GCN-MF: Disease-Gene Association Identification By Graph Convolutional Networks and Matrix Factorization

Peng Han
King Abdullah University of Science and Technology
peng.han@kaust.edu.sa

Peng Yang
Cognitive Computing Lab
Baidu Research USA
yangpeng1985521@gmail.com

Peilin Zhao*
Tencent AI Lab
peilinzhao@hotmail.com

Shuo Shang*
University of Electronic Science and Technology of China
Inception Institute of Artificial Intelligence
jedi.shang@gmail.com

Yong Liu
Alibaba-NTU Singapore Joint Research Institute, Nanyang Technological University
stephenliu@ntu.edu.sg

Jiayu Zhou
Michigan State University
jiayuz@msu.edu

Xin Gao
King Abdullah University of Science and Technology
xin.gao@kaust.edu.sa

Panos Kalnis
King Abdullah University of Science and Technology
panos.kalnis@kaust.edu.sa

ABSTRACT

Discovering disease-gene association is a fundamental and critical biomedical task, which assists biologists and physicians to discover pathogenic mechanism of syndromes. With various clinical biomarkers measuring the similarities among genes and disease phenotypes, network-based semi-supervised learning (NSSL) has been commonly utilized by these studies to address this class-imbalanced large-scale data issue. However, most existing NSSL approaches are based on linear models and suffer from two major limitations: 1) They implicitly consider a local-structure representation for each candidate; 2) They are unable to capture nonlinear associations between diseases and genes. In this paper, we propose a new framework for disease-gene association task by combining *Graph Convolutional Network* (GCN) and matrix factorization, named GCN-MF. With the help of GCN, we could capture nonlinear interactions and exploit measured similarities. Moreover, we define a margin control loss function to reduce the effect of sparsity. Empirical results demonstrate that the proposed deep learning algorithm outperforms all other state-of-the-art methods on most of metrics.

CCS CONCEPTS

• **Computing methodologies** → *Semantic networks.*

*Corresponding Author.

KDD '19, August 4–8, 2019, Anchorage, AK, USA
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6201-6/19/08…$15.00
https://doi.org/10.1145/3292500.3330912

KEYWORDS

graph convolutional networks; deep learning; disease-gene association

ACM Reference Format:
Peng Han, Peng Yang, Peilin Zhao, Shuo Shang, Yong Liu, Jiayu Zhou, Xin Gao, and Panos Kalnis. 2019. GCN-MF: Disease-Gene Association Identification By Graph Convolutional Networks and Matrix Factorization. In *The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '19), August 4–8, 2019, Anchorage, AK, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3292500.3330912

1 INTRODUCTION

Identifying disease genes from human genome is an important and fundamental problem in biomedical research [9, 15, 31, 43]. Despite many publications of machine learning methods have been applied to discover new disease genes, it still remains a challenge. Because the set of genes pleiotropy is large, and the number of confirmed disease genes among whole genome and the genetic heterogeneity of diseases is limited. Recent approaches have applied the concept of 'guilty by association' to investigate the association between a disease phenotype and its causative genes, which means that candidate genes with similar characteristics as known disease genes are more likely to be associated with diseases.

However, due to the imbalance issues (few genes are experimentally confirmed as disease related genes within human genome) in disease-gene identification, semi-supervised approaches, like label propagation approaches and positive-unlabeled learning, are widely used to identify candidate disease-gene links [27, 28]. These methods make use of unknown genes for training typically in the scenario of a small amount of confirmed disease-genes (labeled data) with a large amount of unknown genome (unlabeled data). The performance of disease-gene association models are limited by potential bias of single learning models, incompleteness and noise of single biological data sources. Therefore ensemble learning models are applied via combining multiple diverse biological sources and learning models to obtain better predictive performance.

To learn a predictable model for disease-gene association, some works utilize matrix factorization model [23, 35], which is very popular in the field of recommendation system [40, 41]. In the traditional recommendation task, users and items are represented by latent vectors, which would be learned to get the final recommendation preference. The score, computed by the inner product of one user and one item latent vectors, denotes the preference of this user to that item. Similarly, the relationship between one gene and one disease is computed by the inner product of their latent vectors with matrix factorization method. To utilize high-dimension features, some work [30] projected the features into lower dimension latent vectors before the inner product. However, matrix factorization method only utilize the liner relationship between samples, which cannot truly reflect the complicated relationship between genes and diseases.

With the ability of exploiting the relationship between vertexes in a network, graph-based method has been studied in many areas for its splendid performance by utilizing contextual information. For many graph-based algorithms, Laplacian graph regularization is utilized as the part of loss function, which makes the samples with high similarity have the same labels. However, the similarity in graph may not directly reflect the correlation of labels. To exploit adding information in the graph, Graph convolutional networks(GCN) [22] combined Laplacian graph regularization and artificial neural networks, which outperformed other methods with a large margin in the semi-supervised classification task. By replacing the computation of eigenvectors with the approximation, they reduce a lot of computation time when the graph is too large.

In our work, to learn complicated non-linear relationship from various data sources, we combine GCN and matrix factorization to generate a new framework for disease-gene association task, named GCN-MF. We exploit different data sources to construct the gene-gene similarity and disease-disease similarity. Then we truncate the similarity graph with a fixed number k to generate the k-nn graph, which could reduce a lot of computation and noise. Given a gene and a disease, we use GCN of the corresponding k-nn graph to project their features into new spaces. The computation of the projection includes approximated convolution and non-linear functions. The predicted score of the relationship between one gene and one disease is obtained by the inner product of the corresponding latent vectors. To optimize our model, we use the Frobenius norm between the predicted score matrix and true association labels. Adam method [21] is utilized to train the model with fix learning rate. From the experiments on real dataset, we could see that our framework outperforms all other state-of-the-art methods on most of metrics with a large margin.

Our contributions could be summarized as follows:

- We propose a new framework for disease-gene association by combining GCN and matrix factorization.
- We construct similarity graphs for genes and diseases respectively, based on the domain knowledge from various data sources.
- We analyze different factors that influence the performance of our method.
- We have performed extensive experiments on real datasets, and the experimental results indicate that the proposed method outperform other state-of-the-art algorithms in most of evaluation metrics with a large margin.

2 RELATED WORKS

Disease-gene association is a process by which scientists identify the mutant genotypes responsible for an inherited genetic disorder. Mutations in these genes can include single nucleotide substitutions, single nucleotide additions/deletions, deletion of the entire gene, and other genetic abnormalities. In this section, we will give some related works on this problem with network-based method and integration method. Moreover, we will give recent works about graph convolutional networks.

