NPI-RGCNAE: Fast Predicting ncRNA-Protein Interactions Using the Relational Graph Convolutional Network Auto-Encoder

Han Yu, Zi-Ang Shen, and Pu-Feng Du

***Abstract*—ncRNAs play important roles in a variety of biological processes by interacting with RNA-binding proteins. Therefore, identifying ncRNA-protein interactions is important to understanding the biological functions of ncRNAs. Since experimental methods to determine ncRNA-protein interactions are always costly and time-consuming, computational methods have been proposed as alternative approaches. We developed a novel method NPI-RGCNAE (predicting ncRNA-Protein Interactions by the Relational Graph Convolutional Network Auto-Encoder). With a reliable negative sample selection strategy, we applied the Relational Graph Convolutional Network encoder and the DistMult decoder to predict ncRNA-protein interactions in an accurate and efficient way. By using the 5-fold cross-validation, we found that our method achieved a comparable performance to all state-of-the-art methods. Our method requires less than 10% training time of all state-of-the-art methods. It is a more efficient choice with large datasets in practice.**

***Index Terms*—Bioinformatics, knowledge representation, network theory, neural networks, supervised learning.**

I. Introduction

NcRNAs refer to a kind of RNA with little or no ability to encode proteins. In the human genome, ncRNA genes account for a major part of all genomic DNA, while protein-coding genes occupy only 2% [1]. Originally, ncRNA genes were considered as "junks" or "dark matters" in the genome [2]. Recently, a number of studies on the ncRNAs have revealed that ncRNAs play critical roles in various biological processes [3]–[5]. Emerging evidences supported that ncRNAs are associated with various complex human diseases, such as cancers [6], [7], Alzheimer's disease [8], and diabetic nephropathy [9]. However, the biological functions of most ncRNAs are still largely unknown. Interacting with proteins is a common way for an ncRNA to perform its biological functions [10]. Therefore, determining ncRNA-protein interactions is an important step to the understanding of the ncRNA biological functions.

Experimental methods, such as RIP-Chip [11], HITS-CLIP [12], and RNAcompete [13], have been developed to identify ncRNA-protein interactions. These methods are always costly and time-consuming. Therefore, computational methods have been proposed as alternative approaches to predict ncRNA-protein interactions in recent years.

Computational methods for predicting ncRNA-protein interactions generally fall into two categories, the traditional machine learning-based methods, and the deep learning-based methods. Muppirala *et al.* proposed the RPISeq method based on the SVM (Support Vector Machine) and the RF (Random Forest) algorithms [14]. Wang *et al.* developed a method based on the naive Bayes classifier and sequence-based features [15]. Lu *et al.* designed lncPro by performing Fourier transformation on the physicochemical properties and secondary structures of RNA and protein sequences [16]. Suresh *et al.* proposed RPI-Pred, which utilized information from both sequences and structures of protein and RNA in the SVM model [17].

The above methods relied heavily on sequence-based statistical features. Several other methods explored a different way by incorporating the topological information of various types of biological networks. In general, the heterogeneous network may be composed of ncRNA-protein interactions network, protein-protein interactions network, protein-protein similarity network, or ncRNA-ncRNA similarity network. Li *et al.* proposed LPIHN that utilized the random walk method on a lncRNA-protein heterogeneous network [18]. The work of Yang *et al.* [19], the work of Zheng *et al.* [20], PLPIHS [21], and PLIPCOM [22] all calculated the HeteSim scores on the relevance paths of the heterogeneous network to extract topological information for ncRNA-protein interaction prediction. Zhang *et al.* proposed the PBLPI, which employed depth-first search on the heterogeneous network to predict novel RNA-protein interaction pairs [23].

Over the last few years, deep learning methods have been widely applied in predicting ncRNA-protein interactions from sequences. Several successful results have been achieved. Pan *et al.* proposed IPMiner to extract latent features from RNA and protein sequences based on the Stacked Auto-Encoder Networks [24]. DM-RPIs also used the Stacked Auto-Encoder Networks to extract sequence features for training SVM, RF, and CNN

Manuscript received July 2, 2021; revised September 21, 2021; accepted October 18, 2021. Date of publication October 26, 2021; date of current version April 13, 2022. This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 61872268 and in part by the National Key R&D Program of China under Grant 2018YFC0910405. *(Corresponding author: Pu-Feng Du.)*

The authors are with the College of Intelligence and Computing, Tianjin University, 300350 Tianjin, China (e-mail: yuhan122@tju.edu.cn; ziangshen@tju.edu.cn; pdu@tju.edu.cn).

Digital Object Identifier 10.1109/JBHI.2021.3122527

(Convolutional Neural Networks) classifiers [25]. LPI-Pred constructed the RNA2vec and Pro2vec models to generate RNA and protein embeddings as the input of the RF classifier [26]. Peng *et al.* proposed a hierarchical deep learning model that improved CTF (Conjoint Triad Feature) by adding structural features as supplements for training the models [27]. LPI-CNNCP used the copy-padding trick and high-order one-hot encoding to change protein and RNA sequences into the image-like form that was conveniently handled by CNN [28]. EDLMFC used the CNN and the bi-directional long short-term memory network (BLSTM) to extract useful information from multi-scale features, including sequence features, secondary structure features, and tertiary structure features [29]. All the above methods can automatically learn useful information from sequences. However, the topological information that naturally exists in the ncRNA-protein interaction network is still missing in constructing deep learning-based models. DeepLPI explored an effective way to integrate the sequence-based features and topological information of expression data of protein isoforms and lncRNAs by multimodal deep learning (MDL) and conditional random field (CRF) [30].

In addition, several recent studies successfully combined the Auto-Encoder method and the knowledge graph embeddings to solve bioinformatics problems. For example, Yao *et al.* used the Variational Graph Auto-Encoder to detect protein complexes [31]. Dai *et al.* proposed AAEs (Adversarial Auto-Encoders) to identify drug-drug interactions [32]. Wu *et al.* used a Graph Convolutional Network-based Auto-Encoder to infer the associations between lncRNAs and diseases [33].

The link prediction process using knowledge graph embeddings on a graph can be divided into three steps [34]. Firstly, define entities and relationships in the graph. Secondly, use a score function $f_r(h,t)$ to measure the probability that the entity h and the entity t have a relationship r. There is a variety of scoring functions, such as RESCAL [35], TransE (Translating Embeddings) [36], NTN (Neural Tensor Networks) [37], and DistMult [38]. Thirdly, learn the embeddings of relations and entities, which can maximize the probability of observed relationships in the graph.

In this paper, we proposed a method, namely NPI-RGCNAE (predicting ncRNA-Protein Interactions by Relational Graph Convolutional Network Auto-Encoder), to predict the ncRNA-protein interactions on the ncRNA-protein heterogeneous network. All datasets, source codes and supplementary files of NPI-RGCNAE have been deposited in a public Github repository (https://github.com/Angelia0hh/NPI-RGCNAE). Our model aims at improving prediction performances in two directions. Firstly, effectively and efficiently extract topological information from ncRNA-protein interaction network for novel ncRNA-protein interaction prediction. We achieved this by using the R-GCN [39], [40] encoder and the DistMult [38] decoder, which can effectively aggregate the topological information within a much shorter training procedure. Secondly, generate reliable negative samples to improve prediction performances. In most cases, the relationship between a random pair of ncRNA and protein is unknown. Although most existing methods generate negative samples by randomly pairing ncRNAs and proteins[14], [24], [41], it should be noted that this may count potentially interacting pairs of ncRNAs and proteins as non-interacting pairs in the training dataset. We avoided this problem by using the negatives selection method in [28] to obtain reliable negative samples. By evaluating our methods on different datasets, we find that our predictor can achieve a comparable prediction performance to the state-of-the-art methods with less than 10% training time cost.

TABLE I
FOUR BENCHMARKING DATASETS USED IN OUR STUDY

Dataset	Positive samples	Negative samples[a]	RNAs	Proteins
RPI2241	2241	0	841	2042
RPI369	369	0	331	338
NPInter10412	10412	0	4636	449
RPI7317	7317	7317	1874	118

[a] Only positive samples of original datasets were used in our experiments. Negative samples were generated by the method in the "Negative Sample Selection" section.

II. MATERIALS AND METHODS

In this paper, we proposed a method NPI-RGCNAE. The whole process of this work is as follows. First, we selected a set of reliable negative samples according to the method in the LPI-CNNCP study [28]. Next, we used the R-GCN encoder to learn ncRNA and protein node embeddings together. Finally, node embeddings of each pair of ncRNA and protein were input into the DistMult [38] decoder to calculate interaction scores.

When we used the 5-fold cross-validation, we divided the dataset according to the number of edges rather than that of nodes. Four-fifths of edges in the original dataset took participate in the training process. The loss between true labels and predictions was backpropagated to guide the generation of node embeddings. During the validation process, the rest one-fifth samples were input into the optimized model to make final predictions. The flow chart of our model is illustrated in Fig. 1.

