JCIM JOURNAL OF CHEMICAL INFORMATION AND MODELING

pubs.acs.org/jcim Article

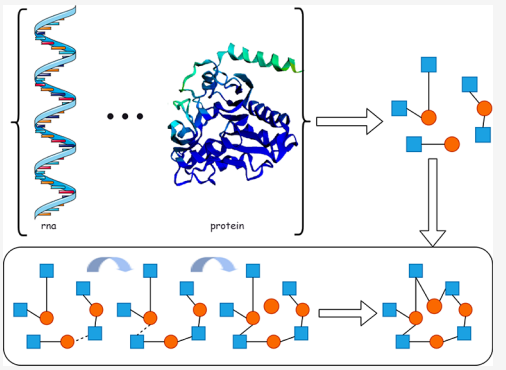
ML-NPI: Predicting Interactions between Noncoding RNA and Protein Based on Meta-Learning in a Large-Scale Dynamic Graph

Tao Wang, Wentao Wang, Xin Jiang, Jiaxing Mao, Linlin Zhuo,* Mingzhe Liu,* Xiangzheng Fu,* and Xiaojun Yao*

Cite This: https://doi.org/10.1021/acs.jcim.3c01238 Read Online

ACCESS | Metrics & More | Article Recommendations

ABSTRACT: Deep learning methods can accurately study noncoding RNA protein interactions (NPI), which is of great significance in gene regulation, human disease, and other fields. However, the computational method for predicting NPI in large-scale dynamic ncRNA protein bipartite graphs is rarely discussed, which is an online modeling and prediction problem. In addition, the results published by researchers on the Web site cannot meet real-time needs due to the large amount of basic data and long update cycles. Therefore, we propose a real-time method based on the dynamic ncRNA−protein bipartite graph learning framework, termed ML-GNN, which can model and predict the NPIs in real time. Our proposed method has the following advantages: first, the meta-learning strategy can alleviate the problem of large prediction errors in sparse neighborhood samples; second, dynamic modeling of newly added data can reduce computational pressure and predict NPIs in real-time. In the experiment, we built a dynamic bipartite graph based on 300000 NPIs from the NPInterv4.0 database. The experimental results indicate that our model achieved excellent performance in multiple experiments. The code for the model is available at https://github.com/taowang11/ML-NPI, and the data can be downloaded freely at http://bigdata.ibp.ac.cn/npinter4.



INTRODUCTION

Noncoding RNAs (ncRNAs) are RNA molecules that do not encode proteins and were once considered as "junk" or "dark matter" in the genome.[1] In recent years, an increasing number of scholars have discovered various types of ncRNAs, including small nuclear RNAs (snRNAs), small nucleolar RNAs (snoRNAs), microRNAs (miRNAs), long noncoding RNAs (lncRNAs), and more.[2−5] These different types of ncRNAs can have a significant impact on many biological processes.[6]

In recent years, increasing evidence has shown that noncoding RNAs (ncRNAs) play a crucial role in various biological processes, such as cell cycle regulation, epigenetic regulation of gene expression, and chromosome modification. In pathological processes, ncRNAs also have a significant impact, such as in cancer,[7−9] diabetes,[10] Alzheimer's disease, and others.[11] In these biological processes, the vast majority of lncRNAs interact with RNA-binding proteins to function. The interaction between ncRNAs and proteins plays a decisive role in the transcription and post-transcription processes. Therefore, developing an efficient and fast method to predict the interaction between noncoding RNAs and proteins is of great significance.

Researchers have proposed many experimental methods to detect the interaction between noncoding RNA and proteins, such as RNAcomputer,[12] RIPChip,[13] which have conducted large-scale biochemical experiments. These methods have high accuracy, but experimental equipment is expensive and time-consuming. Therefore, some new machine learning and deep learning methods have emerged to achieve more efficient prediction.

Bellucci et al. developed catRAPID[14,15] based on the physical properties of protein and RNA secondary structure, hydrogen bonds, and van der Waals forces. Wang et al. proposed a method based on naive Bayes (NB) and extended NB (ENB) classifiers.[16] Lu et al. created a model based on the Fisher linear discriminant method,[17] with secondary structure, hydrogen bond, and van der Waals propensity as input features. The model encodes RNA and protein sequences as numeric vectors and uses the inner product of the vectors to score each RNA−protein pair.

Special Issue: Machine Learning in Bio-cheminformatics

Received: August 6, 2023
Revised: October 16, 2023
Accepted: October 16, 2023

Deep learning has achieved remarkable success in biological domains[18−20] and has also shown promising results in predicting noncoding RNA−protein interactions. For instance, RPI-MDLStack[21] employs an ensemble learning strategy to predict NPI. It first extracts multisource information on RNA and protein and uses a stacking strategy to combine base classification and learn optimal features. IPMiner[22] extracts sequence features using an autoencoder and predicts NPI using a random forest. LPI-BLS[23] employs a general learning system to build a stacking ensemble classifier and then uses a logistic regression classification model to predict noncoding RNA−protein interactions. DeepBind[24] captures sequences using a deep convolutional neural network (CNN) and explores deep RNA−protein interaction information. DeepSEA[25] relies on CNN to analyze sequence information on RNA−protein pairs and predict NPI.

Graph neural networks have been widely used in the field of biology.[26,27] For example, GAMB-GNN[28] trains a graph neural network based on gene structure relationships and Markov chain ordering for cancer classification. This method uses GAT[29] to learn the structure and weights of multiple relationships, thereby obtaining a better representation of sample features. Fout et al. proposed a protein interface recognition method[17] based on graph convolutional networks (GCN).[30] Shen et al. proposed the NPI-GNN model[31] based on the SEAL framework,[32] which converts the NPI prediction task into a binary classification task of subgraphs. Graph neural networks have achieved good results in biological applications, but these methods lack the ability to handle small sample data.

Dynamic graph, also known as temporal graph, is a new research hotspot used to describe a set of relationships that change over time.[33] Dynamic link prediction is also recognized as a basic task in the evolution pattern of dynamic graphs.[34] An important feature of dynamic graphs is that new nodes and links constantly appear as time changes. For example, in exploring the interaction between ncRNA and proteins, a noncoding RNA may involve many peptide bonds in the transcription process, and as the experiment progresses, new peptide bonds participating in the transcription process will be observed. In other words, new NPIs will be constantly observed. Because the new peptide bonds are widely present throughout the entire transcription process and the number of updates is relatively small, it would take a lot of time to rebuild the ncRNA−protein prediction model each time. Therefore, how to predict the continuously updated and relatively small number of NPIs on the dynamic ncRNA−protein bipartite graph has become a problem that must be studied. Furthermore, through investigation, it has been found that no scholars have conducted research on predicting noncoding RNA−protein interactions on dynamic graphs.