2.1 Network-based Approaches

Functionally related genes may have physically interactions in the biological networks. Network-based learning approaches have applied to detect disease-gene associations in the biological networks [26, 36, 38]. The underlying assumption in these studies is that the interacting partners of disease related genes in the network are likely to cause either the same or similar diseases [17]. These methods construct a biological network and subsequently compute the closeness between candidate genes and known disease genes based on network topological information. In particular, Wu et al. [37] built a regression model measuring the correlation between phenotype similarity and gene closeness in the PPI network for prioritizing candidate disease genes. Li et al. [26] designed a global network-based method by formulating constraints on the genes score function that smooth over the whole network.

2.2 Integration Approaches

As more genes are being sequenced and annotated, and gene/protein interaction data are accumulating, an ever-increasing wealth of biological data is now available in public databases. Each data source covers part of the human genome, therefore these data sources are complementary to each other. Adie et al. [2] proposed an integration framework, where gene annotation, gene expression and protein sequence data were used to prioritize disease candidate genes. Recent studies [10, 39] integrated data from several ontologies to discover disease genes associated with disease phenotypes that were of interest to users. Barabasi et al. [5] compiled a functional human gene network that comprised known interactions derived from different databases, i.e., the Kyoto Encyclopedia of Genes and Genomes (KEGG) [19], the Biomolecular Interaction Network Database (BIND) [4], Reactome [18], and the Human Protein Reference Database (HPRD) [11] using a Bayesian classifier. Our proposed approach seamlessly integrates different types of features and therefore provides better generalization to new diseases and new types of evidence.

2.3 Graph Convolutional Networks

GCN [22] was designed for semi-supervised classification task on the data with graph structure. It was pointed out that the traditional assumption, connected nodes in the graph may share same labels, may constrain the model ability. So instead of using Laplacian regularization explicitly in the objective function, GCN utilized the property of Laplacian regularization in the model structure to exploit additional information. To reduce the time of computation, a truncated Chebyshev polynomials was used to approximate the functions of eigenvalues according to the theory in [13].

To speed up the computation of GCN, many works [7, 8, 12] utilized Monte-Carlo approximation to select the neighbours of the training nodes instead of using all the connected samples. In [12] and [8], they used neighbour sampling strategy that randomly select neighbours which were connected with the training samples. In [7], an importance sampling method was utilized, in which fixed number neighbours of the training samples were selected with the probability that was positively correlated to the corresponding graph weights. For other applications, [42] applied GCN on multimedia recommendation task. In [25], they also applied GCN on similar task, but their prediction is based on the edge decoder and they use cross-entropy loss function to optimize the model, that don't consider the data sparsity problem.

3 PRELIMINARY

In this section, we will illustrate some basic knowledge in this work. Firstly, we will introduce the definition of the problem. Then we will give the introduction of matrix factorization. Finally, we will give the details of graph convolutional networks.

3.1 Problem Definition

Given m diseases and n genes, we use $U = \{u_1, u_2, ..., u_m\}$ and $V = \{v_1, v_2, ..., v_n\}$ to denote the set of diseases and genes respectively, and $R \in \mathbb{R}^{m \times n}$ is the association matrix. The entry $R_{ij} = 1$ if disease u_i is linked with gene v_j, otherwise $R_{ij} = 0$. However, $R_{ij} = 0$ doesn't mean that disease u_i has no relation with gene v_i. It may be the reason that the relationship is not found yet. Moreover, we use $V_i^+ = \{v_j | v_j \in V \text{ and } R_{ij} = 1\}$ to denote the discovered linked set of disease u_i and $V_i^- = V / V_i^+$ to denote the non-linked set. $D = \{(u_i, v_j) | R_{ij} = 1\}$ is defined to present all the linked disease and gene pairs. In our work, for every disease u_i, we want to find the gene v_j that $R_{ij} = 0$ and gene v_j is actually related to disease u_i .

3.2 Matrix Factorization

The task of disease-gene association is similar as recommendation problem, where disease is corresponding to the user, gene is equivalent to item and the certificated association is equal to user's visiting/watching/shopping history. Thus, the matrix factorization method, which is very popular for recommendation task, could be directly applied in our problem. Given the latent vector $\mathbf{p}_i$ of disease u_i and the latent vector $\mathbf{q}_j$ of gene v_j, we could get the predicted association score s_{ij} between disease u_i and gene v_i as follows:

$$s_{ij} = \mathbf{p}_i^\top \mathbf{q}_j. \tag{1}$$

In Eq (1), the higher the s_{ij} is, the more possibly gene v_j is linked to disease u_i.

For many recommendation tasks, the sole data is the implicit feedback, the user's visiting history. And matrix factorization only utilize linear relationship between entities, which makes it fit for this kind of simple data. However, for the disease-gene association problem, the relationship of data information are more complicated. For example, we could extract features from gene and disease by domain knowledge that could reflect the similarity between samples, and relationship between features may be non-linear. So the matrix factorization method couldn't fully exploit the data information.

Deep learning is the most popular topic in recent years. Because it could achieve high performance in many applications, and the artificial neural networks could be easily designed and optimized under many frameworks. The key point of high performance in deep learning is that they could exploit the high order and non-linear relationship between instances and labels. Combining deep learning and matrix factorization, NeuMF [14] proposes a deep learning matrix factorization framework for recommendation task with implicit feedback. The goal of NeuMF is to learn embeddings of entities and the regularization weights. With the non-linear functions and layer combinations, they could learn more complicated information from the data. However, their model cannot utilize the contextual information, such as the similarity between different entities and hand-crafted features with domain knowledge.

3.3 Graph-based Methods

Graph-based model is popular in the area of semi-supervised classification, as it could utilize the similarity between different samples. Most of works on this topic assume that similar entities may share the same label. To apply this assumption, many of them use Laplacian regularization as a constraint and combine it into the loss function as followed:

$$L = \|Y - R\|_F + \|R \Delta R^\top\|_F, \tag{2}$$

where the Laplacian term $\|R\Delta R^\top\|_F = \sum_{ij} G_{ij} \|R_{i:} - R_{j:}\|_2$, Y is the label matrix, R is the predicted score matrix, G is the adjacent graph, $\Delta = D - G$ and D is a diagnose matrix where $D_{ii} = \sum_j G_{ij}$. However, in [22], it is pointed out that the Laplacian regularization added in the loss function is too strong to exploit adding information in the adjacent graph.

