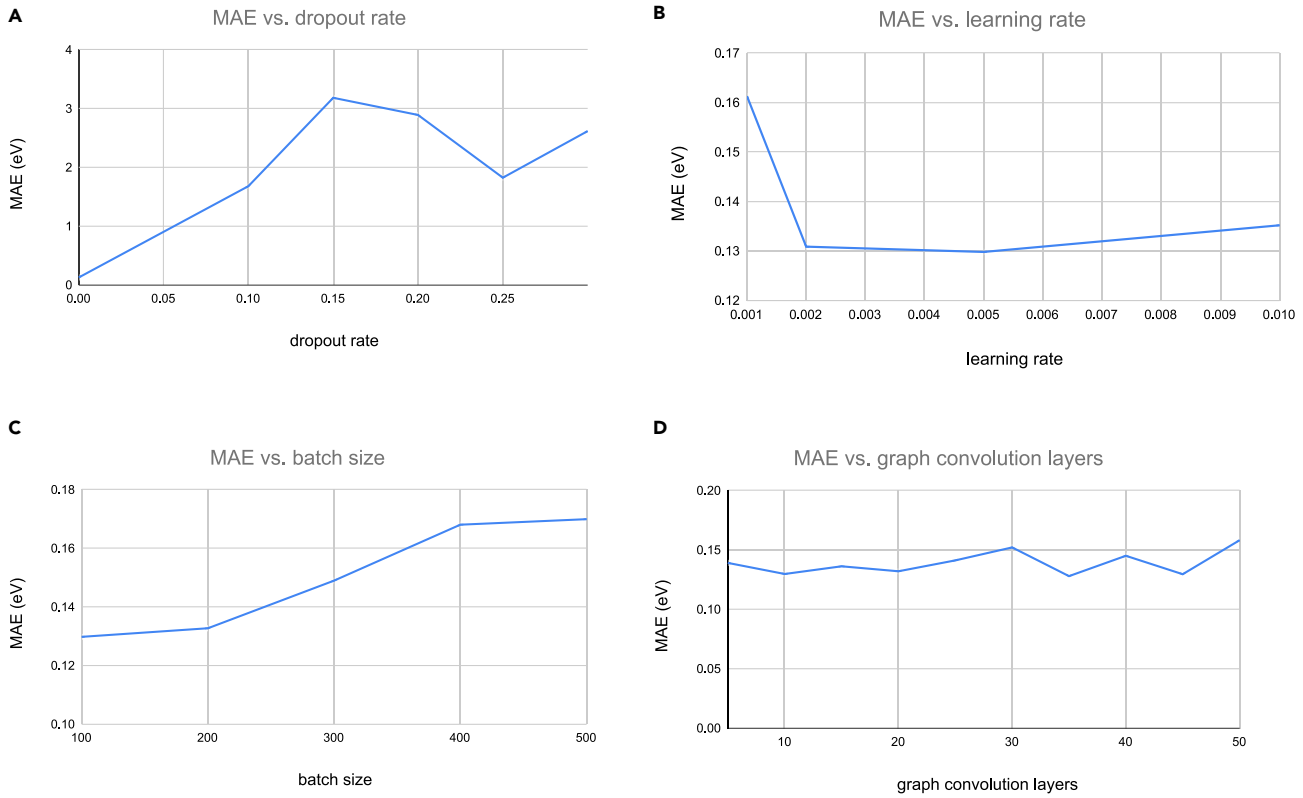


Figure 5. Parameter study of DeeperGATGNN

(A–D) (A) MAE versus dropout rate, (B) MAE versus learning rate, (C) MAE versus batch size (using 10 graph convolution layers), and (D) MAE versus graph convolution layers. All the experiments are done using 500 epochs and 5-fold cross-validation on the Pt-cluster dataset.