Dynamic graph neural networks have achieved advanced performance in node classification and link prediction but cannot efficiently predict links associated with sparse neighborhood nodes. Because the error information corresponding to the sparse neighborhood nodes will be covered by the error information of the overall node with a high probability. In the dynamic ncRNA−protein bipartite graph, the newly added nodes or interactions often cannot be generalized well and the training parameters are not sufficient. This problem is not considered in the existing graph model models. As a technique widely used in sparse sample problems, meta-learning can learn common knowledge in different trainings and can quickly adapt to sparse sample tasks.[35] If the ncRNA−protein dynamic bipartite graph is conveniently regarded as static, then a large amount of time-varying information will be discarded in the training model, which will lead to inaccurate long-term prediction of NPI. Integrating existing meta-learning frameworks with dynamic graphs is not an easy task, so the key is to solve how to subtly integrate meta-learning technology in ncRNA−protein dynamic graphs.

In this study, a model named ML-NPI is proposed based on the meta-learning framework to predict NPI in dynamic ncRNA−protein bipartite graphs. This model can not only adapt to the newly added nodes and interactions in the ncRNA−protein dynamic bipartite graph but also alleviate the NPI prediction problem of sparse neighborhood nodes. At the same time, this is the first time that an incremental modeling strategy has been adopted in a large-scale biomolecular interaction map. After training with a large amount of data, the experimental results proved the effectiveness of our model. Our contributions are summarized in the following aspects:

1. We integrated the dynamic graph neural network model with the meta-learning framework to propose a model that accurately predicts NPI in dynamic ncRNA−protein bipartite graphs. The model can also be applied to predict NPIs associated with nodes in sparse neighborhoods. As far as we know, this is the first time that incremental modeling has been conducted in the field of biomolecular interactions, which can provide constructive references for future researchers to update research results in real-time.
2. Based on the meta-learning framework, a multilevel (node-level and time-level) strategy is used to fine-tune the global parameters of the dynamic GNN model and classifier to improve the expression of newly added nodes. And the residual connection technology is used to make the node embedding more robust.
3. We transformed the NPInterV4.0 data set into a dynamic data set based on time slices and constructed a dynamic ncRNA−protein bipartite graph to capture time-varying information.
4. We conducted multiple sets of comparative experiments to verify the effectiveness of the proposed model. The experimental results show that the performance of the proposed ML-NPI model is higher than that of the current advanced models.

PRELIMINARY

The goal of the model is to predict newly added NPIs in a large-scale dynamic ncRNA−protein bipartite graph over time. For convenience, the following definitions[23] are provided:

Definition 1: The dynamic ncRNA−protein bipartite graph. It can be represented as $G = (V, E)$, where V and E represent the node set and the time-varying edge set, respectively. The temporal edge $e = (v_n, v_p, t) \in E$ represents that the ncRNA node $v_n \in V$ and protein node $v_p \in V$ interact with each other at time t. Note that nodes v_n and v_p can establish multiple edges at different times. Note that for the convenience of using a graph neural network for modeling, ncRNA nodes and protein nodes are considered to be the same type of nodes, but different node indexes are assigned.

With the continuous change in time, new ncRNA (or protein) nodes are constantly added to the dynamic ncRNA−protein bipartite graph. The definition of a new node is given as follows:

https://doi.org/10.1021/acs.jcim.3c01238
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

Definition 2: new nodes in the dynamic ncRNA−protein bipartite graph. In the dynamic ncRNA−protein bipartite graph G, the current time is denoted as t. We define nodes that will be added to G at future time t as new nodes.

Then, the interactions related to the new nodes in the dynamic ncRNA−protein bipartite graph are transformed into the problem of sparse interaction prediction:

Definition 3: NPI prediction of sparse neighborhood nodes in the dynamic ncRNA−protein bipartite graph. Given the dynamic ncRNA−protein bipartite graph $G = (V, E)$ and time s, In our model, we do not differentiate between node types. we denote $Vnew \in V$ as the set of new nodes added after time t. For the newly added node $v \in Vnew$, the goal of the model is to predict the remaining associated NPIs based on its associated K NPIs.

K represents randomly selecting K edges from the existing edges of a node, and it is typically set to a number less than 8, which can be referred to as the $K - shot$ problem in sparse-sample prediction problems. For example, four shots means $K = 4$. In the experiment, we will compare the performance of the proposed model under different values of K.

PROPOSED METHOD

In this section, we propose a meta-learning strategy-based approach to predict NPIs. And this method can be better applied to incremental interaction data sets, which mainly focus on solving the following two problems:

1. How can we alleviate the issue of sparse neighborhoods for certain nodes by incorporating a meta-learning strategy into the dynamic bipartite graph of ncRNA−protein?
2. How can we extract multilevel information from ncRNA and proteins to adapt to the incremental new interaction data?

Meta-Learning Task Setup. When meta-learning training is performed on ncRNA−protein bipartite graphs, the training process can be divided into multiple tasks. Each task has separate training and test sets for one training session. Information shared between different tasks under the meta-learning framework can be learned as global parameters. In this way, in the process of inferring or predicting new NPIs, only some fine-tuning of the basic parameters is required.

General information at the node and time level is extracted for each task of meta-learning. At this time, each task corresponds to a fixed time. Assume that each node plays a different role at different times; for example, there is a new check interaction at a certain new time. Furthermore, each node performs differently in different tasks. In addition, each node has the property of time invariance; for example, the interaction associated with the node will not change or disappear at different times. Therefore, node i (ncRNA or protein) can be set as $\{D_i^1, D_i^2, ..., D_i^k, \cdots, D_i^{num}\}$ according to the time, and the interaction set is also set according to a similar strategy. For the vth task $T_i = (D_i, Q_i)$, its training set D_i can be set as

$$D_i = D_i^1, D_i^2, ..., D_i^k, ..., D_i^{n-1} \tag{1}$$

where n represents the total number of divided tasks or time. D_i^k is the NPI set at the k-th time, Q_i is the NPI set at the num-th time (the last time), and here is the test set of the task. In this way, multilevel information is extracted to adapt to the incremental new interaction data (test set). The next process determines positive and negative NPI samples.

In the training phase of meta-learning, since the number of interactions associated with existing nodes is often greater than that associated with newly added nodes, a small part of the existing interaction set is sampled as the positive sample interactions associated with each node. This will prevent the newly added data distribution from being overwritten by existing data. Specifically, in each batch, K associated interactions are sampled for each node i, and the number of samples is kept approximately the same at different time instants. In addition, in the test set Q_i, the interactions of K different associations are also sampled.