For the adjacent graph in many applications, the score in the graph may not directly reflect the relations of labels between samples and may contain more information. For example, in semi-supervised semantic segmentation task, people would use Gaussian distance between features as the similarity between different superpixels [29]. However, the distance between features cannot directly reflect the label correlations. Because the same object may be decomposed into different superpixels with different patterns, then the image features would vary a lot. And for many different objects, they may include many superpixels with same patterns. So if the similarity in the graph may not directly reflect the correlations of labels, the constraint of Laplacian regularization term in the loss function would be too strong, and cannot exploit extra information in the graph.

To exploit more potential correlation between samples in the graph, Thomas et al. [22] propose a new model which combines first-order approximation of spectral graph convolutions and artificial neural networks, named graph convolutional network (GCN). Instead of adding the Laplacian regularization in the loss function, they utilize the propagation structure in the model to learn new representation of samples.

The key point of GCN is to propagate information between connected samples during the optimization process. For most of previous works about deep learning, the new embedding would only be learned by the feature and label of the corresponding sample. In GCN, to get the new representation of one sample in every layer, the embeddings of the connected samples in the last layer will also be utilized with the corresponding weights. Given the l-th hidden layer $H^{(l)}$ and similarity graph G, GCN would compute the $(l+1)$-th hidden layer $H^{(l+1)}$ as

$$H^{(l+1)} = \sigma(\tilde{A}H^{(l)}W^{(l)}), \tag{3}$$

where $\tilde{A} = \tilde{D}^{-\frac{1}{2}}A\tilde{D}^{-\frac{1}{2}}$, $A = I + G$, $\tilde{D}$ is a diagonal matrix that $\tilde{D}_{ii} = \sum_j A_{ij}$ and I is an identity matrix. $W^{(l)}$ is the learnable weight matrix and σ is the activation function (e.g. *ReLu* function).

4 GCN-MF

In this section, we will give our framework that combines matrix factorization and GCN to solve the disease-gene association task. The problem of disease-gene is similar to the recommendation task. For recommendation task, many works assume that similar users may be interested in the similar items, and similar items may be attractive for the similar users. That is the reason why label propagation method could be successful as the embedding method for recommendation. And many works try to make the latent vectors between similar entities have same pattern by this assumption. However, for disease-gene association task, the genes and diseases similarity contain a lot of noises and the relationship is more sparse. For the movie recommendation task, one popular movie may be liked by thousands of users. But in our problem, one disease may be only linked to three or less (on average 1.231) genes. Instead of assuming the label propagation between similar entities, we try to extract more complicated relationship from similarity graphs. To achieve this goal, we combine GCN and matrix factorization to exploit more information and achieve state-of-the-art performance.

4.1 Model Architecture

To exploit adding information in the similarity graph, we use GCN to extract mutual relation between entities. Given the k_u dimension feature $\mathbf{p}_i$ of disease u_i,the features $\{\mathbf{p}_{i1}, \mathbf{p}_{i2}, \ldots, \mathbf{p}_{ik}\}$ of u_i's connected disease $\{u_{i1}, u_{i2}, \ldots, u_{ik}\}$ and the corresponding normalized similarity weight $\tilde{A}(u)_{i,i1}, \tilde{A}(u)_{i,i2}, \ldots, \tilde{A}(u)_{i,ik}$, we could get the d dimension embedding $\tilde{\mathbf{p}}_i$ of gene u_i as:

$$\tilde{\mathbf{p}}_i = \sigma(\tilde{A}(u)_{i,i}\mathbf{p}_i + \sum_{j=1}^{k} \tilde{A}(u)_{i,ij}\mathbf{p}_{ij})^\top W(u), \tag{4}$$

where $W(u) \in \mathbb{R}^{k_u \times d}$ is the learned parameter to project the combined features into new space and σ is the non-linear activation function, such as *Relu*. Similarly, given k_v dimension feature $\mathbf{q}_j$ of gene v_j, we could get the embedding $\mathbf{q}_j$ for gene v_j as:

$$\tilde{\mathbf{q}}_j = \sigma(\tilde{A}(v)_{j,j}\mathbf{q}_j + \sum_{i=1}^{k} \tilde{A}(v)_{j,ji}\mathbf{q}_{ji})^\top W(v), \tag{5}$$

where $W(v) \in \mathbb{R}^{k_v \times d}$ is the learned parameter for gene's GCN component. Given the disease features matrix $P \in \mathbb{R}^{m \times k_u}$ and gene features matrix $Q \in \mathbb{R}^{n \times k_V}$, we could get their embeddings from GCN as:

$$\begin{aligned} \tilde{P} &= \sigma(\tilde{A}(u)PW(u)), \\ \tilde{Q} &= \sigma(\tilde{A}(v)QW(v)), \end{aligned} \tag{6}$$

where $\tilde{P} \in \mathbb{R}^{m \times d}$ and $\tilde{Q} \in \mathbb{R}^{n \times d}$ are the new representation of diseases and genes respectively. Once we get the embeddings, we could get the preference matrix $S \in \mathbb{R}^{m \times n}$ as

$$S = clip(\tilde{P}\tilde{Q}^\top), \tag{7}$$

where $clip(x)$ is the clip function to make the value of the elements stay in $[0, 1]$. The i-th row in S represents the preference score of disease u_i. In reality, we could test only the top-k genes for a specific disease, which would save a lot of time. And new verified associations could also be added as labels in our model to improve the overall accuracy.

4.2 ϵ-Insensitive Loss Function

There are two popular kinds of loss functions for matrix factorization based methods. One is the pair-wise loss which tries to make positive items rank higher than negative ones. The other is to minimize the Frebious norm of the difference between preference matrix and label matrix. However, our application does not fit for the pair-wise loss. Because most of genes are not linked to any disease, which make them totally out of circulation with the training process. So we couldn't find new associations for unlinked genes with a high probability by pair-wise loss. But for the Frebious norm loss, the sparsity of the dataset would influence the accuracy.

Compare to the number of disease-gene pairs, the number discovered association is far less, which makes the association matrix R extremely sparse. For the dataset we use, there are only 3954 known disease-gene associations, where there are 3209 diseases and more than 10,000 kinds of genes. To overcome this, we control the margin between the predicted score and the label with a fixed variable ϵ. We construct an auxiliary label matrix $\hat{R}$ as follows:

$$\hat{R}_{i,j} = \begin{cases} R_{i,j} & \text{if} \quad R_{i,j} == 1; \\ \epsilon & \text{otherwise.} \end{cases} \tag{8}$$

Thus, instead of minimizing the difference between predicted score matrix S and original association matrix R, we use this auxiliary matrix $\hat{R}$ as the objective goal. With the mean square error of every row, we could get our loss as follows:

$$L = ||S - \hat{R}||_F^2. \tag{9}$$

By controlling the margin of the loss and utilizing the Frebious loss, we could handle the sparsity of disease-gene association dataset.