A. Benchmarking Datasets

In this paper, we took four commonly used datasets as our benchmarking datasets. These datasets are the RPI2241 [14], RPI369 [14], NPInter10412 [42], and RPI7317 [43]. The RPI2241 and the RPI369 datasets were extracted from PRIDB (Protein-RNA Interface Database) [44]. Complexes in the PRIDB with a least atom distance less than 8 angstroms were considered as interaction pairs. The NPInter10412 dataset was curated from the NPInter2.0 database [42], which records experimentally verified interactions between ncRNAs and proteins. The RPI7317 dataset contains 7317 human lncRNA-protein interaction pairs, which were extracted from the NPInter3.0 database [45], as well as 7317 negative samples, which were provided by literature [43]. The details of these datasets are recorded in Table I.

B. Negative Sample Selection

Although many efforts have been made to systematically detect as many ncRNA-protein interactions as possible, we must admit that the records in the public databases may only account

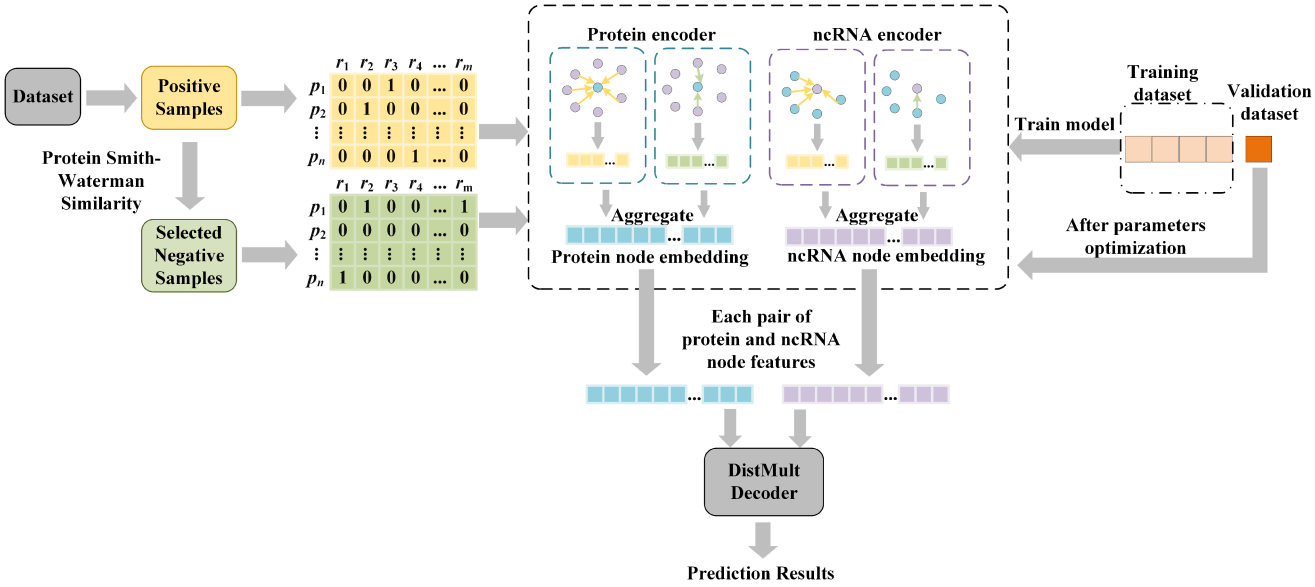

Fig. 1. The flow chart of our NPI-RGCNAE model. Firstly, negative samples were generated according to the interaction scores between each pair of ncRNA and protein. Secondly, positive relation bipartite graph and negative relation bipartite graph were constructed respectively. Thirdly, the R-GCN was used as the encoder to learn protein and RNA node embeddings. For the protein encoder, the input graph was a directed graph, in which all edges pointed from a protein node to an ncRNA node. For the ncRNA encoder, the input graph was a directed graph with the opposite direction to the graph of protein encoder. The information from the neighborhood was aggregated. Features from different relationships were aggregated. Finally, the node features of each pair of protein and ncRNA were fed into the DistMult decoder to predict the relationship between them.

for a small fraction of all ncRNA-protein interactions in nature. Constructing negative samples by randomly paring ncRNAs and proteins is very risky, as it is possible to include unknown interacting pairs as negative samples, which will eventually damage the prediction performances [49]. This may also be the reason why several works chose to use semi-supervised methods rather than supervised methods [50]–[53].

In this work, we chose to use a different strategy to select more reliable negative samples, which was originally introduced by Zhang *et al.* in the LPI-CNNCP study [28].

Given m ncRNAs and n proteins, which can be noted as r_1, r_2, …, r_m and p_1, p_2, …, p_n, respectively. We first calculated the sequence similarity between p_i and p_j, as follows:

$$g_1(p_i, p_j) = \frac{s(p_i, p_j)}{\sqrt{s(p_i, p_i)\, s(p_j, p_j)}}, \tag{1}$$

where $s(.,.)$ is the Smith-Waterman [54] bit-score between two protein sequences.

We calculated the interaction score $g_2(p_i, r_j)$ between protein p_i and RNA r_j based on the known interaction pairs and protein similarities, as follows:

$$g_2(p_i, r_j) = \begin{cases} 1 & p_i \text{ interacts with } r_j \\ \sum_{k=1, k\neq i}^{n} \delta_{k,j} g_1(p_i, p_k) & \text{otherwise} \end{cases}, \tag{2}$$

where

$$\delta_{k,j} = \begin{cases} 1 & p_k \text{ interacts with } r_j \\ 0 & \text{otherwise} \end{cases}. \tag{3}$$

Finally, we sorted the interaction scores of all pairs in an ascending order. Negative samples were selected sequentially from the head of this ascending sorted list with the same number as positives. Although RPI7317 has negative samples, we did not use the original negative samples by randomly pairing. Negative samples of all datasets were generated as above.

C. Node Embeddings

We used the R-GCN [39] to learn the node embeddings. The positive interactions are recorded as a binary matrix $\mathbf{A}_+ = \{a_+(i, j)\}_{n\times m}$, while the negative pairs are noted also as a binary matrix $\mathbf{A}_- = \{a_-(i,j)\}_{n\times m}$, where $a_+(i,j)\in\{0, 1\}$ and $a_-(i,j)\in\{0, 1\}$ are the j-th column of the i-th row of the corresponding matrix. In the $\mathbf{A}_+$ matrix, 1 indicates that the corresponding protein and ncRNA pair interacts. In the $\mathbf{A}_-$ matrix, 1 indicates that the corresponding protein and ncRNA pair does not interact. In addition to existing interacting pairs and selected non-interacting pairs, a number of ncRNA-protein pairs remain unknown. They could be potentially interacting pairs, so it is not appropriate to treat them all as non-interacting pairs. They are marked as 0 in both matrices, as these pairs serve as neither positives nor negatives in our work. They will not participate in the training process.

Given a matrix, the operator Δ converts the binary matrix to a diagonal matrix, representing the row-wise summations. Without losing generality, we take $\mathbf{A}_+$ as an example. The operator Δ can be defined as the follows:

$$\Delta(\mathbf{A}_+) = diag\left(\sum_{j=1}^{m} a_+(1, j) \sum_{j=1}^{m} a_+(2, j) \cdots \sum_{j=1}^{m} a_+(n, j)\right). \tag{4}$$

For convenience, we also define the following notations:

$$\mathbf{A}_{p+} = \mathbf{A}_{+}, \tag{5}$$

$$\mathbf{A}_{p-} = \mathbf{A}_{-}, \tag{6}$$

$$\mathbf{A}_{r+} = \mathbf{A}_{+}^{T}, \tag{7}$$

$$\mathbf{A}_{r-} = \mathbf{A}_{-}^{T}, \tag{8}$$

$$\mathbf{D}_{p+} = \Delta\left(\mathbf{A}_{p+}\right), \tag{9}$$

$$\mathbf{D}_{p-} = \Delta\left(\mathbf{A}_{p-}\right), \tag{10}$$

$$\mathbf{D}_{r+} = \Delta\left(\mathbf{A}_{r+}\right), \text{and} \tag{11}$$

$$\mathbf{D}_{r-} = \Delta\left(\mathbf{A}_{r-}\right), \tag{12}$$

where $\mathbf{A}_{p+}$, $\mathbf{A}_{p-}$, $\mathbf{D}_{p+}$, and $\mathbf{D}_{p-}$ are used to create node embeddings for proteins, while $\mathbf{A}_{r+}$, $\mathbf{A}_{r-}$, $\mathbf{D}_{r+}$ and $\mathbf{D}_{r-}$ for ncRNAs.

The R-GCN has two stages in its information aggregation process. The first stage is to aggregate information from neighbor nodes. The second stage is to aggregate information from different relationships.