In the inference or prediction stage of meta-learning, the K interactions associated with each new node are sampled to train and fine-tune the parameters of the model, and the remaining associated interactions of the new nodes are used for inference, prediction, and evaluation. In addition, during the training process, the same number of negative samples was sampled for training.

Predicting NPIs on ncRNA-Protein Dynamic Bipartite Graphs. In this subsection, a dynamic GNN component is designed based on a meta-learning strategy to predict node-level interactions. This component integrates the topology, time, and ncRNA (or protein) nodes of the ncRNA−protein bipartite graph to obtain embedding of the ncRNA (or protein) nodes. Then the embeddings of ncRNA and protein nodes are assembled and fed into a classifier to predict whether there is an interaction between ncRNA and protein nodes. Overall, this dynamic GNN component consists of a node encoder and an NPI classifier module.

Node Encoder g_φ: At time t, the embedding of an ncRNA (or protein) node aggregates the information on the node's neighbors $N_i(t)$. It includes the node itself and the neighbor nodes before time t. For a node i, its initial node embedding h_i^0 can be obtained by performing a linear transformation on its features. Its initial features can be expressed as $x_i \in R^{d_1}$, and the transformation matrix shared by nodes can be expressed as $W \in d_1 * d_2$, where d_1 and d_2 are feature and embedding dimensions, respectively.

For the layers of the dynamic GNN component, an attention mechanism is employed to compute the weights contributed by different neighbors. Then, the embedding of node i at time t can be written as follows:

$$\begin{aligned} h_i^l(t) &= g_\varphi(i, N_i(t)) \\ &= \sigma_1\left(\left(\sum_{j \in N_i(t)} \alpha_{i,j}^l (h_j^{l-1}(t_{i,j}) \| \Phi(t - t_{i,j}))\right)\right) \end{aligned} \tag{2}$$

where $h_i^l(t)$ represents the node embedding of node i in layer l of the dynamic GNN component at time t, σ_1 represents the RELU activation function, $t_{i,j}$ represents the time when there is an interaction between node i and its neighbor node j, $\|$ indicates the connection operation, $W \in 2d_2 \times d_2$ indicates the sharing matrix of the node, Φ represents a temporal encoder[36,37] that can approximate any positive definite kernel, and $\alpha_{i,j}^l$ represents the attention weight between node i and its temporary neighbor j, which is calculated as

https://doi.org/10.1021/acs.jcim.3c01238
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

$$\alpha_{i,j}^{l} = \mathrm{softmax}(q_{i,j}^{l}) = \frac{\exp(q_{i,j}^{l})}{\sum_{k\in N_i(t)} \exp(q_{i,k}^{l})} \tag{3}$$

$$q_{i,j}^{l} = \hat{a}(h_i^0 \| \Phi(0) \| h_j^{l-1}(t) \| \Phi(t - t_{i,j})) \tag{4}$$

where $q_{i,j}^{l}$ represents the weights of nodes i and j, and $\hat{a} \in 4 \times d_2$ represents the shared attention vector. This value integrates the initial and $l-1$ layer embedding, time encoding. This integration of the embeddings from the zeroth layer and the $l-1$ layer is done to include the initial information and enhance the robustness of the model.

Classifier c_w: Based on the above process, the embedding of ncRNA and protein nodes can be calculated. The representations of ncRNA node n and protein node p at time t are $h_n^L(t)$ and $h_p^L(t)$ respectively, where L is the number of layers of dynamic GNN components, which can affect the effect of GNN model training. Immediately, it can be speculated whether the interaction (n, p, t) between the ncRNA node and the protein node exists:

$$\begin{aligned} P_{n,p} &= c_w(h_n^L(t), h_p^L(t)) \\ &= \sigma_2(MLP(h_n^L(t) \| h_p^L(t))), \end{aligned} \tag{5}$$

where $P_{i,j}$ is the probability that the interaction exists, c_w represents all the relevant parameters of the classifier, which is the activation function (here we use the sigmoid function). $MLP(.)$ represents the multilayer perceptron, and $\|$ represents the concatenate operation. Afterward, the loss based on each interaction is minimized to train the dynamic graph component:

$$L_{n,p} = y_{n,p} \log P_{n,p} - (1 - y_{n,p}) \log(1 - P_{n,p}) \tag{6}$$

if there is an intearaction between node n and node p, its label $y_{n,p} = 1$; if there is no interaction, the label $y_{n,p} = 0$. Compare the proposed dynamic GNN model with typical dynamic GNN models, such as TGAT (using multihead attention mechanism and MLP suboperation for prediction)[37] or EvolveGCN (using LSTM stacking on ordinary GCN model to achieve temporal information and different molecular structure),[38] the proposed dynamic GNN component benefits from the meta-learning framework, and only needs to fine-tune the model parameters for new tasks, which is faster and more convenient.

Hierarchical Meta-Learning Training Model. In this subsection, a hierarchical meta-learning training model is introduced. For a certain task, the global parameters of the dynamic GNN model can be fine-tuned at the time and node level.

Time-level parameter fine-tuning: For a task T_i, its training set under the meta-learning framework is $\{D_i^1, D_i^2, \cdots, D_i^k, \ldots, D_i^{num-1}\}$. Each time can be considered as an independent process, and with the addition of new NPIs, the loss obtained from each process during training is used to fine-tune the global parameters of the nodal encoders. For ncRNA node n and its associated interaction $(n,p,t^k) \in D_n^k$, its node embedding can be written as

$$h_n^L(t^k) = g_\psi(n, N_n(t^k)) \tag{7}$$

where t^k represents the k-th time. The embedding of protein nodes based on their associated interactions is also calculated in the same way. Afterward, the loss value can be calculated based on positive and negative NPI samples associated with ncRNA node n.

$$L(\psi, w, D_n^k) = -\sum_{(n,p)\in D_n^k} \log P_{n,p} - \sum_{(n,p)\in D_n^k - E} \log (1 - P_{n,p}) \tag{8}$$

where $p_{n,p} = c_w(h_n^L(t^k), h_p^L(t^k))$ represents the probability of the existence of the dynamic interaction (n, p, t^k) associated with node n.

Next, at the k-th time, update the global parameters ψ of the node encoder based on the specific ncRNA node n (or protein node) by gradient descent:

$$\psi_n^k = \psi - \beta \frac{\partial L(\psi, w, D_n^k)}{\partial \psi} \tag{9}$$

where β represents the learning rate.