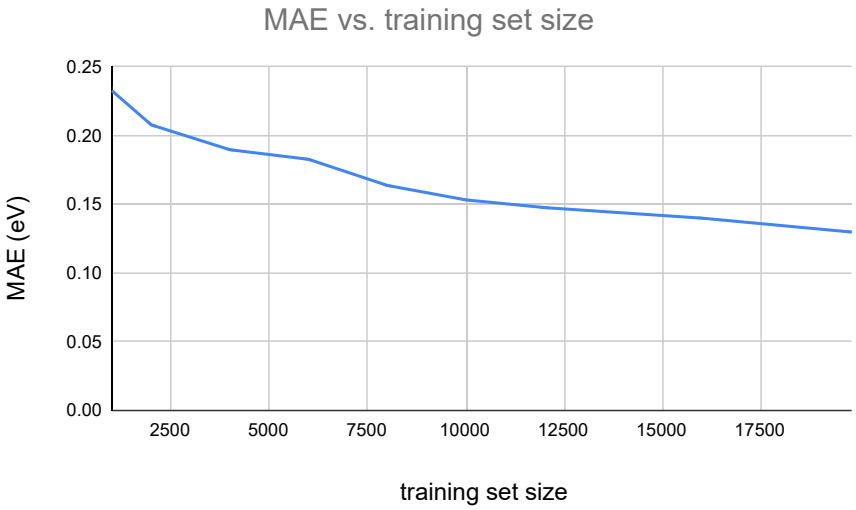


Figure 6. Effect of training set size on DeeperGATGNN's performance

Prediction performance improves with increasing number of training samples. The experiment is done using 500 epochs and 5-fold cross-validation on the Pt-cluster dataset.

Physical insights

We examined whether our DeeperGATGNN model can bring certain physical insights into the materials space. We used the t-distributed stochastic neighbor embedding (t-SNE)[43] for this purpose, which is a widely used non-linear technique for visualizing and interpreting high-dimensional data. The objective of t-SNE is to map higher-dimensional data points to a very low-dimensional one (usually 2D or 3D) so that the pairwise distances between data points are well preserved after mapping.[43,44] So, closer points in the higher dimension tend to remain close after mapping to the lower dimension.

Here, we used the Alloy Surface Adsorption Energy dataset for visualizing the distribution of the learned materials latent representation learned by our model, as well as GATGNN, CGCNN, and MEGNet models. We first trained all the models and then fed the whole dataset to the models and fetched the first layer's output after the final GC layer to generate the t-SNE plots, as shown in Figure 8. Different colors represent different alloy surface adsorption energy levels for the samples represented in the latent space. We can see from Figure 8 that different groups are formed after the two-dimensional mapping and materials (points) in the same cluster have a very high probability of having similar composition and/or structures as the clustering is based on the latent vectors mapped from the crystal structures. Each model might generate different latent spaces, but we can still get a good idea about their prediction pattern by analyzing these clusters. We found that many local areas (clusters) in all the images have been colored with similar colors implying that these compositionally or structurally similar materials tend to have similar surface energies. Compared with the distributions of CGCNN, MEGNet, and GATGNN, the high-energy alloys are more clearly separated from the lower ones. However, it is important to mention that the cluster sizes and the distance between them do not bear much significance in a t-SNE plot.[45]

DISCUSSION

GNNs are increasingly used for solving challenging problems in materials and physics.[28,46–48] Representations in GNNs are inherently rotation and translation invariant, making it ideal to model the atomic relationships. However, we find that there are several key issues in designing and training scalable GNN models.

The first pitfall is that GNN models can easily go under-trained due to the high computational training complexity (some models have several million parameters). The situation becomes even

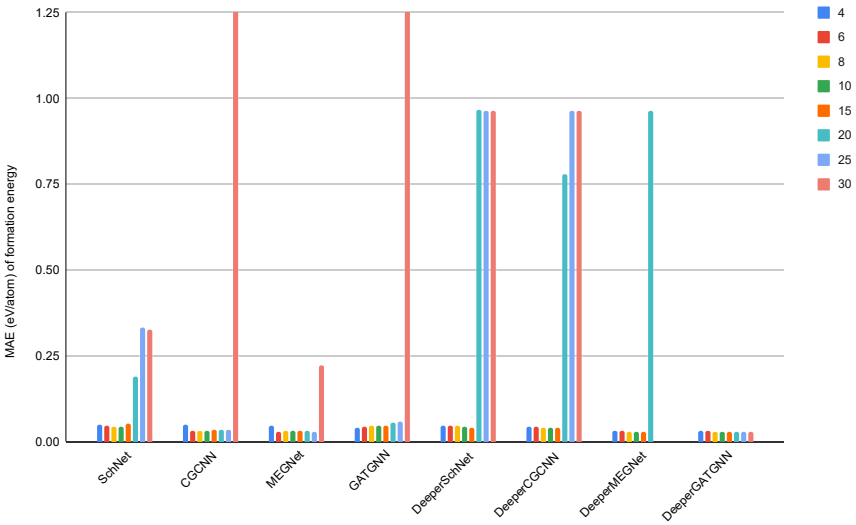


Figure 7. Scalability study of SOTA GNNs

Scalability of GNN models in terms of parameter number as regard to the network depth tested on the Bulk Materials Formation Energy dataset. Different colors represent different numbers of GC layers in the networks. All the models except DeeperGATGNN fail to scale up to at least 30 GC layers.

