OXFORD

Briefings in Bioinformatics, 2022, **23(5)**, 1–14
https://doi.org/10.1093/bib/bbac333
Problem Solving Protocol

MGEGFP: a multi-view graph embedding method for gene function prediction based on adaptive estimation with GCN

Wei Li, Han Zhang, Minghe Li, Mingjing Han and Yanbin Yin

Corresponding author. Han Zhang, College of Artificial Intelligence, Nankai University, Tongyan Road, 300350, Tianjin, China; E-mail: zhanghan@nankai.edu.cn

Abstract

In recent years, a number of computational approaches have been proposed to effectively integrate multiple heterogeneous biological networks, and have shown impressive performance for inferring gene function. However, the previous methods do not fully represent the critical neighborhood relationship between genes during the feature learning process. Furthermore, it is difficult to accurately estimate the contributions of different views for multi-view integration. In this paper, we propose MGEGFP, a multi-view graph embedding method based on adaptive estimation with Graph Convolutional Network (GCN), to learn high-quality gene representations among multiple interaction networks for function prediction. First, we design a dual-channel GCN encoder to disentangle the view-specific information and the consensus pattern across diverse networks. By the aid of disentangled representations, we develop a multi-gate module to adaptively estimate the contributions of different views during each reconstruction process and make full use of the multiplexity advantages, where a diversity preservation constraint is designed to prevent the over-fitting problem. To validate the effectiveness of our model, we conduct experiments on networks from the STRING database for both yeast and human datasets, and compare the performance with seven state-of-the-art methods in five evaluation metrics. Moreover, the ablation study manifests the important contribution of the designed dual-channel encoder, multi-gate module and the diversity preservation constraint in MGEGFP. The experimental results confirm the superiority of our proposed method and suggest that MGEGFP can be a useful tool for gene function prediction.

Keywords: multi-view graph, gene function prediction, graph convolutional network, graph embedding

Introduction

There has been an increasing interest in the research of computational methods for automated gene function prediction. Genes give instructions of different functions during biological process as the basic physical unit of heredity, and the comprehensive insight into various gene functions is critical to biomedical research [1]. High cost and time-consuming biological experiments in the traditional wet-lab annotation methods make automated function annotation very urgent [2]. Specifically, the automated gene function annotation classifies genes into their corresponding functional categories by computational methods, which can be viewed as a multi-label classification problem.

Over the past decade, the rapid progress in the development of high-throughput experimental techniques has led to an explosion of available biological data with different types. Among these data, a variety of genome-scale interaction networks are generated, which describe complex internal interactions among biological molecules and contain a wealth of information for inferring the functional patterns of genes [1, 3, 4]. Therefore, it is important to explore how to effectively integrate multiple heterogeneous networks and extract the comprehensive information for accurate gene function annotation. There have been many remarkable previous works dedicated to gene function prediction task by jointly considering the information from multiple networks [5–9], which can be viewed as a multi-view graph learning problem.

The first category of these methods focuses on integrating multiple networks directly. They firstly fuse all separate networks into a single one and then use the fused network to perform downstream functional

Wei Li is a PhD student in the College of Artificial Intelligence, Nankai University. His research interests include graph neural network, deep learning and bioinformatics.
Han Zhang is a professor in the College of Artificial Intelligence, Nankai University. His research interests include machine learning, deep learning and bioinformatics.
Minghe Li is a postgraduate student in the College of Artificial Intelligence, Nankai University. His research interests include graph neural network and bioinformatics.
Mingjing Han is a PhD student in the College of Artificial Intelligence, Nankai University. Her research interests include graph neural network and multiple kernel learning.
Yanbin Yin is an associate professor in the Department of Food Science and Technology, University of Nebraska - Lincoln. His research interests include bioinformatics and computational biology.
Received: May 11, 2022. **Revised:** July 2, 2022. **Accepted:** July 21, 2022

prediction [6, 10–14]. ProMK [15] is a kernel-based method, which combines multiple kernels into the best composite kernel and trains a multi-label classifier on the obtained kernel. SNF [16] takes diverse sources of data as input and constructs networks for each type. Then it fuses all separate networks into a common view in a nonlinear combination manner. GeneMANIA [6, 17, 18] integrates multiple heterogeneous networks into a single kernel, and then infers gene functions via Gaussian label propagation. [13] is a probabilistic method which utilizes Bayesian inference to integrate the edges from various networks. However, the fused network which is directly integrated by multiple networks in above methods may lead to the loss of many valuable information and bring in noises. To remedy this issue, the multi-view joint feature learning methods are carried out to generate more accurate representation. Mashup [1] characterizes gene patterns across multiple networks jointly. It obtains a canonical low-dimensional feature from original multiplex topological structures based on matrix factorization and effectively fuses the information in multiple views. Then it utilizes the off-the-shelf support vector machine (SVM) classifier for each function classification. Further considering the noise problem in the original biological networks, EnMUGR [19] proposes a denoised diffusion method and utilizes the joint regularized decomposition, which leads to a more robust representation. Nevertheless, most of the above methods only use shallow models, which limit the capability of capturing node representations in nonlinear gene networks.

Due to the powerful feature extraction ability, deep learning is used to obtain comprehensive gene representation for multi-view fusion and circumvent the above issues. DeepNF [20] firstly proposes a deep learning-based approach which fuses multiple gene networks through a designed multimodal deep autoencoder. In particular, it stacks multiple encoder layers to extract representations for each view at the beginning. Then it designs a bottleneck layer to connect all layers and obtain the compressed features. Finally, the network is trained by minimizing the reconstruction error between each original and reconstructed feature matrix. Furthermore, DeepMNE-CNN [9] proposes a semi-supervised autoencoder. It calculates the pairwise gene similarity based on Pearson Correlation Coefficient value among different networks and utilizes these informative constraints as the supervised signal for next encoder layer. Then it concatenates the features from all views and predicts functions using a multi-scale convolutional neural network. Although these deep learning-based methods have achieved better performance, they fail to fully take into account the neighboring relationship between genes in the network, which is very crucial for generating accurate representations.

In recent years, numerous remarkable deep learning-based models dedicated to multi-view graph representation learning, which are not designed specifically for gene function prediction, have also been proposed. Among them, many unsupervised methods are conducted to deal with the situation that the node labels are scarce and noisy. DMNE [21] proposes the first deep learning-based multi-network embedding method based on an autoencoder framework and designs a co-regularized loss to leverage the cross-network associations. MEGAN [22] develops a generative adversarial network-based model. It devises a generator to effectively generate adversarial node pairs for multi-view graph, and the discriminator enforces the generator to produce more accurate embeddings. VANE [23] learns a more robust representation based on two well-designed adversarial minimax games. The first game aims to extract the view-independent pattern, while the second one enhances the robustness of the representations. One2Multi [24] is a method based on graph autoencoder. It selects the most informative view and performs graph embedding technique on the chosen graph. Then it reconstructs multiple networks by a trainable multi-view graph decoder. DMGI [25] learns the view-specific embedding by maximizing the mutual information between the global representation and the local network patches for each view individually, then designs a regularization framework to obtain the consensus representation. However, these methods all suffer from a lack of scientific and interpretable partition for the multi-view representation space, which will benefit the learning process of multi-view integration. Using current tools, it is difficult to accurately estimate the contribution of each view during the fusion process.

Recently, Graph Neural Network (GNN) provides a powerful learning model for handling graph-structured data [26]. It has made significant achievements in numerous graph data analysis tasks such as graph classification [27, 28], node classification [29, 30] and link prediction [31, 32], and has been applied in various domains of bioinformatics including single-cell RNA-seq analysis [33, 34], peptide toxicity prediction [35], drug discovery [36, 37], etc. Graph Convolutional Network (GCN) [38] is a representative method of GNNs, which performs convolution operation on the graph and updates the representation of nodes by iteratively aggregating the feature from their neighbors. Therefore, GCN can effectively capture the neighborhood relationship between nodes. It has been proven that genes with connections or similar topological roles in the interaction networks tend to have related functions [39, 40]. Therefore, GCN can be a highly appropriate method for feature extraction in gene networks and facilitates the downstream function annotation task. Recently, several methods have been proposed to leverage GCN to learn the latent pattern in single gene network for function annotation [41–43]. However, how to exploit GCN to effectively extract features from multiple gene networks for function prediction remains to be investigated.

In this paper, we propose a Multi-view Graph Embedding method for Gene Function Prediction, named MGEGFP, to learn the comprehensive representations of

genes across multiple networks for accurate functional annotation. Our major contributions are as follows:

- To our best knowledge, MGEGFP makes the first attempt to adopt GCN to jointly analyze multiple gene interaction networks for inferring gene function. MGEGFP is proposed to adaptively estimate the contributions of different views from disentangled representations learned by the designed GCN encoder.
- We design a dual-channel GCN encoder to explicitly disentangle the view-specific information and the common pattern across diverse graphs, which leads to more accurate gene embeddings. This allows us to consider that different views have not only their own view-specific knowledge, but also have consistent message.
- We devise a multi-gate module to adaptively estimate the contributions of different views during each reconstruction process, and further introduce a diversity preservation constraint to prevent the overfitting problem of this multi-gate mechanism, so as to take full advantages of the multi-view graph.
- Experiments on yeast and human datasets are conducted for gene function annotation task. We compare our method with seven state-of-the-art models in five evaluation metrics. The results show that MGEGFP outperforms other methods on both datasets and manifest the effectiveness of our model.