Node-level parameter fine-tuning: The global parameter w of the classifier model is fine-tuned based on the node encoder ψ_n^k of a specific node. In this process, classifier parameter w is projected to obtain parameter w_n. For each set D_n^k, the following two steps are performed to obtain the classifier parameter w_n^k:

$$w_n = w + h_n^0 \cdot W_w \tag{10}$$

$$w_n^k = w_n - \lambda \frac{\partial L(\psi_n^k, w_n, D_n^k)}{\partial w_n} \tag{11}$$

where W_w represents the projected matrix, λ represents the learning rate optimization. After the above two processes, based on a specific ncRNA node n (or protein p) at a specific time k, the global parameters can be fine-tuned from θ to $\theta_n^k = \{\psi_n^k, w_n^k\}$. In order to further optimize the meta-learning parameters based on the test set Q_n, the specific parameters obtained for each node n in all train set D_n are fused:

$$\psi_n^* = \sum_{k=1}^{n-1} a_n^k \psi_n^k, \qquad w_n^* = \sum_{k=1}^{n-1} a_n^k w_n^k \tag{12}$$

where $a_n^k = \mathrm{softmax}(-L(\psi_n^k, w_n^k, D_n))$; it is calculated by performing softmax based on the entire training set D_n, and it is used to represent the weight of θ_n^k. Compared with other aggregation methods, it is very convenient to calculate the weight a_n^k without an extra process, which can more effectively accelerate the convergence of the entire meta-learning training process.

After the above work is finished, the goal shifts to minimize the loss of the test set Q_n, and further update the global parameter $\theta = \{\psi, w\}$ and the projection matrix W_w through back-propagation.

$$\theta = \theta - \eta_\theta \sum_{n\in V} L(\psi_n^*, w_n^*, Q_n) \tag{13}$$

$$W_w = W_w - \eta_{W_w} \sum_{n\in V} L(\psi_n^*, w_n^*, Q_n) \tag{14}$$

where η is the learning rate for meta-learning in the above process.

Model Architecture. Overall, the learned global parameter ψ encodes information spanning multiple ncRNA (or protein) nodes and multiple times; At different times, the classifier parameter w_n encodes specific invariant knowledge of ncRNA

https://doi.org/10.1021/acs.jcim.3c01238
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

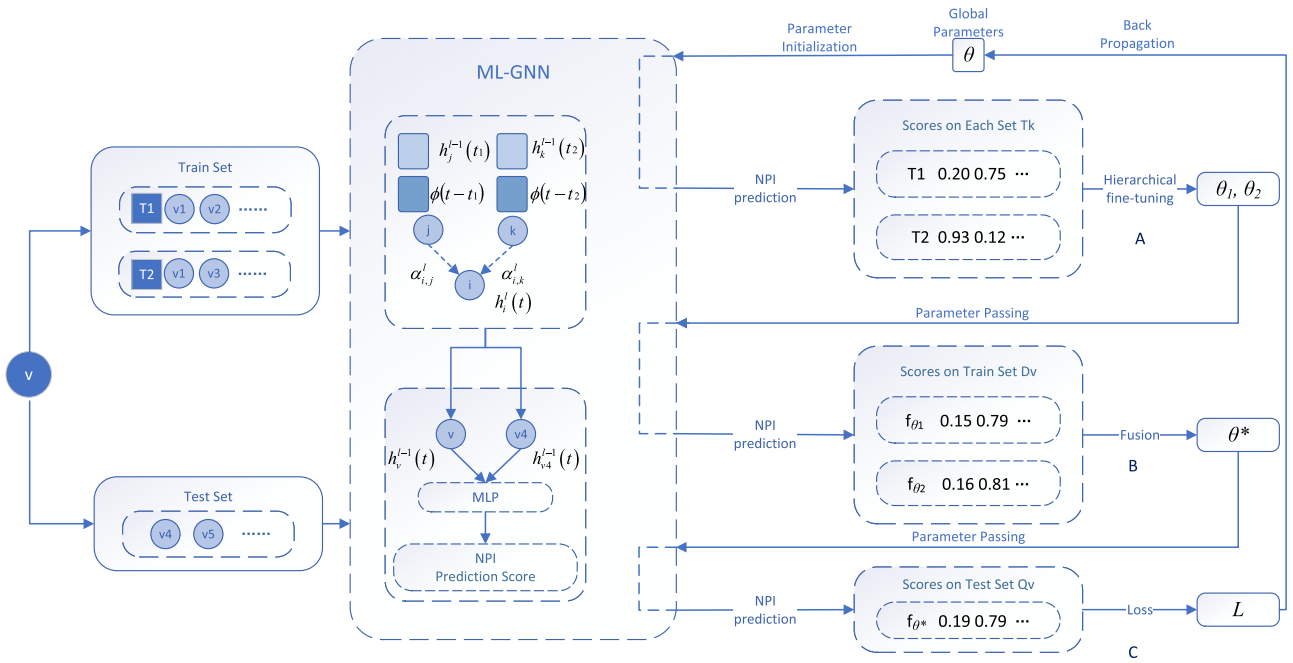


Figure 1. A simple flowchart for processing node v using the ML-NPI model. The left side of the graph shows the division of training and Test sets for each of our tasks, the middle part displays the process of Graph Neural Networks (GNN) handling nodes and time information, and the right side illustrates the process of meta-learning.

(or protein) node n. In other words, the edges already associated with nodes will not disappear over time, which can be achieved through classifiers. The detailed process of meta learning training in the algorithm ML-NPI is shown in Figure 1. A brief example will demonstrate the process of the ML-NPI model. In Figure 1, the ML-NPI module displays the entire model architecture, which mainly includes two parts: a dynamic graph encoder and a classifier. For node v, its associated NPI set is divided into two parts based on the T1 and T2 times. The node pair (v, $v4$) constitutes an NPI. In step A, fine-tune global parameters at the time and node levels; In Step B, we merge the parameters of the training set at different time periods in node v, as per Equation 12; In step C, we optimize the entire model parameters based on the test set of node v.

EXPERIMENTAL RESULTS

In order to evaluate the performance of the proposed method, 300,000 pieces of data were selected from the public NPInterv4.0 database for experiments. The proposed ML-NPI model is compared with three types of traditional dynamic GNN models and finally discussed in detail in the following four aspects: (RQ1) Among all the models based on the ncRNA−protein dynamic bipartite graph, which model is more effective in predicting NPI; (RQ2) how does the ML-NPI model perform in sparse neighborhoods; (RQ3) what is the impact of parameter changes on the ML-NPI method; (RQ4) what is the impact of parameter fine-tuning on the node level and time level on the performance of ML-NPI.