4.3 Weight Regularization

The drawback of the data sparsity in the deep learning is that it would cause overfitting. To avoid overfitting, dropout is utilized in many works. With a fixed probability, this technology would set each dimension of the weight as zero in the training process randomly. So for every epoch, the model only updates parts of the weights. And most of the works utilizing dropout includes a large training set, then the instability caused by dropout would

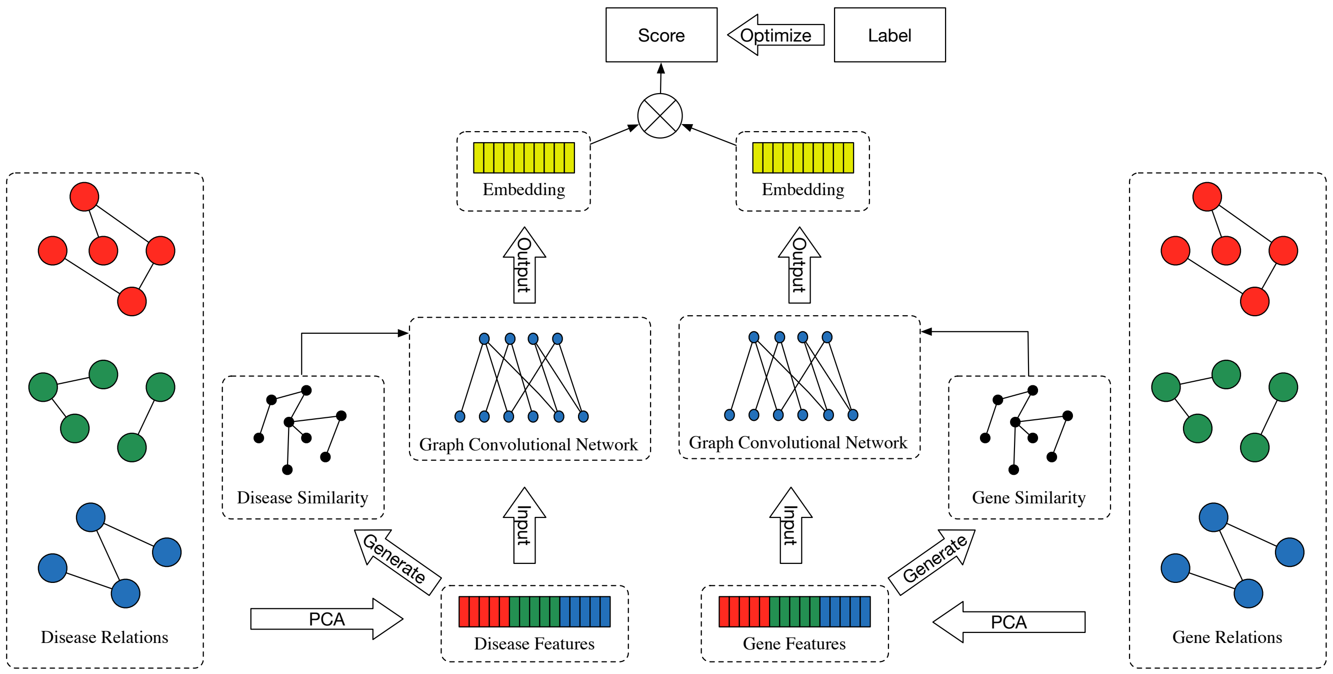


Figure 1: Framework of GCN-MF. We firstly extract the features of genes and diseases. Then their features are given to the GCNs constructed by their corresponding similarity graphs. After that, we could get the embeddings of genes and disease. Finally, we could get the predicted score matrix with the inner product of them. The objective function is to minimize the difference between the predicted score and discovered disease-gene association.

be neutralized by enough data. However, the instability caused by dropout in our dataset will result in worse consequence than overfitting, which could be observed in our experiment results.

Another way to avoid overfitting is adding a regularization term. With the regularization term, the parameters would be limited to be general. Then we could get our loss function as follows:

$$L = \|S - \hat{R}\|_F^2 + \lambda \|[W(u); W(v)]\|_F^2, \tag{10}$$

where λ is the weight to balance the regularization term and $[W(u); W(v)]$ is the concatenation of $W(u)$ and $W(v)$.

4.4 Graph Construction

The core of GCN is the graph that reflects relationship between entities. From the motivation of GCN [22], we know that the relation in graph doesn't need to denote the label correlation, that the connected samples may not share the same label. And GCN can exploit the extra information from the graph by huge number of parameters and non-linear functions. Both of the properties are suitable for disease-gene association dataset. Firstly, it is difficult to construct graphs for genes and diseases respectively that could propagate the link relation, because the association is not enough to find the explicit patterns. Moreover, there are a lot of works [2, 24] which are specific on finding the relationships in genes or diseases respectively. Many domain knowledge are utilized in these works, which may contain implicit connection to the disease-gene association. With GCN, we could avoid the strong requirements for graph and exploit additional information.

There are many data sources for gene and diseases respectively, which could be used to extract similarity information. To combine them without losing significant information and introducing noises, we do principal component analysis (PCA) on each data source and make concatenation for PCA vectors. The concatenated vectors are used as the features for input and construction of graph. All data sources and details of the feature extraction could be found in Section 5.1.