Let d be the desired dimensions of node embedding after (t-1) layers of the R-GCN. The R-GCN uses the following iterative process to aggregate neighboring information:

$$\mathbf{H}_{p+}(t) = \mathbf{D}_{p+}^{-1}\mathbf{A}_{p+}\mathbf{H}_{r}(t-1)\,\mathbf{W}_{+}(t), \tag{13}$$

$$\mathbf{H}_{p-}(t) = \mathbf{D}_{p-}^{-1}\mathbf{A}_{p-}\mathbf{H}_{r}(t-1)\,\mathbf{W}_{-}(t), \tag{14}$$

$$\mathbf{H}_{r+}(t) = \mathbf{D}_{r+}^{-1}\mathbf{A}_{r+}\mathbf{H}_{p}(t-1)\,\mathbf{W}_{+}(t), \text{and} \tag{15}$$

$$\mathbf{H}_{r-}(t) = \mathbf{D}_{r-}^{-1}\mathbf{A}_{r-}\mathbf{H}_{p}(t-1)\,\mathbf{W}_{-}(t), \tag{16}$$

where $\mathbf{H}_{p+}(t)\in\mathbf{R}^{n\times d/4}$, $\mathbf{H}_{p-}(t)\in\mathbf{R}^{n\times d/4}$, $\mathbf{H}_{r+}(t)\in\mathbf{R}^{m\times d/4}$ and $\mathbf{H}_{r-}(t)\in\mathbf{R}^{m\times d/4}$ represent the t-th layer node feature matrices of proteins and ncRNAs in positive and negative relationships respectively, $\mathbf{W}_{+}(t)\in\mathbf{R}^{d\times d/4}$and $\mathbf{W}_{-}(t)\in\mathbf{R}^{d\times d/4}$ the corresponding transformation matrices in the t-th layer, and $\mathbf{H}_{p}(t\text{-}1)\in\mathbf{R}^{n\times d}$ and $\mathbf{H}_{r}(t\text{-}1)\in\mathbf{R}^{m\times d}$ the aggregated feature embedding matrices of the (t-1)-th layer. The aggregated embedding matrices $\mathbf{H}_{p}(t)\in\mathbf{R}^{n\times d/2}$ and $\mathbf{H}_{r}(t)\in\mathbf{R}^{m\times d/2}$ of the t-th layer can be defined by the following iterative process:

$$\mathbf{H}_{p}(t) = \sigma\left[\psi\left(\mathbf{H}_{p+}(t), \mathbf{H}_{p-}(t)\right) + \mathbf{H}_{p}(t-1)\,\mathbf{W}_{0}(t)\right], \text{and} \tag{17}$$

$$\mathbf{H}_{r}(t) = \sigma\left[\psi\left(\mathbf{H}_{r+}(t), \mathbf{H}_{r-}(t)\right) + \mathbf{H}_{r}(t-1)\,\mathbf{W}_{0}(t)\right], \tag{18}$$

where $\mathbf{W}_{0}(t)\in\mathbf{R}^{d\times d/2}$ is the parameter matrix for the self-loop [39], ψ(.,.) the column-wise concatenation operator on matrices, and σ(.) the Rectified Linear Unit (ReLU) activation function.

The final node feature matrices $\mathbf{Z}_p$ and $\mathbf{Z}_r$ can be computed as follows:

$$\mathbf{Z}_{p} = \sigma(\mathbf{H}_{p}(t)\mathbf{W}_{p}), \text{and} \tag{19}$$

$$\mathbf{Z}_{r} = \sigma(\mathbf{H}_{r}(t)\mathbf{W}_{r}), \tag{20}$$

where $\mathbf{W}_p$ and $\mathbf{W}_r$ are two fully connected layers for protein and ncRNA node embeddings, respectively.

D. DistMult Decoder

During the training process, the DistMult [38] decoder is used to reconstruct the observed relationships to obtain effective node embeddings. During the validating process, it is used to predict unknown relationships.

Let $\mathbf{Z}_r(i)$ and $\mathbf{Z}_p(j)$ denote the node features of ncRNA r_i and protein p_j, respectively. The DistMult decoder function for the positive and negative relationships can be defined as:

$$f_{+}\left(\mathbf{Z}_{r}(i), \mathbf{Z}_{p}(j)\right) = \mathbf{Z}_{r}(i)\,\mathbf{Q}_{+}\mathbf{Z}_{p}^{T}(j), \text{and} \tag{21}$$

$$f_{-}\left(\mathbf{Z}_{r}(i), \mathbf{Z}_{p}(j)\right) = \mathbf{Z}_{r}(i)\,\mathbf{Q}_{-}\mathbf{Z}_{p}^{T}(j), \tag{22}$$

where $\mathbf{Q}_+$ and $\mathbf{Q}_-$ are the trainable diagonal matrices of the corresponding type of relationships.

By stitching all values of f_+ from different i and j, we can have a vector $\mathbf{s}_+$, while another vector $\mathbf{s}_-$ for f_-, as follows:

$$\mathbf{s}_{+} = \left[s_{+}(1)\ s_{+}(2)\ \cdots\ s_{+}(w)\right]^{T}, \text{and} \tag{23}$$

$$\mathbf{s}_{-} = \left[s_{-}(1)\ s_{-}(2)\ \cdots\ s_{-}(w)\right]^{T}, \tag{24}$$

where

$$s_{+}(k) = f_{+}\left(\mathbf{Z}_{r}(i), \mathbf{Z}_{p}(j)\right), \text{and} \tag{25}$$

$$s_{-}(k) = f_{-}\left(\mathbf{Z}_{r}(i), \mathbf{Z}_{p}(j)\right), \tag{26}$$

w the total number of ncRNA-protein pairs involved in the training or validating process, and k the index of the k-th ncRNA-protein pair in the training dataset or validation dataset.

A score matrix $\mathbf{S}$ can be defined as:

$$\mathbf{S} = \left[\mathbf{s}_{+}\ \mathbf{s}_{-}\right]. \tag{27}$$

The probability matrix $\mathbf{S}$' can be calculated as follows:

$$\mathbf{S}' = \sigma\left(\mathbf{S}\mathbf{W}_{c}\right), \tag{28}$$

where $\mathbf{W}_c\in\mathbf{R}^{2\times2}$ is a transformation matrix. The values in $\mathbf{S}$' were further normalized by a softmax function, as follows:

$$p_{+}(k) = \frac{\exp\left[s'_{+}(k)\right]}{\exp\left[s'_{+}(k)\right] + \exp\left[s'_{-}(k)\right]}, \text{and} \tag{29}$$

$$p_{-}(k) = \frac{\exp\left[s'_{-}(k)\right]}{\exp\left[s'_{+}(k)\right] + \exp\left[s'_{-}(k)\right]}, \tag{30}$$

where $s'_{+}(k)$ and $s'_{-}(k)$ are the k-th row of the positive and negative column in the matrix $\mathbf{S}'$, respectively. If $p_{+}(k) > p_{-}(k)$, the k-th protein and ncRNA pair is predicted as interacting, otherwise, non-interacting.

E. Model Training and Parameter Calibrations

To train the model, we chose cross-entropy as the loss function and used the back-propagation algorithm [55] to minimize this loss. Only observed edges participated in the optimization process. The objective is to minimize the following loss function:

$$L = -\sum_{k}\left(I_{+}(k)\log p_{+}(k) + I_{-}(k)\log p_{-}(k)\right), \tag{31}$$

where the k-th pair is an observed interacting pair or non-interacting pair in the matrix $\mathbf{A}_+$ or $\mathbf{A}_-$, and

$$I_+(k) = \begin{cases} 1 & \text{The } k-\text{th pair is observed in } \mathbf{A}_+ \\ 0 & \text{otherwise} \end{cases}, \text{ and} \quad (32)$$

$$I_-(k) = \begin{cases} 1 & \text{The } k-\text{th pair is observed in } \mathbf{A}_- \\ 0 & \text{otherwise} \end{cases}. \quad (33)$$

We implemented NPI-RGCNAE with Pytorch 1.7.0. During the training process, we applied the node dropout trick [56] and L2 regularization to avoid overfitting and improve the robustness of the model. We chose the full batch to learn and the Adam [57] optimizer to train our model. Adam needs a learning rate and an L2 regularization penalty factor for weight decay. We used the StepLR strategy to adjust the learning rate. The learning rate will be updated after the number of *step_size* epochs as follows:

$$new_{lr} = initial_{lr} \times \gamma^{epoch//step\_size}, \quad (34)$$

where // represents the operation of division and rounding down, $initial_{lr}$ the initial learning rate, new_{lr} the learning rate after updating, *epoch* the times of forward-propagation and back-propagation on all training samples, and γ the multiplication factor.

To summarize, the hyper-parameters involved in our model include the dropout ratio, the initial learning rate, the weight decay factor, *step_size*, γ, and epochs. Considering that different datasets vary in scale, we used different hyper-parameters on different datasets. We listed the hyper-parameters of our model, IPMiner, and RPITER in the Supplementary File Table S1, Table S2 and Table S3, respectively.

F. Performance Measures

Four statistics, including accuracy (*ACC*), sensitivity (*Sn*), specificity (*Sp*), and Matthews Correlation Coefficient (*MCC*), were employed to measure the prediction performance of our model, as well as the Area Under the Receiver Operating Characteristic curve (AUROC). These statistics can be defined as follows:

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}, \quad (35)$$

$$Sn = \frac{TP}{TP + FN}, \quad (36)$$

$$Sp = \frac{TN}{TN + FP}, \text{ and} \quad (37)$$

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}, \quad (38)$$

where *TP*, *TN*, *FP*, and *FN* are the numbers of true positives, true negatives, false positives, and false negatives respectively in the 5-fold cross-validation.