Methods

Notations

Consider a multi-view undirected gene association network as $G = \{V, E^{(1)}, \ldots, E^{(M)}\}$, where each view represents a specific type of relationship among genes. M is the number of views. $V = \{v_i\}_{i=1}^{N}$ is the set of gene nodes which are shared across all views, and N is the number of gene nodes. $E^{(m)}$ denotes the set of all edges in view m, where $m \in \{1, 2, \ldots, M\}$. The topological structure of each graph can be specified by a set of symmetric adjacency matrices $\{A^{(m)}\}_{m=1}^{M}$. The value of $A_{ij}^{(m)}$ varies from 0 to 1, which represents the weight of edge linking node i and j corresponding to view m.

Intra-view graph embedding

Input feature learning

For intra-view learning, we explore how to effectively learn the embedding for each network independently. As shown in Figure 1(A), we initialize the input features $X^{(m)}$ for each individual gene network with random walk with restart (RWR) [44]. It has been proved that RWR tends to capture global associations between nodes in the interaction network [1, 9, 45]. GCN is capable of capturing features among local neighborhoods in the message passing process [26]. Therefore, the global features extracted by RWR can be complemented with the local characteristics captured by the following GCN module. The process of RWR starting from node i in view m can be expressed as

$$x_i^{(m)}[t] = \gamma \bar{A}^{(m)} x_i^{(m)}[t-1] + (1-\gamma) x_i^{(m)}[0], \tag{1}$$

where $x_i^{(m)}[t]$ is the feature vector of node i, which reflects the probability of visiting node i after t step random walks. $x_i^{(m)}[0]$ is an N-dimensional initial vector of node i, which is a one-hot feature vector with the corresponding ith entry equal to 1 and other elements equal to 0. γ is a number in the range $(0, 1)$ and $1-\gamma$ denotes the probability of restart. $\bar{A}^{(m)}$ is the transition matrix which is obtained from the original adjacency matrix as

$$\bar{A}_{ij}^{(m)} = \frac{A_{ij}^{(m)}}{\sum_i A_{ij}^{(m)}} \tag{2}$$

and $\bar{A}_{ij}^{(m)}$ represents the probability of going from node j to node i. After T steps of RWR, we embed each node into an N-dimensional feature vector, which is used as the input feature for subsequent learning model. The same procedure is performed in each view and we can finally obtain the set of all input feature matrices $\{X^{(m)}\}_{m=1}^{M}$.

View-specific graph autoencoder

Considering the superior ability of GCN to model graph data, we utilize it as the graph encoder which embeds the original gene network topology and the obtained node features into a d_s-dimensional embedding space for each view. The process of intra-view feature learning is the same for all views, so here we omit the superscript of the view number for easier reading. In particular, given the network structure and initial feature matrix, the view-specific representation can be learned by an L-layer GCN encoder as

$$Z_s^{l+1} = f\left(Z_s^l, A \mid W_s^l\right) = \delta\left(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} Z_s^l W_s^l\right), \tag{3}$$

where $\tilde{A} = A + I$ and I is the identity matrix. $\tilde{D}$ is the degree matrix of $\tilde{A}$ which is calculated by $\tilde{D}_{ii} = \sum_i \tilde{A}_{ij}$. W_s^l is the layer-specific trainable weight matrix. Z_s^l represents the learned representations by lth layer and $Z_s^0 = X$. δ denotes the nonlinear activation function. To generate a more informative embedding and alleviate the over-smoothing problem, we employ the jump connection [46, 47] and aggregate the representations obtained from all layers to get the final embedding as

$$Z_s = Agg\left(Z_s^1, Z_s^2, \ldots, Z_s^L\right). \tag{4}$$

In order to facilitate the layer aggregation operation, the dimension of all hidden layers is set to d_s. Agg can be any layer aggregation function such as Vector-concatenation, Max-pooling and Mean-pooling. Subsequently, we exploit the decoder to reconstruct the original topological structure from the learned representation. Specifically, we use the Inner-product

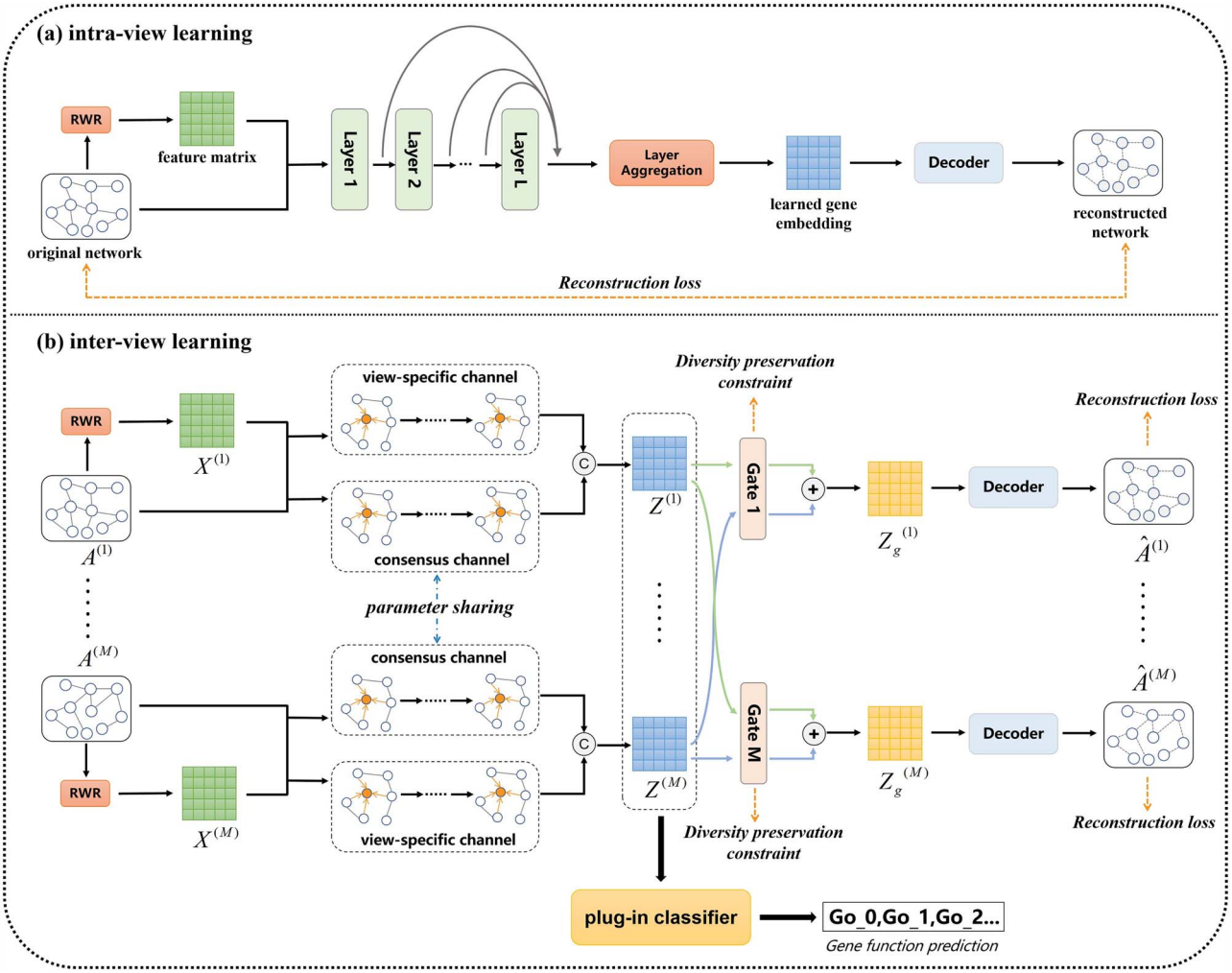

Figure 1. The overall framework of MGEGFP model. **(A)** Intra-view graph representation learning. For each individual view, we exploit RWR to obtain the high-quality initial feature representations. Then the original adjacency matrix and the initial features undergo an *L*-layer GCN equipped with a jump connection to generate the learned representation matrix, which is then fed into the decoder to calculate the reconstructed adjacency matrix. Then we minimize the reconstruction error to train the model. **(B)** Inter-view graph representation learning on the basis of intra-view learning. For simplicity, we use two views as an example. In multi-view representation learning, a dual-channel GCN encoder is constructed to disentangle the view-specific information and the common pattern across all views. Then the obtained embeddings in each view pass through the multi-gate module and the outputs are used to decode the topology structure of each view. Finally, the learned gene representations are concatenated and used to train the plug-in classifier to annotate gene functions.

operation and the *sigmoid* activation function to predict whether there exists an edge between two gene nodes:

$$p\left(\hat{A}_{ij} \mid z_i, z_j\right) = sigmoid\left(z_i^T z_j\right), \tag{5}$$

where $\hat{A}$ represents the reconstructed adjacency matrix and $p(\hat{A}_{ij})$ is the value of $\hat{A}_{ij}$, which denotes the probability that the edge between node i and j exists in the network. For model training, we minimize the reconstruction error between $\hat{A}$ and A. The binary cross entropy is utilized as the loss function:

$$L_{rec}^{(m)} = -\frac{1}{N \times N}\left(\sum_{e_{ij} \in E} \log \hat{A}_{ij} + \sum_{e_{ij} \notin E} \log\left(1 - \hat{A}_{ij}\right)\right), \tag{6}$$

where $L_{rec}^{(m)}$ represents the reconstruction loss of view m.