Data Sets and Evaluation Strategies. In the experiment, the data set NPInterv4.0 was used, which integrated 6 million new experimentally identified noncoding RNA interactions. NPIs includes data for multiple RNAs, such as 658,171 lncRNA-related interactions, 488,025 miRNA-related interactions, 61,700 snoRNA-related interactions, 12,789 snRNA-related interactions, and 335 circular RNA-related interactions. As for the construction of the dynamic graph, we divided it into a series of time frames based on the random time marks in the NPI data. In each frame, their interactions are used as edges to construct the graph. As time progresses, new nodes and edges are added to the graph, forming a dynamic graph.

Two static GNN models, GraphSAGE[39] and GAT,[29] and two dynamic GNN models, TGN[40] and TGAT,[37] will be selected as baseline comparison methods. GraphSAGE[39] is a GNN model based on an inductive strategy, which can effectively predict new nodes. GAT[29] is an attention-based GNN model that calculates attention weights between two nodes. For these two static graph models, we treat each moment in the graph as an independent static graph. TGAT[37] is a dynamic graph model that integrates the attention mechanism of GAT and the sampling strategy of GraphSAGE for selecting and aggregating time-related information. TGN[37] combines RNN and dynamic graph technology, which can effectively predict new nodes and their interactions.

The goal is to make predictions about future ncRNA node (or protein) associated interactions. In the following experiments, the data set is divided by the following two steps: (1) Sort all NPIs in chronological order, first select the nodes in the NPIs that satisfy K interactions, set K to 2−4, and set the step size to 1. Then, the early appearing NPI nodes are divided into the training set, the later appearing NPI nodes into the validation set, and the latest appearing nodes into the test set. For example, represent all data as N, and the three times of t_{train}, t_{val}, and t_{test} are set, and N can be divided into three parts: Nt_{train}, Nt_{val}, and Nt_{test} according to the time. There are no identical nodes between Nt_{train}, Nt_{val}, and Nt_{test}; (2) each node T_i is divided into the training set D_i and the test set Q_i according to time. For each node's D_i in Nt_{train}, with the addition of new NPIs, we will fine-tune the parameters. This fine-tuning does not change the initial parameters of the model; it is only used to predict future NPIs in Q_i of Nt_{train}. and the initial parameters are updated by the loss obtained in Q_i. In

https://doi.org/10.1021/acs.jcim.3c01238
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

Table 1. Performance of all methods on NPInterV4.0 dataset

Models	AUC	AUPR	F1-score	Accuracy	Precision	Recall	Specification
GAT[29]	62.96	66.20	63.06	56.77	51.03	82.50	30.15
GraphSAGE[39]	79.35	79.84	69.09	68.52	59.38	82.61	58.06
TGAT[37]	88.07	80.19	57.87	74.63	58.90	59.43	89.83
TGN[40]	94.00	92.27	88.26	88.60	83.80	**94.53**	82.67
ML-NPI	**95.82**	**96.32**	**91.01**	**91.11**	**90.85**	91.33	**90.71**

Table 2. Performance of ML-NPI and TGN Models on Nodes with Sparse Neighborhoods

K-interactions	Models	AUC	AUPR	F1-score	Accuracy	Precision	Recall	Specification
2-interactions	TGN[40]	94.49	93.06	87.72	87.92	82.67	**94.72**	81.13
	ML-NPI	**94.93**	**96.11**	**91.14**	**91.50**	**91.08**	91.36	**90.95**
3-interactions	TGN[40]	94.00	92.27	88.26	88.60	83.80	**94.53**	82.67
	ML-NPI	**95.82**	**96.32**	**91.01**	**91.11**	**90.85**	91.33	**90.71**
4-interactions	TGN[40]	94.41	92.84	87.58	87.70	82.25	**94.80**	80.59
	ML-NPI	**95.10**	**95.64**	**90.07**	**90.08**	**89.89**	90.48	**89.67**

Nt_{val} and Nt_{test}, the parameters in Nt_{train}, and only use the training set D_i for parameter fine-tuning, to predict Q_i future NPIs. The evaluation strategy uses common AUC, AUPR, F1-score, Accuracy, Recall, Precision, and Specification. In order to ensure the fairness of the experiment, we have the same data set division and evaluation strategy for each model.

RQ1: Comparison with Other Methods. The performance of the proposed ML-NPI model is compared to other baseline models on the NPInterV4.0 database, and the *K* value is set to 3. As shown in Table 1, the optimal results and suboptimal results have been shown in bold and italics.

In the comparative experiment of ML-NPI model with GraphSAGE[39] and GAT[29] static GNN model, the prediction effect of NPI is significantly better than other comparative experimental methods. Therefore, the model parameters are fine-tuned and the adaptability of the model to new data is improved. In dynamic models, the time factor plays an important role.

Compared with TGAT[37] and TGN[40] dynamic models, the ML-NPI model can extract general information at the node level and time level, so that it can quickly fine-tune the entire model parameters during incremental or dynamic modeling. In the other dynamic GNN models mentioned above, once the global parameters are trained, they are no longer updated but the RNN technology is used to update the node embedding. Therefore, they cannot adapt well to new samples and have a poor generalization ability. In addition, we found that these dynamic GNN models are even inferior to static GNN models in some indicators. After analysis, the reason may be that the above model only focuses on the newly added nodes and does not consider some general information, because it cannot cope with nodes with sparse topological neighborhoods.

RQ2: Performance on Sparse Neighborhoods. We construct a set of comparative experiments to verify the performance of ML-NPI on sparse neighborhoods. Specifically, the ML-NPI model is compared with the current state-of-the-art dynamic graph model TGN on the NPInterV4.0 data set for multiple indicators. Similar to Section, the goal is to predict all remaining associated interactions of new nodes based on its *K* associated interactions. The *K* values were respectively set to 2, 3, and 4 for experiments, and the experimental results are shown in Table 2. Obviously, regardless of the value of *K*, the performance of ML-NPI can surpass the TGN model for most indicators. This may benefit from the meta-learning strategy, which enables the ML-NPI model to better adapt to new nodes and sparse neighborhoods. The ML-NPI model only lags behind TGN in the Recall indicator, which shows that TGN is more inclined to predict samples as positive samples, while ML-NPI has higher requirements for positive samples.