Once we have obtained the disease features P and gene features Q, we could construct the similarity matrix of diseases and genes respectively. The entry of this matrix is computed as followed:

$$\begin{aligned} Sim(u_i, u_j) &= p_i^\top p_j, \\ Sim(v_i, v_j) &= q_i^\top q_j, \end{aligned} \tag{11}$$

where p_i is i-th row vector of P and q_i is i-th column vector of Q. From the definition of our structure in Eq. 6, we could know that the computation is very time-consuming if we directly use the dense similarity matrix as the graph. Moreover, the dense graph would also incur noises which could harm the model performance. To speedup the computation, some works [7, 8, 12] utilize Monte-Carlo approximation to select the neighbours of the training nodes instead of using all the connected samples. In our work, we avoid the time-consuming computation by constructing a sparse graph. To obtain the sparse graph, we use k-nn neighbours graph to denote the similarity between diseases $G(u) \in \mathbb{R}^{m\times m}$ and genes $G(v) \in \mathbb{R}^{n\times n}$

as following:

$$G(u)_{ij} = \begin{cases} Sim(u_i, u_j) & \text{if} \quad u_j \in \mathcal{K}(u_i) \quad \text{or} \quad u_i \in \mathcal{K}(u_j) \\ 0 & \text{otherwise} \end{cases},$$
$$G(v)_{ij} = \begin{cases} Sim(v_i, v_j) & \text{if} \quad v_j \in \mathcal{K}(v_i) \quad \text{or} \quad v_i \in \mathcal{K}(v_j) \\ 0 & \text{otherwise} \end{cases},$$
$$(12)$$

where $\mathcal{K}(x)$ is the the neighbour set of entity x. In our work, we select 10 most similar samples to construct the graph. As the k-nn neighbours graph in our model is hand-crafted, the weights of neighbours may not reflect the similarity accurately. So we construct the adjacent matrix A in our work with a weight to control the influence of neighbours as follows:

$$\begin{aligned} A(u) &= I + \alpha_u \text{Norm}(G(u)) \\ A(v) &= I + \alpha_v \text{Norm}(G(v)) \end{aligned} \quad (13)$$

where Norm(G) is the normalization function that every entry will be divided by the l1-norm of its corresponding row vector, α_u and α_v are the weights to control the influences of neighbours for diseases and genes respectively. After that, we could get the approximated convolutional graph kernel for diseases and genes as:

$$\begin{aligned} \tilde{A}(u) &= \tilde{D}(u)^{-\frac{1}{2}} A(u) \tilde{D}(u)^{-\frac{1}{2}} \\ \tilde{A}(v) &= \tilde{D}(v)^{-\frac{1}{2}} A(v) \tilde{D}(v)^{-\frac{1}{2}} \end{aligned} \quad (14)$$

where $\tilde{D}(u)$ and $\tilde{D}(v)$ are diagonal matrices, $\tilde{D}(u)_{i,i} = \sum_j A(u)_{ij}$, and $\tilde{D}(v)_{i,i} = \sum_j A(v)_{ij}$.

4.5 Optimization

We use Adam optimization method to optimize the loss function Eq. (10) by Tensorflow [1]. For Adam, we set the learning rate η with different values, and the parameter selection strategy could be found in Section 5.2.

5 EXPERIMENTS

In this section, we will illustrate all the details of our experiment settings and results. We also give the observations from the comparison of our method and other baselines.

5.1 Dataset and Features

5.1.1 Disease-gene associations. Gene-phenotype associations are assembled from the Online Mendelian Inheritance in Man (OMIM) database [3]. There are 3954 known gene-phenotype associations extracted, spanning 3209 disease phenotypes and 12231 genes.

5.1.2 Gene features. Gene features are from three data sources: (1) Gene expression dataset, obtained from BioGPS (www.biogps.org) and Connectivity Map (www.broadinstitute.org/cmap), includes 8755 genes' expression levels from 4536 different tissue cell types. (2) HumanNet is functional gene network data, which collects 21 data sources to measure the functional associations between genes [24]. (3) Gene orthology is a set of disease-gene associations extracted from 8 different ((non-human) species, collected by [34]. For each data source, we can form a gene association matrix. We perform principal component analysis (PCA) on each matrix to obtain the low-dimensional feature space for data representation. We use the leading 100 singular vectors from each of three matrices as features for genes.

5.1.3 Disease features. Similar with latent features generation for genes, we extract 100 latent disease features from two disease similarity networks: MimMiner [2] and OMIM diseases. As in the method by van Driel [2], we construct the phenotype network by applying a text-mining approach to evaluate the similarity among OMIM phenotypes using Medical Subject Headings (MeSH) controlled vocabulary as standardized phenotypic feature terms. We exploit the logical function to re-weight the similarity score and then use PCA to retain the top 100 principal components of the feature space.

5.2 Experiment Setting

To evaluate the performance of our framework, we use three fold cross validation method. In our dataset, we randomly split discovered associations into three non-intersecting subsets with equal size. For every subset, we set it as the testing set and remaining subsets as training set. For every fold, embedding size k is selected in $\{30, 50, 100, 200\}$, learning rate η is set from $\{0.001, 0.01, 0.1, 1\}$, margin loss parameter ϵ is selected from $\{0.001, 0.01, 0.1\}$, running epoch is chosen in $\{100, 125, 150, 175, 200\}$, regularization term λ is set from $\{0.001, 0.01, 0.1\}$, weight parameters α_u and α_v are set in $\{0.1, 0.2, \dots, \}$. All experiments are repeated for 5 times, each with a different random seed. The results reported are average of the 5 runs.

The accuracy of a recommendation model is measured by using five metrics, namely MAP, AUC, NDCG@N, precision@N and recall@N. The former two metrics are widely adopted for evaluation of ranking accuracy. Because we are usually interested in a few top-ranked items, NDCG@N is used to compare the top-N recommendation performance. Sometimes the sequence of the recommendation list is not important, then precision@N and recall@N are utilized to evaluate the quality of the recommendation list content. In our experiments, we set $N = 10$ for NDCG and $N = \{1, 5, 10, 15\}$ for precision and recall. For each metric, we first compute the accuracy for each disease on the testing data, and then report the averaged accuracy over all diseases.

5.3 Baselines

We evaluate GCN-MF with seven baseline methods: PopRank, WRMF, BPRMF, NeuMF, IMC, Catapult and Katz.

- PopLink is a naïve baseline that link genes to diseases purely based on the popularity of genes.
- WRMF [16] is a state-of-the-art approach designed for item recommendation with users' implicit feedback. It gives an analytical solution of the alternating least square problem for matrix factorization.
- BPRMF [33] is a state-of-the-art approach designed for item recommendation with users' implicit feedback. It is a pairwise approach and has outperformed pointwise methods [16, 32], especially in terms of the ranking metric AUC.
- NeuMF [14] propose a deep learning matrix factorization framework for recommendation task with implicit feedback. To compare with NeuMF fairly, we initialize the embeddings the same features in our model.

