Data Engineering and Applications

Research Paper

Application of Parallelized Graph Neural Networks in Predicting Molecular Properties from Large-Scale Chemical Databases

Piotr Tomaszewski[1,*] and Adam Wojcik[2]

[1] Faculty of Electrical and Computer Engineering, Wrocaw University of Science and Technology, Wrocaw 50-370, Poland
[2] Faculty of Electrical Engineering, Automatics, Computer Science and Biomedical Engineering, University of Silesia in Katowice, Katowice 40-007, Poland
*Corresponding author: piotr.tomaszewski@pwr.edu.pl

Abstract. A parallel GNN architecture is proposed to solve the problems of large computation and poor scalability in molecular property prediction of large-scale chemical databases. This study uses data-level and model-level parallel methods, distributed training and memory optimization methods to learn graph-structured molecular data. To make GNN training as efficient as possible on heterogeneous datasets on multi-node GPU clusters, we optimize graph batching, adaptive sampling, workload balancing, and communication-efficient synchronization. According to the results of the standard chemical benchmark experiment, the parallelized GNN obtained a great acceleration ratio, a small peak memory usage, and still had a high prediction accuracy compared with the basic method. The comprehensive analysis analyzes how using better partitioning and sampling techniques affects how many conversations and what is needed. In computational chemistry and materials science, these distributed and parallel GNN methods seem to have good large-scale molecular prediction capabilities.

Keywords: *Parallel Computing, Graph Neural Networks, Molecular Property Prediction, Chemical Databases*

Received on 19 July 2024, Accepted on 23 November 2024, Published on 13 Jan2025

Introduction

The rapid growth of chemical databases, a result of high-throughput computation and high-throughput experiments, is changing the way molecules are found and materials are made. Accurately predicting molecular properties is now important in drugs, materials and pollution [1]. However, traditional ways of thinking, such as using mathematical connectivity models to understand how different components in a drug work (quantitative structure-activity relationship) or using simple computer learning concepts to understand how various things interact with each other in nature, are still unable to understand complex interactions. To become good learners of large and convoluted molecular data, increasingly large molecular datasets will require more powerful and spacious computational settings algorithms [2].

Graph neural networks (GNNs) have emerged as an effective tool for predicting molecular properties [4] due to their natural fit with the graph form of molecules. GNNs can learn node and edge representations to capture local and global structural information through message passing architectures [5]. Despite these benefits, using GNNs on large chemical databases poses significant computational challenges. Deeper networks with more expressive power are needed, and all of these, and more, due to complexity, will lead to memory problems and longer times to process data [6]. In practice, these factors limit the scalability of GNNs and prevent them from being rapidly applied in commercial and academic settings where a large number of predictions are required [7,8].

DOI: 10.64972/dea.2025.v3i1.68

This paper details the approach to predicting molecular properties from large chemical databases using a parallel GNN architecture. To improve the computational speed and scalability, the state-of-the-art parallelization techniques such as data-level and model-level parallelism, distributed training and memory optimization are studied and implemented. Comprehensive experiments are conducted on some standard chemical datasets. The results show that the parallelized GNN can be both much faster and more accurate in prediction. Significant progress has been made in the field of molecular informatics, with the proposal of large-scale chemical analysis as a scalable method for the practical application of GNNs.

Related Work

Graph Neural Networks for Molecular Data

Since molecular data is a kind of chemical data with natural graph structure, graph neural network (GNN) is very suitable for processing molecular data. GNNs allow molecules to be directly treated as graphs, where atoms and bonds are represented as nodes and edges, respectively. This allows molecules to be directly processed as graphs, unlike traditional machine learning methods based on hand-crafted molecular descriptors. Due to this structural alignment, GNNs can extract local and global features through an iterative process of message passing and node aggregation [9]. As a result, GNNs have outperformed traditional descriptor-driven models in predicting molecular properties such as solubility, toxicity, and bioactivity. [10].

In recent years, many different molecular learning GNN architectures have emerged. Different MPNN variants, GAT variants, and GIN variants capture different amounts of chemical information. Because these models recursively collect data from nearby nodes and edges, they are well-suited to describing important chemical environments that are difficult to distinguish. In the field of computational chemistry and drug discovery pipelines, the ability of GNNs to model complex molecular interactions without explicit feature engineering is at the forefront.

Challenges in Large-Scale Molecular Property Prediction

Applying GNNs to large-scale molecular datasets is not without algorithmic and computational challenges. Public chemical repositories such as PubChem, ChEMBL, and ZINC store millions to hundreds of millions of distinct chemical substances. These chemicals vary in molecular size, topology, and chemical functional groups [11]. This heterogeneity poses a barrier to computational load. In addition, it is challenging to create a model that is widely used in different molecular scaffolds. The dynamic and unstable structure of molecular graphs also requires personalized data loading and batch processing, as well as memory management methods to ensure efficient training and reasoning [12].

Then, when more complex models are used, the problems become more serious, such as whether scaling up is effective. Deeper GNN architectures can become over-smoothed, which causes node representations to become indistinguishable from each other and leads to vanishing gradients. Table 1 shows that the scale and diversity of modern chemistry datasets make the situation worse, as molecules of various sizes and structures need to be considered. To address this, better memory optimization and efficient dynamic mini-batching and graph sampling are needed. The high throughput requirements of modern drug discovery and materials search require inference pipelines to be not only accurate but also able to process millions of molecules in minutes [13]. To achieve this, algorithms and systems must be continuously improved.

Table 1. Statistics of Large-Scale Chemical Datasets

Dataset	Number of Molecules	Median Nodes per Molecule	Median Edges per Molecule	Main Property Types
PubChem	>110 million	24	26	Bioactivity; Physicochemical
ChEMBL	>2 million	29	32	Drug-likeness; Activity
ZINC	>230 million	23	25	Lead-likeness; Reactivity
QM9	134,000	9	8	Quantum properties
OGB-MolPCBA	437,929	26	28	Bioactivity (128 tasks)

Predicting molecular properties presents a variety of opportunities and computational hurdles in today's era of diverse and large chemical datasets. Due to the diversity in size, structure, and property types of molecules, structural types, and property types, we need to select machines with acceptable data bias of different

DOI: 10.64972/dea.2025.v3i1.68

distributions. As chemical databases increase, models need to learn and generalize across a wide chemical space. Therefore, the importance of graph neural network architecture size is evident, being able to handle both large and small molecules well. In addition, having methods to quickly process large amounts of data is critical to determining the correct property outcomes and accelerating discovery times, such as in drug manufacturing. Current advances in data preparation, feature creation and graph representation are important to ensure that computer models become larger and more diverse in the types of chemical information they can handle.

Parallel and Distributed Machine Learning in Chemistry

Scaling GNNs to large industrial chemical datasets requires parallel distributed machine learning approaches. Model parallelism splits the neural network into different devices or nodes, while data parallelism allows simultaneous training on multiple data shards [14]. Frameworks such as DeepSpeed and Horovod enable us to train large GNNs in GPU clusters, which helps reduce training wall-clock time and memory bottlenecks [15].These are critical for handling all the computations for predicting contemporary molecular properties.

However, parallelizing GNNs with chemicals brings new technical challenges. The inseparability of data and model updates, and the irregular and dynamic nature of data structures and data, lead to load imbalance and communication. To achieve efficient distributed training, it is necessary to consider the graph partitioning algorithm, the memory consistency protocol, and which elements the model architecture interacts with. Hardware. Research into these issues remains very active, as deep learning is essential to unlocking the full potential of large, complex molecules [16].