Inter-view graph embedding

Dual-channel graph encoder

Then we explore how to generate comprehensive representation by multi-view collaborative learning. Each individual network has its view-specific context, and at the same time, there exists consistent latent pattern among all networks [48]. To this end, we further extend the single-channel GCN encoder into a dual-channel GCN encoder to explicitly disentangle both contents from original multi-view gene network. In particular, the first channel is the same as the setting in the intra-view learning process, which captures the view-specific pattern within each individual network. Then, we devise an additional channel called *consensus* channel, which aims to extract the common pattern across all views.

In this *consensus* channel, we encode each input graph structure information and corresponding node features using a parameter-sharing GCN function. Like the view-specific channel, we also stack L layers of continuous graph convolution operation and use the jump connection. The output common representation of this channel for each view can be obtained as

$$Z_c^{l+1} = f\left(Z_c^l, A \mid W_c^l\right) = \delta\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}Z_c^l W_c^l\right) \tag{7}$$

$$Z_c = Agg\left(Z_c^1, Z_c^2, \ldots, Z_c^L\right). \tag{8}$$

Here, we also omit the superscript of the view number for easier reading. W_c^l is the parameter-sharing weight matrix across all views, and the dimension of all hidden layers in this channel is set to d_c. The design of shared weight matrix allows filtering out shared features from multiple views, and the combination of view-specific embedding and consensus embedding can obtain larger expressive capability for genes. The input features for both *consensus* channel and view-specific channel are from RWR. As shown in Figure 1(B), we finally get the representation $Z^{(m)}$ for each view by concatenating the output of both channels:

$$Z^{(m)} = combine(Z_c^{(m)}, Z_s^{(m)}), \tag{9}$$

which is used for downstream functional classification.

Multi-gate module

Once we obtain the node representation $Z^{(m)}$ for each view, a straightforward idea is to concatenate all features directly which is then fed into each view-specific decoder to reconstruct the original graph structures. However, this way ignores the different role of each view during each reconstruction process and fails to capture the multiplexity of the multi-view graph. To model the inter-view relationship more accurately and improve the representation quality, we design a multi-gate network fusion module, considering that each view provides a different contribution in each reconstruction task. Specifically, the gating module is implemented by MLP with a nonlinear activation function. Here, we focus on gate m and we calculate the gate score $\alpha^{(m)(j)}$ for embedding from view j as

$$s^{(m)(j)} = g\left(W^{(m)(j)}Z^{(j)} + b^{(m)(j)}\right) \tag{10}$$

$$\alpha^{(m)(j)} = \frac{\exp\left(s^{(m)(j)}\right)}{\sum_i \exp\left(s^{(m)(i)}\right)}, \tag{11}$$

where the value ranges from 0 to 1. $W^{(m)(j)}$ and $b^{(m)(j)}$ are the trainable weight matrix and bias vector for view j in gate m, respectively. g denotes the nonlinear activation function. $s^{(m)(j)}$ represents the information score. Then we take a linear combination of the representations of each view with the learned gate weight, and obtain the fused representation for view m as

$$Z_g^{(m)} = \sum_{j=1}^{M} \alpha^{(m)(j)} Z^{(j)}. \tag{12}$$

$Z_g^{(m)}$ can be seen as the weighted sum of the representations from all views, and is then used to reconstruct the original graph structure. With this mechanism, the model becomes more accurate by adaptive estimation. By the aid of disentangled representations, the multi-gate module can adaptively estimate the contributions of different views during each reconstruction process. Therefore, each reconstruction task can evaluate the importance of the disentangled representations from different views, rather than taking information from all views equally. Correspondingly, during the backpropagation process, the gradients of multiple reconstruction tasks will be propagated to the graph encoder. Thus, a well-designed graph reconstruction mechanism will supervise the encoder of each view to better disentangle the view-specific and consensus pattern, and generate more comprehensive feature, which is beneficial for downstream tasks. As the aforementioned intra-view learning, we also employ the Inner-product operation with the *sigmoid* activation function as the decoder to calculate the reconstructed adjacency matrices. The total reconstruction loss is the sum of all views:

$$L_{rec} = \sum_{m=1}^{M} L_{rec}^{(m)}. \tag{13}$$

Diversity preservation constraint

In the gated fusion process, we want to preserve the multiplexity in each reconstruction task as much as possible to prevent the over-fitting problem. Hence, to further enhance the capability of the multi-gate module, we propose a corresponding diversity preservation constraint which takes the following form:

$$L_{dpc} = \frac{1}{M}\sum_{i=1}^{M}\sum_{j=1}^{M} S(Z_g^{(i)}, Z^{(j)}), \tag{14}$$

where S is a function which measures the similarity between the gated fusion embedding Z_g and the view embedding Z. It can be calculated by cosine similarity, Manhattan Distance, Euclidean distance or other similarity measure.

Specifically, for each reconstruction task, representations from its own view are usually more suitable for reconstruction than representations from other views. Hence, each gate may tend to give an extremely large weight to the embedding of its own view in the training

process, which can cause the model training to fall into the local minima and result in the over-fitting problem. In other words, $Z_g^{(m)}$ may be quite similar to $Z^{(m)}$, which ignores the message from other views, thereby weakening the effect of the multi-gate module and failing to fully take advantage of the multi-view graph. By minimizing this term, the representation of each view can be fully considered during each reconstruction process and we can arrive at a more comprehensive embedding. The final objective function of the inter-view node representation learning is formulated as

$$L = L_{rec} + \beta L_{dpc}, \tag{15}$$

where $\beta > 0$ is a coefficient which controls the importance of L_{dpc}.

Gene function prediction

After we obtain the representation $Z^{(m)}$ of each view, we concatenate them together and use an out-of-the-box classifier to annotate gene functions.

The overall process of MGEGFP is summarized in Supplementary Algorithm S1. We formulate the gene function prediction as a multi-label problem and use Light Gradient Boosting Machine (LightGBM) [49] as the classifier. LightGBM is a gradient boosting framework and utilizes decision tree algorithm for learning. Compared with the commonly used SVM classifier for functional annotation, LightGBM has a faster training speed and lower memory usage, which offers a competitive predictive performance. For each function, we train a binary LightGBM model on the training set, and finally obtain a predicted probability for each unlabeled gene.

Experiments

Experimental setup

The parameters in MGEGFP are all initialized by Xaiver initialization [50] and we optimize the parameters with Adam [51]. We introduce the dropout [52] technique to prevent the over-fitting problem. The nonlinear activation function in the GCN encoder is tanh, except for the last layer which is equipped with a linear activation function. The similarity function used for diversity preservation in Equation 14 is the cosine similarity, which takes two vectors as the input and outputs the similarity value. The learning rate is set to 0.005, and the balanced factor β is tuned in the range of [0.5, 1.0].

Besides, we investigate the model performance as a function of embedding dimensions and the number of encoder layers, which are shown in Supplementary Figure S3 and Figure S4. We also conduct analysis for different decoders (detailed descriptions are illustrated in Supplementary Section S2), layer aggregation functions and initial feature extractors. The results can be found in Supplementary Table S2, Table S3 and Table S4, respectively. The nonlinear activation function used in the multi-gate module is ReLU, and we also investigate the effect of other different activation functions in Supplementary Table S5.

For each downstream gene annotation task, we first randomly shuffle the entire labeled genes. Then we take 80% of the genes as the training set, 10% as the validation set and evaluate the model performance on the remaining 10% genes as the test set. We randomly perform this data split process 10 times for each method and take the mean value as the predictive result. We consider the following five evaluation metrics:

- **acc**: The percentage of genes which are correctly classified in the category with their highest prediction score.
- **f1**: The F-measure for multi-label classification.
- **m-aupr**: micro-aupr calculates the average aupr (area under precision-recall curve) value using predicted labels and known labels for all classes.
- **M-aupr**: Macro-aupr calculates the aupr value for each class independently and then averages over all classes.
- **subset 0–1 loss**: The fraction of incorrectly predicted samples. For each sample, subset 0–1 loss considers it correctly classified only if the entire set of its labels are all correctly predicted.