Table 1: Experiment Results on OMIM Fold1

Method	MAP	AUC	NDCG@10	P@1	P@5	P@10	P@15	R@1	R@5	R@10	R@15
PopLink	0.0051	0.3917	0.0075	0.0009	0.0021	0.0017	0.0017	0.0008	0.0088	0.0158	0.0229
WRMF	0.0127	0.0742	0.0222	0.0113	0.0047	0.0037	0.0025	0.0094	0.0196	0.0271	0.0271
BPRMF	0.0161	0.4931	0.0179	0.0130	0.0042	0.0025	0.0017	0.0116	0.0173	0.0182	0.0182
NeuMF	0.0076	0.6750	0.0091	0.0026	0.0024	0.0019	0.0018	0.0022	0.0090	0.0147	0.0219
Katz	0.0315	0.6711	0.0387	0.0252	0.0103	0.0063	0.0045	0.0203	0.0386	0.0477	0.0511
Catapult	0.0211	0.6492	0.0265	0.0122	0.0068	0.0051	0.0044	0.0095	0.0263	0.0394	0.0525
IMC	0.0410	**0.7848**	0.0493	0.0209	0.0111	0.0094	0.0076	0.0181	0.0503	0.0831	0.0995
GCN-MF	**0.0586**	0.7657	**0.0760**	**0.0278**	**0.0190**	**0.0152**	**0.0119**	**0.0252**	**0.0833**	**0.1327**	**0.1535**

Table 2: Experiment Results on OMIM Fold2

Method	MAP	AUC	NDCG@10	P@1	P@5	P@10	P@15	R@1	R@5	R@10	R@15
PopLink	0.0059	0.3955	0.0079	0.0017	0.0021	0.0018	0.0015	0.0017	0.0098	0.0171	0.0216
WRMF	0.0065	0.0714	0.0157	0.0068	0.0031	0.0030	0.0020	0.0033	0.0104	0.0214	0.0214
BPRMF	0.0078	0.4971	0.0099	0.0051	0.0024	0.0015	0.0010	0.0039	0.0091	0.0108	0.0108
NeuMF	0.0080	0.6745	0.0090	0.0034	0.0019	0.0017	0.0013	0.0030	0.0081	0.0144	0.0161
Katz	0.0345	0.6639	0.0440	0.0239	0.0121	0.0075	0.0055	0.0195	0.0468	0.0574	0.0611
Catapult	0.0240	0.6494	0.0304	0.0128	0.0070	0.0056	0.0049	0.0115	0.0295	0.0450	0.0587
IMC	0.0355	**0.7867**	0.0435	0.0137	0.0113	0.0088	0.0072	0.0124	0.0531	0.0790	0.0974
GCN-MF	**0.0555**	0.7707	**0.0678**	**0.0273**	**0.0173**	**0.0125**	**0.0103**	**0.0263**	**0.0771**	**0.1131**	**0.1386**

Table 3: Experiment Results on OMIM Fold3

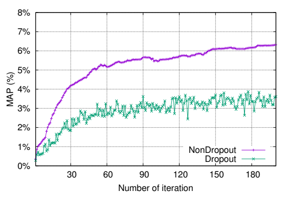
Method	MAP	AUC	NDCG@10	P@1	P@5	P@10	P@15	R@1	R@5	R@10	R@15
PopLink	0.0053	0.3876	0.0070	0.0009	0.0010	0.0017	0.0015	0.0009	0.0051	0.0170	0.0230
WRMF	0.0094	0.0792	0.0192	0.0060	0.0046	0.0039	0.0026	0.0041	0.0188	0.0316	0.0316
BPRMF	0.0143	0.5057	0.0159	0.0128	0.0034	0.0020	0.0014	0.0109	0.0133	0.0158	0.0158
NeuMF	0.0101	0.6780	0.0112	0.0026	0.0036	0.0020	0.0020	0.0026	0.0148	0.0174	0.0250
Katz	0.0315	0.6584	0.0403	0.0230	0.0111	0.0066	0.0049	0.0190	0.0392	0.0460	0.0520
Catapult	0.0246	0.6422	0.0324	0.0170	0.0089	0.0058	0.0045	0.0132	0.0335	0.0440	0.0524
IMC	0.0377	**0.7781**	0.0462	0.0170	0.0112	0.0091	0.0075	0.0133	0.0489	0.0806	0.0994
GCN-MF	**0.0500**	0.7592	**0.0643**	**0.0255**	**0.0163**	**0.0124**	**0.0099**	**0.0226**	**0.0712**	**0.1094**	**0.1319**

- Katz denotes one of the methods in [34] that utilizes the truncated Katz [20] metric to construct information propagation method.
- Catapult is another method in [34] that makes the propagation method become a prediction model by changing the damping weights into learnable parameters.
- IMC [30] propose a matrix factorization method by projecting the features to a low dimension.

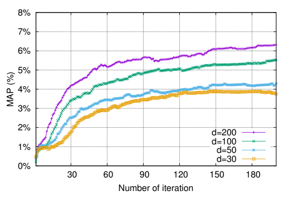
5.4 Performance Evaluation

The results of GCN-MF and other baselines could be found in Table 1, 2 and 3. From those results, we could make the following observations:

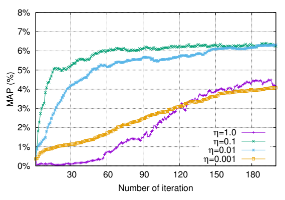
- From the results of PopLink, we could verify that our dataset is too sparse and every gene may only be related to few diseases.
- From the comparison of GCN-MF and traditional recommendation algorithms (BPRMF, WRMF, NeuMF), we could get that our framework GCN-MF outperforms a lot than them by exploiting the gene-gene and disease-disease similarities. And only utilizing discovered associations is not enough to find new associations.
- From the comparison of GCN-MF and graph-based methods (Catapult and Katz), we could find that only label propagation method cannot work well for this task. Because the similarity in the graph may not directly reflect the similarity of association between instances.
- From the comparison of GCN-MF and IMC, we could find that GCN-MF are better than IMC on most of metrics by learning non-linear relationship. And linear relationship is not enough for disease-gene association task.



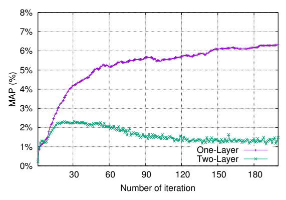
(a) Dropout



(b) Embedding Size



(c) Learning Rate



(d) Number of Layer

Figure 2: Parameter Analysis

In summary, GCN-MF outperforms all other state-of-the-art algorithms with a large margin on most of metrics. Because it could utilize more kinds of information and exploit the non-linear relationship between the same kind and different kinds of entities.

6 PARAMETER ANALYSIS

In this section, we will give the analysis of some parameters in our model to show the influence of them. Moreover we also give results of some technologies that are not fit for our model. Due to the space limitation, we just give the results with MAP, instead of giving the results with all metrics.