Parallelization Strategies and Algorithmic Optimizations

Data-Level and Model-Level Parallelism

Scaling GNN computation requires parallelizing large molecular datasets. Data-level parallelism distributes independent training samples (e.g., molecular graphs) to multiple processing units. The former approach will be effective when the sample dataset is huge and the computational load of each sample is fairly balanced. Instead, model-level parallelism assigns different layers or parts of the GNN architecture to specific devices. This is especially important for memory-intensive models that exceed the depth of a single device [17]. Figure 1 shows the conceptual difference between model-level parallelism and data-level parallelism for GNNs operating on molecular graphs.

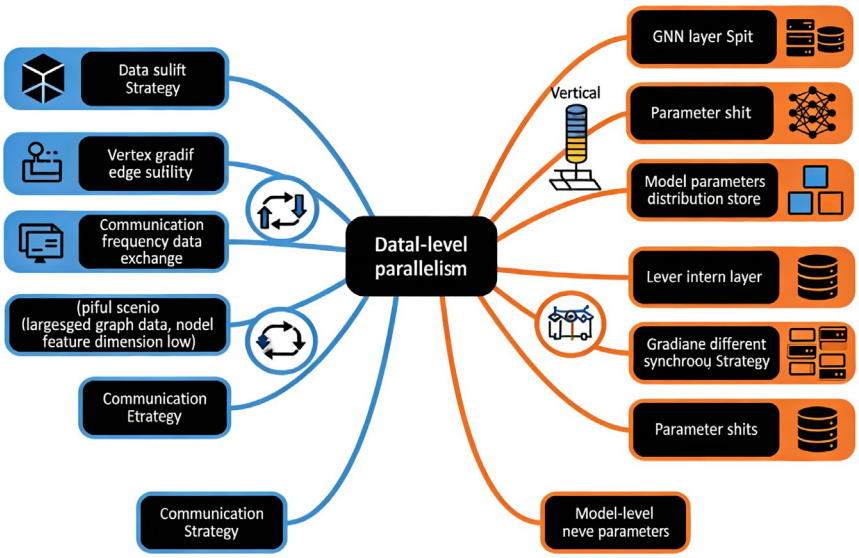

Figure 1. Comparison between data-level and model-level parallelism for GNNs.

DOI: 10.64972/dea.2025.v3i1.68

The scalability and efficiency of chemical graph neural network training are affected by the differences between data-level and model-level parallelism. Data-level parallelism is excellent when the dataset is large and relatively uniform; each compute node bears the same workload, and scaling is easy. On the contrary, model-level parallelism is necessary to train more complex and memory-intensive architectures, as single-device memory is becoming increasingly scarce. By assigning different parts of the network to devices, the latest GNN models can be created and used to complete very complex molecular prediction tasks. Most importantly, the most effective parallelization strategy often depends on the size of the dataset, the variance of the graphs, and the depth of the network. Resource and quantity transfer can also be improved by changing the mixed strategy of data and model segmentation ratio.

The following is an example of data-level parallelism. Suppose the dataset D is divided into n partitions, which are $D_1, D_2, \ldots, D_n$, and each partition is independently processed on n devices:

$$\text{For each device } i \in \{1, \ldots, n\}: \theta_i \leftarrow \text{Update}(\theta_i, D_i) \quad \text{Eq. (1)}$$

In Eq.(1), θ_i denotes the model parameters on device i. Periodic synchronization steps ensure consistency across devices. Model-level parallelism, on the other hand, maps layers $L_1, L_2, \ldots, L_k$ to different devices. The forward and backward passes are pipelined across devices, which can be represented as Eq.(2):

$$h^{(l+1)} - f^{(l)}\left(h^{(l)}\right), \text{ where } f^{(l)} \text{ executes on device } d_l \quad \text{Eq. (2)}$$

This structure can train larger GNNs while keeping the device memory limit [18]. As the complexity and expressiveness of graph neural networks increase, horizontal model parallelism becomes more and more timely. With the deepening of the research on network architecture and molecular features, the blocks of this model can be distributed on multiple devices. This allows us to push the depth and width of the model further than ever before, capturing huge, sprawling molecular structural features and long-range dependencies.

Distributed Training Frameworks

To be able to effectively train large-scale parallel GNNs, a powerful distributed training platform is needed. These frameworks help different devices communicate with each other, determine which parts of the data and model are handed over, and decide when to synchronize everything. Well-known frameworks such as DeepSpeed and Horovod also support data parallelism and hybrid methods, which can be scaled to hundreds of GPUs or even distributed clusters [19]. Figure 2 shows the detailed image of the distributed GNN training system. It shows how data, model parameters and gradients flow in the system.

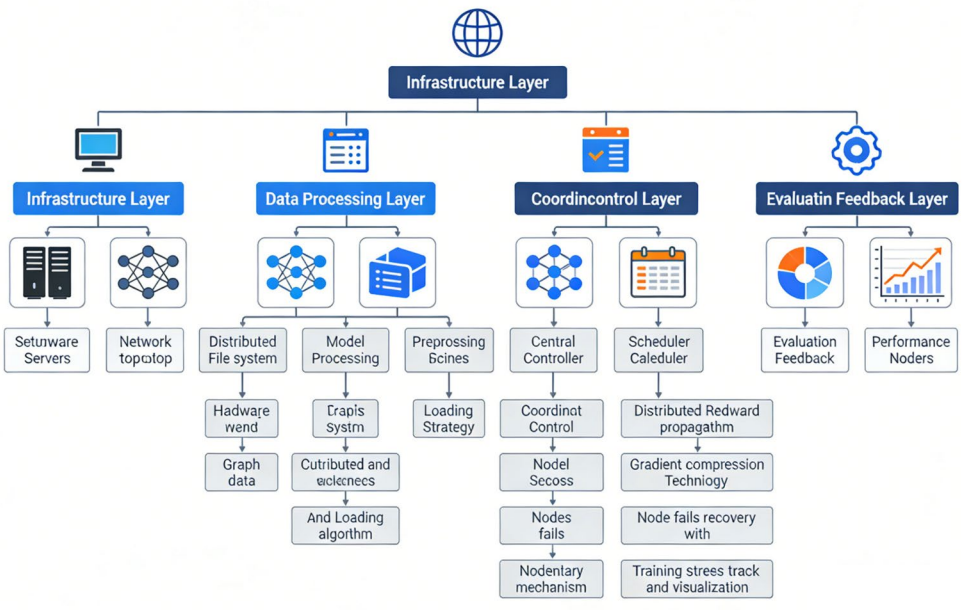

Figure 2. System architecture of distributed GNN training across multiple compute nodes.

DOI: 10.64972/dea.2025.v3i1.68

Distributed GNN training systems are built to maximize computational throughput and communication efficiency. To address the inherent scaling issues in deep learning molecular prediction, these systems coordinate data flow, parameter synchronization, and gradient updates across multiple compute nodes. Keeping everything consistent and convergent in a distributed training setup requires synchronization barriers and good communication protocols. The inter-node bottleneck problem will be greatly improved by using some more advanced optimizers such as gradient compression and asynchronous update. Modern flexible distributed frameworks can easily adapt to various hardware environments, from small clusters to large clouds. Finally, the effectiveness of distributed training depends on finding a balance between parallel computation and communication overhead. In this case, it is necessary to use a very efficient GNN model to quickly analyze large and complex chemical datasets. The mathematical form of a typical distributed training step is as follows:

$$\theta \leftarrow \theta - \eta \cdot \frac{1}{n}\sum_{i=1}^{n} \nabla_{\theta}\mathcal{L}(D_i;\theta) \quad \text{Eq. (3)}$$

In Eq.(3), η is the learning rate, $\mathcal{L}$ is the loss function, and D_i represents the data partition on device i. Synchronization barriers are inserted at each step to ensure that parameter updates are consistent across the distributed system.