III. Results

We first compared our method with the state-of-the-art sequence-based deep learning methods, IPMiner [24] and RPITER [27]. Both methods utilized sequence-based statistical features. We performed 5-fold cross-validation on the RPI2241, RPI369, NPInter10412, and RPI7317 datasets.

TABLE II
COMPARISON BETWEEN OUR METHOD AND OTHER SEQUENCE-BASED DEEP LEARNING METHODS ON FOUR BENCHMARKING DATASETS

Dataset	Method	*ACC*[a](%)	*Sn*[b](%)	*Sp*[c](%)	*MCC*[d](%)	*AUROC*[e](%)
RPI2241	NPI-RGCNAE	96.7±0.3	96.5±0.6	96.8±0.9	93.3±0.7	98.9±0.1
	IPMiner	98.5±0.1	98.2±0.1	98.8±0.1	97.0±0.1	98.5±0.1
	RPITER	**99.1±0.1**[f]	**98.7±0.1**	**99.4±0.1**	**98.2±0.1**	**99.9±0.0**
RPI369	NPI-RGCNAE	**94.0±0.5**	92.0±0.8	**95.9±0.9**	**88.0±1.0**	96.2±0.4
	IPMiner	**94.0±0.4**	**92.9±0.4**	95.1±0.5	**88.0±0.8**	94.0±0.4
	RPITER	88.6±7.4	90.0±9.1	87.2±9.9	77.5±14.8	**99.2±0.2**
NPInter 10412	NPI-RGCNAE	98.0±0.1	97.5±0.3	98.6±0.4	96.0±0.2	99.7±0.0
	IPMiner	99.4±0.2	**99.9±0.0**	99.0±0.5	98.9±0.5	99.4±0.2
	RPITER	**99.8±0.0**	99.7±0.0	**99.9±0.0**	**99.6±0.0**	**99.9±0.0**
RPI7317	NPI-RGCNAE	98.6±0.2	98.2±0.3	99.1±0.2	97.3±0.3	99.6±0.1
	IPMiner	**99.5±0.1**	**99.8±0.1**	**99.2±0.1**	99.0±0.2	99.5±0.1
	RPITER	99.3±0.0	**99.8±0.1**	98.6±0.1	**99.9±0.0**	**99.8±0.1**

[a] *ACC* stands for accuracy.
[b] *Sn* stands for sensitivity.
[c] *Sp* stands for specificity.
[d] *MCC* stands for Matthews Correlation Coefficient.
[e] *AUROC* stands for the Area Under the ROC curve.
[f] The boldface indicates the best performance metric on each dataset among different methods. Results are displayed in the form of "mean±standard deviation".

RPITER used the RNAfold program in the ViennaRNA Package version 2.4.3 [58] to predict RNA secondary structures, but the RNAfold can only deal with RNAs with a length shorter than 10000 nt. We discarded five RNAs in the RPI7317 dataset, which are longer than 10000 nt.

The performance values in Table II are average values of 10 times 5-fold cross-validations. Our model achieved slightly lower, but yet comparable, performances on all datasets to all state-of-the-art methods. Particularly, on the RPI369 dataset, the performances of all methods had decreased. But our method still obtained the best *ACC* of 94.0%, the best *Sp* of 95.9%, and the best *MCC* of 88.0% among the three methods.

All computational methods in the comparison were re-trained and evaluated on identical datasets to make a fair comparison. Negative samples were generated as described in the "Negative Sample Selection" section. It should be noted that the performance values of other state-of-the-art methods are much higher than their original reports. This is the result of our negative sample selection method. This observation indicates that our negative sample selection method is not only effective for our model but also works for other models.

Although the prediction accuracy of our method seems to be slightly lower on three datasets, our method has a great advantage in training efficiency. To compare the training efficiency between different sequence-based deep learning methods, we implemented IPMiner [24], RPITER [27], as well as our method, on a workstation with 32GB memory and a Xeon 4-core processor. We compared the training time of our method to the other two methods in Fig. 2. On the NPInter10412 dataset, the

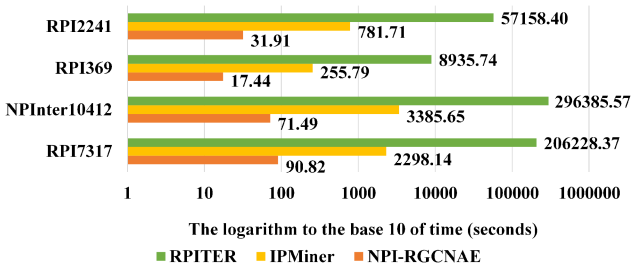

Fig. 2. Comparison of training time between our method and sequence-based deep learning methods on four benchmarking datasets. The X-axis is the logarithm to the base 10 of time in seconds. Green, yellow, and orange bars represent the running time of RPITER, IPMiner, and NPI-RGCNAE, respectively. The original RPITER used the TensorFlow-GPU to train the model. Restricted by the machine configuration, we used the TensorFlow of CPU to run RPITER.

TABLE III
COMPARISON BETWEEN OUR METHOD AND OTHER NETWORK-BASED METHODS ON THE NPINTER4158 DATASET BY 5-FOLD CROSS-VALIDATION

Method	*Sn*[a](%)	*AUROC*[b](%)
NPI-RGCNAE	**92.5**[c]	**98.1**
PBLPI	62.3	92.4
RWR	35.4	83.3
LPBNI	41.4	85.9

[a] Sn stands for sensitivity.
[b] AUROC stands for the Area Under the ROC curve.
[c] We used boldface to indicate the best performance metric among different models.

RPITER took over 3 days to finish the training process. IPMiner also required nearly an hour. However, our method required only 71.49 seconds. In addition, our method required only 31.91 seconds on the RPI2241 dataset, 17.44 seconds on the RPI369 dataset, and 90.82 seconds on the RPI7317 dataset, which were all less than 1% of the training time of RPITER, and 10% of IPMiner. Therefore, by sacrificing less than 3% prediction accuracy, we gained 10∼100 times efficiency.

In addition to the sequence-based deep learning method, the network-based semi-supervised method is also a kind of popular method to predict ncRNA-protein interactions. So we compared our model with other network-based methods, including PBLPI [23], RWR [18], and LPBNI [59]. Due to the difficulties in retraining these methods and limited time for this study, we took the results in literature [23] directly for comparison. Since literature [23] only reported performances on the NPInter4158 dataset, we also evaluated our method on this dataset.

The NPInter4158 dataset contains only positive samples. Therefore, we generated the negative samples as in the "Negative Sample Selection" section. Due to nature of semi-supervised methods, PBLPI, RWR, and LPBNI take all unknown ncRNA-protein pairs as non-interacting pairs, resulting in a highly-imbalanced working dataset. Their performances have a high specificity and an over-estimated accuracy. The negative samples for the comparing methods and ours are different. To give a fair-enough comparison, only *AUROC*, which is a balanced performance measure, and *Sn*, which only concerns positive samples, were used to measure the performances in Table III.

TABLE IV
RESULTS OF INDEPENDENT TEST ON NPINTER10412 AND RPI7317

Dataset	*ACC*[a](%)	*Sn*[b](%)	*Sp*[c](%)	*MCC*[d](%)	*AUROC*[e](%)
NPInter10412	98.2±0.3[f]	97.7±0.5	98.6±0.6	96.3±0.7	99.8±0.0
RPI7317	98.8±0.2	98.4±0.3	99.3±0.2	97.7±0.3	99.7±0.1

[a] *ACC* stands for accuracy.
[b] *Sn* stands for sensitivity.
[c] Sp stands for specificity.
[d] MCC stands for Matthews Correlation Coefficient.
[e] AUROC stands for the Area Under the ROC curve.
[f] Results are displayed in the form of "mean±standard deviation".

Our method obtained 92.5% *Sn* and 98.1% *AUROC*, which outperformed all three existing methods. The values in Table III for our method were obtained by averaging 10 times 5-fold cross-validations on the NPInter4158 dataset.

To further evaluate the performance of NPI-RGCNAE in predicting ncRNA-protein interactions, we performed an independent dataset test on the NPInter10412 and the RPI7317, as they are large enough to be divided into training and independent testing datasets.

We randomly divided the NPInter10412 and the RPI7317 datasets into two parts with the 8:2 ratio, respectively. We trained our model with the 80% data and tested it with the 20% data. There is no overlap between the training and the testing datasets. All testing samples are set to be unknown before training, and therefore would never be involved in the training procedures. We listed the average results of 10 times independent tests in Table IV. The prediction performances of independent dataset test are similar to the cross-validation results, which indicates that NPI-RGCNAE is not over-optimized.

IV. DISCUSSION

A. Sequence-Based Features are Ignorable

Sequence-based features, like *k*-mer frequencies, can be added to the R-GCN based model as side information. However, we found that this kind of side information is ignorable in our method. We compared the performances of our model with and without *k*-mer frequencies as side information. We performed 10 times 5-fold cross-validations on four benchmarking datasets. As in Fig. 3, the performance differences are minimal and can be ignored for simplicity of the whole algorithm. The details about generating and integrating *k*-mer frequencies can be found in Supplementary File 1. This observation implied that if the network information is well aggregated, node embeddings solely can work well in predicting interactions without sequence information.