6.1 Dropout

Dropout nearly is the default module to construct the deep learning model, for its function of avoiding overfitting. However we have mentioned in Section 4.3 that the instability caused by dropout in our dataset will result in worse consequence than overfitting. From the results in Fig. 2a, we could see that dropout would lead to instability and low performance in our dataset. Under our analysis, we think the potential reason is that the association is too sparse in our dataset, where the density of our dataset is only 0.01%. But many other tasks, which could be applied by matrix factorization, contain much more labeled data. For instance, the density of Netflix data for movie recommendation is 1.2% [6], which is much higher than that of disease-gene association data.

6.2 Embedding size

Embedding size is an important factors for matrix factorization methods, which could directly influence the performance of the model. If the embedding size is not big enough, the model may lead to underfitting and be hard to converge. And if the dimension of the embedding is too large, the model would incur overfitting problem. In this comparison, we fixed all other parameters, and changed embedding size in $\{30, 50, 100, 200\}$. From Fig. 2b, we could see that different embedding sizes would lead to different final results but similar convergence rate. This could prove the stability of our model, that our results would not be fluctuated too much if the embedding size is selected in an appropriate range. And this also indicates that our model could exploit as much information as it can to achieve good result.

6.3 Learning rate

Learning rate is the parameter that control the step size of gradient descent in the optimization process. Controlling the learning rate is related to whether the algorithm can achieve optimal solution. If the learning rate is too small, the algorithm would be easily stuck in a poor local minimum. And if the step size is too large, the model would be unstable and cannot converge. In our comparison, we keep all other parameters fixed and change the learning rate from $\{0.001, 0.01, 0.1, 1\}$. From Fig. 2c, we could see that all relatively small learning rates could achieve convergence, but the middle two could obtain better results. The reason of learning rate in a wide range resulting in good convergence is that Adam can adaptively tune the learning during the learning process. This result also proves that our algorithm could achieve stable final results if the learning rate is in a proper range.

6.4 Layer Construction

For many works about GCN, they usually use two layer networks. From the definition of our framework in Eq. 6, we could know that there is one layer GCN in our model. Moreover, we also tried two layers structure. For the two layers model, embeddings of diseases and genes could be got as:

$$\tilde{P}' = \sigma(\tilde{A}(u)\sigma(\tilde{A}(u)PW(u)^{(1)})W(u)^{(2)}$$
$$\tilde{Q}' = \sigma(\tilde{A}(v)\sigma(\tilde{A}(v)QW(v)^{(1)})W(v)^{(2)},$$

where $W^{(i)}$ is the weight of i-th layer. However, from the results in Fig. 2d, we could see that two layer model cannot achieve good performance as our one layer setting. The drawback of the multiple layers GCN is that it could bring in more noises if the graph itself contains many uncertainty. For example, if we want to compute the preference score s_{ij} by two layer GCN model, we need gene u_i's all neighbours to compute u_i's final embedding and its' neighbours' neighbours to get the first layer. As our graphs are hand-crafted and don't include explicit association similarity, two layer structure would include too much noises to find new disease-gene associations.

7 CONCLUSION AND FUTURE WORK

In this work, we propose a new framework for disease-gene association task by combining Graph Convolutional Networks and matrix factorization, named GCN-MF. With the structure of GCN, we could utilize many existing relations inner genes and diseases respectively. For every relation, we extract corresponding features by PCA. Then for genes and diseases respectively, we concatenate all the features as the input feature. Dense similarities matrices are constructed by the inner product of features. To make the graph sparse, we construct a k-nn graph by selecting k closest neighbours for every entity with these similarities. Moreover, we construct the graph kernel for GCN with a weight to control the influence of neighbours. Once we got embeddings of genes and diseases by

GCNs, we use the inner product of them to get the preference score matrix. An ϵ-Insensitive loss is proposed to reduce the effect of data sparsity. We add Frobenius norm in the loss function to overcome overfitting and Adam method is used to optimize this loss function. From our experiments, we could see that GCN-MF outperforms all other state-of-the-art methods a lot on most of metrics.

In the analysis of our results, we could see that extra information is the key point to improve the performance in disease-gene association task. In the future, we would try to mine more accurate correlations by learning method. Moreover, we will apply our method on more applications that include similarity information and contain sparse data, such as POI recommendation.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: a system for large-scale machine learning.. In *USENIX Symposium on Operating Systems Design and Implementation*, Vol. 16. 265–283.

[2] Euan A Adie, Richard R Adams, Kathryn L Evans, David J Porteous, and Ben S Pickard. 2006. SUSPECTS: enabling fast and effective prioritization of positional candidates. *Bioinformatics* 22, 6 (2006), 773–774.

[3] Joanna S Amberger, Carol A Bocchini, François Schiettecatte, Alan F Scott, and Ada Hamosh. 2014. OMIM. org: Online Mendelian Inheritance in Man (OMIM®), an online catalog of human genes and genetic disorders. *Nucleic acids research* 43, D1 (2014), D789–D798.

[4] Gary D Bader, Doron Betel, and Christopher WV Hogue. 2003. BIND: the biomolecular interaction network database. *Nucleic acids research* 31, 1 (2003), 248–250.

[5] Albert-László Barabási, Natali Gulbahce, and Joseph Loscalzo. 2011. Network medicine: a network-based approach to human disease. *Nature reviews genetics* 12, 1 (2011), 56.

[6] Robert M. Bell and Yehuda Koren. 2007. Lessons from the Netflix prize challenge. *SIGKDD Explorations* (2007).

[7] Jie Chen, Tengfei Ma, and Cao Xiao. 2018. FastGCN: fast learning with graph convolutional networks via importance sampling. *arXiv preprint arXiv:1801.10247* (2018).

[8] Jianfei Chen, Jun Zhu, and Le Song. 2018. Stochastic Training of Graph Convolutional Networks with Variance Reduction.. In *International Conference on Machine Learning*. 941–949.

[9] Joshua C Denny, Marylyn D Ritchie, Melissa A Basford, Jill M Pulley, Lisa Bastarache, Kristin Brown-Gentry, Deede Wang, Dan R Masys, Dan M Roden, and Dana C Crawford. 2010. PheWAS: demonstrating the feasibility of a phenome-wide scan to discover gene–disease associations. *Bioinformatics* 26, 9 (2010), 1205–1210.

[10] Kyle J Gaulton, Karen L Mohlke, and Todd J Vision. 2007. A computational system to select candidate genes for complex human traits. *Bioinformatics* 23, 9 (2007), 1132–1140.