In real life, the speed of distributed training is limited by the cost of passing gradients. Gradients compression and asynchronous updates strategies are often used to solve the bottleneck problem [20].

Efficient Graph Batching, Sampling, and Memory Optimization

Parallelizing GNNs is difficult for large irregular molecular graphs. Graph batching groups multiple graphs into a batch for computation, which helps vectorize operations and better GPU utilization. However, the variation in graph sizes can lead to padding and memory fragmentation. Efficient batching methods sort graphs by topology or size to reduce padding and balance workload [21]. Sampling methods such as subgraph extraction or neighbor sampling reduce the amount of computation required for each batch by using representative substructures.

$$\mathcal{N}_v^{(l)} \sim \text{SampleNeighbors}(v,s), h_v^{(l+1)} - \sigma\left(W^{(l)} \cdot \text{AGG}\left(\left\{h_u^{(l)} : u \in \mathcal{N}_v^{(l)}\right\}\right)\right) \quad \text{Eq. (4)}$$

In Eq.(4), at the l layer, the sampling neighborhood of node v is$\mathcal{N}_v^{(l)}$, the sampling size is s, the activation function is σ, and the aggregation operator is AGG[22]. Techniques such as mixed precision training, checkpointing, and efficient tensor representation help with memory optimization. The memory footprint M of a batch can be approximated as Eq.(5):

$$M - \sum_{i=1}^{B} (|V_i| \cdot d_v + |E_i| \cdot d_e) \quad \text{Eq. (5)}$$

B is the batch size, $|V_i|$ and $|E_i|$ are the numbers of nodes and edges in graph i, and d_v, d_c are the dimensions of node and edge features.

For the speed and size of GNN-based molecular property prediction systems, deciding which batching and sampling strategy to use is critical. So as long as you add a lot of MolGraphs, these GPUs will start to do more and align with each other, except for the natural differences in size and the location of the connection points. As a result, you sometimes get more memory padding, and in some mini-batches, some do more work than others. Unlike other sampling methods such as neighbor, layer, and subgraph sampling, the model is able to focus the computation on more important graph substructures. This is done to avoid memory and computation waste. However, without proper tuning, important long-range connections may be missed, increasing the sampling cost. Sampling, partitioning, and batching are trade-offs in terms of throughput and accuracy for large-scale GNN training.

Figure 3 shows the quantitative comparison of sampling efficiency and batch processing efficiency of different graph partitioning schemes. The GPU utilization and average batch memory usage are shown. Sampling redundancy by using bar graphs. The structure-aware approach was found to be worthwhile for large-scale GNN training on large and diverse molecular data when the topology-aware partitioning reduced the amount of

DOI: 10.64972/dea.2025.v3i1.68

sampling redundancy and memory usage. Optimal partitioning improves computational throughput, reduces wasted resources, and supports the application of parallel GNN architectures in large-scale cheminformatics'.

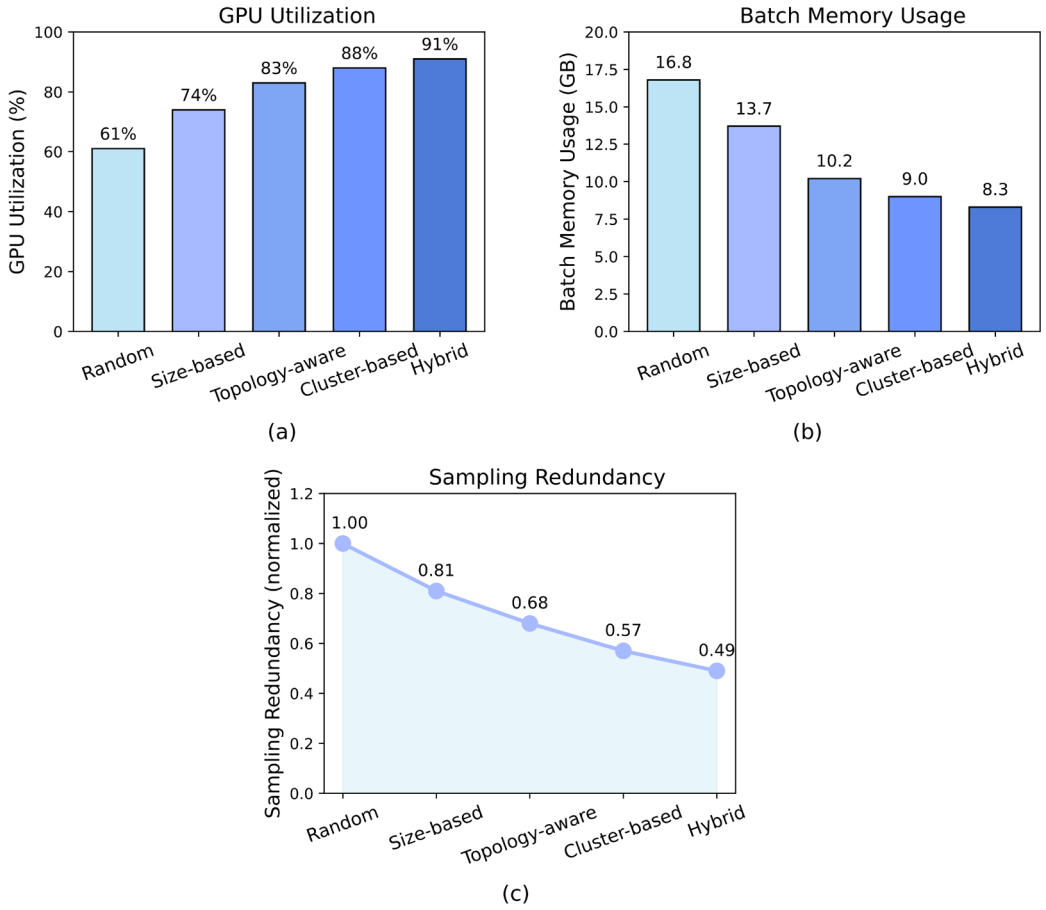

Figure 3. Comparison of batching and sampling efficiency across different graph partitioning strategies. (a) GPU utilization under five partitioning strategies, using blue gradient colors. (b) Average batch memory usage (GB), showing memory decrease with advanced partitioning. (c) Normalized sampling redundancy, where structure-aware and hybrid strategies yield the lowest redundancy.

Maximizing the performance of GNNs on large heterogeneous chemical datasets requires efficient sampling and batching. Choosing the appropriate batching strategy affects GPU memory footprint, training convergence, and utilization efficiency. For example, neighbor sampling or subgraph sampling helps the model to focus its computational power on the graph, which is considered to contain the most information. As a result, the computation converges faster and some computations are no longer necessary. However, care must be taken to account for any important long-range dependencies in the molecular graph. The graph structure can improve processing speed and model accuracy when dealing with different molecular skeletons. As the things discovered in chemistry become larger and more complex, these data processing methods need to be constantly improved to build larger and better GNN pipelines. Incorporating domain knowledge and adaptively tuning sampling parameters seem to be good ways to predict and improve efficiency.