B. Comparison Between Different Numbers of R-GCN Layers

We further explored the effect of different numbers of R-GCN layers in our model. We applied the R-GCN with different numbers of layers, including 1, 2, 3, and 4 layers. The prediction performances on four benchmarking datasets were recorded in Fig. 4. For each case, 10 times 5-fold cross-validations were performed. On the RPI2241 dataset, as the number of layers

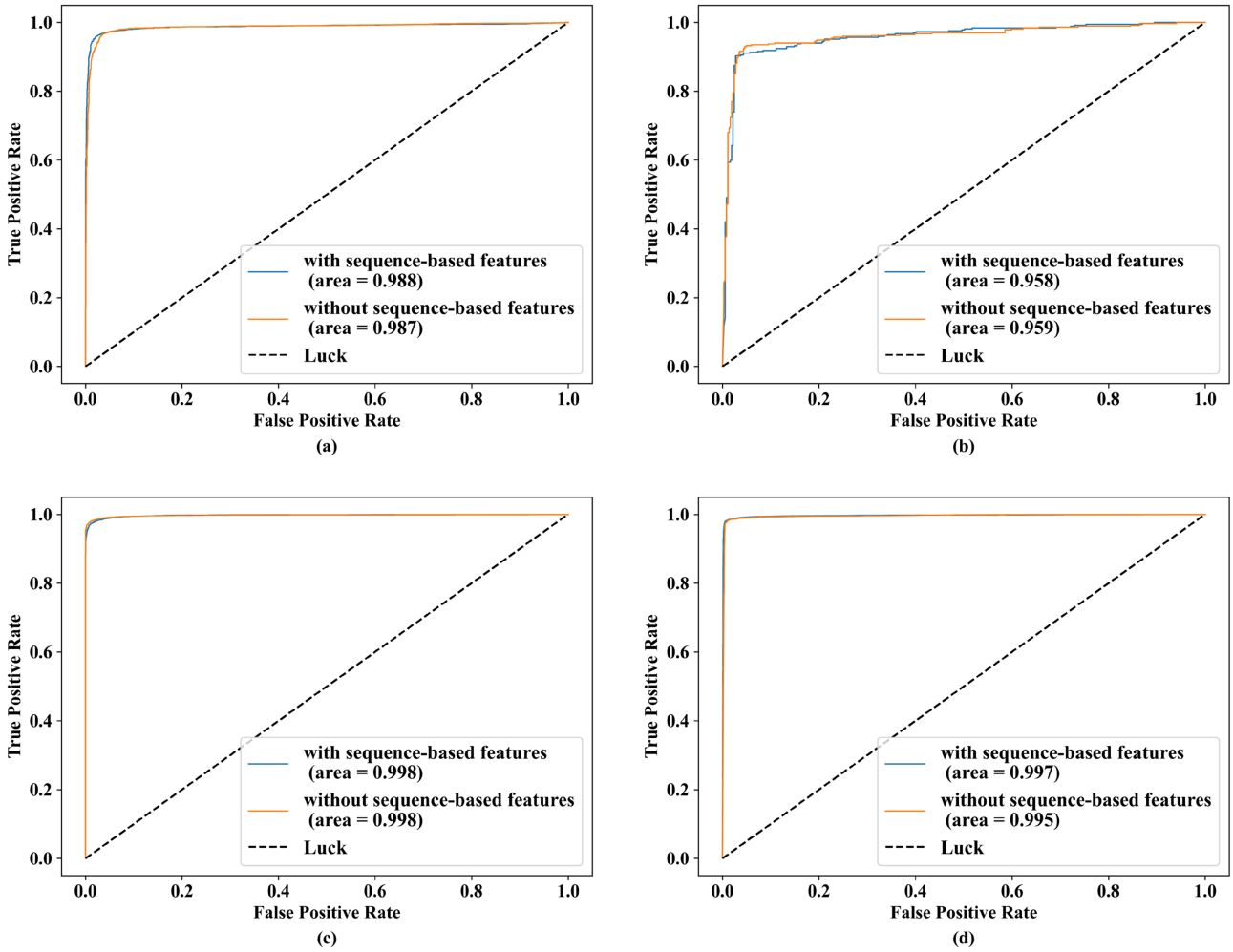

Fig. 3. The AUROC curve of different node features on four benchmarking datasets. "with sequence-based features" or "without sequence-based features" means whether sequence-based features are considered as a part of final node features. (a) The AUROC curve of different node features on the RPI2241 dataset; (b) The AUROC curve of different node features on the RPI369 dataset; (c) The AUROC curve of different node features on the NPInter10412 dataset; (d) The AUROC curve of different node features on the RPI7317 dataset.

increased, *ACC*, *Sn*, and *MCC* were decreased, while *Sp* was slightly increased, and *AUROC* remained almost unchanged. On the RPI369, *ACC*, *MCC*, and *AUROC* were generally decreased. Although some fluctuations of *Sn* and *Sp* can be observed they also gave a down-fall trend. On the NPInter10412, the performances of all metrics decreased generally or remained almost unchanged. On the RPI7317, all metrics were only slightly improved. Therefore, increasing the number of layers does not bring sufficient advantage to our model. One layer of R-GCN was used finally.

C. The Effectiveness of the Negative Sample Selection

To validate the effectiveness of the negative sample selection, we generated negative samples using two other strategies. The default strategy is the one that is described in the "Negative Sample Selection" section, which is called the "sort" strategy. The second strategy is composed by three steps: (1) Sort interaction scores g_2 in an ascending order. (2) Pick negative samples sequentially from the head of the list in an ascending order with twice the number of positive samples. (3) Randomly select half of the negative samples as the final negative sample set. This strategy is named as the "sort_random". The third one generates the negative sample set by randomly pairing RNAs and proteins, with exceptions on known interacting pairs. This strategy is called the "random".

We evaluated the prediction performance of our method using the above three different negative sample selection strategies. Ten times 5-fold cross-validations were performed on every dataset with every strategy, respectively. The results are compared in Fig. 5.

Generally, our model performed better with the "sort" or the "sort_random" strategies than with the "random" strategy. For large datasets, like NPInter10412 and RPI7317, the increments of performance values are not significant. However, for the relatively smaller datasets, like RPI2241 and RPI369, the "sort" strategy almost doubled the prediction performances of the "random" strategy. There was a huge gap in all performance indicators between the "random" strategy and the "sort" strategy. That is probably because our model relies heavily on the

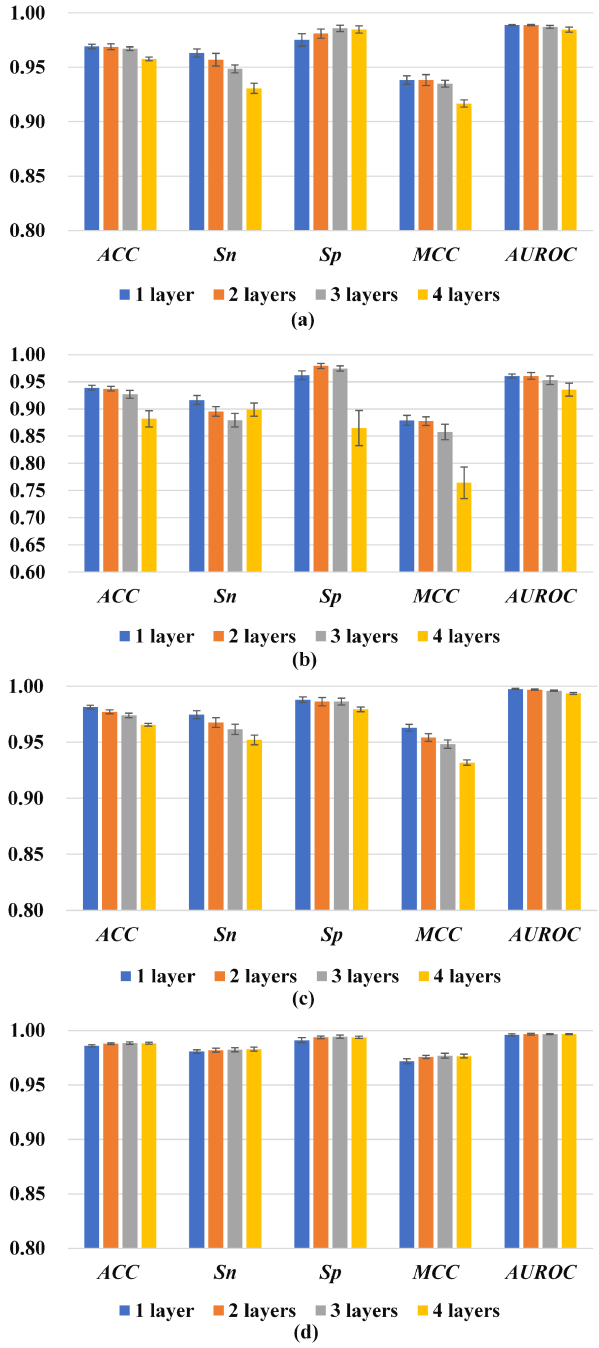

Fig. 4. Performances of different numbers of R-GCN layers. We compared the performances of our model when the number of layers varies from 1 to 4. (a) Performances of different numbers of R-GCN layers on the RPI2241 dataset; (b) Performances of different numbers of R-GCN layers on the RPI369 dataset; (c) Performances of different numbers of R-GCN layers on the NPInter10412 dataset; (d) Performances of different numbers of R-GCN layers on the RPI7317 dataset.