[11] Renu Goel, HC Harsha, Akhilesh Pandey, and TS Keshava Prasad. 2012. Human Protein Reference Database and Human Proteinpedia as resources for phosphoproteome analysis. *Molecular bioSystems* 8, 2 (2012), 453–463.

[12] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*. 1024–1034.

[13] David K Hammond, Pierre Vandergheynst, and Rémi Gribonval. 2011. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis* 30, 2 (2011), 129–150.

[14] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 173–182.

[15] Kristina M Hettne, Mark Thompson, Herman HHBM van Haagen, Eelke Van Der Horst, Rajaram Kaliyaperumal, Eleni Mina, Zuotian Tatum, Jeroen FJ Laros, Erik M Van Mulligen, Martijn Schuemie, et al. 2016. The implicitome: a resource for rationalizing gene-disease associations. *PloS one* 11, 2 (2016), e0149621.

[16] Yifan Hu, Florham Park, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *IEEE International Conference on Data Mining*.

[17] Trey Ideker and Roded Sharan. 2008. Protein networks in disease. *Genome research* 18, 4 (2008), 644–652.

[18] G Joshi-Tope, Marc Gillespie, Imre Vastrik, Peter D'Eustachio, Esther Schmidt, Bernard de Bono, Bijay Jassal, GR Gopinath, GR Wu, Lisa Matthews, et al. 2005. Reactome: a knowledgebase of biological pathways. *Nucleic acids research* 33, suppl_1 (2005), D428–D432.

[19] Minoru Kanehisa, Susumu Goto, Yoko Sato, Masayuki Kawashima, Miho Furumichi, and Mao Tanabe. 2013. Data, information, knowledge and principle: back to metabolism in KEGG. *Nucleic acids research* 42, D1 (2013), D199–D205.

[20] Leo Katz. 1953. A new status index derived from sociometric analysis. *Psychometrika* 18, 1 (1953), 39–43.

[21] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[22] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[23] Yehuda Koren, Robert M. Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* (2009).

[24] Insuk Lee, U Martin Blom, Peggy I Wang, Jung Eun Shim, and Edward M Marcotte. 2011. Prioritizing candidate disease genes by network-based boosting of genome-wide association data. *Genome research* (2011), gr–118992.

[25] Yu Li, Hiroyuki Kuwahara, Peng Yang, Le Song, and Xin Gao. 2019. PGCN: Disease gene prioritization by disease and gene embedding through graph convolutional neural networks. *bioRxiv* (2019). https://doi.org/10.1101/532226 arXiv:https://www.biorxiv.org/content/early/2019/01/28/532226.full.pdf

[26] Yongjin Li and Jagdish C Patra. 2010. Genome-wide inferring gene–phenotype relationship by walking on the heterogeneous network. *Bioinformatics* 26, 9 (2010), 1219–1224.

[27] Yong Liu, Min Wu, Chenghao Liu, Xiaoli Li, and Jie Zheng. 2019. SL 2 MF: Predicting Synthetic Lethality in Human Cancers via Logistic Matrix Factorization. *IEEE/ACM transactions on computational biology and bioinformatics* (2019).

[28] Yong Liu, Min Wu, Chunyan Miao, Peilin Zhao, and Xiao-Li Li. 2016. Neighborhood regularized logistic matrix factorization for drug-target interaction prediction. *PLoS computational biology* 12, 2 (2016), e1004760.

[29] Zhiwu Lu, Zhenyong Fu, Tao Xiang, Peng Han, Liwei Wang, and Xin Gao. 2017. Learning from weak and noisy labels for semantic segmentation. *IEEE transactions on pattern analysis and machine intelligence* 39, 3 (2017), 486–500.

[30] Nagarajan Natarajan and Inderjit S Dhillon. 2014. Inductive matrix completion for predicting gene–disease associations. *Bioinformatics* 30, 12 (2014), i60–i68.

[31] Arzucan Özgür, Thuy Vu, Güneş Erkan, and Dragomir R Radev. 2008. Identifying gene-disease associations using centrality on a literature mined gene-interaction network. *Bioinformatics* 24, 13 (2008), i277–i285.

[32] Rong Pan, Yunhong Zhou, Bin Cao, Nathan Nan Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. 2008. One-class collaborative filtering. In *IEEE International Conference on Data Mining*. IEEE, 502–511.

[33] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Conference on Uncertainty in Artificial Intelligence*. 452–461.

[34] U Martin Singh-Blom, Nagarajan Natarajan, Ambuj Tewari, John O Woods, Inderjit S Dhillon, and Edward M Marcotte. 2013. Prediction and validation of gene-disease associations using methods inspired by social network analyses. *PloS one* 8, 5 (2013), e58977.

[35] Qi Wang, Mengying Sun, Liang Zhan, Paul Thompson, Shuiwang Ji, and Jiayu Zhou. 2017. Multi-Modality Disease Modeling via Collective Deep Matrix Factorization. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1155–1164.

[36] Xiujuan Wang, Natali Gulbahce, and Haiyuan Yu. 2011. Network-based methods for human disease gene prediction. *Briefings in functional genomics* 10, 5 (2011), 280–293.

[37] Xuebing Wu, Rui Jiang, Michael Q Zhang, and Shao Li. 2008. Network-based global inference of human disease genes. *Molecular systems biology* 4, 1 (2008), 189.

[38] Peng Yang, Xiaoli Li, Min Wu, Chee-Keong Kwoh, and See-Kiong Ng. 2011. Inferring gene-phenotype associations via global protein complex network propagation. *PloS one* 6, 7 (2011), e21502.

[39] Peng Yang, Xiao-Li Li, Jian-Ping Mei, Chee-Keong Kwoh, and See-Kiong Ng. 2012. Positive-unlabeled learning for disease gene identification. *Bioinformatics* 28, 20 (2012), 2640–2647.

[40] Peng Yang, Peilin Zhao, Yong Liu, and Xin Gao. 2018. Robust Cost-Sensitive Learning for Recommendation with Implicit Feedback. In *Proceedings of the 2018 SIAM International Conference on Data Mining*. SIAM, 621–629.

[41] Peng Yang, Peilin Zhao, Vincent W Zheng, Lizhong Ding, and Xin Gao. 2018. Robust Asymmetric Recommendation via Min-Max Optimization. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 1077–1080.

[42] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. *arXiv preprint arXiv:1806.01973* (2018).

[43] Hongyi Zhou and Jeffrey Skolnick. 2016. A knowledge-based approach for predicting gene–disease associations. *Bioinformatics* 32, 18 (2016), 2831–2838.