Load Balancing and Communication Overhead

In a large-scale distributed training environment, the computation speed of graph neural networks largely depends on how we distribute all computation tasks to different parts of a set of machines that can help. Due to the size and complexity differences of molecular graphs, simple or static partitioning strategies are often not effective; some devices handle too large graphs, while others are underutilized. Therefore, a dynamic load balancing is needed to ensure that the number and amount of computation of the computation graph and each computation node can obtain an equal part from the entire workload. Advanced scheduling algorithms can monitor device usage in real time and move tasks when needed. This makes it possible to avoid bottlenecks and

DOI: 10.64972/dea.2025.v3i1.68

maximize the use of everything you have. When you have hundreds of GPUs or distributed pieces of a computer system, good load balancing helps optimize the hardware and provides more stable and predictable training dynamics. With this approach, parallel training of GNNs provides the basis for scale and speed.

To achieve the best results in parallel distributed training of GNNs, it is necessary to achieve a proper balance of load and reduce communication costs at the same time. The workload imbalance of each device will lead to the waste of computing resources and the decline of performance. The load L_i on device i can be estimated as Eq.(6):

$$L_i - \sum_{j \in D_i} \left(\left| V_j \right| \cdot c_v + \left| E_j \right| \cdot c_e \right) \quad \text{Eq. (6)}$$

where c_v and c_e are the per-node and per-edge computation costs, and D_i is the set of graphs assigned to device i. The efficiency E of parallel execution is impacted by the maximum load across devices, Eq.(7):

$$E - \frac{\sum_{i-1}^{n} L_i}{n \cdot \max_i L_i} \quad \text{Eq. (7)}$$

Communication overhead, particularly in distributed settings, is a primary source of scaling inefficiency. The total communication time T_{comm} is often modeled as, Eq.(8):

$$T_{\text{commn}} - \alpha + \beta \cdot S \quad \text{Eq. (8)}$$

α is the latency per message, β is the transfer cost per byte, and S is the size of the synchronized data [23]. Balancing these factors is essential for maintaining high throughput as system scale increases.

Experimental Design and Evaluation Datasets, Implementation, and Baselines

Datasets, Implementation, and Baselines

The gains are attributed to the algorithm rather than experimental noise by using a state-of-the-art distributed training framework and systematic hyperparameter tuning. The study lays a solid foundation for determining the scale and effect of parallel GNNs, and provides useful suggestions for more research on using computers to observe molecules. Our experiments will use benchmark chemical datasets such as PubChem, ChEMBL, ZINC, and OGB-MolPCBA. All datasets were processed with molecular graph representation and feature normalization. Horovod was used for grid search on a multi-node GPU cluster. The GNN was parallelized and then compared with Cluster-GCN, Vanilla GCN, and GraphSAGE under the same settings. Communication, RAM, AUC-ROC, speedup ratio, and RMSE are used to evaluate the performance. Table 2 summarizes the comparison results, which prove that the method in this paper is more accurate.

Table 2. Comparative performance of mainstream methods and the proposed approach across chemical datasets.

Method	RMSE	AUC-ROC	Speedup Ratio	Peak Memory (GB)	Comm. Overhead (s)
GCN	0.621	0.812	1.0	18.2	-
GraphSAGE	0.597	0.825	1.2	17.7	-
Cluster-GCN	0.580	0.834	2.8	14.1	-
Ours (Parallel)	0.561	0.846	6.5	8.9	1.3

Scalability and System Efficiency

Scalability was evaluated by measuring the speedup ratio S_r as the number of compute nodes N increases,Eq.(9):

$$S_r(N) - \frac{T_1}{T_N} \quad \text{Eq. (9)}$$

T_N represents the training time on one node, and T_1 represents the training time on another node. As shown in Figure 4, the method exhibits near-linear speedup as the number of nodes increases. However, the efficiency gradually decreases as the number of nodes increases due to the increase in overhead.

DOI: 10.64972/dea.2025.v3i1.68

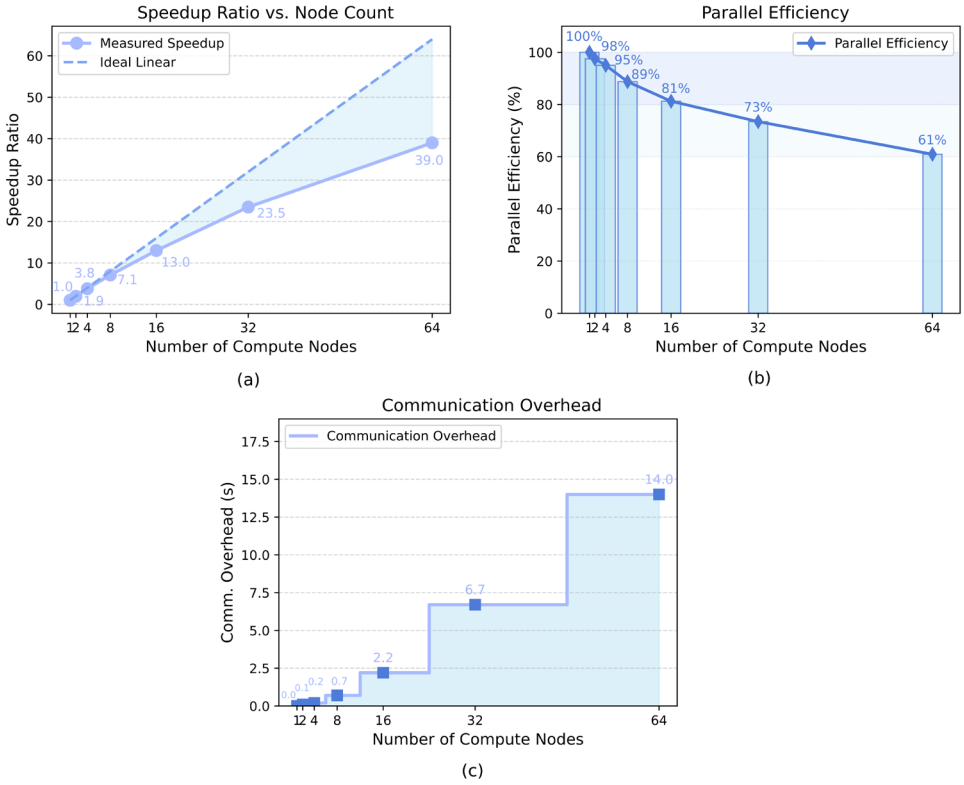

Figure 4. Speedup ratio as a function of compute node count (PNG).
(a) Speedup ratio compared to ideal linear scaling, with shaded area indicating deviation from perfect scaling.(b) Parallel efficiency, showing the percentage of ideal speedup achieved at each scale, with color bands for efficiency regions.(c) Communication overhead increases with node count, visualized as a step area plot.

Scalability remains an important part of computational chemistry workflows as more molecules are added to databases, resulting in millions of different chemicals. The ability to make good use of distributed resources to accelerate training will lead to faster cycles and more complete molecular screening. Near-linear speedup in a distributed setting means lower talk time and even workloads. Both are critical to developing stable, high-yielding models. This level of efficiency requires constant innovation as the number of compute nodes increases. These innovations include workload partitioning, resource allocation tuning, and system-level tuning. In addition to the hardware scalability issues, it also involves software architecture, algorithms, and methods for preprocessing data. In the future, parallel GNN training may focus more on adaptive expansion mechanisms, intelligent resource management, and close integration with multiple computing environments. This will make these powerful large-scale molecular models more accessible.

In addition, communication costs and system memory efficiency are improved. Memory used when calculating each maximum:

$$M - \sum_{i=1}^{B} (|V_i| \cdot d_v + |E_i| \cdot d_r) \qquad \text{Eq. (10)}$$

From Eq.(10), B is the batch size, $|V_i|$ and $|E_i|$ denote the number of nodes and edges in each graph, and d_v, d_c are the dimensions of node and edge features. Communication overhead T_{comm} was modeled as:

$$T_{\text{commin}} - \alpha + \beta \cdot S \qquad \text{Eq. (11)}$$

In Eq.(11), where S is the size of the synchronized data, α is the latency per message, and β is the transmission cost per byte. Figure 5 shows the quantitative comparison of peak memory usage and inter-node communication volume as the number of computing nodes increases. The results show that the distributed framework is more memory-efficient, because the peak memory consumption of each node decreases sharply as the number of nodes increases. However, the cost of synchronizing and transferring data becomes increasingly high due to the sharp increase in communication costs for more than 16 nodes. This indicates that a balance between memory

DOI: 10.64972/dea.2025.v3i1.68

saving and communication cost is necessary in scaling up parallel GNN training for large-scale molecular property prediction.

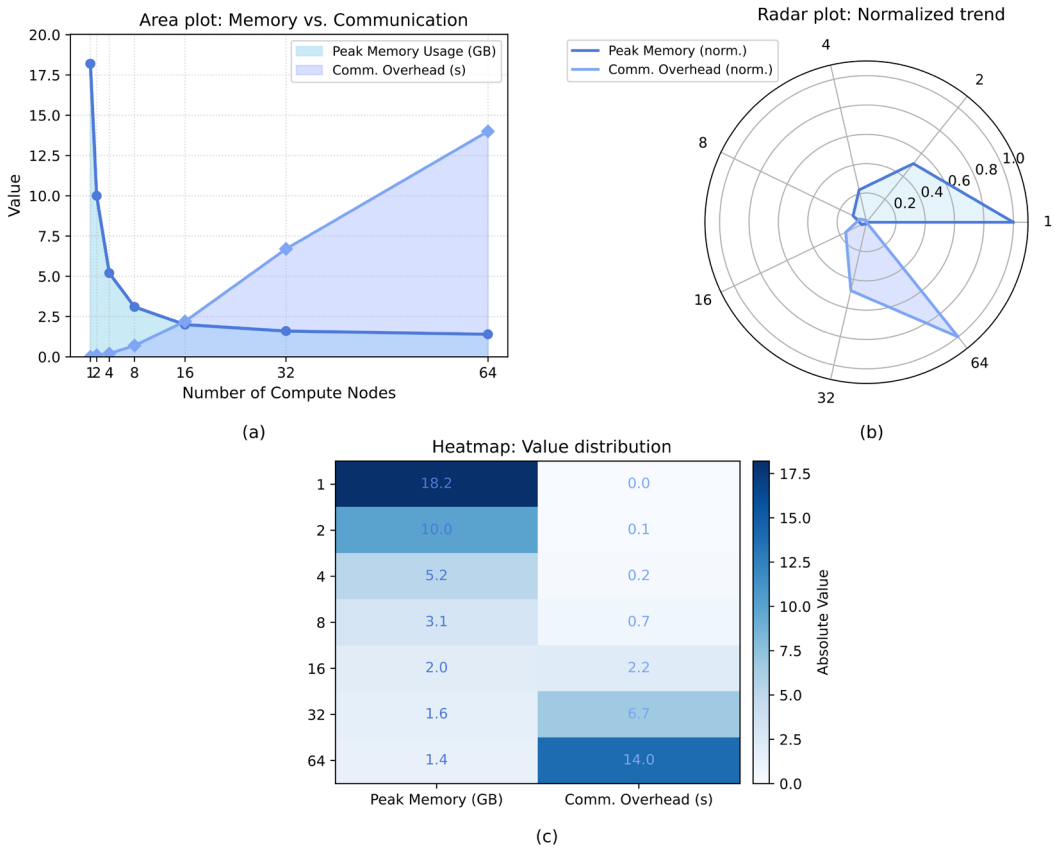

Figure 5. Comparison of peak memory usage and communication overhead across node counts (PNG).
(a) Area plot comparing the trends of peak memory usage and communication overhead as compute node count increases. (b) Radar plot of normalized peak memory and communication overhead, revealing their contrasting scalability patterns in a circular scale. (c) Heatmap visualizing absolute values of both metrics for each node count, enabling direct cross-metric and cross-scale comparison.

Prediction Performance and Case Analysis

To ensure the credibility and good results of graph neural network models in molecular property prediction, rigorous evaluation must be performed. Due to the inevitable natural differences and complexities of different chemical datasets, it is necessary to ask whether the correct answer can be given, and also to ask what the expectations of the model are with the emergence of new types of molecules and new properties. By using a variety of metrics, further insight can be gained into the strengths and weaknesses of the model. The RMSE and MAE metrics provide information on the average amount of error that occurs in the predictions, and can provide a sense of the type of systematic bias that may be present, as well as some random volatility. On the other hand, AUC-ROC is useful in classification models, which helps to understand the ability of classification models to distinguish different classes under different thresholds. By making the most of this data and spending a lot of time planning experiments, researchers will be able to ensure that the models they create and build meet the stringent standards required to apply these models in areas such as drug manufacturing, materials development and other areas that require large amounts of data. Then, a thorough evaluation can help determine how the algorithm should be improved next time and where it should be used in real life. RMSE, MAE, AUC-ROC are used to evaluate the prediction performance. The regression loss function is as follows Eq.(12):

$$\mathcal{L}_{\text{MSE}} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \qquad \text{Eq. (12)}$$

For classification, the binary cross-entropy loss was Eq.(13):

DOI: 10.64972/dea.2025.v3i1.68

$$\mathcal{L}_{\mathrm{BCE}} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log\left(\hat{y}_i\right) + (1-y_i)\log\left(1-\hat{y}_i\right)\right] \qquad \text{Eq. (13)}$$

The AUC-ROC metric was computed as Eq.(14):

$$\mathrm{AUC-ROC} = \int_0^1 TPR(FPR^{-1}(x))dx \qquad \text{Eq. (14)}$$

Figure 6 shows the structured comparison of AUC-ROC prediction accuracy of different GNN methods as the number of computing nodes increases. In addition, GCN, GraphSAGE, and cluster-gcn only have a slight decrease at a very large scale, while the effect after parallelization is the highest and most stable. In addition, the scalability and robustness of the parallelization framework in large-scale tasks of molecular property prediction are also proved.