Table 4. Results (MAE) of ablation experiments of DeeperGATGNN to perceive the impact of each component

Architecture	Bulk materials (Eform)	GCL	% improve	Alloy surfaces (Adsorption energy)	GCL	% improve
GATGNN	0.04544	5	–	0.08063	5	–
GATGNN + DGN	0.03934	5	13.42	0.06924	5	14.13
GATGNN + SC	0.0344	5	24.30	0.05525	5	31.48
GATGNN + DGN + SC (shallow)	0.03332	5	26.67	0.0507	5	37.12
GATGNN + DGN + SC (deep) = DeeperGATGNN	**0.02955**	15	**34.97**	**0.04086**	20	**49.32**

The experiments are conducted on the Bulk Materials Formation Energy and Alloy Surface Adsorption Energy datasets. GCL and SC denotes the numxber of GC layers and skip connection, respectively.

worse when one has to do large-scale hyperparameter tuning. For example, in the benchmark study of GNNs,[1] the hyperparameters include three encoding dimensions, GC layer number, fully connected layer number, pooling methods, the learning rate, and the batch size. The study found that existing GNNs tend to achieve optimal performance for different datasets using different hyperparameter sets. This expensive hyperparameter search process forced the authors to use only 200 epochs for evaluation. However, our analysis in Figure 2 shows that their networks are all under-trained, which will not stagnate until 500 epochs. This has led to their severe under-estimation of the GNN performances for all the results they reported. For example, for the Alloy Surface dataset, CGCNN's MAE with 250 epochs of training is 0.06 eV, which is 40% larger than the result (0.042 eV) when trained with 500 epochs. Since running more epochs with huge hyperparameter space is infeasible, it is more desirable to use GNN models that can achieve more stable results with default or minor parameter tuning. For example, our DeeperGATGNN models achieved SOTA results across five datasets using the same architecture except with a varying number of GC layers.

The second pitfall for GNNs is that they usually suffer from the over-smoothing issue, which leads to their performance degradation when too many layers are used.[49] This is clearly shown in our scalability study and Figure 7. All existing GNN models, except our DeeperGATGNN, have significant performance degradation when 30 GC layers are used. It is interesting to find that, while DGN and skip connections have effectively helped our DeeperGATGNN to address this issue, the same strategy does not work equally well for SchNet, CGCNN, and MEGNet, even though it does help to improve their performance too.

Another limitation to our model performance is the scarcity of data or the information input to our models. For example, in the 2D Materials dataset, which has only 3,814 samples, MEGNet achieved the best result with 4 GC layers. Our DeeperGATGNN with 25 GC layers achieved the SOTA result on this dataset. In this case, it seems that to further improve the performance, additional information, such as the angular information of the materials structures, is needed. Also other methods, such as pre-activated skip connections[50] or DenseNet[31] as a skip connection method, can also be applied.

In conclusion, we have shown in this work that existing GNNs for materials property prediction have so far all suffered from the over-smoothing issue and cannot scale up to very deep networks without significant performance degradation. Our proposed DeeperGATGNN achieved SOTA prediction results with up to 10% performance improvement over the previous best results for five out of the six diverse benchmark datasets. This is all achieved with a single neural architecture and hyperparameter (except for the GC layer). This makes it much simpler in practical materials property prediction without the need for expensive hyperparameter tuning. Our model can also scale up to more than 30 GC layers, while all other models show dramatically degraded prediction performance. Our deeper GNN enabling strategies, such as skip connections and DGN, have shown to be able to also improve other GNNs' performances (e.g., MEGNet, SchNet, and CGCNN), but only on special datasets while their scalability remains an issue.

EXPERIMENTAL PROCEDURES

Resource availability

Lead contact

Further information and requests for resources should be directed to and will be fulfilled by the lead contact, Jianjun Hu (jianjunh@cse.sc.edu).

Materials availability

This study did not generate unique materials.

Data and code availability

The data that support the findings of this study are openly downloadable as stated in Fung et al.[1] The Bulk Materials Band Gap dataset is downloaded from Materials Project database at https://www.materialsproject.org. The source code of our work is freely accessible at http://github.com/usccolumbia/deeperGATGNN. The DOI for our code is https://doi.org/10.5281/zenodo.6336185.