For acc, f1, m-aupr and M-aupr, larger values represent better performance. In contrast, for the subset 0–1 loss, smaller values indicate better performance.

Data preparation

We evaluate the performance of MGEGFP on both yeast and human datasets, which have been also used in [1, 9, 20]. We describe the properties of the datasets in Table 1, and more detailed descriptions can be found in Supplementary Table S1. The networks we use are all downloaded from the STRING database [13]. The STRING database is based on diverse data sources and is designed to provide a comprehensive perspective for protein–protein interaction information. There are six heterogeneous graphs in total, and each view represents a specific type of associations between genes: neighborhood (conserved genomic neighborhood), fusion (gene fusion events), cooccurrence (phylogenetic co-occurrence), experiment (high-throughput experiments), database (curated Protein–Protein Interaction databases) and coexpression. Each edge in the networks is assigned a weight with a value between 0 and 1, which represents the probability of the existence of this edge.

In yeast dataset, each original network contains 6400 gene nodes. To make fair comparisons with the previously published methods, we also get the annotations from Munich Information Center for Protein Sequences (MIPS) [53], which are then divided into three functional categories: level 1, level 2 and level 3. As shown in Table 1, level 1 contains 17 most general functional categories, level 2 includes 74 categories and level 3 consists of 154 most specific categories [1].

Table 1. The statistics of datasets used in this work. # Views represents the number of input views and # Nodes denotes the number of gene nodes. # Edges represents the number of edges in coexpression, cooccurrence, database, experiment, fusion and neighborhood network, respectively. # Classes represents the number of classes in each subset. We describe the properties of all individual networks in more detail in the Supplementary Table S1.

Dataset	# Views	# Nodes	# Edges	# Classes
Yeast	6	6400	314 013/2664/33 486/219 995/1361/45 610	level 1 (17) level 2 (74) level 3 (154)
Human	6	18 362	1576 332/36 128/319 004/618 574/3760/104 958	MF: 11–30 (153) 31–100 (72) 101–300 (18) CC: 11–30 (82) 31–100 (46) 101–300 (20) BP: 11–30 (262) 31–100 (100) 101–300 (28)

Table 2. The results for ablation study on yeast level 1 and human MF: 31–100 datasets. We analyze the performance of our model with single-channel and dual-channel encoder. Moreover, we explore the effect of different fusion strategies and the proposed diversity preservation constraint.

	Yeast: Level 1			Human: MF: 31–100		
Model	f1	m-aupr	M-aupr	f1	m-aupr	M-aupr
MGEGFP	0.621	0.762	0.642	0.342	0.395	0.327
MGEGFP-1	0.613	0.751	0.631	0.331	0.379	0.319
MGEGFP-2	0.610	0.748	0.632	0.323	0.372	0.315
MGEGFP-3	0.598	0.722	0.613	0.301	0.362	0.296
MGEGFP-4	0.615	0.753	0.637	0.332	0.382	0.318

In human dataset, the number of gene nodes is 18 362 for each graph. The annotations for human are obtained from Gene Ontology database (GO) [54], which includes three domains, i.e. Molecular Function (MF), Cellular Component (CC) and Biological Process (BP). As shown in Table 1, according to the number of annotated genes by GO term, each functional domain is further categorized into three subsets to make a fair comparison with [1, 9, 19, 20]. In each domain, there are 11–30, 31–100 and 101–300 genes annotated by GO terms in three subsets, respectively.

Ablation experiments

In this section, a comprehensive ablation study is carried out to investigate the contribution of each major component in MGEGFP. We present the ablation experimental results in Table 2 and the details of variant models can be found in Supplementary Table S6. Firstly, to demonstrate the effect of dual-channel GCN encoder, we remove the *consensus* channel and create a variant model with single view-specific channel named MGEGFP-1. By comparing the results of MGEGFP and MGEGFP-1, we can observe that the performance of MGEGFP is better, which implies that the *consensus* channel improves the performance of gene annotation task. This is consistent with our analysis that the designed weight-shared GCN channel can effectively model the common attributes of all views and facilitates to learn high-quality representations.

To validate that our multi-gate fusion mechanism can effectively integrate multi-view information, we eliminate the designed fusion module and create two variant models: MGEGFP-2 and MGEGFP-3. MGEGFP-2 is equipped with a single-gate fusion mechanism which is shared for all views, and MGEGFP-3 is a straightforward way which directly concatenates the features learned by each view without any gated fusion module. As shown in Table 2, MGEGFP-2 achieves a better performance than MGEGFP-3 by an improvement of 1.2%, 2.6% and 1.9% on f1, m-aupr and M-aupr in yeast level 1, indicating that fusing multiple networks with weighted gating schemes instead of simple concatenation can lead to a better performance. In addition, we can observe that MGEGFP further outperforms MGEGFP-2. In human MF: 31–100 dataset, we can observe that MGEGFP outperforms MGEGFP-2 by an improvement of 1.9%, 2.3% and 1.2% on three metrics. This verifies the effectiveness of the devised multi-gate module. The reason is that the weight distribution of each view varies when reconstructing different networks. The single-gate mechanism simply assumes that the weight distribution is the same in each reconstruction process, which is apparently suboptimal, while the multi-gate mechanism takes the different contribution of each view in each reconstruction into account, and helps improve the model capacity.

Especially, we remove the diversity preservation constraint in MGEGFP to explore its effect on the over-fitting problem, and name the variant model as MGEGFP-4. From Table 2 we can observe that, with the help of the designed constraint, our multi-gate mechanism can be further improved and result in a better predictive performance. Besides, we visualize the gate weights in the database view of gene YPL139C in Figure 2 as a concrete example. We run 10 times of MGEGFP-4 and MGEGFP, and the weight distributions of all views are shown in Figure 2(A) and Figure 2 (B), respectively. As can be seen from Figure 2(A), without the diversity preservation constraint, the gate tends to give its own view a larger weight for easier reconstruction process, which results in the over-fitting problem. With the diversity preservation constraint, this problem can be alleviated (Figure 2B), and it is beneficial to better capture the multiplexity in the multi-view graph.

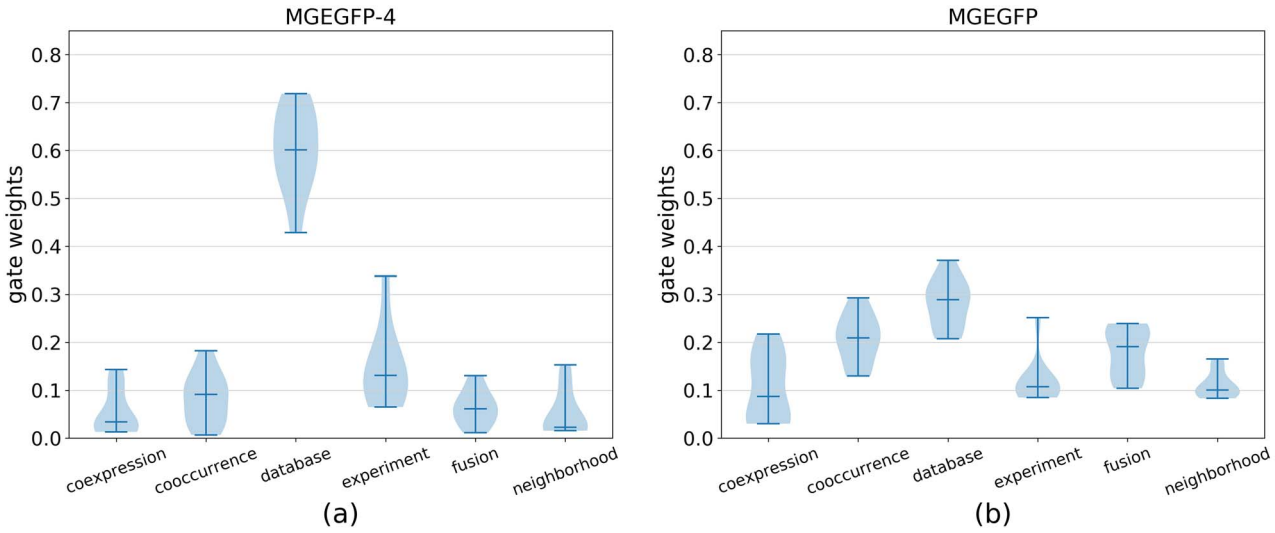

Figure 2. The gate weights of the database view for gene node YPL139C. **(A)** Weight distribution of each view in MGEGFP. **(B)** Weight distribution of each view in MGEGFP-4, which is not equipped with the diversity preservation constraint. Moreover, from Table 2 we can observe the quantitative results of the predictive performance for MGEGFP and MGEGFP-4. In yeast level 1, the values of three metrics (f1, m-aupr, M-aupr) are (0.621, 0.762, 0.642) for MGEGFP, (0.615, 0.753, 0.637) for MGEGFP-4. In human MF: 31–100, the metrics are (0.342, 0.395, 0.327) for MGEGFP, (0.332, 0.382, 0.318) for MGEGFP-4. It can be seen that with the diversity preservation constraint, our model effectively alleviates the over-fitting problem and the predictive performance is enhanced.