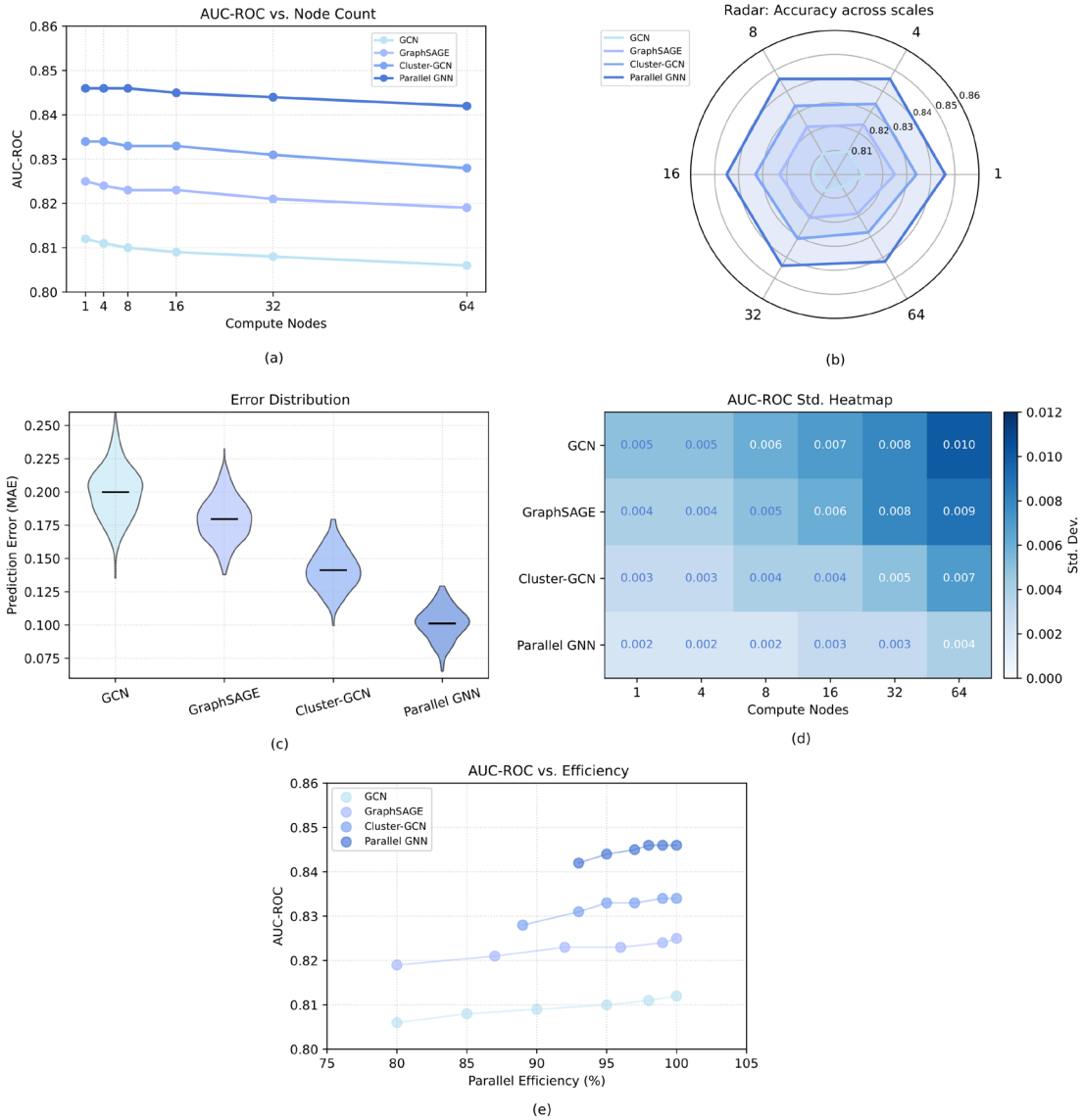

Figure 6. Accuracy vs. scalability for different GNN methods.
(a) AUC-ROC across node counts. (b) Radar plot of accuracy scalability. (c) Violin plot of prediction error distributions (MAE). (d) Heatmap of AUC-ROC standard deviation. (e) Scatter plot of AUC-ROC vs. parallel efficiency.

To meet the practical application needs of GNN in molecular property prediction, it is crucial to maintain authenticity under larger system scales. From the previous experimental results, the results are relatively stable and more extensive, whether in the intensity of the batch size, the number of computing nodes, or the changes in the collection of different data sets. The consistent high accuracy obtained in the extended experiment also proves the success of memory optimization and parallelization strategies. This means that the efficiency improvement brought by parallel and distributed training will not affect the scientific Ty and reliability of the

DOI: 10.64972/dea.2025.v3i1.68

model. When more complex and diverse chemical datasets emerge, it will make GNN models more adaptable and interpretable. This will make it robust across a variety of chemical spaces and applications.

Figure 7 shows a quantitative comparison of predicted and experimental molecular property values for some representative molecules in the test set. The parallel GNN model shows consistency for various structures of compounds with low average error. The accuracy and robustness of the model, as well as its ability to identify cases with the smallest and largest prediction errors, were also demonstrated. Therefore, it has important practical significance for predicting molecular properties.

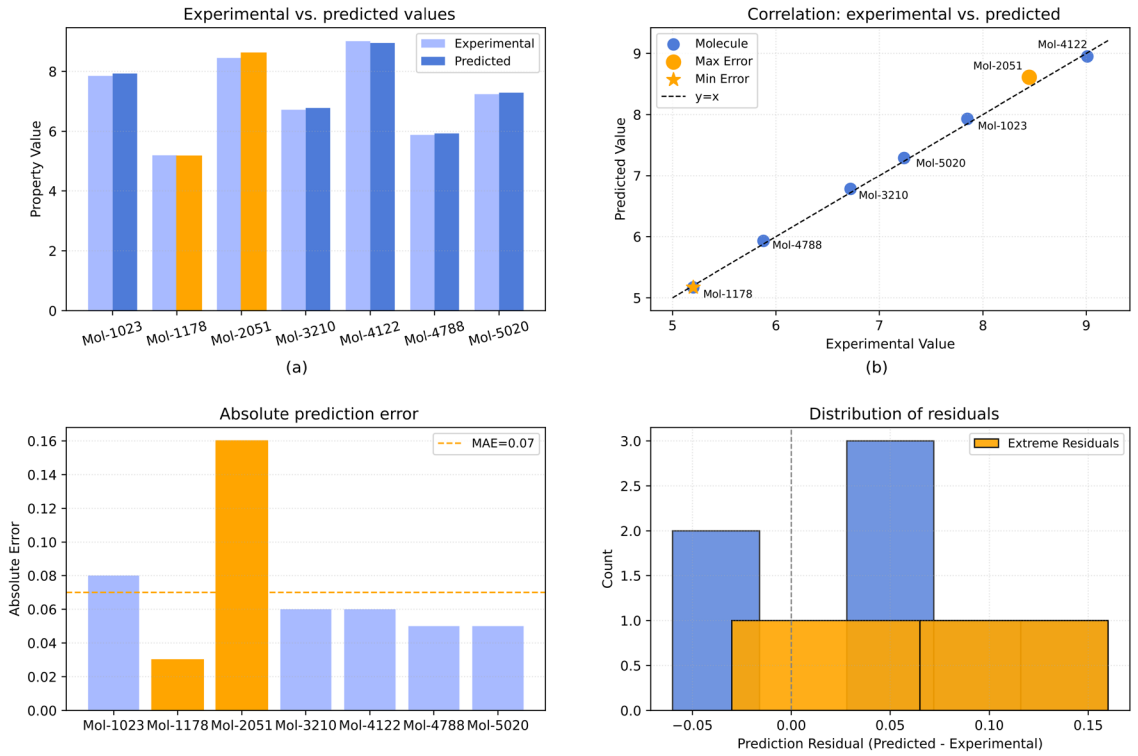

Figure 7. Case study results: Predicted vs. experimental molecular properties.
(a) Experimental and predicted values; best and worst predictions in orange. (b) Experimental vs. predicted scatter; extremes in orange. (c) Absolute errors per molecule; min/max in orange. (d) Residuals histogram; extremes in orange.