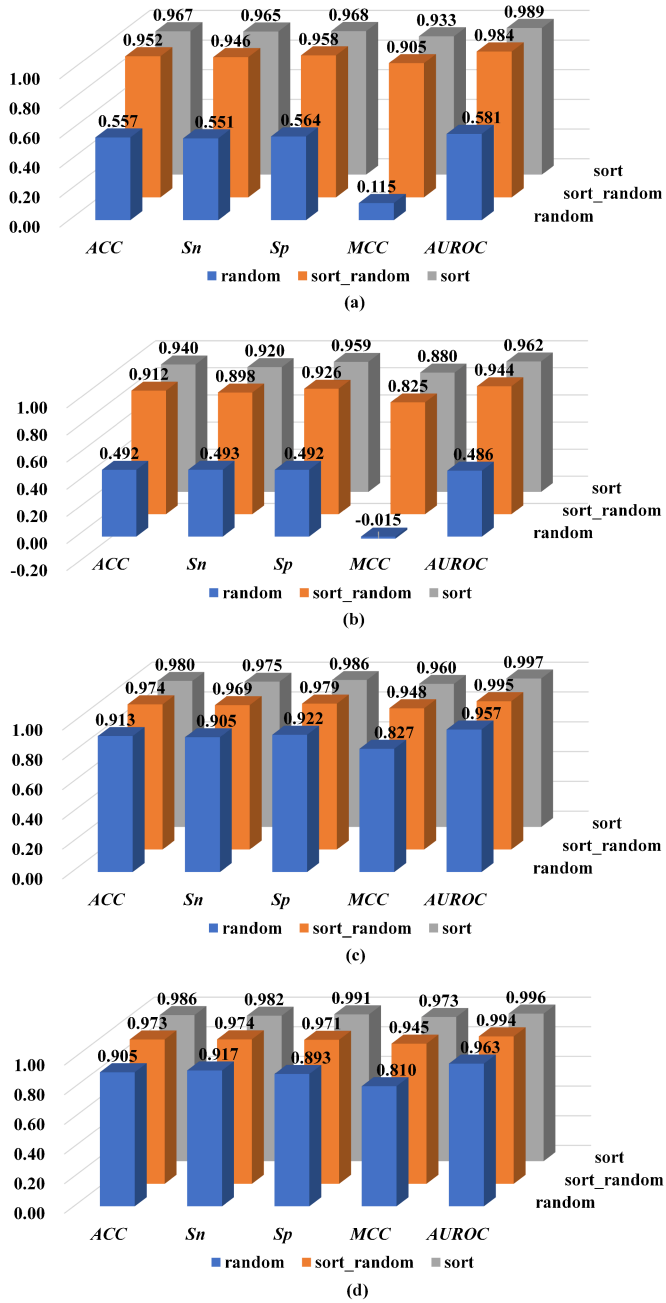

Fig. 5. Performances of different negative sample selection methods of our method. Blue, orange, and gray pillars respectively represent the performances of the "random", "sort_random" and "sort" negative sample selection strategies. (a) Performances of different negative sample selection strategies on the RPI2241 dataset; (b) Performances of different negative sample selection strategies on the RPI369 dataset; (c) Performances of different negative sample selection strategies on the NPInter10412 dataset; (d) Performances of different negative sample selection strategies on the RPI7317 dataset.

neighborhood information, while the "sort" strategy happens to preserve much more neighborhood information than the "random" strategy.

To quantitively investigate the relationship between the neighborhoods and the prediction performances, we defined the "average neighbors" indicator. This indicator is the average number of directly connected neighbors of each ncRNA-protein pair in the graph.

As illustrated in Fig. 6, the accuracy with the "random" strategy on different datasets has a linear correlation with the logarithm to the base 10 of the "average neighbors". The correlation coefficient R^2 is 0.988. Therefore, the more first-order

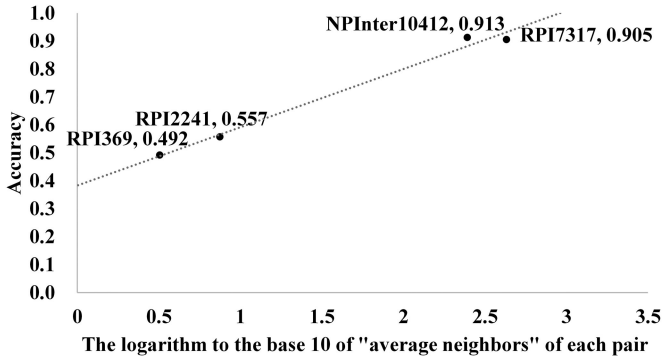

Fig. 6. The linear relationship between the average accuracy of 10 times 5-fold cross-validations of our model and the logarithm to the base 10 of "average neighbors" on datasets with the "random" strategy.

TABLE V
AVERAGE NEIGHBORS OF EACH PAIR AND ACCURACY ON DIFFERENT DATASETS

Dataset	The negative sample selection method	Average neighbors	*ACC*[a](%)
RPI2241	random	7	55.7
	sort_random	37	95.2
	sort	46	96.7
RPI369	random	3	49.2
	sort_random	14	91.2
	sort	18	94.0
NPInter10412	random	246	91.3
	sort_random	552	97.4
	sort	857	98.0
RPI7317	random	428	90.5
	sort_random	474	97.3
	sort	530	98.6

[a]*ACC* stands for accuracy. Results are the average of 10 times 5-fold validations.

neighbors for each pair, the better performance would be achieved.

We listed the value of "average neighbors" and the prediction accuracy on each dataset with different strategies in Table V. We can find that the "average neighbors" on the RPI369 and the RPI2241 datasets with the "random" strategy are 3 and 7, respectively. This may explain why our model has lower performances on these two datasets than the other two. For each dataset, the "sort" strategy always achieves the best performances with the highest "average neighbors". This observation implied that the "sort" strategy may help GNN (Graph Neural Networks) to better learn hidden topological information for link predictions. In addition, the performance differences between the "sort" strategy and the "sort_random" strategy are always less than that between the "sort_random" and the "random", indicating that the "sort" strategy is robust to a certain degree of randomness.

D. The Advantage and Limitation of Our Method

The biggest advantage of our model is that the training time is extremely short. To analyze the complexity of our method, we randomly selected 20%, 40%, 60%, and 80% of the data from the NPInter10412 dataset and measured the wall-clock running time of NPI-RGCNAE, IPMiner, and RPITER on the same platform. As illustrated in Fig. 7, the running time of all models increased along with the data size, but the increasing trends were different. For every 20% increment in the dataset, the training time of our model increased by 21.09 seconds, 9.44 seconds, and 7 seconds, respectively. With the growth of data size, the increase of training time of our model was decreased, while IPMiner had almost a linear increase, and the increase in training time of RPITER was increased. Although we did not formally analyze the computational complexity of three algorithms, this wall-clock time analysis also suggests that IPMiner may have linear complexity, while our method has a sublinear complexity, and RPITER has a super-linear complexity.

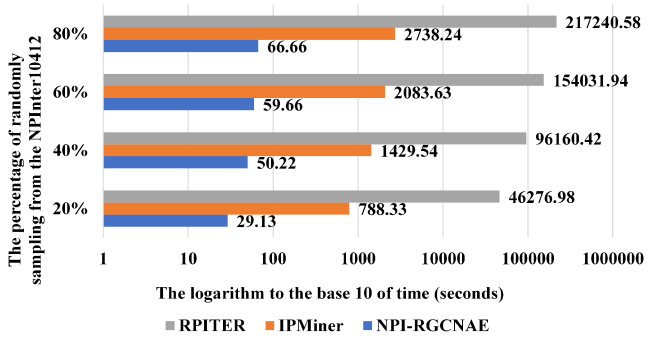

Fig. 7. The running time of the RPITER, IPMiner, and NPI-RGCNAE on datasets with different sizes. We randomly selected 20%, 40%, 60%, and 80% of data from the NPInter10412 dataset. The X-axis is the logarithm to the base 10 of time in seconds. Gray, orange, and blue bars represent the running time of RPITER, IPMiner, and NPI-RGCNAE, respectively.

The GNN based method may have an efficiency advantage over other deep learning-based methods, particularly CNN. The results of our method consist with the existing report in protein structure predictions[60]. This training efficiency makes our method much easier to be applied in practice, particularly on a dataset with a considerable size.

It should be mentioned that our method has a limitation, comparing to sequence-based methods. Our method relies on aggregating existing link information for prediction, which makes it difficult to achieve good performances on sparse networks, where neighborhoods of a node are insufficiently defined. Therefore, although sequence-based methods are not as efficient as ours, they do have slightly better prediction performances in comparison and the ability to work without sufficient link information.

V. CONCLUSION

In this work, we introduced a novel ncRNA-protein interaction prediction method, namely NPI-RGCNAE. The prediction performances of our method are comparable to all state-of-the-art methods. Particularly, our method is much more efficient than existing methods. The training time of our method is generally less than 10% of all state-of-the-art methods. This advantage makes our method more feasible in practice with increasingly large datasets. To overcome the

unreliable negative sample problem, we introduced a negative sample selection strategy. This strategy not only improved prediction performances of our method, but also all methods in comparison. It should be noted that NPI-RGCNAE does not work for orphan nodes in the network with no interactions to other nodes, which is a common limitation of all network-based methods. Besides that, we can safely state that our method, the NPI-RGCNAE, is an efficient, accurate, and robust method for predicting ncRNA-protein interactions.

Moreover, our method is a proof of the effectiveness of deep learning algorithms in modeling biological phenomenon. Along with existing methods, which all performed almost perfect in prediction, we can conclude that the deep learning method can provide an accurate working model to predict ncRNA-protein interactions. However, the interpretation of the results is difficult, as the deep learning model is still a black-box. Without shedding the light of deep learning interpretation theory, which is still not comprehensively available, how this black-box should be applied in practical studies, and how this model can be used to replace or partially replace wet-experiments are still open questions.

Appendix

Supplementary File 1: *k*-mer features as side-information.

Supplementary Table S1: Hyper-parameters of NPI-RGCNAE.

Supplementary Table S2: Hyper-parameters of IPMiner in comparison.

Supplementary Table S3: Hyper-parameters of RPITER in comparison.

References

[1] S. Knowling and K. V. Morris, "Non-coding RNA and antisense RNA. Nature's trash or treasure?," *Biochimie*, vol. 93, no. 11, pp. 1922–1927, Nov. 2011, doi: 10.1016/j.biochi.2011.07.031.

[2] K. Schaukowitch and T.-K. Kim, "Emerging epigenetic mechanisms of long non-coding RNAs," *Neuroscience*, vol. 264, pp. 25–38, Apr. 2014, doi: 10.1016/j.neuroscience.2013.12.009.

[3] S. Ghafouri-Fard, H. Shoorei, F. T. Anamag, and M. Taheri, "The role of non-coding RNAs in controlling cell cycle related proteins in cancer cells," *Front Oncol.*, vol. 10, 2020, Art. no. 608975, doi: 10.3389/fonc.2020.608975.

[4] M. Morlando, M. Ballarino, A. Fatica, and I. Bozzoni, "The role of long noncoding RNAs in the epigenetic control of gene expression," *Chem. Med. Chem.*, vol. 9, no. 3, pp. 505–510, Mar. 2014, doi: 10.1002/cmdc.201300569.

[5] B. Zhang *et al.*, "Dysregulation of long Non-coding RNAs and mRNAs in plasma of clear cell renal cell carcinoma patients using microarray and bioinformatic analysis," *Front. Oncol.*, vol. 10, Nov. 2020, Art. no. 559730 doi: 10.3389/fonc.2020.559730.

[6] M. Kitagawa, Y. Kotake, and T. Ohhata, "Long non-coding RNAs involved in cancer development and cell fate determination," *Curr. Drug Targets*, vol. 13, no. 13, pp. 1616–1621, Dec. 2012, doi: 10.2174/138945012803530026.

[7] Y.-P. Zhu *et al.*, "Long noncoding RNA expression signatures of bladder cancer revealed by microarray," *Oncol. Lett.*, vol. 7, no. 4, pp. 1197–1202, Apr. 2014, doi: 10.3892/ol.2014.1843.

[8] M. A. Faghihi *et al.*, "Expression of a noncoding RNA is elevated in Alzheimer's disease and drives rapid feed-forward regulation of beta-secretase," *Nat. Med.*, vol. 14, no. 7, pp. 723–730, Jul. 2008, doi: 10.1038/nm1784.

[9] M. L. Alvarez and J. K. DiStefano, "Functional characterization of the plasmacytoma variant translocation 1 gene (PVT1) in diabetic nephropathy," *PLoS One*, vol. 6, no. 4, Apr. 2011, Art. no. e18671, doi: 10.1371/journal.pone.0018671.

[10] J. Zhu, H. Fu, Y. Wu, and X. Zheng, "Function of lncRNAs and approaches to lncRNA-protein interactions," *Sci. China Life Sci.*, vol. 56, no. 10, pp. 876–885, Oct. 2013, doi: 10.1007/s11427-013-4553-6.

[11] J. D. Keene, J. M. Komisarow, and M. B. Friedersdorf, "RIP-Chip: The isolation and identification of mRNAs, microRNAs and protein components of ribonucleoprotein complexes from cell extracts," *Nat. Protoc.*, vol. 1, no. 1, pp. 302–307, Jun. 2006, doi: 10.1038/nprot.2006.47.

[12] R. B. Darnell, "HITS-CLIP: Panoramic views of protein–RNA regulation in living cells," *WIREs RNA*, vol. 1, no. 2, pp. 266–286, Sep. 2010, doi: 10.1002/wrna.31.

[13] D. Ray *et al.*, "Rapid and systematic analysis of the RNA recognition specificities of RNA-binding proteins," *Nat. Biotechnol.*, vol. 27, no. 7, pp. 667–670, Jul. 2009, doi: 10.1038/nbt.1550.

[14] U. K. Muppirala, V. G. Honavar, and D. Dobbs, "Predicting RNA-Protein interactions using only sequence information," *BMC Bioinf.*, vol. 12, no. 1, Dec. 2011, Art. no. 489, doi: 10.1186/1471-2105-12-489.

[15] Y. Wang *et al.*, "De novo prediction of RNA–protein interactions from sequence information," *Mol. BioSyst.*, vol. 9, no. 1, pp. 133–142, 2013, doi: 10.1039/C2MB25292A.

[16] Q. Lu *et al.*, "Computational prediction of associations between long non-coding RNAs and proteins," *BMC Genomic.*, vol. 14, no. 1, 2013, Art. no. 651 doi: 10.1186/1471-2164-14-651.

[17] V. Suresh, L. Liu, D. Adjeroh, and X. Zhou, "RPI-Pred: Predicting ncRNA-protein interaction using sequence and structural information," *Nucleic Acids Res.*, vol. 43, no. 3, pp. 1370–1379, Feb. 2015, doi: 10.1093/nar/gkv020.

[18] A. Li, M. Ge, Y. Zhang, C. Peng, and M. Wang, "Predicting long noncoding RNA and protein interactions using heterogeneous network model," *BioMed. Res. Int.*, vol. 2015, pp. 1–11, 2015, doi: 10.1155/2015/671950.

[19] J. Yang, A. Li, M. Ge, and M. Wang, "Relevance search for predicting lncRNA–protein interactions based on heterogeneous network," *Neurocomputing*, vol. 206, pp. 81–88, Sep. 2016, doi: 10.1016/j.neucom.2015.11.109.

[20] X. Zheng *et al.*, "Fusing multiple protein-protein similarity networks to effectively predict lncRNA-protein interactions," *BMC Bioinf.*, vol. 18, no. S12, Oct. 2017, Art. no. 420, doi: 10.1186/s12859-017-1819-1.

[21] Y. Xiao, J. Zhang, and L. Deng, "Prediction of lncRNA-protein interactions using hetesim scores based on heterogeneous networks," *Sci. Rep.*, vol. 7, no. 1, Dec. 2017, Art. no. 3664, doi: 10.1038/s41598-017-03986-1.

[22] L. Deng, J. Wang, Y. Xiao, Z. Wang, and H. Liu, "Accurate prediction of protein-lncRNA interactions by diffusion and hetesim features across heterogeneous network," *BMC Bioinf.*, vol. 19, no. 1, Dec. 2018, Art. no. 370, doi: 10.1186/s12859-018-2390-0.

[23] H. Zhang, Z. Ming, C. Fan, Q. Zhao, and H. Liu, "A path-based computational model for long non-coding RNA-protein interaction prediction," *Genomics*, vol. 112, no. 2, pp. 1754–1760, Mar. 2020, doi: 10.1016/j.ygeno.2019.09.018.

[24] X. Pan, Y.-X. Fan, J. Yan, and H.-B. Shen, "IPMiner: Hidden ncRNA-protein interaction sequential pattern mining with stacked autoencoder for accurate computational prediction," *BMC Genomic.*, vol. 17, no. 1, Dec. 2016, Art. no. 582, doi: 10.1186/s12864-016-2931-8.

[25] S. Cheng, L. Zhang, J. Tan, W. Gong, C. Li, and X. Zhang, "DM-RPIs: Predicting ncRNA-protein interactions using stacked ensembling strategy," *Comput. Biol. Chem.*, vol. 83, Dec. 2019, Art. no. 107088, doi: 10.1016/j.compbiolchem.2019.107088.

[26] H.-C. Yi *et al.*, "Learning distributed representations of RNA and protein sequences and its application for predicting lncRNA-protein interactions," *Comput. Struct. Biotechnol. J.*, vol. 18, pp. 20–26, 2020, doi: 10.1016/j.csbj.2019.11.004.

[27] C. Peng, S. Han, H. Zhang, and Y. Li, "RPITER: A hierarchical deep learning framework for ncRNA–Protein interaction prediction," *IJMS*, vol. 20, no. 5, Mar. 2019, Art. no. 1070 doi: 10.3390/ijms20051070.

[28] S.-W. Zhang, X.-X. Zhang, X.-N. Fan, and W.-N. Li, "LPI-CNNCP: Prediction of lncRNA-protein interactions by using convolutional neural network with the copy-padding trick," *Anal. Biochem.*, vol. 601, Jul. 2020, Art. no. 113767, doi: 10.1016/j.ab.2020.113767.

[29] J. Wang *et al.*, "EDLMFC: An ensemble deep learning framework with multi-scale features combination for ncRNA-protein interaction prediction," *BMC Bioinf.*, vol. 22, no. 1, Mar. 2021, Art. no. 133 doi: 10.1186/s12859-021-04069-9.

[30] D. Shaw, H. Chen, M. Xie, and T. Jiang, "DeepLPI: A multimodal deep learning method for predicting the interactions between lncRNAs and protein isoforms," *BMC Bioinf.*, vol. 22, Jan. 2021, Art. no. 24, doi: 10.1186/s12859-020-03914-7.
[31] H. Yao, J. Guan, and T. Liu, "Denoising protein–protein interaction network via variational graph auto-encoder for protein complex detection," *J. Bioinform. Comput. Biol.*, vol. 18, no. 3, Jun. 2020, Art. no. 2040010, doi: 10.1142/S0219720020400107.
[32] Y. Dai, C. Guo, W. Guo, and C. Eickhoff, "Drug–drug interaction prediction with wasserstein adversarial Autoencoder-based knowledge graph embeddings," *Brief. Bioinf.*, Oct. 2020, Art. no. bbaa256, doi: 10.1093/bib/bbaa256.
[33] X. Wu, W. Lan, Q. Chen, Y. Dong, J. Liu, and W. Peng, "Inferring Lncrna-disease associations based on graph autoencoder matrix completion," *Comput. Biol. Chem.*, vol. 87, Aug. 2020, Art. no. 107282 doi: 10.1016/j.compbiolchem.2020.107282.
[34] Q. Wang, Z. Mao, B. Wang, and L. Guo, "Knowledge graph embedding: A survey of approaches and applications," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 12, pp. 2724–2743, Dec. 2017, doi: 10.1109/TKDE.2017.2754499.
[35] M. Nickel, V. Tresp, and H.-P. Kriegel, "A three-way model for collective learning on multi-relational data," in *Proc. Int. Council Machinery Lubrication*, Bellevue, WA, USA, 2011, pp. 809–816. [Online]. Available: https://icml.cc/2011/papers/438_icmlpaper.pdf
[36] A. Bordes, N. Usunier, A. García-Durán, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling Multi-relational data," in *Proc. NeurIPS*, Lake Tahoe, NV, USA, 2013, pp. 2787–2795. [Online]. Available: https://proceedings.neurips.cc/paper/2013/hash/1cecc7a77928ca8133fa24680a88d2f9-Abstract.html
[37] R. Socher, D. Chen, C. D. Manning, and A. Y. Ng, "Reasoning with neural tensor networks for knowledge base completion," in *Proc. NeurIPS*, Lake Tahoe, NV, USA, 2013, pp. 926–934. [Online]. Available: https://proceedings.neurips.cc/paper/2013/hash/b337e84de8752b27eda3a12363109e80-Abstract.html
[38] B. Yang, W. Yih, X. He, J. Gao, and L. Deng, " Embedding entities and relations for learning and inference in knowledge bases," Presented at Proc. 3rd Int. Conf. Learn. Representations, San Diego, CA, USA, May 2015, pp. 1–13. [Online]. Available: http://arxiv.org/abs/1412.6575
[39] M. S. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *Proc. 15th ESWC*, Heraklion, Crete, GR, Jun. 2018, vol. 10843, pp. 593–607, doi: 10.1007/978-3-319-93417-4_38.
[40] R. van den Berg, T. N. Kipf, and M. Welling, "Graph convolutional matrix completion," in *Proc. KDD'18*, London, U.K., 2018, Art. no. 7. [Online]. Available: https://www.kdd.org/kdd2018/files/deep-learning-day/DLDay18_paper_32.pdf
[41] H. Hu *et al.*, "HLPI-Ensemble: Prediction of human lncRNA-protein interactions based on ensemble strategy," *RNA Biol.*, pp. 1–10, Jun. 2018, doi: 10.1080/15476286.2018.1457935.
[42] J. Yuan, W. Wu, C. Xie, G. Zhao, Y. Zhao, and R. Chen, "NPInter v2.0: An updated database of ncRNA interactions," *Nucl. Acids Res.*, vol. 42, no. D1, pp. D104–D108, Jan. 2014, doi: 10.1093/nar/gkt1057.
[43] X.-N. Fan and S.-W. Zhang, "LPI-BLS: Predicting lncRNA–protein interactions with a broad learning system-based stacked ensemble classifier," *Neurocomputing*, vol. 370, pp. 88–93, Dec. 2019, doi: 10.1016/j.neucom.2019.08.084.
[44] B. A. Lewis *et al.*, "PRIDB: A protein-RNA interface database," *Nucleic Acids Res.*, vol. 39, pp. D277–D282, Jan. 2011, doi: 10.1093/nar/gkq1108.
[45] Y. Hao *et al.*, "NPInter v3.0: An upgraded database of noncoding RNA-associated interactions," *Database*, vol. 2016, 2016, Art. no. baw057 doi: 10.1093/database/baw057.
[46] The UniProt Consortium, "UniProt: A worldwide hub of protein knowledge," *Nucleic Acids Res.*, vol. 47, no. D1, pp. D506–D515, Jan. 2019, doi: 10.1093/nar/gky1049.
[47] Y. Zhao *et al.*, "NONCODE 2016: An informative and valuable data source of long non-coding RNAs," *Nucleic Acids Res.*, vol. 44, no. D1, pp. D203–D208, Jan. 2016, doi: 10.1093/nar/gkv1252.
[48] J. Harrow *et al.*, "GENCODE: The reference human genome annotation for the ENCODE project," *Genome Res.*, vol. 22, no. 9, pp. 1760–1774, Sep. 2012, doi: 10.1101/gr.135350.111.
[49] X. Pan, L. Zhu, Y.-X. Fan, and J. Yan, "Predicting protein–RNA interaction amino acids using random forest based on submodularity subset selection," *Comput. Biol. Chem.*, vol. 53, pp. 324–330, Dec. 2014, doi: 10.1016/j.compbiolchem.2014.11.002.
[50] Y. Ma, T. He, and X. Jiang, "Projection-based neighborhood non-negative matrix factorization for lncRNA-Protein interaction prediction," *Front. Genet.*, vol. 10, Nov. 2019, Art. no. 1148, doi: 10.3389/fgene.2019.01148.
[51] T. Zhang, M. Wang, J. Xi, and A. Li, "LPGNMF: Predicting long non-coding RNA and protein interaction using graph regularized nonnegative matrix factorization," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 17, no. 1, pp. 189–197, Jan. 2020, doi: 10.1109/TCBB.2018.2861009.
[52] C. Shen, Y. Ding, J. Tang, L. Jiang, and F. Guo, "LPI-KTASLP: Prediction of LncRNA-Protein interaction by semi-supervised link learning with multivariate information," *IEEE Access*, vol. 7, pp. 13486–13496, 2019, doi: 10.1109/ACCESS.2019.2894225.
[53] Q. Zhao, Y. Zhang, H. Hu, G. Ren, W. Zhang, and H. Liu, "IRWNRLPI: Integrating random walk and neighborhood regularized logistic matrix factorization for lncRNA-Protein interaction prediction," *Front. Genet.*, vol. 9, Art. no. 239 Jul. 2018, doi: 10.3389/fgene.2018.00239.
[54] T. F. Smith and M. S. Waterman, "Identification of common molecular subsequences," *J. Mol. Biol.*, vol. 147, no. 1, pp. 195–197, Mar. 1981, doi: 10.1016/0022-2836(81)90087-5.
[55] D. E. Rumelhart, G. E. Hintont, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986, doi: 323533a0.
[56] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.
[57] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," Presented at Proc. 3rd Int. Conf. Learn. Representations, San Diego, CA, USA, May 2015, pp. 1–11. [Online]. Available: http://arxiv.org/abs/1412.6980
[58] R. Lorenz *et al.*, "ViennaRNA package 2.0," *Algorithms Mol. Biol.*, vol. 6, no. 1, Dec. 2011, Art. no. 26, doi: 10.1186/1748-7188-6-26.
[59] M. Ge, A. Li, and M. Wang, "A bipartite network-based method for prediction of long non-coding RNA–protein interactions," *Genomic., Proteomic. Bioinf.*, vol. 14, no. 1, pp. 62–71, Feb. 2016, doi: 10.1016/j.gpb.2016.01.004.
[60] X. Jing and J. Xu, "Fast and effective protein model refinement using deep graph neural networks," *Nat. Comput. Sci.*, vol. 1, no. 7, pp. 462–469, Jul. 2021, doi: 10.1038/s43588-021-00098-9.