Table 3. The detailed descriptions of baseline methods.

Methods	Method descriptions
SNF [16]	A method that uses a nonlinear combination to fuse various type of networks into one network, which represents the information across all views.
DeepNF [20]	DeepNF is a deep learning-based method which designs a multi-modal autoencoder to capture the latent patterns from multiple graphs.
Mashup [1]	Mashup learns a comprehensive low-dimensional feature from multiple topological structures based on matrix factorization.
DeepMNE-CNN [9]	It designs a semi-supervised autoencoder which takes the correlations from multiple networks into consideration, and uses multi-scale CNN for gene function annotation.
EnMUGR [19]	EnMUGR uses the denoised diffusion technique to alleviate the noise problem in each network and then utilizes the joint regularized decomposition to learn a robust common embedding across all views.
One2Multi [24]	It is a graph autoencoder-based framework which firstly selects the most informative view, and then uses the chosen view to reconstruct multiple networks with a GCN-based encoder and multiple decoders.
DMGI [25]	It learns the view-specific representation by maximizing the mutual information between the global representation and the local patches of the network, and designs a regularization framework to obtain the consensus embedding.

Comparison with the state-of-the-art methods

Baseline methods

As shown in Table 3, the performance of MGEGFP is compared against seven state-of-the-art algorithms, including five baseline models dedicated to gene function annotation: SNF [16], DeepNF [20], Mashup [1], DeepMNE-CNN [9], EnMUGR [19] and two representative unsupervised multiple graphs embedding methods: One2Multi [24], DMGI [25].

For fair comparisons, all the baseline models are equipped with the same off-the-shelf LightGBM classifier in the downstream function prediction task, and the parameters are tuned to achieve the best performance for each model. For DeepMNE-CNN, we always use the originally designed CNN classifier. Besides, the initial features in One2Multi and DMGI are also learned by RWR as well as MGEGFP for fairness. Moreover, for One2Multi, we choose the densest network as the input graph, which is coexpression for both yeast and human datasets, and adopt the Bilinear decoder used in the original paper for reconstruction task.

Performance on yeast dataset

Intra-view

We firstly show the performance of intra-view learning for yeast dataset. As shown in Figure 3, MGEGFP outperforms all other methods in cooccurrence, database, experiment and neighborhood networks, and achieves the second in coexpression and fusion networks. Take cooccurrence network as an example, the m-aupr value of MGEGFP achieves 0.394, which is higher than DeepNF (0.330), Mashup (0.378), DeepMNE-CNN (0.355), EnMUGR (0.372) and DMGI (0.366). Besides, it can be seen that the GCN-based methods (MGEGFP and DMGI) consistently perform well in all individual networks. Especially in fusion network, MGEGFP and DMGI achieve 0.568 and

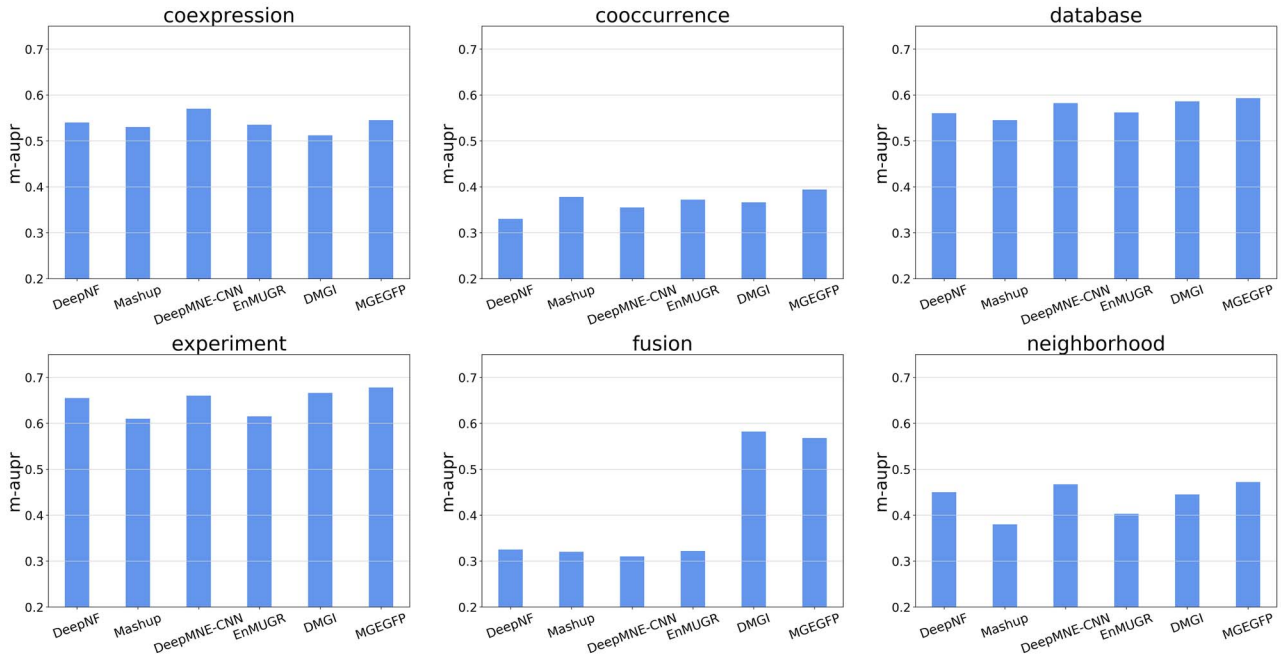

Figure 3. The performance of intra-view gene function annotation on yeast dataset. We take level 1 as a concrete example to illustrate. We compare our method with DeepNF, Mashup, DeepMNE-CNN, EnMUGR and DMGI. The x-axis denotes the methods, and the y-axis denotes the value of m-aupr.

0.582 for m-aupr value, which outperform other models by a wide margin. The overall performance of intra-view learning indicates that GCN is suitable for extracting features in gene networks. This is a great foundation for the subsequent inter-view learning to obtain more comprehensive embeddings for the downstream annotation task.

Inter-view

We then report the average value along with standard deviation of each metric across various models in Table 4, and the *P*-value of significance test in the results between MGEGFP and other baseline models is shown in Supplementary Table S7. The results show a considerable improvement in the function prediction performance with MGEGFP. In level 1, we can observe MGEGFP significantly performs better than all comparing methods in m-aupr, M-aupr, f1 and subset 0–1 loss (t-test, *P*-value<0.05, Supplementary Table S7), while still has a competitive performance in acc. Compared with DeepMNE-CNN, MGEGFP obtains a significantly better performance by an improvement of 2.8%, 1.6%, 3.1% and 4.5% on f1, m-aupr M-aupr and subset 0–1 loss. For level 2 and level 3 task, MGEGFP still shows a better capacity for function annotation. Moreover, MGEGFP consistently outperforms other methods by a wide margin on the strictest metric subset 0–1 loss, which considers a gene to be incorrectly classified if the predicted classes do not entirely match all the true labels. This suggests that our method can more accurately annotate all functional classes of each gene at the same time.

Besides, among GCN-based methods, MGEGFP achieves clearly better results. One2Multi only uses a single view for encoding, which may result in losing a lot of important information. Although DMGI achieves competitive performance in intra-view learning, it does not fully take into account the relationship among views during inter-view fusion process.

In summary, the experimental results compared with all the methods demonstrate that MGEGFP can effectively fuse the information of multiple views and extract more comprehensive features which are helpful for function prediction. In addition, a comparison with the results of intra-view learning in Figure 3 shows that the performance after integrating multiple graphs is obviously better than the performance using single network, which proves the effectiveness of the multi-view collaborative learning.

Model complexity and running time analysis Similar to the calculation process in [55] and [56], we analyze the computational complexity for two key components of our MGEGFP framework. For the dual-channel GCN encoder, $O(L\left|E^{(m)}\right|(d_s+d_c)+LN(d_s^2+d_c^2))$ calculations are needed to obtain the gene representations in each view, where $\left|E^{(m)}\right|$ represents the number of edges in view m. For our designed multi-gate module, the time complexity is $O(NM^2(d_s+d_c))$.

To further evaluate the running time of MGEGFP, we conduct a running time comparison with other baseline methods. All the experimental results reported are conducted on a server with a Tesla P100 GPU under CentOS Linux release 7.6.1810. The baseline methods are re-implemented according to their publicly released code and we use the default hyperparameter settings as recommended. The results are shown in Figure 4. It can be seen that non-deep learning-based methods (SNF, Mashup and EnMUGR) obviously run faster than other deep learning-based methods. This is reasonable because deep learning-based models tend to be more complex,

Table 4. The comparison results of MGEGFP and other baseline methods for yeast dataset. The mean value and standard deviation of each metric are computed from 10 random data splits. For each metric, we bold the best performance, and the runner-up is underlined. For the subset 0–1 loss, the lower value indicates the better model performance. In contrast, for other metrics the higher values represent better predictive performance.

	Methods	acc (↑)	f1 (↑)	m-aupr (↑)	M-aupr (↑)	subset 0–1 loss (↓)
level 1	SNF	0.785 (0.015)	0.566 (0.008)	0.701 (0.018)	0.550 (0.016)	0.751 (0.012)
	DeepNF	0.787 (0.012)	0.578 (0.010)	0.711 (0.008)	0.577 (0.012)	0.733 (0.017)
	Mashup	0.811 (0.011)	0.585 (0.012)	0.727 (0.011)	0.601 (0.020)	0.724 (0.015)
	DeepMNE-CNN	**0.837 (0.016)**	0.593 (0.015)	0.746 (0.009)	0.611 (0.018)	0.713 (0.015)
	EnMUGR	0.786 (0.018)	0.561 (0.006)	0.701 (0.009)	0.581 (0.004)	0.735 (0.008)
	One2Multi	0.788 (0.011)	0.572 (0.018)	0.721 (0.006)	0.583 (0.013)	0.745 (0.018)
	DMGI	0.791 (0.011)	0.583 (0.005)	0.717 (0.005)	0.590 (0.021)	0.704 (0.015)
	MGEGFP	0.833 (0.015)	**0.621 (0.014)**	**0.762 (0.019)**	**0.642 (0.022)**	**0.668 (0.022)**
level 2	SNF	0.742 (0.021)	0.520 (0.012)	0.613 (0.011)	0.439 (0.018)	0.818 (0.019)
	DeepNF	0.745 (0.015)	0.530 (0.012)	0.633 (0.005)	0.456 (0.013)	0.801 (0.007)
	Mashup	0.772 (0.013)	0.528 (0.009)	0.630 (0.018)	0.454 (0.011)	0.807 (0.001)
	DeepMNE-CNN	0.782 (0.018)	0.550 (0.008)	0.670 (0.012)	**0.492 (0.012)**	0.788 (0.008)
	EnMUGR	0.761 (0.011)	0.515 (0.016)	0.622 (0.008)	0.443 (0.018)	0.819 (0.016)
	One2Multi	0.751 (0.011)	0.519 (0.016)	0.622 (0.009)	0.461 (0.019)	0.812 (0.008)
	DMGI	0.745 (0.015)	0.521 (0.008)	0.625 (0.016)	0.431 (0.012)	0.805 (0.013)
	MGEGFP	**0.804 (0.013)**	**0.553 (0.009)**	**0.691 (0.011)**	0.481 (0.012)	**0.753 (0.009)**
level 3	SNF	0.674 (0.011)	0.472 (0.018)	0.548 (0.020)	0.329 (0.014)	0.821 (0.005)
	DeepNF	0.715 (0.018)	0.508 (0.012)	0.597 (0.008)	0.363 (0.019)	0.798 (0.014)
	Mashup	0.726 (0.015)	0.501 (0.009)	0.601 (0.018)	0.362 (0.004)	0.809 (0.016)
	DeepMNE-CNN	0.739 (0.019)	0.520 (0.008)	0.622 (0.009)	0.373 (0.012)	0.791 (0.011)
	EnMUGR	0.728 (0.016)	0.511 (0.009)	0.624 (0.012)	0.368 (0.017)	0.788 (0.014)
	One2Multi	0.711 (0.013)	0.499 (0.019)	0.605 (0.008)	0.352 (0.014)	0.808 (0.006)
	DMGI	0.702 (0.008)	0.498 (0.012)	0.602 (0.011)	0.358 (0.017)	0.801 (0.009)
	MGEGFP	**0.743 (0.019)**	**0.533 (0.012)**	**0.643 (0.011)**	**0.379 (0.016)**	**0.757 (0.009)**

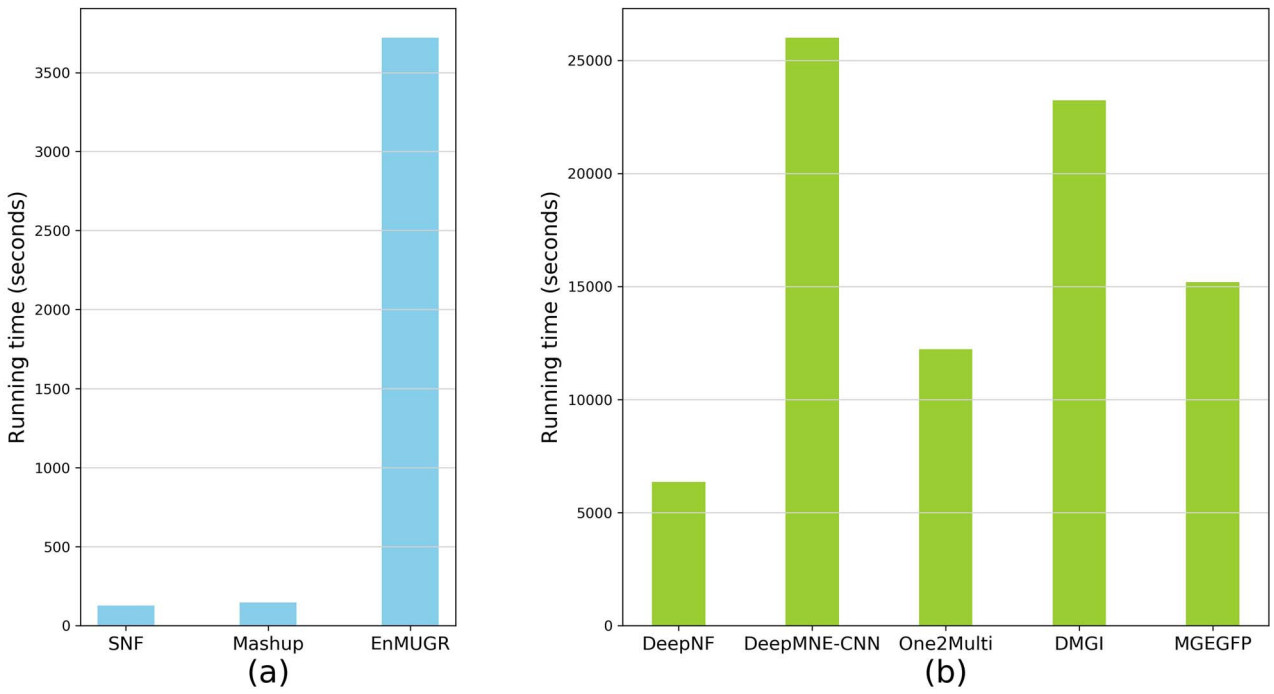

Figure 4. Comparison of running time. **(A)** Non-deep learning-based methods. **(B)** Deep learning-based methods. The running time of deep learning-based methods is longer, which can be attributed to that these models usually have a large number of parameters to train and need hundreds of epochs to converge.

which have a large number of parameters to train and take hundreds of epochs to converge.

Among deep learning-based methods, DeepNF has the shortest running time, but its performance in the downstream functional prediction task is inferior to MGEGFP and DeepMNE-CNN. Besides, compared with DeepMNE-CNN, which is the best-performing baseline model among deep learning-based methods, the running

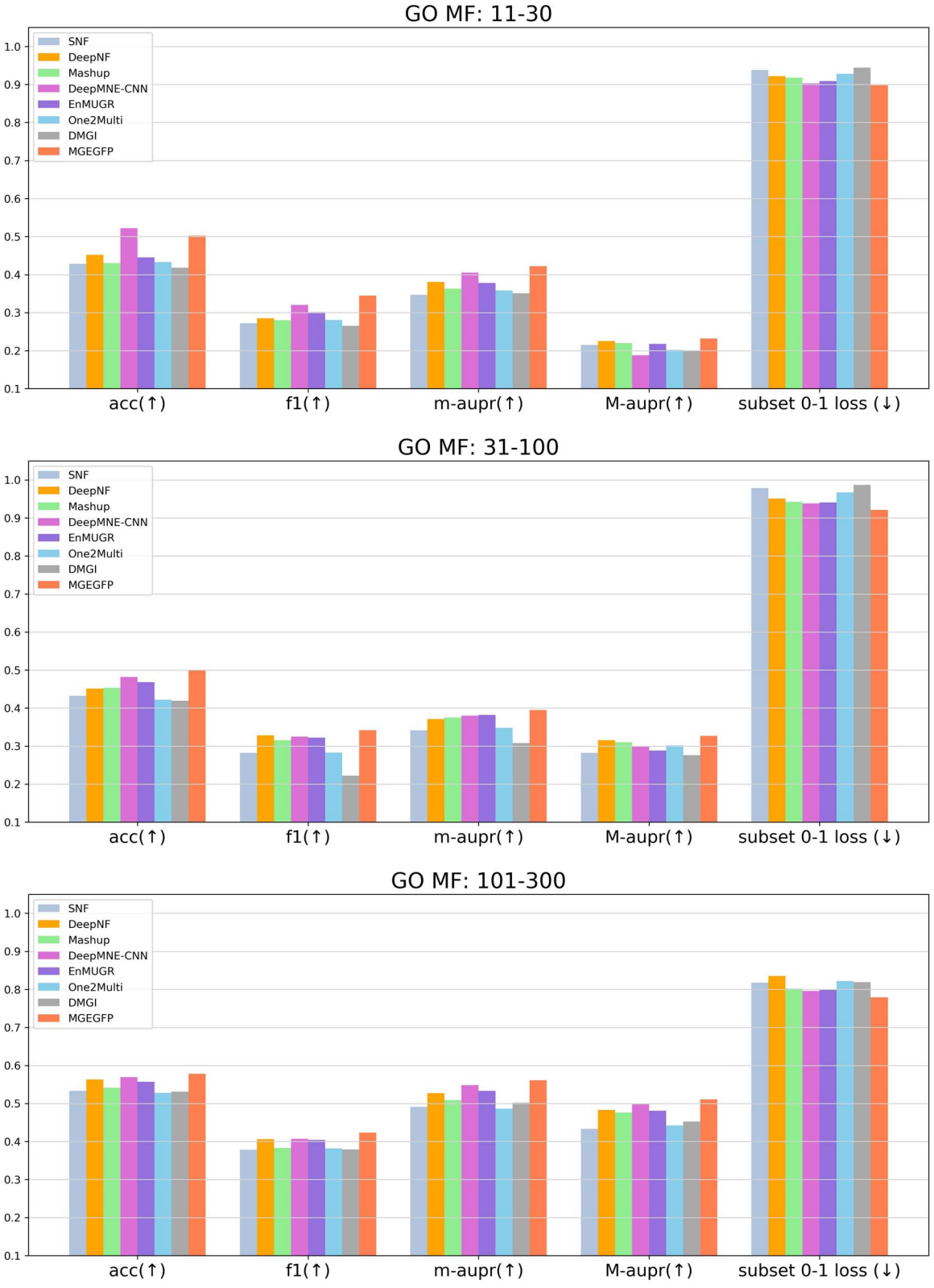

Figure 5. The predictive performance of MGEGFP and other methods for GO Molecular Function terms on human dataset. Three subsets are used: 11–30, 31–100, 101–300. The x-axis represents various metrics, and the y-axis shows the value of corresponding metrics. Except for subset 0–1 loss, the higher values mean better model capability.

time of MGEGFP is shorter. This might be attributed to that DeepMNE-CNN is an iteratively stacked model, and the process of constraints extraction for semi-supervised autoencoder is time-consuming. In conclusion, considering the better performance of MGEGFP on gene function prediction, such a running time is acceptable.

Performance on human dataset

The experimental results of MGEGFP and other baseline models on human dataset are presented in Figure 5. We present the performance for GO Molecular Function here, and the detailed results for Cellular Component and Biological Process terms can be found in Supplementary Figure S1 and Figure S2.

In 11–30 subset of MF, MGEGFP obtains the best result in four metrics. In terms of f1 and M-aupr, MGEGFP outperforms DeepMNE-CNN by 7.81% and 23.40%, and outperforms EnMUGR by 14.62% and 6.42%. In 31–100 mini dataset, MGEGFP outperforms all comparing models in all five metrics. The subset 0–1 loss of MGEGFP is 0.921, which is lower than 0.942 for Mashup, 0.941 for EnMUGR and 0.938 for DeepMNE-CNN. Moreover, for 101–300 subset, our method also achieves best results among all methods, and shows a considerable improvement in all metrics.

Overall, the MGEGFP performance on human dataset is consistent with the yeast dataset, which validates the effectiveness of our model. In contrast to other methods, MGEGFP adequately exploits the neighborhood information in the graph and uses GCN to capture latent patterns. The dual-channel GCN encoder explicitly models both the view-specific information and consensus pattern, which leads to a more comprehensive embedding. In addition, the multi-gate module effectively leverages the collaborative correlation between views. Together, these advantages enable MGEGFP to learn a more accurate representation and have a better function prediction performance.

Discussion

Motivated by the observations that genes with similar topological roles are more likely to share correlated functions, we propose MGEGFP here, which employs GCN to extract embeddings from multiple heterogeneous networks for gene function prediction, and constructs adaptive contribution estimation model based on disentangled representation. GCN is capable of capturing proximity relationship between nodes, and leads to a highly predictive representation for functional annotation. More importantly, the designed dual-channel GCN encoder can simultaneously disentangle both the complementary and common information across multiple views, which leads to a more comprehensive representation. Furthermore, the multi-gate module together with diversity preservation constraint are designed to adaptively estimate the different contributions of each individual network during each reconstruction process, and can help better model the correlations of diverse networks. These designed mechanisms together lead to a more accurate gene representation which is finally fed into an out-of-the-box classifier to predict gene functions.

Experimental results demonstrate MGEGFP outperforms all the state-of-the-art methods, and manifest the effectiveness of our model. We further perform an ablation study to manifest the effectiveness of the major components in MGEGFP. In the future, we will consider to incorporate more intrinsic biological characteristics of genes, such as sequence and structure knowledge as the input node features to further enhance the predictive performance. Besides, inspired by several methods that focus on alleviating the noise problem based on multiplex networks [57, 58], a possible direction of our future work is to design a more robust model. Furthermore, since GCN-based methods always suffer from the over-smoothing problem, we will try to develop a more efficient model to tackle this problem.

Key points

- To our best knowledge, MGEGFP is the first method which adopts GCN to extract latent features from multi-view gene interaction networks for inferring gene function, which enhances the performance on both intra-view and inter-view learning tasks.
- To jointly disentangle and extract view-specific and consensus semantic information, a dual-channel GCN encoder is designed for each view.
- We devise a multi-gate module for adaptive estimation to take advantage of the correlations of multiple graphs, which leads to a more comprehensive representation for genes. Furthermore, we propose a corresponding diversity preservation constraint to alleviate the over-fitting problem.
- The experimental results demonstrate that MGEGFP achieves the best performance in both yeast and human datasets against all other state-of-the-art methods, and confirm that our model can be very useful to gene function prediction task.

Acknowledgments

The authors thank the anonymous reviewers for their valuable suggestions.

Supplementary data

Supplementary data are available online at https://academic.oup.com/bib.

Funding

This work was supported by National Natural Science Foundation of China [61973174] and Key project of the Natural Science Foundation of Tianjin City [No. 21JCZDJC00140].

Data Availability

The implementation of MGEGFP is freely available at https://github.com/zhanglabNKU/MGEGFP.

References

1. Cho H, Berger B, Peng J. Compact integration of multi-network topology for functional analysis of genes. *Cell systems* 2016;**3**(6):540–8.
2. Berger B, Peng J, Singh M. Computational solutions for omics data. *Nat Rev Genet* 2013;**14**(5):333–46.
3. Zitnik M, Nguyen F, Wang B, *et al.* Machine learning for integrating data in biology and medicine: Principles, practice, and opportunities. *Information Fusion* 2019;**50**:71–91.
4. Chen Q, Li Y, Tan K, *et al.* Network-based methods for gene function prediction. *Brief Funct Genomics* 2021;**20**(4): 249–57.
5. Re M, Valentini G. Integration of heterogeneous data sources for gene function prediction using decision templates and ensembles of learning machines. *Neurocomputing* 2010;**73**(7–9):1533–7.
6. Mostafavi S, Ray D, Warde-Farley D, *et al.* GeneMANIA: a real-time multiple association network integration algorithm for predicting gene function. *Genome Biol* 2008;**9**(1):1–15.
7. Yu G, Fu G, Wang J, *et al.* Predicting protein function via semantic integration of multiple networks. *IEEE/ACM Trans Comput Biol Bioinform* 2015;**13**(2):220–32.
8. Zhang J, Deng L. Integrating multiple interaction networks for gene function inference. *Molecules* 2018;**24**(1):30.
9. Peng J, Xue H, Wei Z, *et al.* Integrating multi-network topology for gene function prediction using deep neural networks. *Brief Bioinform* 2021;**22**(2):2096–105.
10. Lee I, Blom UM, Wang PI, *et al.* Prioritizing candidate disease genes by network-based boosting of genome-wide association data. *Genome Res* 2011;**21**(7):1109–21.
11. Tsuda K, Shin H, Schölkopf B. Fast protein classification with multiple networks. *Bioinformatics* 2005;**21**(suppl_2):ii59–65.
12. Lanckriet GR, De Bie T, Cristianini N, *et al.* A statistical framework for genomic data fusion. *Bioinformatics* 2004;**20**(16):2626–35.
13. Franceschini A, Szklarczyk D, Frankild S, *et al.* STRING v9. 1: protein-protein interaction networks, with increased coverage and integration. *Nucleic Acids Res* 2012;**41**(D1):D808–15.
14. Wong AK, Krishnan A, Yao V, *et al.* IMP 2.0: a multi-species functional genomics portal for integration, visualization and prediction of protein functions and networks. *Nucleic Acids Res* 2015;**43**(W1):W128–33.
15. Yu G, Rangwala H, Domeniconi C, *et al.* Predicting protein function using multiple kernels. *IEEE/ACM Trans Comput Biol Bioinform* 2014;**12**(1):219–33.
16. Wang B, Mezlini AM, Demir F, *et al.* Similarity network fusion for aggregating data types on a genomic scale. *Nat Methods* 2014;**11**(3):333–7.
17. Mostafavi S, Morris Q. Fast integration of heterogeneous data sources for predicting gene function with limited annotation. *Bioinformatics* 2010;**26**(14):1759–65.
18. Mostafavi S, Morris Q. Combining many interaction networks to predict gene function and analyze gene lists. *Proteomics* 2012;**12**(10):1687–96.
19. Zhang X, Wang W, Ren CX, *et al.* Learning representation for multiple biological networks via a robust graph regularized integration approach. *Brief Bioinform* 2022;**23**(1):bbab409.
20. Gligorijević V, Barot M, Bonneau R. deepNF: deep network fusion for protein function prediction. *Bioinformatics* 2018;**34**(22):3873–81.
21. Ni J, Chang S, Liu X, *et al.* Co-regularized deep multi-network embedding. In: Pierre-Antoine, C *et al.* (eds). *Proceedings of the 2018 World Wide Web Conference*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 2018, 469–78.
22. Sun Y, Wang S, Hsieh TY, *et al.* Megan: A generative adversarial network for multi-view network embedding. In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. International Joint Conferences on Artificial Intelligence Organization, 2019. p. 3527–33.
23. Fu D, Xu Z, Li B, *et al.* A view-adversarial framework for multi-view network embedding. In: d'Aquin M, *et al.* (eds). *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. Association for Computing Machinery, New York, NY, USA, 2020, 2025–8.
24. Fan S, Wang X, Shi C, *et al.* One2multi graph autoencoder for multi-view graph clustering. In: Huang Y, *et al.* (eds). *Proceedings of The Web Conference 2020*. Association for Computing Machinery, New York, NY, USA, 2020, 3070–6.
25. Park C, Kim D, Han J, *et al.* Unsupervised attributed multiplex network embedding. In: Rossi F, *et al.* (eds). *Proceedings of the AAAI Conference on Artificial Intelligence*. The AAAI Press, Palo Alto, California USA, Vol. **34**, 2020, 5371–8.
26. Wu Z, Pan S, Chen F, *et al.* A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems* 2020;**32**(1):4–24.
27. Ying Z, You J, Morris C, *et al.* Hierarchical graph representation learning with differentiable pooling. *Advances in neural information processing systems* 2018;**31**:4800–10.
28. Lee J, Lee I, Kang J. Self-attention graph pooling. In: Xing E, *et al.* (eds). *International conference on machine learning*. PMLR, Long Beach, California, USA, 2019, 3734–43.
29. Veličković P, Cucurull G, Casanova A, *et al.* Graph attention networks. In: *International Conference on Learning Representations*, 2018. pp. 1–12.
30. Hamilton W, Ying Z, Leskovec J. Inductive representation learning on large graphs. *Advances in neural information processing systems.* 2017;**30**:1025–35.
31. Zhang M, Chen Y. Link prediction based on graph neural networks. *Advances in neural information processing systems.* 2018;**31**:5165–75.
32. Kipf TN, Welling M. Variational graph auto-encoders. In: *Conference and Workshop on Neural Information Processing Systems NIPS* 2016;**1050**:1–3.
33. Wang J, Ma A, Chang Y, *et al.* scGNN is a novel graph neural network framework for single-cell RNA-Seq analyses. *Nat Commun* 2021;**12**(1):1–11.
34. Ma Q, Xu D. Deep learning shapes single-cell data analysis. *Nat Rev Mol Cell Biol* 2022;**23**:303–4.
35. Wei L, Ye X, Xue Y, *et al.* ATSE: a peptide toxicity predictor by exploiting structural and evolutionary information based on graph neural network and attention mechanism. *Brief Bioinform* 2021;**22**(5):bbab041.
36. Li J, Cai D, He X. Learning graph-level representation for drug discovery. arXiv preprint arXiv:1709.03741, 2017.
37. Li P, Wang J, Qiao Y, *et al.* An effective self-supervised framework for learning expressive molecular global representations to drug discovery. *Brief Bioinform* 2021;**22**(6):bbab109.
38. Kipf TN, Welling M. Semi-supervised classification with graph convolutional networks. In: *5th International Conference on*

Learning Representations, ICLR 2017, Toulon, France, 2017. Conference Track Proceedings. pp. 1609–16.
39. Milenković T, Pržulj N. Uncovering biological network function via graphlet degree signatures. *Cancer informatics* 2008;**6**:CIN–S680.
40. Sharan R, Ulitsky I, Shamir R. Network-based prediction of protein function. *Mol Syst Biol* 2007;**3**(1):88.
41. You R, Yao S, Mamitsuka H, *et al.* DeepGraphGO: graph neural network for large-scale, multispecies protein function prediction. *Bioinformatics* 2021;**37**(Supplement_1):i262–71.
42. Gligorijević V, Renfrew PD, Kosciolek T, *et al.* Structure-based protein function prediction using graph convolutional networks. *Nat Commun* 2021;**12**(1):1–14.
43. Zhao C, Liu T, Wang Z. PANDA2: protein function prediction using graph neural networks. *NAR Genomics and Bioinformatics* 2022;**4**(1):1–11.
44. Tong H, Faloutsos C, Pan JY. Fast random walk with restart and its applications. In: Christopher W. Clifton, *et al.* (eds). *Sixth international conference on data mining (ICDM'06)*. Institute for Electrical and Electronics Engineers (IEEE), New York City, United States, 2006, 613–22.
45. Cao M, Pietras CM, Feng X, *et al.* New directions for diffusion-based network prediction of protein function: incorporating pathways with confidence. *Bioinformatics* 2014;**30**(12): i219–27.
46. He X, Deng K, Wang X, *et al.* Lightgcn: Simplifying and powering graph convolution network for recommendation. In: Huang J, *et al.* (eds). *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. Association for Computing Machinery, New York, NY, USA, 2020, 639–48.
47. Xu K, Li C, Tian Y, *et al.* Representation learning on graphs with jumping knowledge networks. In: Bach F, *et al.* (eds). *International Conference on Machine Learning*. PMLR, Stockholmsmässan, Stockholm SWEDEN, 2018, 5453–62.
48. Zhang Y, Huang Q, Zhang B, *et al.* Deep multiview clustering via iteratively self-supervised universal and specific space learning. *IEEE Transactions on Cybernetics* 2021;1–13. https://doi.org/10.1109/TCYB.2021.3086153.
49. Ke G, Meng Q, Finley T, *et al.* Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*. 2017;**30**:3146–54.
50. Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks. In: Yee Whye Teh, *et al.* (eds). *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings. JMLR, Inc. and Microtome Publishing, United States, 2010. p. 249–56.
51. Kingma DP, Ba J. Adam: A method for stochastic optimization. In: *3rd International Conference on Learning Representations*. San Diego, CA, USA, 2015. arXiv.org, Ithaca, NY. pp. 1–13.
52. Srivastava N, Hinton G, Krizhevsky A, *et al.* Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 2014;**15**(1):1929–58.
53. Ruepp A, Zollner A, Maier D, *et al.* The FunCat, a functional annotation scheme for systematic classification of proteins from whole genomes. *Nucleic Acids Res* 2004;**32**(18):5539–45.
54. Ashburner M, Ball CA, Blake JA, *et al.* Gene ontology: tool for the unification of biology. *Nat Genet* 2000;**25**(1):25–9.
55. Chen M, Wei Z, Ding B, *et al.* Scalable graph neural networks via bidirectional propagation. *Advances in neural information processing systems* 2020;**33**:14556–66.
56. Vaswani A, Shazeer N, Parmar N, *et al.* Attention is all you need. *Advances in neural information processing systems*. 2017;**30**:5998–6008.
57. Gligorijević V, Panagakis Y, Zafeiriou S. Non-negative matrix factorizations for multiplex network analysis. *IEEE Trans Pattern Anal Mach Intell* 2018;**41**(4):928–40.
58. Xiong H, Yan J, Pan L. Contrastive Multi-View Multiplex Network Embedding with Applications to Robust Network Alignment. In: Zhu F, *et al.* (eds). *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. Association for Computing Machinery, New York, NY, USA, 2021, 1913–23.