The case studies use real-world molecular predictions to demonstrate the model's applicability beyond traditional benchmarks. The model can impact molecular discovery and materials development because it correctly predicts the experimentally measured properties of molecules that are structurally different and have never been seen before. It is important how it works mathematically, but it has been tested very thoroughly because it uses fairly good mathematics, and it is now available. Furthermore, in-depth analysis of individual predictions will uncover model errors and point the way to training or architectural improvements. These practical demonstrations give confidence in the scalability and robustness of this parallel GNN. It may find applications in many high-impact areas such as environmental chemistry, catalyst discovery and drug design.

Ablation Study and Error Analysis

More of this type of research is needed, where each part of the training pipeline is taken apart one by one, so that the function of each part can be understood more clearly. By gradually disabling or modifying certain parts, such as advanced sampling methods, memory optimization methods, or communication processes, the direct impact between prediction accuracy and computing power can be clearly seen. Hidden relationships or synergies between different parts of the system were also discovered, and it was determined which optimizations would have the greatest effect. Error analysis is used to assess the distribution and nature of prediction residuals. These provide useful suggestions for improving the model structure and training methods. Knowing whether errors only occur in specific molecular subtypes or property ranges will help make the necessary specific changes. To make the proposed parallel GNN framework robust and suitable for large-scale deployment in complex chemical analysis environments, rigorous ablation and error analysis must be performed.

DOI: 10.64972/dea.2025.v3i1.68

Ablation experiments illustrate the function of each optimization, and error analysis illustrates the remaining performance differences. As an important indicator of classification evaluation, F1 score was computed as Eq.(15):

$$\text{F1 Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \qquad \text{Eq. (15)}$$

Figure 8 shows the comprehensive quantitative evaluation of the ablation experiment. The results show that the orange part has the largest accuracy drop, the largest prediction error increase, and the longest training time, except for the advanced sampling or load balancing part. The model's efficiency and performance are also helped by other optimizations, but these optimizations are less obvious. These findings suggest that we need all of these for fast, high-accuracy, stable error distribution, and large-scale parallel GNN training.

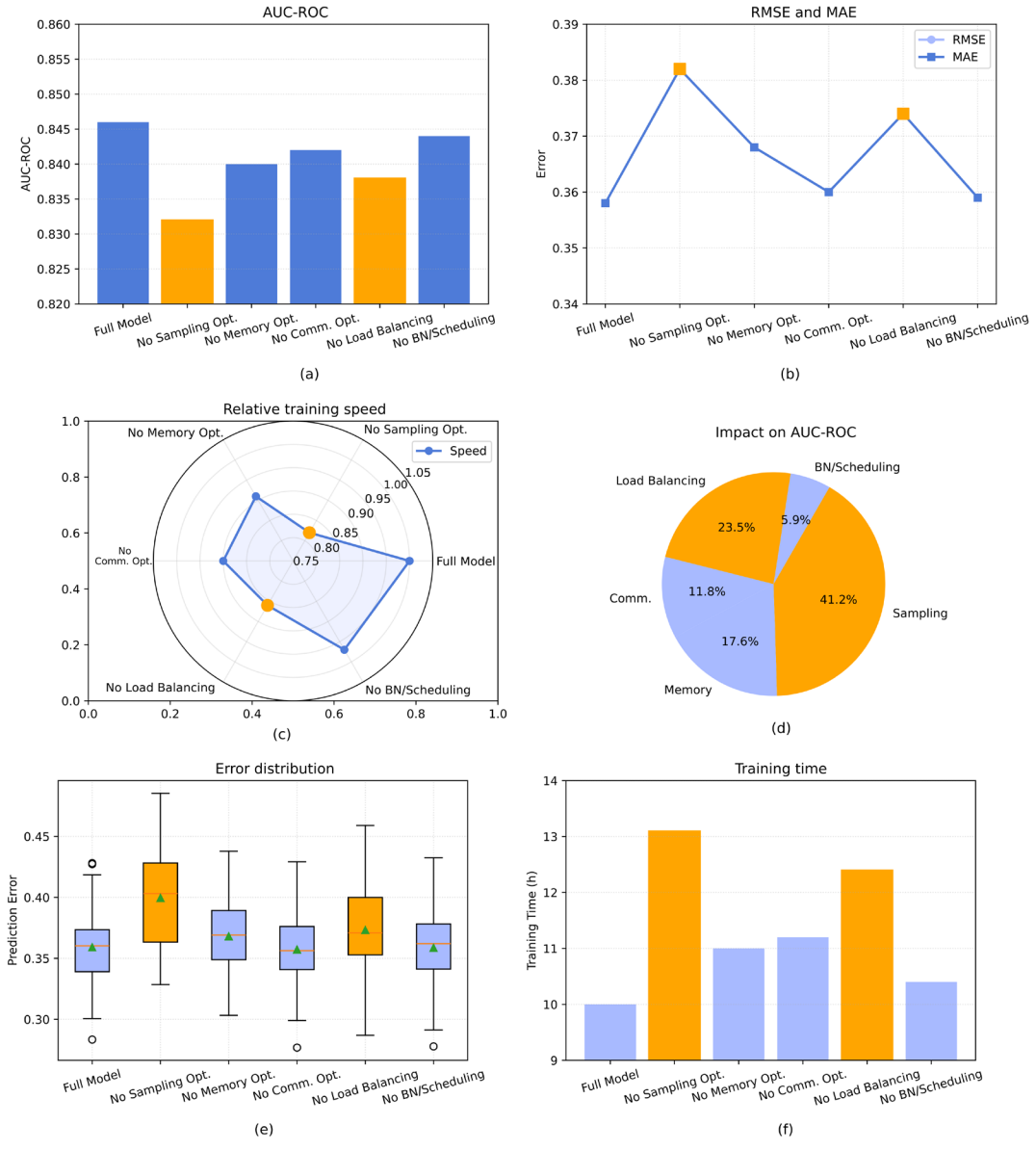

Figure 8. Ablation study: Impact of key optimizations on overall performance.
(a) AUC-ROC. (b) RMSE and MAE.; (c) Training speed (radar). (d) Impact on AUC-ROC (pie). (e) Prediction error distribution (boxplot). (f) Training time.

Ablation studies and error analysis are needed to distinguish the work done by different parts of the system and optimization techniques. By carefully evaluating the impact of each module, more impactful strategies can be found to improve efficiency and accuracy. The results can help improve sampling and partitioning methods as

DOI: 10.64972/dea.2025.v3i1.68

well as memory management, especially when dealing with strange or rare molecular structures. It helps to better understand how algorithm choices affect model performance, allowing us to build a broader GNN framework. As parallel and distributed learning systems evolve, and to cope with increasingly complex cheminformatics problems, continuous evaluation through ablation and error analysis will become best practice.

Conclusion and Outlook

In this study, we systematically explore the potential of parallel graph neural networks (GNNs) in accelerating large-scale molecular property prediction. The core challenges in memory management, workload balancing, and communication efficiency are addressed by designing and implementing a parallel GNN architecture suitable for distributed GPU environments. The proposed framework is rigorously evaluated on a variety of chemical datasets, and the results show that the parallel GNN can achieve near-linear acceleration with the increase of computing resources, while greatly reducing the peak memory usage and communication overhead. Importantly, these efficiency gains do not come at the cost of predictive accuracy, with consistently strong performance on both regression and classification tasks compared to leading baseline models.