RQ3: Analysis of Parameters. Since the parameters affecting the performance of ML-NPI mainly involve the number of neighbor nodes, The number of divisions for task, embedding dimension, and batch size, an experimental analysis based on these parameters was constructed. With parameters not as variables, we set the embedding size to 64, the batch size to 48, the number of neighbors to 16, and the number of divisions for the task to 2. Specifically, the influence of the number of neighborhoods on the model performance is tested, and the results are shown in Figure 2; the impact of batch size

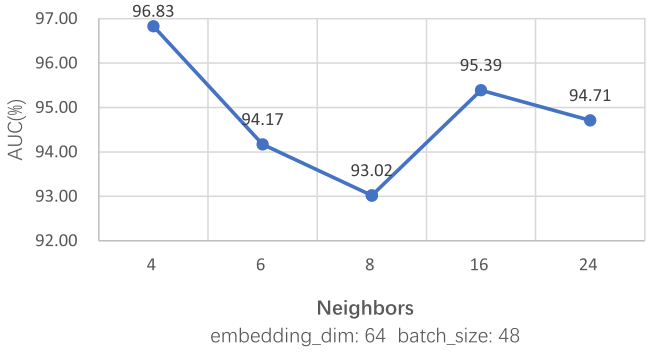


Figure 2. Impact of the number of neighbors on the performance of the ML-NPI model.

on the model is tested, and the results are shown in Figure 3; and the impact of embedding dimension on model performance is tested, and the results are shown in Figure 4. The impact of the number of divisions for the task on model performance is tested, and the results are shown in Figure 5.

The results show that the average AUC in Figure 2 is 94.82, and the standard deviation is 1.42. The average AUC in Figure 2 is 95.95 with a standard deviation of 0.64; the average AUC in Figure 2 is 94.90 with a standard deviation of 1.22. As the number of neighbors increases, the AUC index first decreases and then increases, which may be caused by other parameter settings. Because when the number of neighbor nodes is small, the performance of the model may be more affected by the

https://doi.org/10.1021/acs.jcim.3c01238
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

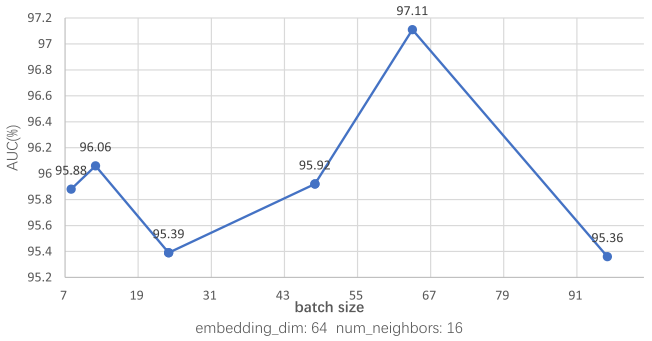


Figure 3. Impact of batch size on the performance of the ML-NPI model.

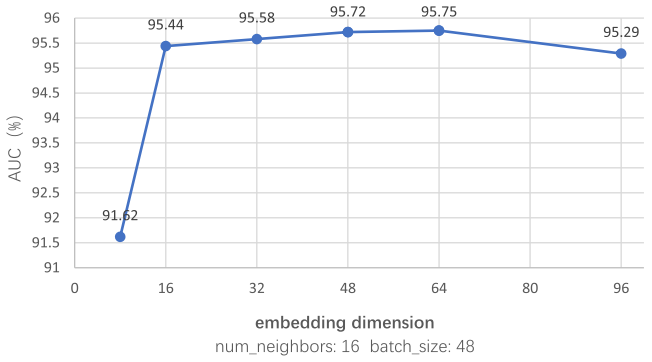


Figure 4. Impact of embedding size on the performance of the ML-NPI model.

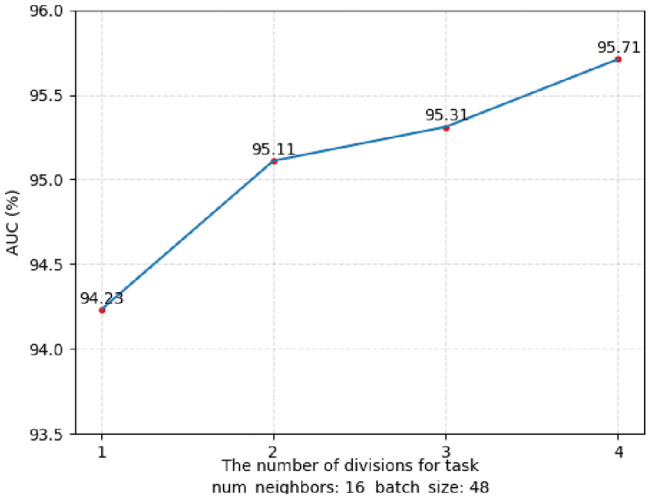


Figure 5. Influence of the number of divisions for task on the performance of the ML-NPI model.

embedding size and batch size. As the batch size increases, the model performance improves steadily; when the batch size is >64, the model performance decreases, which may be related to sampling. The performance of the model continues to improve as the embedding dimension increases and tends to be stable after reaching 16. As the number of divisions in the task increases, the AUC also shows a stable upward trend, which suggests that integrating learning parameters from more levels often contributes to more effective prediction of new edges. The results can prove that the overall performance of the model is relatively stable as the parameters change, and it is easy to find the appropriate number of neighbor nodes, embedding dimension, batch size, and the impact of the number of divisions for task.

RQ4: Ablation Experiment. The proposed model adopts a meta-learning strategy and a hierarchical optimization training scheme to predict the interaction of future new nodes in the ncRNA−protein dynamic bipartite graph. The model mainly includes fine-tuning of global parameters at the node and time levels under the meta-learning framework. We will construct an ablation experiment to verify the respective roles of each part. The ML-NPI model adopts the scheme of fine-tuning the global parameters at the node level and the time level; the ML-NPI-BN model does not adopt the scheme of fine-tuning the global parameters at the node level and the time level; the ML-NPI-N model only adopts the scheme of fine-tuning the global parameters only at the node level; the ML-NPI-I model represents the scheme of fine-tuning the global parameters only at the time level; The ML-NPI-R model does not adopt the strategy of using residual connection technology.

The experimental results are shown in Figure 6. Clearly, the ML-NPI model outperforms all models, and the ML-NPI-BN

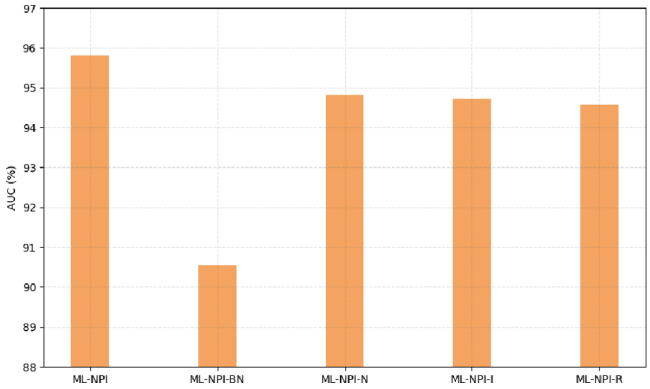


Figure 6. Influence of parameter fine-tuning at node level and time level on the performance of the ML-NPI model.

model is the worst performing. The performance of the ML-NPI-N model is slightly higher than that of the ML-NPI-N model. It can be seen that the fine-tuning schemes at the node and time levels both promote the performance of the model, and the two are mutually reinforcing. In addition, the node-level fine-tuning scheme is slightly better than the time-level fine-tuning scheme, which may be because the original NPInterV4.0 data set is a static data set. Moreover, ML-NPI-R performs slightly less well than ML-NPI-I, indicating that the use of residual strategies has a positive effect on the optimization of the model. In summary, designing a fine-tuning scheme based on node and time levels under the meta-learning framework is very suitable for NPI prediction on ncRNA−protein dynamic bipartite graphs.