DGN

One of the major issues in training a deep GNN architecture is the over-smoothing problem, in which the representation vectors of all nodes of a graph become indistinguishable as the number of GC layers increases.[49,51–53] This problem restricts GNNs to a very few layers for better performance.[35,54,55] For example, both GAT[35] and GCN[54] perform best when the number of layers is limited to only two and thus they fail to utilize higher-order neighbors' features.[56] Many approaches have already been adopted to improve upon the problem.[57–61] Traditional normalization techniques used to reduce this problem include batch normalization[62] or measures based on node pair distances, such as pair normalization.[60] But these techniques do not take account of the global graph structure that results in sub-optimal performance when the number of GC layers of the GNN is large.[61] Recently, Zhou et al.[61] addressed this issue by proposing the DGN. The main procedure of DGN is to first use a cluster assignment matrix to cluster the nodes of a graph to different clusters and then normalize each cluster separately. This will make the nodes' representations within the same community/class to be similar while those of different classes to be separated, leading to effective control of the over-smoothing issue. More specifically, the DGN works as follows:

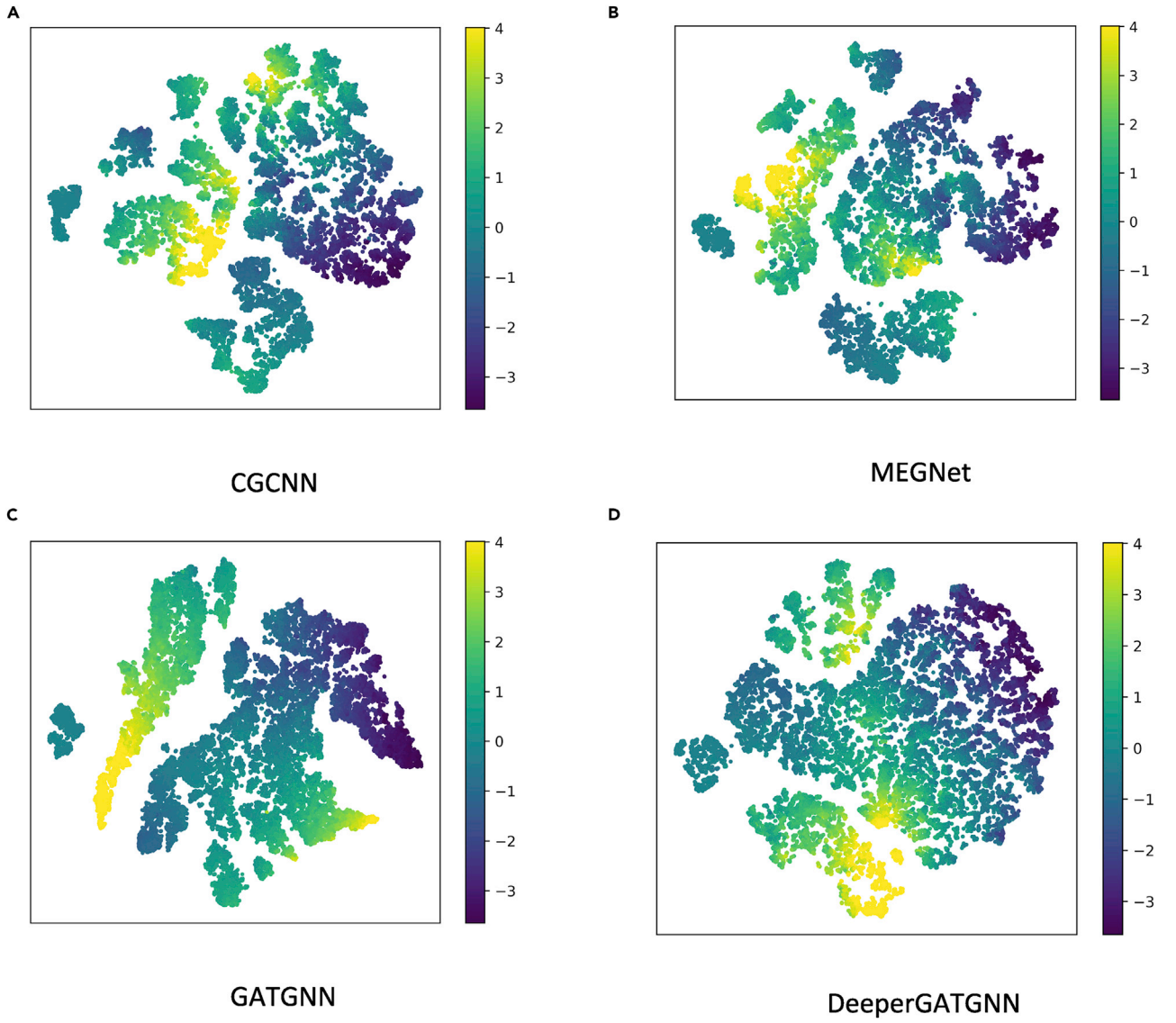


Figure 8. Alloy surface adsorption energy distribution in the latent space

(A–D) t-SNE plot of the representations after the final GC layer for (A) CGCNN, (B) MEGNet, (C) GATGNN, and (D) DeeperGATGNN trained on the Alloy Surface Adsorption Energy dataset. Different colors represent different adsorption energy levels in the latent space. Each point indicates a distinct alloy material.