Our experimental analysis highlights several key findings. First, the efficient sampling and partitioning strategies help ensure the balance of computation and the minimization of redundant data transfer, thus directly improving the scalability. Second, the integration of advanced distributed memory techniques effectively alleviates the bottlenecks associated with large-scale heterogeneous molecular graphs. Third, ablation studies confirm that each optimization, from micro-batch scheduling to memory and communication management, plays a critical role in achieving high throughput and robust model generalization. These results collectively demonstrate that the proposed parallel GNN method is a powerful tool for accelerating large-scale complex molecular analysis.

Looking ahead, the benefits of parallel GNNs will extend far beyond traditional molecular property prediction. As the size and complexity of molecular datasets continue to grow, the ability to efficiently leverage large-scale computing infrastructure will become increasingly important for discovering novel compounds, optimizing drug candidates, and advancing materials science. Future work will focus on incorporating domain-specific chemical knowledge into the GNN learning process, further reducing distributed training overhead, and extending these methods to a broader class of graph-structured data. In summary, this study provides strong evidence for the transformative impact of scalable parallel GNNs in computational chemistry, laying a solid foundation for the next generation of molecular discovery and design.

References

[1] Dara, O. N., Mohammed, T. A., & Ibrahim, A. A. (2024). Evaluating the Effectiveness of Graph Convolutional Network for Detection of Healthcare Polypharmacy Side Effects. Intelligent Automation & Soft Computing, 39(6). https://doi.org/10.32604/iasc.2024.058736

[2] Yu, L., Su, Y., Liu, Y., & Zeng, X. (2021). Review of unsupervised pretraining strategies for molecules representation. Briefings in functional genomics, 20(5), 323-332. https://doi.org/10.1093/bfgp/elab036

[3] Md, V., Misra, S., Ma, G., Mohanty, R., Georganas, E., Heinecke, A.,& Avancha, S. (2021, November). Distgnn: Scalable distributed training for large-scale graph neural networks. In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (pp. 1-14). https://doi.org/10.1145/3476483

[4] Chen, J., Gao, J., Lyu, T., Oloulade, B. M., & Hu, X. (2022). AutoMSR: auto molecular structure representation learning for multi-label metabolic pathway prediction. IEEE/ACM transactions on computational biology and bioinformatics, 20(6), 3430-3439. https://doi.org/10.1109/TCBB.2022.3198119

[5] Baran, K., & Kloskowski, A. (2023). Graph neural networks and structural information on ionic liquids: a cheminformatics study on molecular physicochemical property prediction. The Journal of Physical Chemistry B, 127(49), 10542-10555. https://doi.org/10.1021/acs.jpcb.3c05521

[6] Atz, K., Grisoni, F., & Schneider, G. (2021). Geometric deep learning on molecular representations. Nature Machine Intelligence, 3(12), 1023-1032. https://doi.org/10.1038/s42256-021-00418-8

[7] Qin, M., Sun, Z., Feng, L., Han, C., Xia, J., & Han, L. (2025). Molecule Former is a GCN-transformer architecture for molecular property prediction. Communications Biology, 8(1), 1668. https://doi.org/10.1038/s42003-025-09064-x

[8] Oulladji, L., Saadallah, M., Guellil, Z., Abbad, H., & Bekhti, H. (2025). AI-Driven molecule generation and bioactivity prediction: A multi-model approach combining VAE, graph and language-based neural networks. Computational Biology and Chemistry, 108532. https://doi.org/10.1016/j.compbiolchem.2025.108532

[9] Wang, Y., Li, Z., & Barati Farimani, A. (2023). Graph neural networks for molecules. In Machine learning in molecular sciences (pp. 21-66). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-031-37196-7_2

[10] Sypetkowski, M., Wenkel, F., Poursafaei, F., Dickson, N., Suri, K., Fradkin, P., & Beaini, D. (2024). On the scalability of gnns for molecular graphs. Advances in Neural Information Processing Systems, 37, 19870-19906. https://doi.org/10.52202/079017-0626

[11] Chiang, W. L., Liu, X., Si, S., Li, Y., Bengio, S., & Hsieh, C. J. (2019, July). Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining (pp. 257-266). https://doi.org/10.1145/3292500.3330925

[12] Cao, Z., Sciabola, S., & Wang, Y. (2024). Large-scale pretraining improves sample efficiency of active learning-based virtual screening. Journal of Chemical Information and Modeling, 64(6), 1882-1891. https://doi.org/10.1021/acs.jcim.3c01938

[13] Jiang, D., Wu, Z., Hsieh, C. Y., Chen, G., Liao, B., Wang, Z., & Hou, T. (2021). Could graph neural networks learn better molecular representation for drug discovery? A comparison study of descriptor-based and graph-based models. Journal of cheminformatics, 13(1), 12. https://doi.org/10.1186/s13321-020-00479-8

[14] Xie, Y., Chen, C., Gong, M., Li, D., & Qin, A. K. (2021). Graph embedding via multi-scale graph representations. Information Sciences, 578, 102-115. https://doi.org/10.1016/j.ins.2021.07.026

[15] Shen, Y., Zhang, J., Song, S. H., & Letaief, K. B. (2022). Graph neural networks for wireless communications: From theory to practice. IEEE Transactions on Wireless Communications, 22(5), 3554-3569. https://doi.org/10.1109/TWC.2022.3219840

[16] Ma, H., Bian, Y., Rong, Y., Huang, W., Xu, T., Xie, W., & Huang, J. (2022). Cross-dependent graph neural networks for molecular property prediction. Bioinformatics, 38(7), 2003-2009. https://doi.org/10.1093/bioinformatics/btac039

[17] Li, C., Ye, Z., Pasini, M. L., Choi, J. Y., Wan, C., & Balaprakash, P. (2025, June). Scaling laws of graph neural networks for atomistic materials modeling. In 2025 62nd ACM/IEEE Design Automation Conference (DAC) (pp. 1-7). IEEE. https://doi.org/10.1109/DAC63849.2025.11132864

[18] Tang, M., Li, B., & Chen, H. (2023). Application of message passing neural networks for molecular property prediction. Current Opinion in Structural Biology, 81, 102616. https://doi.org/10.1016/j.sbi.2023.102616

[19] Lu, Y., Jiang, X., Fang, Y., & Shi, C. (2021, May). Learning to pre-train graph neural networks. In Proceedings of the AAAI conference on artificial intelligence (Vol. 35, No. 5, pp. 4276-4284). https://doi.org/10.1609/aaai.v35i5.16552

[20] Yasin, Q., Liu, B., Sun, M., Sohail, G. M., Ismail, A., Majdanski, M., & Fu, X. (2024). Automatic pore structure analysis in organic-rich shale using FIB-SEM and attention U-Net. Fuel, 358, 130161. https://doi.org/10.1016/j.fuel.2023.130161

[21] Harnik, Y., & Milo, A. (2024). A focus on molecular representation learning for the prediction of chemical properties. Chemical science,15(14), 5052-5055. https://doi.org/10.1039/D4SC90043J

[22] Zhang, Q., Sun, Y., Hu, Y., Wang, S., & Yin, B. (2023). A subgraph sampling method for training large-scale graph convolutional network. Information Sciences, 649, 119661. https://doi.org/10.1016/j.ins.2023.119661

[23] Deng, D., Lei, Z., Hong, X., Zhang, R., & Zhou, F. (2022). Describe molecules by a heterogeneous graph neural network with transformer-like attention for supervised property predictions. ACS omega, 7(4), 3713-3721. https://doi.org/10.1021/acsomega.1c06389