CONCLUSION

This article investigates traditional NPI forecasting methods and proposes a dynamic NPI forecasting problem. At the same time, the existing static and dynamic GNN technology to deal with the model of dynamic graph data is investigated, and a brief analysis is carried out. Static GNN techniques cannot capture time-varying information, resulting in a lower performance. Dynamic GNN technology only relies on other timing-based methods such as RNN to update node embedding but

https://doi.org/10.1021/acs.jcim.3c01238
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

ignores general information, resulting in low generalization ability. In this article, the ML-NPI model is proposed to solve the NPI prediction problem on ncRNA−protein dynamic bipartite graphs. The concept of time is added to the model, which can capture the information on ncRNA nodes (or protein nodes) changing over time. Based on the multilevel training process, the nodes can fine-tune the global parameters at the node level and the time level and have better generalization ability for new nodes. The most important thing is that the meta-learning strategy is added to the ncRNA−protein dynamic model so that the model has the ability to handle small sample data. At the same time, we constructed dynamic NPI data based on the latest NPInterV4.0 database. In the comparison with other models in the NPInterv4.0 data set, the performance of the ML-NPI model has achieved obvious advantages. Both parametric and ablation experiments demonstrate the stability of the model and the effectiveness of the model strategy. Overall, this is the first time that an incremental modeling strategy has been adopted in the field of ncRNA−protein interactions and it performed well. We hope to provide a reliable reference for incremental modeling of biomolecular interactions (including related Web site updates) in the future.

■ ASSOCIATED CONTENT

Data Availability Statement

Our data and code are publicly available at https://github.com/taowang11/ML-NPI.

■ AUTHOR INFORMATION

Corresponding Authors

Linlin Zhuo − *Neher's Biophysics Laboratory for Innovative Drug Discovery, State Key Laboratory of Quality Research in Chinese Medicine, Macau Institute for Applied Research in Medicine and Health, Macau University of Science and Technology, 999078 Macao, China; Wenzhou University of Technology, 325000 Wenzhou, China;* orcid.org/0000-0002-6586-0533; Email: zhuoninnin@163.com

Mingzhe Liu − *Wenzhou University of Technology, 325000 Wenzhou, China*; Email: liumz@cdut.edu.cn

Xiangzheng Fu − *Neher's Biophysics Laboratory for Innovative Drug Discovery, State Key Laboratory of Quality Research in Chinese Medicine, Macau Institute for Applied Research in Medicine and Health, Macau University of Science and Technology, 999078 Macao, China*; Email: xzfu@must.edu.mo

Xiaojun Yao − *Neher's Biophysics Laboratory for Innovative Drug Discovery, State Key Laboratory of Quality Research in Chinese Medicine, Macau Institute for Applied Research in Medicine and Health, Macau University of Science and Technology, 999078 Macao, China;* orcid.org/0000-0001-9958-8438; Email: xjyao@must.edu.mo

Authors

Tao Wang − *Neher's Biophysics Laboratory for Innovative Drug Discovery, State Key Laboratory of Quality Research in Chinese Medicine, Macau Institute for Applied Research in Medicine and Health, Macau University of Science and Technology, 999078 Macao, China; Wenzhou University of Technology, 325000 Wenzhou, China*

Wentao Wang − *Wenzhou University of Technology, 325000 Wenzhou, China*

Xin Jiang − *Wenzhou University of Technology, 325000 Wenzhou, China*

Jiaxing Mao − *Central South University of Forestry and Technology, 410000 Changsha, China*

Complete contact information is available at: https://pubs.acs.org/10.1021/acs.jcim.3c01238

Notes

The authors declare no competing financial interest.

■ ACKNOWLEDGMENTS

Tao Wang and Wentao Wang contributed equally.

■ REFERENCES

(1) Schaukowitch, K.; Kim, T. K. Emerging epigenetic mechanisms of long non-coding RNAs. *Neuroscience* **2014**, *264*, 25−38.

(2) Koch, L. Screening for lncRNA function. *Nat. Rev. Genet.* **2017**, *18*, 70.

(3) Zhu, J.; Fu, H.; Wu, Y.; Zheng, X. Function of lncRNAs and approaches to lncRNA-protein interactions. *Sci. China Life Sci.* **2013**, *56*, 876−885.

(4) Bridges, M. C.; Daulagala, A. C.; Kourtidis, A. LNCcation: lncRNA localization and function. *J. Cell Biol.* **2021**, *220*, e202009045.

(5) Mathieu, E.-L.; Belhocine, M.; Dao, L.; Puthier, D.; Spicuglia, S. Functions of lncRNA in development and diseases. *Medecine Sciences: M/S.* **2014**, *30*, 790−796.

(6) Louro, R.; Smirnova, A. S.; Verjovski-Almeida, S. Long intronic noncoding RNA transcription: expression noise or expression choice? *Genomics* **2009**, *93*, 291−298.

(7) Dunn, G. P.; Old, L. J.; Schreiber, R. D. The immunobiology of cancer immunosurveillance and immunoediting. *Immunity* **2004**, *21*, 137−148.

(8) Bailar, J. C.; Gornik, H. L. Cancer undefeated. *N. Engl. J. Med.* **1997**, *336*, 1569−1574.

(9) Pantel, K.; Alix-Panabières, C.; Riethdorf, S. Cancer micrometastases. *Nat. Rev. Clin. Oncol.* **2009**, *6*, 339−351.

(10) Reyes Sanamé, F. A.; Pérez Álvarez, M. L.; Alfonso Figueredo, E.; Ramírez Estupiñan, M.; Jiménez Rizo, Y. Tratamiento actual de la diabetes mellitus tipo 2. *Correo científico médico* **2016**, *20*, 98−121.

(11) Mayeux, R.; Sano, M. Treatment of alzheimer's disease. *N. Engl. J. Med.* **1999**, *341*, 1670−1679.

(12) Ray, D.; Kazan, H.; Chan, E. T.; Castillo, L. P.; Chaudhry, S.; Talukder, S.; Blencowe, B. J.; Morris, Q.; Hughes, T. R. Rapid and systematic analysis of the RNA recognition specificities of RNA-binding proteins. *Nat. Biotechnol.* **2009**, *27*, 667−670.