Let, n be the number of nodes, G be the number of clusters specified. Let, $H^{(l)} \in \mathbb{R}^{n \times d^{(l)}}$ be the embedding matrix derived after the l-th layer of a GNN where $d(l)$ is the embedding dimension of layer l. Then the cluster assignment matrix $S^{(l)}$ can be calculated using the following equation where $U^{(l)} \in \mathbb{R}^{d^{(l)} \times G}$ is a trainable parameter:

$$S^{(l)} = softmax\left(H^{(l)}U^{(l)}\right). \quad \text{(Equation 3)}$$

The cluster assignment matrix $S^{(l)} \in \mathbb{R}^{n \times G}$ stores the probabilities of each node of the graph being assigned to each cluster. It then places the nodes into different groups using the following equation:

$$H_i^{(l)} = S^{(l)}[:, i] \circ H^{(l)}, \quad \text{(Equation 4)}$$

where $H_i^{(l)}$ denotes the embedding matrix for cluster i and $\circ$ is the row-wise multiplication operator in Equation 4. Each cluster is then normalized separately using the following equation:

$$\tilde{H}_i^{(l)} = \gamma_i\left(\frac{H_i^{(l)} - \mu_i}{\sigma_i}\right) + \beta_i, \quad \text{(Equation 5)}$$

where μ_i and σ_i mean the mean and standard deviation of each group i in Equation 5, and γ_i and β_i denote two trainable parameters.

Finally, the final embedding matrix can be calculated using the following equation:

$$\tilde{H}^{(l)} = H^{(l)} + \lambda \sum_{i=1}^{G} \tilde{H}_i^{(l)} \in \mathbb{R}^{n \times d^{(l)}}, \quad \text{(Equation 6)}$$

where λ denotes a balancing factor and $\tilde{H}^{(l)} = [\tilde{H}_1^{(l)}, \tilde{H}_2^{(l)}, \ldots, \tilde{H}_G^{(l)}]$ denotes the final embedding matrix in Equation 6.

The two main reasons why DGN is so successful in preventing the over-smoothing issue are: (1) that each group is normalized separately using Equation 5, so that each group will have a different mean and standard deviation

and thus the probability of the representation vectors of nodes of different groups being similar will decrease and (2) that input embedding is preserved in Equation 6 to prevent over-normalization.

Skip connections for enabling scalable GNNs

One of the major enabling techniques in DL for training very deep networks is the residual skip connection, which was first introduced in the ResNet framework.[30] It has allowed training networks with even more than 1,200 layers.[63] The key idea of residual skip connection is to learn to achieve identity mapping where the input x is added to the output of stacked layers $F(x)$. So, we have $H(x) = F(x) + x$. Instead of learning the underlying mapping $H(x)$ function, the stacked layers are used to learn the residual mapping $F(x) = H(x) - x$. The major benefit is that, if the identity mapping is already optimal and the stacked layers cannot learn more salient information, the training procedure can push the residual mapping to zero to avoid the degradation problem. Residual connections have been introduced into GNNs for training deeper networks.[50,64,65] In this work, we implement the layer-wise residual skip connections in the GNN models' deeper versions, which is similar to that of IRNet.[33] The difference in the skip connection method between ResNet and our implementation is shown in Figure S3.

Evaluation criterion

To evaluate the models' performances, we use MAE, which is a standard evaluation criterion for materials property prediction problems that is also used as the primary evaluation criterion in the benchmark study of Fung et al.[1] We use both 5-fold cross-validation and hold-out tests for performance evaluations depending on specific experiments. The baseline models' parameters are specified in the supplemental information of Fung et al.[1] Our DeeperGATGNN model's hyperparameters are also provided in Note S2.

SUPPLEMENTAL INFORMATION

Supplemental information can be found online at https://doi.org/10.1016/j.patter.2022.100491.

ACKNOWLEDGMENTS

Research reported in this work was supported in part by NSF under grants 1940099 and 1905775. The views, perspective, and content do not necessarily represent the official views of NSF. We thank Daniel Varivoda and Xeerak Agha for proofreading the paper.

AUTHOR CONTRIBUTIONS

Conceptualization, J.H.; methodology, J.H., S.S.O., and S.-Y.L.; software, S.S.O. and S.-Y.L.; validation, S.S.O. and J.H.; investigation, J.H., S.S.O., S.-Y.L., N.F., L.W., S.D., R.D., and Q.L.; resources, J.H.; data curation, J.H. and S.S.O.; writing – original draft, J.H., S.S.O., S.-Y.L., and L.W.; writing – review & editing, J.H., S.S.O., N.F., S.D., R.D., and Q.L.; visualization, J.H., S.S.O., and S.-Y.L.; supervision, J.H.; funding acquisition, J.H.

DECLARATION OF INTERESTS

The authors declare no competing interests.

Received: November 27, 2021
Revised: February 22, 2022
Accepted: March 18, 2022
Published: April 27, 2022

REFERENCES

1. Fung, V., Zhang, J., Juarez, E., and Sumpter, B.G. (2021). Benchmarking graph neural networks for materials chemistry. npj Comput. Mater. *7*, 1–8.
2. Bergerhoff, G., Hundt, R., Sievers, R., and Brown, I. (1983). The inorganic crystal structure data base. J. Chem. Inf. Comput. Sci. *23*, 66–69.
3. Oganov, A.R., Pickard, C.J., Zhu, Q., and Needs, R.J. (2019). Structure prediction drives materials discovery. Nat. Rev. Mater. *4*, 331–348.
4. Kim, S., Noh, J., Gu, G.H., Aspuru-Guzik, A., and Jung, Y. (2020). Generative adversarial networks for crystal structure prediction. ACS Cent. Sci. *6*, 1412–1420.
5. Zhang, Z., Li, M., Flores, K., and Mishra, R. (2020). Machine learning formation enthalpies of intermetallics. J. Appl. Phys. *128*, 105103.
6. Dinic, F., Singh, K., Dong, T., Rezazadeh, M., Wang, Z., Khosrozadeh, A., Yuan, T., and Voznyy, O. (2021). Applied machine learning for developing next-generation functional materials. Adv. Funct. Mater. *31*, 2104195.
7. Noh, J., Gu, G.H., Kim, S., and Jung, Y. (2020). Machine-enabled inverse design of inorganic solid materials: promises and challenges. Chem. Sci. *11*, 4871–4881.
8. Dan, Y., Zhao, Y., Li, X., Li, S., Hu, M., and Hu, J. (2020). Generative adversarial networks (gan) based efficient sampling of chemical composition space for inverse design of inorganic materials. npj Comput. Mater. *6*, 1–7.
9. Zhao, Y., Al-Fahdi, M., Hu, M., Siriwardane, E.M., Song, Y., Nasiri, A., and Hu, J. (2021). High-throughput discovery of novel cubic crystal materials using deep generative neural networks. Adv. Sci. *8*, 2100566.
10. Chen, C., Zuo, Y., Ye, W., Li, X., Deng, Z., and Ong, S.P. (2020). A critical review of machine learning of energy materials. Adv. Energy Mater. *10*, 1903242.
11. Goodall, R.E., and Lee, A.A. (2020). Predicting materials properties without crystal structure: deep representation learning from stoichiometry. Nat. Commun. *11*, 1–9.
12. Wang, A.Y.-T., Kauwe, S.K., Murdock, R.J., and Sparks, T.D. (2021). Compositionally restricted attention-based network for materials property predictions. npj Comput. Mater. *7*, 1–10.
13. Xie, T., and Grossman, J.C. (2018). Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties. Phys. Rev. Lett. *120*, 145301.
14. Chen, C., Ye, W., Zuo, Y., Zheng, C., and Ong, S.P. (2019). Graph networks as a universal machine learning framework for molecules and crystals. Chem. Mater. *31*, 3564–3572.
15. Dunn, A., Wang, Q., Ganose, A., Dopp, D., and Jain, A. (2020). Benchmarking materials property prediction methods: the matbench test set and automatminer reference algorithm. npj Comput. Mater. *6*, 1–10.
16. Jain, A., et al. (2013). Commentary: the materials project: a materials genome approach to accelerating materials innovation. APL Mater. *1*, 011002.
17. Nouira, A., Sokolovska, N., and Crivello, J.-C. (2018). Crystalgan: learning to discover crystallographic structures with generative adversarial networks. Preprint at arxiv. https://arxiv.org/abs/1810.11203.
18. Li, S., Liu, Y., Chen, D., Jiang, Y., Nie, Z., and Pan, F. (2022). Encoding the atomic structure for machine learning in materials science. Wiley Interdiscip. Rev. Comput. Mol. Sci. *12*, e1558.
19. Rupp, M., Tkatchenko, A., Müller, K.-R., and Von Lilienfeld, O.A. (2012). Fast and accurate modeling of molecular atomization energies with machine learning. Phys. Rev. Lett. *108*, 058301.
20. Zhao, Y., Yuan, K., Liu, Y., Louis, S.-Y., Hu, M., and Hu, J. (2020). Predicting elastic properties of materials from electronic charge density using 3d deep convolutional neural networks. J. Phys. Chem. C *124*, 17262–17273.
21. Faber, F., Lindmaa, A., von Lilienfeld, O.A., and Armiento, R. (2015). Crystal structure representations for machine learning models of formation energies. Int. J. Quan. Chem. *115*, 1094–1101.
22. Faber, F.A., Lindmaa, A., Von Lilienfeld, O.A., and Armiento, R. (2016). Machine learning energies of 2 million elpasolite (a b c 2 d 6) crystals. Phys. Rev. Lett. *117*, 135502.
23. Ward, L., Liu, R., Krishna, A., Hegde, V.I., Agrawal, A., Choudhary, A., and Wolverton, C. (2017). Including crystal structure attributes in machine learning models of formation energies via voronoi tessellations. Phys. Rev. B *96*, 024104.

24. Schütt, K.T., Sauceda, H.E., Kindermans, P.-J., Tkatchenko, A., and Müller, K.-R. (2018). Schnet–a deep learning architecture for molecules and materials. J. Chem. Phys. *148*, 241722.

25. Sendek, A.D., Yang, Q., Cubuk, E.D., Duerloo, K.-A.N., Cui, Y., and Reed, E.J. (2017). Holistic computational structure screening of more than 12000 candidates for solid lithium-ion conductor materials. Energy Environ. Sci. *10*, 306–320.

26. Rosen, A.S., Iyer, S.M., Ray, D., Yao, Z., Aspuru-Guzik, A., Gagliardi, L., Notestein, J.M., and Snurr, R.Q. (2021). Machine learning the quantum-chemical properties of metal–organic frameworks for accelerated materials discovery. Matter *4*, 1578–1597.

27. Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., and Dahl, G.E. (2017). Neural message passing for quantum chemistry. Int. Conf. Mach. Learn. *70*, 1263–1272.

28. Park, C.W., and Wolverton, C. (2020). Developing an improved crystal graph convolutional neural network framework for accelerated materials discovery. Phys. Rev. Mater. *4*, 063801.

29. Louis, S.-Y., Zhao, Y., Nasiri, A., Wang, X., Song, Y., Liu, F., and Hu, J. (2020). Graph convolutional neural networks with global attention for improved materials property prediction. Phys. Chem. Chem. Phys. *22*, 18141–18148.

30. He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (IEEE), pp. 770–778.

31. Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K.Q. (2017). Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (IEEE), pp. 4700–4708.

32. Brown, T., et al. (2020). Language models are few-shot learners. Adv. Neural Inf. Process. Syst. *33*, 1877–1901.

33. Jha, D., Gupta, V., Ward, L., Yang, Z., Wolverton, C., Foster, I., Liao, W.-k., Choudhary, A., and Agrawal, A. (2021). Enabling deeper learning on big data for materials informatics applications. Sci. Rep. *11*, 1–12.

34. Yang, Z., Papanikolaou, S., Reid, A.C., Liao, W.-k., Choudhary, A.N., Campbell, C., and Agrawal, A. (2020). Learning to predict crystal plasticity at the nanoscale: deep residual networks and size effects in uniaxial compression discrete dislocation simulations. Sci. Rep. *10*, 1–14.

35. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., and Bengio, Y. (2018). Graph attention networks. In International Conference on Learning Representations.

36. Wang, X., Ji, H., Shi, C., Wang, B., Ye, Y., Cui, P., and Yu, P.S. (2019). Heterogeneous graph attention network. In The World Wide Web Conference, pp. 2022–2032.

37. Liu, Z., Chen, C., Li, L., Zhou, J., Li, X., Song, L., and Qi, Y. (2019). Geniepath: graph neural networks with adaptive receptive paths. Proc. AAAI Conf. Artif. Intelligence *33*, 4424–4431.

38. Mamun, O., Winther, K.T., Boes, J.R., and Bligaard, T. (2019). High-throughput calculations of catalytic properties of bimetallic alloy surfaces. Sci. Data *6*, 1–9.

39. Fung, V., and Jiang, D.-e. (2017). Exploring structural diversity and fluxionality of pt n (n= 10–13) clusters from first-principles. J. Phys. Chem. C *121*, 10796–10802.

40. Haastrup, S., Strange, M., Pandey, M., Deilmann, T., Schmidt, P.S., Hinsche, N.F., Gjerding, M.N., Torelli, D., Larsen, P.M., Riis-Jensen, A.C., and Gath, J. (2018). The computational 2d materials database: high-throughput modeling and discovery of atomically thin crystals. 2D Mater. *5*, 042002.

41. LeCun, Y.A., Bottou, L., Orr, G.B., and Müller, K.-R. (2012). Efficient Backprop. Neural Networks: Tricks of the Trade (Springer), pp. 9–48.

42. Keskar, N.S., Mudigere, D., Nocedal, J., Smelyanskiy, M., and Tang, P.T.P. (2017). On large-batch training for deep learning: Generalization gap and sharp minima. In International Conference on Learning Representations.

43. Van der Maaten, L., and Hinton, G. (2008). Visualizing data using t-sne. J. Mach. Learn. Res. *9*, 2579–2605.

44. Li, W., Cerise, J.E., Yang, Y., and Han, H. (2017). Application of t-sne to human genetic data. J. Bioinform. Comput. Biol. *15*, 1750017.

45. Wattenberg, M., Viégas, F., and Johnson, I. (2016). How to use t-sne effectively. Distill *1*, e2.

46. Shlomi, J., Battaglia, P., and Vlimant, J.-R. (2020). Graph neural networks in particle physics. Mach. Learn. Sci. Technol. *2*, 021001.

47. Sanchez-Gonzalez, A., Godwin, J., Pfaff, T., Ying, R., Leskovec, J., and Battaglia, P. (2020). Learning to simulate complex physics with graph networks. In International Conference of Machine Learning, pp. 8459–8468.

48. Park, C.W., Kornbluth, M., Vandermause, J., Wolverton, C., Kozinsky, B., and Mailoa, J.P. (2021). Accurate and scalable graph neural network force field and molecular dynamics with direct force architecture. npj Comput. Mater. *7*, 1–9.

49. Li, Q., Han, Z., and Wu, X.-M. (2018). Deeper insights into graph convolutional networks for semi-supervised learning. In Thirty-Second AAAI Conference on Artificial Intelligence.

50. Li, G., Xiong, C., Thabet, A., and Ghanem, B. (2020). Deepergcn: all you need to train deeper gcns. Preprint at axriv. https://arxiv.org/abs/2006.07739.

51. Oono, K., and Suzuki, T. (2020). Graph neural networks exponentially lose expressive power for node classification. In International Conference on Learning Representations.

52. Chen, D., Lin, Y., Li, W., Li, P., Zhou, J., and Sun, X. (2020). Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. Proc. AAAI Conf. Artif. Intell. *34*, 3438–3445.

53. Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., and Sun, M. (2020). Graph neural networks: a review of methods and applications. AI Open *1*, 57–81.

54. Kipf, T.N., and Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In International Conference on Learning Representations.

55. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Philip, S.Y. (2020). A comprehensive survey on graph neural networks. IEEE Trans. Neural Networks Learn. Syst. *32*, 4–24.

56. Chen, M., Wei, Z., Huang, Z., Ding, B., and Li, Y. (2020). Simple and deep graph convolutional networks. In International Conference on Machine Learning, pp. 1725–1735.

57. Rong, Y., Huang, W., Xu, T., and Huang, J. (2020). Dropedge: towards deep graph convolutional networks on node classification. In International Conference on Learning Representations.

58. Xu, K., Li, C., Tian, Y., Sonobe, T., Kawarabayashi, K.-i., and Jegelka, S. (2018). Representation learning on graphs with jumping knowledge networks. In International Conference on Machine Learning, pp. 5453–5462.

59. Li, G., Muller, M., Thabet, A., and Ghanem, B. (2019). Deepgcns: can gcns go as deep as cnns? In Proceedings of the IEEE/CVF International Conference on Computer Vision (IEEE), pp. 9267–9276.

60. Zhao, L., and Akoglu, L. (2020). Pairnorm: Tackling oversmoothing in gnns. In International Conference on Learning Representations.

61. Zhou, K., Huang, X., Li, Y., Zha, D., Chen, R., and Hu, X. (2020). Towards deeper graph neural networks with differentiable group normalization. Adv. Neural Inf. Process. Syst. *33*, 4917–4928.

62. Ioffe, S., and Szegedy, C. (2015). Batch normalization: accelerating deep network training by reducing internal covariate shift. In International Conference on Machine Learning, pp. 448–456.

63. Huang, G., Sun, Y., Liu, Z., Sedra, D., and Weinberger, K.Q. (2016). Deep networks with stochastic depth. In European Conference on Computer Vision (Springer), pp. 646–661.

64. Li, G., Müller, M., Ghanem, B., and Koltun, V. (2021). Training graph neural networks with 1000 layers. In International Conference on Machine Learning, pp. 6437–6449.

65. Xu, K., Zhang, M., Jegelka, S., and Kawaguchi, K. (2021). Optimization of graph neural networks: implicit acceleration by skip connections and more depth. In International Conference on Machine Learning, pp. 11592–11602.