(13) Keene, J. D.; Komisarow, J. M.; Friedersdorf, M. B. RIP-chip: the isolation and identification of mRNAs, microRNAs and protein components of ribonucleoprotein complexes from cell extracts. *Nat. Protoc.* **2006**, *1*, 302−307.

(14) Bellucci, M.; Agostini, F.; Masin, M.; Tartaglia, G. G. Predicting protein associations with long noncoding RNAs. *Nat. Methods.* **2011**, *8*, 444−445.

(15) Agostini, F.; Zanzoni, A.; Klus, P.; Marchese, D.; Cirillo, D.; Tartaglia, G. G. CatRAPID omics: a web server for large-scale prediction of protein-RNA interactions. *Bioinformatics* **2013**, *29*, 2928−2930.

(16) Wang, Y.; Chen, X.; Liu, Z. P.; Huang, Q.; Wang, Y.; Xu, D.; Zhang, X. S.; Chen, R.; Chen, L. De novo prediction of RNA−protein interactions from sequence information. *Mol. BioSyst.* **2013**, *9*, 133−142.

(17) Lu, Q.; Ren, S.; Lu, M.; Zhang, Y.; Zhu, D.; Zhang, X.; Li, T. Computational prediction of associations between long non-coding RNAs and proteins. *BMC genomics* **2013**, *14*, 1−10.

(18) Liu, W.; Yang, Y.; Lu, X.; Fu, X.; Sun, R.; Yang, L.; Peng, L. NSRGRN: a network structure refinement method for gene regulatory network inference. *Brief. Bioinform.* **2023**, *24*, bbad129.

(19) Chen, X.; Xie, D.; Wang, L.; Zhao, Q.; You, Z.-H.; Liu, H. BNPMDA: bipartite network projection for miRNA−disease association prediction. *Bioinformatics* **2018**, *34*, 3178−3186.
(20) Chen, X.; Yan, C. C.; Zhang, X.; You, Z.-H.; Deng, L.; Liu, Y.; Zhang, Y.; Dai, Q. WBSMDA: within and between score for miRNA-disease association prediction. *Sci. Rep.* **2016**, *6*, 21106.
(21) Yu, B.; Wang, X.; Zhang, Y.; Gao, H.; Wang, Y.; Liu, Y.; Gao, X. RPI-MDLstack predicting RNA-protein interactions through deep learning with stacking strategy and LASSO. *Appl. Soft Comput.* **2022**, *120*, 108676.
(22) Pan, X.; Fan, Y. X.; Yan, J.; Shen, H. B. IPMiner: hidden ncRNA-protein interaction sequential pattern mining with stacked autoencoder for accurate computational prediction. *BMC genomics* **2016**, *17*, 1−14.
(23) Fan, X.-N.; Zhang, S.-W. LPI-BLS Predicting lncRNA−protein interactions with a broad learning system-based stacked ensemble classifier. *Neurocomputing* **2019**, *370*, 88−93.
(24) Alipanahi, B.; Delong, A.; Weirauch, M. T.; Frey, B. J. Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. *Nat. Biotechnol.* **2015**, *33*, 831−838.
(25) Jannasch, H. W.; Taylor, C. D. Deep-sea microbiology. *Annu. Rev. Microbiol.* **1984**, *38*, 487.
(26) Xu, J.; Xu, J.; Meng, Y.; Lu, C.; Cai, L.; Zeng, X.; Nussinov, R.; Cheng, F. Graph embedding and gaussian mixture variational autoencoder network for end-to-end analysis of single-cell RNA sequencing data. *Cells Rep. Methods.* **2023**, *3*, 100382.
(27) Wei, J.; Zhuo, L.; Pan, S.; Lian, X.; Yao, X.; Fu, X. Headtailtransfer: an efficient sampling method to improve the performance of graph neural network method in predicting sparse ncRNA−protein interactions. *Comput. Biol. Med.* **2023**, *157*, 106783.
(28) Zhang, S.; Xie, W.; Li, W.; Wang, L.; Feng, C. GAMB-GNN graph neural networks learning from gene structure relations and markov blanket ranking for cancer classification in microarray data. *Chemom. Intell. Lab. Syst.* **2023**, *232*, 104713.
(29) Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; Bengio, Y. Graph attention networks. *arXiv* 2017; arXiv 1710.10903.
(30) Kipf, T. N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv* 2016; arXiv 1609.02907.
(31) Shen, Z.-A.; Luo, T.; Zhou, Y.-K.; Yu, H.; Du, P.-F. NPI-GNN: predicting ncRNA−protein interactions with deep graph neural networks. *Brief. Bioinform.* **2021**, *22*, bbab051.
(32) Zhang, M.; Chen, Y. Link prediction based on graph neural networks. *Adv. Neural Inf. Process. Syst.* **2018**, *31*, 1.
(33) Holme, P.; Saramäki, J. Temporal networks. *Phys. Rep.* **2012**, *519*, 97−125.
(34) Divakaran, A.; Mohan, A. Temporal link prediction: a survey. *New Gener. Comput.* **2020**, *38*, 213−258.
(35) Vanschoren, J. Meta-learning. *Autom. Mach. Learn.: Methods, Syst., Chall.* **2019**, 35−61.
(36) Kazemi, S. M.; Goel, R.; Eghbali, S.; Ramanan, J.; Sahota, J.; Thakur, S.; Wu, S.; Smyth, C.; Poupart, P.; Brubaker, M. Time2vec: learning a vector representation of time. *arXiv* 2019; arXiv 1907.05321.
(37) Xu, D.; Ruan, C.; Korpeoglu, E.; Kumar, S.; Achan, K. Inductive representation learning on temporal graphs. *arXiv* 2020; arXiv 2002.07962.
(38) Pareja, A.; Domeniconi, G.; Chen, J.; Ma, T.; Suzumura, T.; Kanezashi, H.; Kaler, T.; Schardl, T.; Leiserson, C. Evolvegcn: evolving graph convolutional networks for dynamic graphs. *Assoc. Adv. Artif. Intell.* **2020**, *34*, 5363−5370.
(39) Hamilton, W.; Ying, Z.; Leskovec, J. Inductive representation learning on large graphs. *Adv. Neural Inf. Process. Syst.* **2017**, *30*.1
(40) Rossi, E.; Chamberlain, B.; Frasca, F.; Eynard, D.; Monti, F.; Bronstein, M. Temporal graph networks for deep learning on dynamic graphs. *arXiv* 2020; arXiv 2006.10637.

https://doi.org/10.1021/acs.jcim.3c01238
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX