Cross-Dependency Inference in Multi-Layered Networks: A Collaborative Filtering Perspective

CHEN CHEN and HANGHANG TONG, Arizona State University
LEI XIE, City University of New York
LEI YING, Arizona State University
QING HE, University at Buffalo

The increasingly connected world has catalyzed the fusion of networks from different domains, which facilitates the emergence of a new network model—multi-layered networks. Examples of such kind of network systems include critical infrastructure networks, biological systems, organization-level collaborations, cross-platform e-commerce, and so forth. One crucial structure that distances multi-layered network from other network models is its cross-layer dependency, which describes the associations between the nodes from different layers. Needless to say, the cross-layer dependency in the network plays an essential role in many data mining applications like system robustness analysis and complex network control. However, it remains a daunting task to know the exact dependency relationships due to noise, limited accessibility, and so forth. In this article, we tackle the cross-layer dependency inference problem by modeling it as a collective collaborative filtering problem. Based on this idea, we propose an effective algorithm FASCINATE that can reveal unobserved dependencies with linear complexity. Moreover, we derive FASCINATE-ZERO, an online variant of FASCINATE that can respond to a newly added node timely by checking its neighborhood dependencies. We perform extensive evaluations on real datasets to substantiate the superiority of our proposed approaches.

Categories and Subject Descriptors: H.2.8 [**Database Applications**]: Data Mining

General Terms: Algorithm, Experimentation

Additional Key Words and Phrases: Multi-layered network, cross-layer dependency, graph mining

ACM Reference Format:
Chen Chen, Hanghang Tong, Lei Xie, Lei Ying, and Qing He. 2017. Cross-dependency inference in multi-layered networks: A collaborative filtering perspective. ACM Trans. Knowl. Discov. Data 11, 4, Article 42 (June 2017), 26 pages.
DOI: http://dx.doi.org/10.1145/3056562

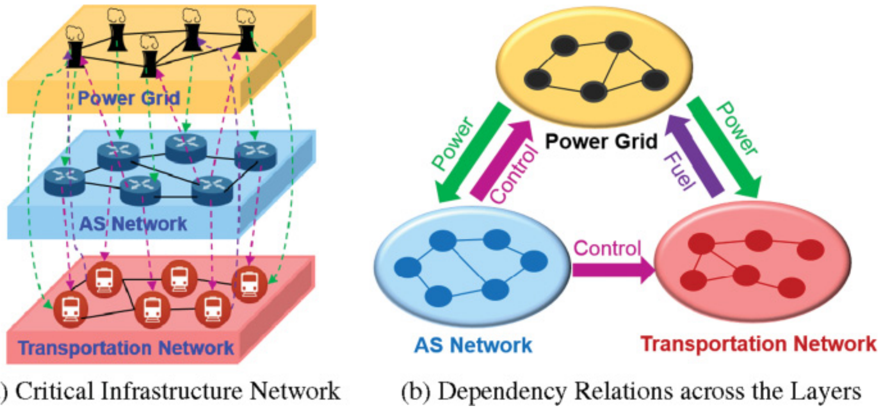
1. INTRODUCTION

Networks are prevalent and naturally appear in many high-impact domains, including infrastructure constructions, academic research, social collaboration, and many more. As the world is becoming highly connected, cross-domain interactions are more

This work is supported by DTRA under the grant number HDTRA1-16-0017, National Science Foundation under Grant No. IIS-1651203, Army Research Office under the contract number W911NF-16-1-0168, National Institutes of Health under the grant number R01LM011986, Region II University Transportation Center under the project number 49997-33 25 and a Baidu gift.

Authors' addresses: C. Chen, Brickyard Engineering (BYENG) 411AC, 699 S. Mill Ave. Tempe, AZ 85281; email: chen_chen@asu.edu; H. Tong, Brickyard Engineering (BYENG) 416, 699 S. Mill Ave. Tempe, AZ 85281; email: hanghang.tong@asu.edu; L. Xie, Hunter College, the City University of New York, 695 Park Ave. New York, NY 10065; email: lei.xie@hunter.cuny.edu; L. Ying, Goldwater Center (GWC) 436, 650 E. Tyler Mall Tempe, AZ 85281; email: lei.ying.2@asu.edu; Q. He, 225 Ketter Hall Buffalo, NY 14260; email: qinghe@buffalo.edu.

DOI: http://dx.doi.org/10.1145/3056562



(a) Critical Infrastructure Network (b) Dependency Relations across the Layers

Fig. 1. An illustrative example of multi-layered networks. In Figure 1(b), each ellipse corresponds to a critical infrastructure network in Figure 1(a) (i.e., power grid, AS network, and transportation network). The arrows between two ellipses indicate cross-layer dependency relationships between the corresponding two networks (e.g., a router in the AS network depends on one or more power plants in the power grid).

frequently observed in numerous applications, catalyzing the emergence of a new network model—*multi-layered networks* [Buldyrev et al. 2010; Gao et al. 2012; Parshani et al. 2010; Sen et al. 2014; Shao et al. 2011]. One typical example of such type of multi-layered networks is critical infrastructure network as illustrated in Figure 1. In an infrastructure network system, the full functioning of the autonomous system network (AS network) and the transportation network is dependent on the power supply from the power grid. While for the gas-fired and coal-fired generators in the power grid, their functioning is fully dependent on the gas and coal supplies from the transportation network. Moreover, to keep the whole complex system working in order, extensive communications are needed between the nodes in the networks, which are supported by the AS network. Another example is *citation networks*, where papers that belong to the same domain can be viewed as the nodes from one layer, and the cross-domain paper citations can be considered as cross-layer dependencies. While in the biological system, the protein–protein interaction network (PPI/gene network) is naturally linked to the disease similarity network by the known disease–gene associations, and the disease network is in turn coupled with the drug network by drug–disease associations. Multi-layered networks also appear in many other application domains, such as organization-level collaboration platform [Chen et al. 2015] and cross-platform e-commerce [Chen et al. 2013; Li et al. 2009; Lu et al. 2013; Yang et al. 2015].

One crucial topological structure that differentiates multi-layered network from other network models is its *cross-layer dependency*, which describes the associations/dependencies between the nodes from different layers. For example, in infrastructure networks, the full functioning of the AS network depends on the sufficient power supply from the power grid layer, which in turn relies on the functioning of the transportation network (e.g., to deliver the sufficient fuel). Similarly, in the biological systems, the dependency is represented as the associations among diseases, genes, and drugs. In practice, the cross-layer dependency often plays a central role in many multi-layered network mining tasks. For example, in the critical infrastructure network, the intertwined cross-layer dependency is considered as a major factor of system robustness. This is because a small failure on the supporting network (e.g., power station malfunction in power grid) may be amplified in all its dependent networks through cross-layer dependencies, resulting in a catastrophic/cascading failure of the entire system. On the other hand, the cross-layer dependency in the biological system is often the key to new discoveries, such as new treatment associations between existing drugs and new diseases.

In spite of its key importance, it remains a daunting task to know the exact cross-layer dependency structure in a multi-layered network, due to noise, incomplete data sources, and limited accessibility to network dynamics. For example, an extreme weather event might significantly disrupt the power grid, the transportation network, and the cross-layer dependencies in between at the epicenter. Yet, due to limited accessibility to the damaged area during or soon after the disruption, the cross-layer dependency structure might only have a probabilistic and/or coarse-grained description. On the other hand, for a newly identified chemical in the biological system, its cross-layer dependencies w.r.t. proteins and/or the diseases might be completely unknown due to clinical limitations (i.e., the *zero-start* problem).

In this article, we aim to tackle the above challenges by developing effective and efficient methods to infer cross-layer dependency on multi-layered networks. The main contributions of the article can be summarized as (1) *Problem Formulations*, we define the cross-layer dependency inference problem as a regularized optimization problem. The key idea behind this formulation is to collectively leverage the within-layer topology and the observed cross-layer dependency to infer a latent, low-rank representation for each layer, which can be used to infer the missing cross-layer dependencies in the network. (2) *Algorithms and Analysis*, we propose an effective algorithm—FASCINATE—to infer the cross-layer dependency on multi-layered networks, and analyze its optimality, convergence, and complexity. We further present its variants and generalizations, including an online algorithm to address the *zero-start* problem. (3) *Evaluations*, we perform extensive experiments on five real datasets to substantiate the effectiveness, efficiency, and scalability of our proposed algorithms. Specifically, our experimental evaluations show that the proposed algorithms outperform their best competitors by 8.2%–41.9% in terms of inference accuracy, while enjoying linear complexity. Moreover, the proposed FASCINATE-ZERO algorithm can achieve up to $10^7\times$ speedup with barely any compromise on accuracy.

The rest of the article is organized as follows. Section 2 gives the formal definitions of the cross-layer dependency inference problems. Section 3 proposes FASCINATE algorithm with its analysis. Section 4 introduces the *zero-start* algorithm FASCINATE-ZERO. Section 5 presents the experiment results. Section 6 reviews the related works. Section 7 summarizes the article.

2. PROBLEM DEFINITION

In this section, we give the formal definitions of the cross-layer dependency inference problems. The main symbols used throughout the article are listed in Table I. Following the convention, we use bold upper-case for matrices (e.g., $\mathbf{A}$), bold lower-case for vectors (e.g., $\mathbf{a}$), and calligraphic for sets (e.g., $\mathcal{A}$). $\mathbf{A}'$ denotes the transpose of matrix $\mathbf{A}$. We use the ˆ sign to denote the notations after a new node is accommodated to the system (e.g., $\hat{\mathcal{J}}$, $\hat{\mathbf{A}}_1$), and the ones without the ˆ sign as the notations before the new node arrives.

While several multi-layered network models exist in the literature (see Section 6 for a review), we will focus on a recent model proposed in Chen et al. [2015], due to its flexibility to model more complicated cross-layer dependency structure. We refer the readers to Chen et al. [2015] for its full details. For the purpose of this article, we mainly need the following notations to describe a multi-layered network with g layers. First, we need a $g \times g$ layer–layer dependency matrix $\mathbf{G}$, where $\mathbf{G}(i, j) = 1$ if layer-j depends on layer-i, and $\mathbf{G}(i, j) = 0$ otherwise. Second, we need a set of g within-layer

Table I. Main Symbols

Symbol	Definition and description
$\mathbf{A}, \mathbf{B}$	The adjacency matrices (bold upper case)
$\mathbf{a}, \mathbf{b}$	Column vectors (bold lower case)
$\mathcal{A}, \mathcal{B}$	Sets (calligraphic)
$\mathbf{A}(i, j)$	The element at ith row jth column in matrix $\mathbf{A}$
$\mathbf{A}(i, :)$	The ith row of matrix $\mathbf{A}$
$\mathbf{A}(:, j)$	The jth column of matrix $\mathbf{A}$
$\mathbf{A}'$	Transpose of matrix $\mathbf{A}$
$\hat{\mathbf{A}}$	The adjacency matrix of $\mathbf{A}$ with the newly added node
$\mathbf{G}$	The layer–layer dependency matrix
$\mathcal{A}$	Within-layer connectivity matrices of the network $\mathcal{A} = \{\mathbf{A}_1, \ldots, \mathbf{A}_g\}$
$\mathcal{D}$	Cross-layer dependency matrices $\mathcal{D} = \{\mathbf{D}_{i,j}\ i, j = 1, \ldots, g\}$
$\mathbf{W}_{i,j}$	Weight matrix for $\mathbf{D}_{i,j}$
$\mathbf{F}_i$	Low-rank representation for layer-i ($i = 1, \ldots, g$)
m_i, n_i	Number of edges and nodes in graph $\mathbf{A}_i$
$m_{i,j}$	Number of dependencies in $\mathbf{D}_{i,j}$
g	Total number of layers
r	The rank for $\{\mathbf{F}_i\}_{i=1,\ldots,g}$
t	The maximal iteration number
ξ	The threshold to determine the iteration

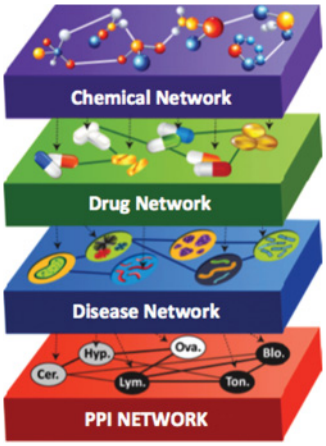


Fig. 2. A simplified four-layered network for biological systems.

connectivity matrices: $\mathcal{A} = \{\mathbf{A}_1, \ldots, \mathbf{A}_g\}$ to describe the connectivities/similarities between nodes within the same layer. Third, we need a set of cross-layer dependency matrices $\mathcal{D} = \{\mathbf{D}_{i,j}\ i, j = 1, \ldots, g\}$, where $\mathbf{D}_{i,j}$ describes the dependencies between the nodes from layer-i and the nodes from layer-j if these two layers are directly dependent (i.e., $\mathbf{G}(i, j) = 1$). When there is no direct dependencies between the two layers (i.e., $\mathbf{G}(i, j) = 0$), the corresponding dependency matrix $\mathbf{D}_{i,j}$ is absent. Taking the multilayered network in Figure 2 for an example, the abstract layer–layer dependency network $\mathbf{G}$ of this biological system can be viewed as a line graph. The four within-layer similarity matrices in $\mathcal{A}$ are the *chemical network* ($\mathbf{A}_1$), the *drug network* ($\mathbf{A}_2$), the *disease network* ($\mathbf{A}_3$), and the *PPI network* ($\mathbf{A}_4$). Across those layers, we have three non-empty dependency matrices, including the *chemical–drug* dependency matrix ($\mathbf{D}_{1,2}$), the *drug–disease* interaction matrix ($\mathbf{D}_{2,3}$), and the *disease–protein* dependency matrix ($\mathbf{D}_{3,4}$).[1]

As mentioned earlier, it is often very hard to accurately know the cross-layer dependency matrices $\{\mathbf{D}_{i,j}\ i, j = 1, \ldots, g\}$. In other words, such *observed* dependency matrices are often incomplete and noisy. Inferring the missing cross-layer dependencies is an essential prerequisite for many multi-layered network mining tasks. On the other hand, real-world networks are evolving over time. Probing the cross-layer dependencies is often a time-consuming process in large complex networks. Thus, a newly added node could have no observed cross-layer dependencies for a fairly long period of time since its arrival. Therefore, inferring the dependencies of such kind of *zero-start* nodes is an important problem that needs to be solved efficiently. Formally, we define the cross-layer dependency inference problem (CODE) and its corresponding *zero-start* variant (CODE-ZERO) as follows.

PROBLEM 1. *(*CODE*) Cross-Layer Dependency Inference*

Given: a multi-layered network with (1) layer–layer dependency matrix $\mathbf{G}$*; (2) within-layer connectivity matrices* $\mathcal{A} = \{\mathbf{A}_1, \ldots, \mathbf{A}_g\}$*; and (3) observed cross-layer dependency matrices* $\mathcal{D} = \{\mathbf{D}_{i,j}\ i, j = 1, \ldots, g\}$*;*

Output: the true cross-layer dependency matrices $\{\tilde{\mathbf{D}}_{i,j}\ i, j = 1, \ldots, g\}$.

PROBLEM 2. *(*CODE-ZERO*) Cross-Layer Dependency Inference for zero-start Nodes*

Given: (1) a multi-layered network $\{\mathbf{G}, \mathcal{A}, \mathcal{D}\}$*; (2) a newly added node* p *in the* l*th layer; (3) a* $1 \times n_l$ *vector* $\mathbf{s}$ *that records the connections between* p *and the existing* n_l *nodes in layer* l*;*

Output: the true dependencies between node p *and nodes in dependent layers of layer-*l*, i.e.,* $\tilde{\mathbf{D}}_{l,j}(p, :)$ $(j = 1, \ldots, g,\ \mathbf{G}(l, j) = 1)$.

3. FASCINATE FOR PROBLEM 1

In this section, we present our proposed solution for Problem 1 (CODE). We start with the proposed optimization formulation, and then present our algorithm (FASCINATE), followed by some effectiveness and efficiency analysis.

3.1. FASCINATE: Optimization Formulation

The key idea behind our formulation is to treat Problem 1 as a *collective collaborative filtering* problem. To be specific, if we view (1) nodes from a given layer (e.g., power plants) as objects from a given domain (e.g., users/items), (2) the within-layer connectivity (e.g., communication networks) as an object–object similarity measure, and (3) the cross-layer dependency (e.g., dependencies between computers in the communication layer and power plants in power grid layer) as the “ratings” from objects of one domain to those of another domain, then inferring the missing cross-layer dependencies can be viewed as a task of inferring the missing ratings between the objects (e.g., users and items) across different domains. Having this analogy in mind, we propose to formulate Problem 1 as the following regularized optimization problem:

[1] More complicated dependency relationships may exist across the layers in real settings, which can be addressed with our model as well.

$$\min_{\mathbf{F}_i \geq \mathbf{0}(i=1,\ldots,g)} J = \underbrace{\sum_{i,j:\ \mathbf{G}(i,j)=1} \|\mathbf{W}_{i,j} \odot (\mathbf{D}_{i,j} - \mathbf{F}_i\mathbf{F}_j{}')\|_F^2}_{\text{C1: Matching Observed Cross-Layer Dependencies}} + \underbrace{\alpha \sum_{i=1}^{g} \mathrm{tr}(\mathbf{F}_i{}'(\mathbf{T}_i - \mathbf{A}_i)\mathbf{F}_i)}_{\text{C2: Node Homophily}} + \underbrace{\beta \sum_{i=1}^{g} \|\mathbf{F}_i\|_F^2}_{\text{C3: Regularization}}, \tag{1}$$

where $\mathbf{T}_i$ is the diagonal degree matrix of $\mathbf{A}_i$ with $\mathbf{T}_i(u,u) = \sum_{v=1}^{n_i} \mathbf{A}_i(u,v)$; $\mathbf{W}_{i,j}$ is an $n_i \times n_j$ weight matrix to assign different weights to different entries in the corresponding cross-layer dependency matrix $\mathbf{D}_{i,j}$; and $\mathbf{F}_i$ is the low-rank representation for layer i. For now, we set the weight matrices as follows: $\mathbf{W}_{i,j}(u,v)$:
$=1$ if $\mathbf{D}_{i,j}(u,v)$ is observed, and $\mathbf{W}_{i,j}(u,v) \in [0,1]$ if $\mathbf{D}_{i,j}(u,v)$
$=0$ (i.e., unobserved). To simplify the computation, we set the weights of all unobserved entries to a global value w. We will discuss alternative choices for the weight matrices in Section 3.3.

In this formulation (Equation (1)), we can think of $\mathbf{F}_i$ as the low-rank representations/features of the nodes in layer i in some latent space, which is shared among different layers. The cross-layer dependencies between the nodes from two dependent layers can be viewed as the inner product of their latent features. Therefore, the intuition of the first term (i.e., C1) is that we want to match all the cross-layer dependencies calibrated by the weight matrix $\mathbf{W}_{i,j}$. The second term (i.e., C2) is used to achieve node homophily, which says that for a pair of nodes u and v from the same layer (say layer-i), their low-rank representations should be similar (i.e., small $\|\mathbf{F}_i(u,:) - \mathbf{F}_i(v,:)\|_2$) if the within-layer connectivity between these two nodes is strong (i.e., large $\mathbf{A}_i(u,v)$). The third term (i.e., C3) is to regularize the norm of the low-rank matrices $\{\mathbf{F}_i\}_{i=1,\ldots,g}$ to prevent over-fitting.

Once we solve Equation (1), for a given node u from layer-i and a node v from layer-j, the cross-layer dependency between them can be estimated as $\tilde{\mathbf{D}}_{i,j}(u,v) = \mathbf{F}_i(u,:)\mathbf{F}_j(v,:)'$.

3.2. FASCINATE: Optimization Algorithm

The optimization problem defined in Equation (1) is non-convex. Thus, we seek to find a local optima by the block coordinate descent method, where each $\mathbf{F}_i$ naturally forms a "block." To be specific, if we fix all other $\mathbf{F}_j (j = 1, \ldots, g, j \neq i)$ and ignore the constant terms, Equation (1) can be simplified as

$$\underset{\mathbf{F}_i \geq \mathbf{0}}{J_i} = \sum_{j:\ \mathbf{G}(i,j)=1} \|\mathbf{W}_{i,j} \odot (\mathbf{D}_{i,j} - \mathbf{F}_i\mathbf{F}_j{}')\|_F^2 + \alpha \mathrm{tr}(\mathbf{F}_i{}'(\mathbf{T}_i - \mathbf{A}_i)\mathbf{F}_i) + \beta\|\mathbf{F}_i\|_F^2. \tag{2}$$

The derivative of J_i w.r.t. $\mathbf{F}_i$ is

$$\frac{\partial J_i}{\partial \mathbf{F}_i} = 2\left(\sum_{j:\ \mathbf{G}(i,j)=1} [-(\mathbf{W}_{i,j} \odot \mathbf{W}_{i,j} \odot \mathbf{D}_{i,j})\mathbf{F}_j + (\mathbf{W}_{i,j} \odot \mathbf{W}_{i,j} \odot (\mathbf{F}_i\mathbf{F}_j{}'))\mathbf{F}_j] \right. \tag{3}$$
$$\left. + \alpha\mathbf{T}_i\mathbf{F}_i - \alpha\mathbf{A}_i\mathbf{F}_i + \beta\mathbf{F}_i \right).$$

A fixed-point solution of Equation (3) with non-negativity constraint on $\mathbf{F}_i$ leads to the following multiplicative updating rule for $\mathbf{F}_i$:

$$\mathbf{F}_i(u,v) \leftarrow \mathbf{F}_i(u,v)\sqrt{\frac{\mathbf{X}(u,v)}{\mathbf{Y}(u,v)}}, \tag{4}$$

where

$$\mathbf{X} = \sum_{j:\ \mathbf{G}(i,j)=1} (\mathbf{W}_{i,j} \odot \mathbf{W}_{i,j} \odot \mathbf{D}_{i,j})\mathbf{F}_j + \alpha \mathbf{A}_i \mathbf{F}_i \tag{5}$$

$$\mathbf{Y} = \sum_{j:\ \mathbf{G}(i,j)=1} (\mathbf{W}_{i,j} \odot \mathbf{W}_{i,j} \odot (\mathbf{F}_i \mathbf{F}_j{}'))\mathbf{F}_j + \alpha \mathbf{T}_i \mathbf{F}_i + \beta \mathbf{F}_i.$$

Recall that we set $\mathbf{W}_{i,j}(u,v) = 1$ when $\mathbf{D}_{i,j}(u,v) > 0$, and $\mathbf{W}_{i,j}(u,v) = w$ when $\mathbf{D}_{i,j}(u,v) = 0$. Here, we define $\mathbf{I}^O_{i,j}$ as an indicator matrix for the observed entries in $\mathbf{D}_{i,j}$, that is, $\mathbf{I}^O_{i,j}(u,v) = 1$ if $\mathbf{D}_{i,j}(u,v) > 0$, and $\mathbf{I}^O_{i,j}(u,v) = 0$ if $\mathbf{D}_{i,j}(u,v) = 0$. Then, the estimated dependencies over the observed data can be represented as $\tilde{\mathbf{R}}_{i,j} = \mathbf{I}^O_{i,j} \odot (\mathbf{F}_i \mathbf{F}_j)$. With these notations, we can further simplify the update rule in Equation (5) as follows:

$$\mathbf{X} = \sum_{j:\ \mathbf{G}(i,j)=1} \mathbf{D}_{i,j}\mathbf{F}_j + \alpha \mathbf{A}_i \mathbf{F}_i \tag{6}$$

$$\mathbf{Y} = \sum_{j:\ \mathbf{G}(i,j)=1} ((1-w^2)\tilde{\mathbf{R}}_{i,j} + w^2 \mathbf{F}_i \mathbf{F}_j{}')\mathbf{F}_j + \alpha \mathbf{T}_i \mathbf{F}_i + \beta \mathbf{F}_i. \tag{7}$$

The proposed FASCINATE algorithm is summarized in Algorithm 1. First, it randomly initializes the low-rank matrices for each layer (line 1–line 3). Then, it begins the iterative update procedure. In each iteration (line 4–line 10), the algorithm alternatively updates $\{\mathbf{F}_i\}_{i=1,\ldots,g}$ one by one. We use two criteria to terminate the iteration: (1) either the Frobenius norm between two successive iterations for all $\{\mathbf{F}_i\}_{i=1,\ldots,g}$ is less than a threshold ξ, or (2) the maximum iteration number t is reached.

ALGORITHM 1: The FASCINATE algorithm

Input: (1) a multi-layered network with (a) layer–layer dependency matrix $\mathbf{G}$, (b) within-layer connectivity matrices $\mathcal{A} = \{\mathbf{A}_1, \ldots, \mathbf{A}_g\}$, and (c) observed cross-layer node dependency matrices $\mathcal{D} = \{\mathbf{D}_{i,j}\ i, j = 1, \ldots, g\}$; (2) the rank size r; (3) weight w; (4) regularized parameters α and β;

Output: low-rank representations for each layer $\{\mathbf{F}_i\}_{i=1,\ldots,g}$

```
 1: for $i = 1$ to $g$ do
 2:     initialized $\mathbf{F}_i$ as $n_i \times r$ non-negative random matrix
 3: end for
 4: while not converge do
 5:     for $i = 1$ to $g$ do
 6:         compute $\mathbf{X}$ as Equation (6)
 7:         compute $\mathbf{Y}$ as Equation (7)
 8:         update $\mathbf{F}_i$ as Equation (4)
 9:     end for
10: end while
11: return $\{\mathbf{F}_i\}_{i=1,\ldots,g}$
```

3.3. Proof and Analysis

Here, we analyze the proposed Fascinate algorithm in terms of its effectiveness as well as its efficiency.

Effectiveness Analysis. In terms of effectiveness, we show that the proposed Fascinate algorithm indeed finds a local optimal solution to Equation (1). To see this, we first give the following theorem, which says that the fixed point solution of Equation (4) satisfies the *KKT* condition.

Theorem 3.1. *The fixed point solution of Equation* (4) *satisfies the KKT condition.*

Proof. The Lagrangian function of Equation (2) can be written as

$$L_i = \sum_{j:\ \mathbf{G}(i,j)=1} \|\mathbf{W}_{i,j} \odot (\mathbf{D}_{i,j} - \mathbf{F}_i{\mathbf{F}_j}')\|_F^2 \\ + \alpha \mathrm{tr}({\mathbf{F}_i}'\mathbf{T}_i\mathbf{F}_i) - \alpha \mathrm{tr}({\mathbf{F}_i}'\mathbf{A}_i\mathbf{F}_i) + \beta\|\mathbf{F}_i\|_F^2 - \mathrm{tr}(\Lambda'\mathbf{F}_i), \tag{8}$$

where Λ is the Lagrange multiplier. Setting the derivative of L_i w.r.t. $\mathbf{F}_i$ to 0, we get

$$2\left(\sum_{j:\ \mathbf{G}(i,j)=1} \left[-(\mathbf{W}_{i,j} \odot \mathbf{W}_{i,j} \odot \mathbf{D}_{i,j})\mathbf{F}_j + (\mathbf{W}_{i,j} \odot \mathbf{W}_{i,j} \odot (\mathbf{F}_i{\mathbf{F}_j}'))\mathbf{F}_j\right] \\ + \alpha\mathbf{T}_i\mathbf{F}_i - \alpha\mathbf{A}_i\mathbf{F}_i + \beta\mathbf{F}_i\right) = \Lambda. \tag{9}$$

By the KKT complementary slackness condition, we have

$$\underbrace{[\sum_{j:\ \mathbf{G}(i,j)=1} (\mathbf{W}_{i,j} \odot \mathbf{W}_{i,j} \odot (\mathbf{F}_i{\mathbf{F}_j}'))\mathbf{F}_j + \alpha\mathbf{T}_i\mathbf{F}_i + \beta\mathbf{F}_i}_{\mathbf{Y}} \\ - (\underbrace{\sum_{j:\ \mathbf{G}(i,j)=1} (\mathbf{W}_{i,j} \odot \mathbf{W}_{i,j} \odot \mathbf{D}_{i,j})\mathbf{F}_j + \alpha\mathbf{A}_i\mathbf{F}_i}_{\mathbf{X}})](u,v)\mathbf{F}_i(u,v) = 0. \tag{10}$$

Therefore, we can see that the fixed point solution of Equation (4) satisfies the above equation. □

The convergence of the proposed Fascinate algorithm is given by the following lemma.

Lemma 3.2. *Under the updating rule in Equation* (4)*, the objective function in Equation* (2) *decreases monotonically.*

Proof. By expending the Frobius norms and dropping constant terms, Equation (2) can be further simplified as

$$J_i = \sum_{\mathbf{G}(i,j)=1} (\underbrace{-2\mathrm{tr}((\mathbf{W}_{i,j} \odot \mathbf{W}_{i,j} \odot \mathbf{D}_{i,j})\mathbf{F}_j\mathbf{F}_i')}_{T_1} + \underbrace{\mathrm{tr}((\mathbf{W}_{i,j} \odot \mathbf{W}_{i,j} \odot (\mathbf{F}_i\mathbf{F}_j'))\mathbf{F}_j\mathbf{F}_i'))}_{T_2} \\ + \underbrace{\alpha\mathrm{tr}(\mathbf{F}_i'\mathbf{T}_i\mathbf{F}_i)}_{T_3} \underbrace{-\alpha\mathrm{tr}(\mathbf{F}_i'\mathbf{A}\mathbf{F}_i)}_{T_4} + \underbrace{\beta\mathrm{tr}(\mathbf{F}_i\mathbf{F}_i')}_{T_5}. \tag{11}$$

Following the auxiliary function approach in Lee and Seung [2001], the auxiliary function $H(\mathbf{F}_i, \tilde{\mathbf{F}}_i)$ of J_i must satisfy

$$H(\mathbf{F}_i, \mathbf{F}_i) = J_i,\ \ H(\mathbf{F}_i, \tilde{\mathbf{F}}_i) \geq J_i. \tag{12}$$

Define

$$\mathbf{F}_i^{(t+1)} = \arg\min_{\mathbf{F}_i} H\big(\mathbf{F}_i, \mathbf{F}_i^{(t)}\big), \tag{13}$$

by this construction, we have

$$J_i^{(t)} = H\big(\mathbf{F}_i^{(t)}, \mathbf{F}_i^{(t)}\big) \geq H\big(\mathbf{F}_i^{(t+1)}, \mathbf{F}_i^{(t)}\big) \geq J_i^{(t+1)}, \tag{14}$$

which proves that $J_i^{(t)}$ decreases monotonically.

Next, we prove that (1) we can find an auxiliary function that satisfies the above constraints and (2) the updating rule in Equation (4) leads to global minimum solution to the auxiliary function.

First, we show that the following function is one of the auxiliary functions of Equation (11):

$$H(\mathbf{F}_i, \tilde{\mathbf{F}}_i) = \sum_{\mathbf{G}_{(i,j)}=1} (T_1' + T_2') + T_3' + T_4' + T_5', \tag{15}$$

where

$$T_1' = -2\sum_{u=1}^{n_i}\sum_{k=1}^{r}[(\mathbf{W}_{i,j} \odot \mathbf{W}_{i,j} \odot \mathbf{D}_{i,j})\mathbf{F}_j](u,k)\tilde{\mathbf{F}}_i(u,k)(1 + \log\frac{\mathbf{F}_i(u,k)}{\tilde{\mathbf{F}}_i(u,k)}) \tag{16}$$

$$T_2' = \sum_{u=1}^{n_i}\sum_{k=1}^{r}\frac{[(\mathbf{W}_{i,j} \odot \mathbf{W}_{i,j} \odot (\tilde{\mathbf{F}}_i\mathbf{F}_j'))\mathbf{F}_j](u,k)\mathbf{F}_i^2(u,k)}{\tilde{\mathbf{F}}_i(u,k)} \tag{17}$$

$$T_3' = \sum_{u=1}^{n_i}\sum_{k=1}^{r}\frac{[\alpha\mathbf{T}_i\tilde{\mathbf{F}}_i](u,k)\mathbf{F}_i^2(u,k)}{\tilde{\mathbf{F}}_i(u,k)} \tag{18}$$

$$T_4' = -\sum_{u=1}^{n_i}\sum_{v=1}^{n_i}\sum_{k=1}^{r}\alpha\mathbf{A}_i(u,v)\tilde{\mathbf{F}}_i(v,k)\tilde{\mathbf{F}}_i(u,k)(1 + \log\frac{\mathbf{F}_i(v,k)\mathbf{F}_i(u,k)}{\tilde{\mathbf{F}}_i(v,k)\tilde{\mathbf{F}}_i(u,k)}) \tag{19}$$

$$T_5' = \sum_{u=1}^{n_i}\sum_{k=1}^{r}\beta\mathbf{F}_i^2(u,k). \tag{20}$$

Here, we prove that $T_i' \geq T_i$ for $i = 1, \ldots, 5$ term by term.

Using the inequality $z \geq 1 + \log z$, we have

$$T_1' \geq -2\sum_{u=1}^{n_i}\sum_{k=1}^{r}[(\mathbf{W}_{i,j} \odot \mathbf{W}_{i,j} \odot \mathbf{D}_{i,j})\mathbf{F}_j](u,k)\mathbf{F}_i(u,k) = T_1 \tag{21}$$

$$T_4' \geq -\sum_{u=1}^{n_i}\sum_{v=1}^{n_i}\sum_{k=1}^{r}\alpha\mathbf{A}_i(u,v)\mathbf{F}_i(v,k)\mathbf{F}_i(u,k) = T_4.$$

Expanding T_2', we can rewrite it as

$$T_2' = \sum_{u=1}^{n_i}\sum_{v=1}^{n_j}\sum_{k=1}^{r}\sum_{l=1}^{r}\frac{\mathbf{W}_{i,j}^2(u,v)\tilde{\mathbf{F}}_i(u,l)\mathbf{F}_j'(l,v)\mathbf{F}_j(v,k)\mathbf{F}_i^2(u,k)}{\tilde{\mathbf{F}}_i(u,k)}. \tag{22}$$

Let $\mathbf{F}_i(u,k) = \tilde{\mathbf{F}}_i(u,k)\mathbf{Q}_i(u,k)$, and then

$$\begin{aligned} T_2' &= \sum_{u=1}^{n_i}\sum_{v=1}^{n_j}\sum_{k=1}^{r}\sum_{l=1}^{r} \mathbf{W}_{i,j}^2(u,v)\mathbf{F}_j'(l,v)\mathbf{F}_j(v,k)\tilde{\mathbf{F}}_i(u,l)\tilde{\mathbf{F}}_i(u,k)\mathbf{Q}_i^2(u,k) \\ &= \sum_{u=1}^{n_i}\sum_{v=1}^{n_j}\sum_{k=1}^{r}\sum_{l=1}^{r} \mathbf{W}_{i,j}^2(u,v)\mathbf{F}_j'(l,v)\mathbf{F}_j(v,k)\tilde{\mathbf{F}}_i(u,l)\tilde{\mathbf{F}}_i(u,k)\left(\frac{\mathbf{Q}_i^2(u,k)+\mathbf{Q}_i^2(u,l)}{2}\right) \\ &\geq \sum_{u=1}^{n_i}\sum_{v=1}^{n_j}\sum_{k=1}^{r}\sum_{l=1}^{r} \mathbf{W}_{i,j}^2(u,v)\mathbf{F}_j'(l,v)\mathbf{F}_j(v,k)\tilde{\mathbf{F}}_i(u,l)\tilde{\mathbf{F}}_i(u,k)\mathbf{Q}_i(u,k)\mathbf{Q}_i(u,l) \\ &= \sum_{u=1}^{n_i}\sum_{v=1}^{n_j}\sum_{k=1}^{r}\sum_{l=1}^{r} \mathbf{W}_{i,j}^2(u,v)\mathbf{F}_i(u,l)\mathbf{F}_j'(l,v)\mathbf{F}_j(v,k)\mathbf{F}_i(u,k) \\ &= T_2. \end{aligned} \tag{23}$$

For T_3', by using the following inequality in Ding et al. [2006]

$$\sum_{i=1}^{n}\sum_{p=1}^{k} \frac{[\mathbf{A}\mathbf{S}^*\mathbf{B}](i,p)\mathbf{S}^2(i,p)}{\mathbf{S}^*(i,p)} \geq \mathrm{tr}(\mathbf{S}'\mathbf{A}\mathbf{S}\mathbf{B}), \tag{24}$$

where $\mathbf{A} \in \mathbb{R}_+^{n\times n}$, $\mathbf{B} \in \mathbb{R}_+^{k\times k}$, $\mathbf{S} \in \mathbb{R}_+^{n\times k}$, $\mathbf{S}^* \in \mathbb{R}_+^{n\times k}$, and $\mathbf{A}$, $\mathbf{B}$ are symmetric, we have

$$T_3' \geq \alpha\mathrm{tr}(\mathbf{F}_i'\mathbf{T}_i\mathbf{F}_i) = T_3. \tag{25}$$

For T_5', we have $T_5' = T_5$. Putting the above inequalities together, we have $H(\mathbf{F}_i, \tilde{\mathbf{F}}_i) \geq J_i^s(\mathbf{F}_i)$.

Next, we find the global minimum solution to $H(\mathbf{F}_i, \tilde{\mathbf{F}}_i)$. The gradient of $H(\mathbf{F}_i, \tilde{\mathbf{F}}_i)$ is

$$\begin{aligned} \frac{1}{2}\frac{\partial H(\mathbf{F}_i, \tilde{\mathbf{F}}_i)}{\partial \mathbf{F}_i(u,k)} = &- \frac{[(\mathbf{W}_{i,j}\odot\mathbf{W}_{i,j}\odot\mathbf{D}_{i,j})\mathbf{F}_j](u,k)\tilde{\mathbf{F}}_i(u,k)}{\mathbf{F}_i(u,k)} \\ &+ \frac{[(\mathbf{W}_{i,j}\odot\mathbf{W}_{i,j}\odot(\tilde{\mathbf{F}}_i\mathbf{F}_j'))\mathbf{F}_j](u,k)\mathbf{F}_i(u,k)}{\tilde{\mathbf{F}}_i(u,k)} \\ &+ \frac{[\alpha\mathbf{T}_i\tilde{\mathbf{F}}_i](u,k)\mathbf{F}_i(u,k)}{\tilde{\mathbf{F}}_i(u,k)} - \frac{[\alpha\mathbf{A}_i\tilde{\mathbf{F}}_i](u,k)\tilde{\mathbf{F}}_i(u,k)}{\mathbf{F}_i(u,k)} + \beta\mathbf{F}_i(u,k). \end{aligned} \tag{26}$$

From the gradient of $H(\mathbf{F}_i, \tilde{\mathbf{F}}_i)$, we can easily get its Hessian matrix, which is a positive diagonal matrix. Therefore, the global minimum of $H(\mathbf{F}_i, \tilde{\mathbf{F}}_i)$ can be obtained by setting its gradient Equation (26) to zero, which leads to

$$\mathbf{F}_i^2(u,k) = \tilde{\mathbf{F}}_i^2(u,k)\frac{[(\mathbf{W}_{i,j}\odot\mathbf{W}_{i,j}\odot(\tilde{\mathbf{F}}_i\mathbf{F}_j'))\mathbf{F}_j + \alpha\mathbf{A}_i\tilde{\mathbf{F}}_i](u,k)}{[(\mathbf{W}_{i,j}\odot\mathbf{W}_{i,j}\odot\mathbf{D}_{i,j})\mathbf{F}_j + \alpha\mathbf{T}_i\tilde{\mathbf{F}}_i + \beta\tilde{\mathbf{F}}_i](u,k)}. \tag{27}$$

Recall that we have set $\mathbf{F}_i^{(t+1)} = \mathbf{F}_i$ and $\mathbf{F}_i^{(t)} = \tilde{\mathbf{F}}_i$. The above equation proves that the updating rule in Equation (2) decreases monotonically. □

According to Theorem 3.1 and Lemma 3.2, we conclude that Algorithm 1 converges to a local minima solution for Equation (2) w.r.t. each individual $\mathbf{F}_i$.

Efficiency Analysis. In terms of efficiency, we analyze both the time complexity and the space complexity of the proposed FASCINATE algorithm, which are summarized in Lemmas 3.3 and 3.4. We can see that FASCINATE scales linearly w.r.t. the size of the entire multi-layered network.

LEMMA 3.3. *The time complexity of Algorithm 1 is* $O([\sum_{i=1}^{g}(\sum_{j:\ \mathbf{G}(i,j)=1}(m_{i,j}r + (n_i + n_j)r^2) + m_i r)]t)$.

PROOF. In each iteration in Algorithm 1 for updating $\mathbf{F}_i$, the complexity of calculating $\mathbf{X}$ by Equation (6) is $O(\sum_{j:\ \mathbf{G}(i,j)=1} m_{i,j}r + m_i r)$ due to the sparsity of $\mathbf{D}_{i,j}$ and $\mathbf{A}_i$. The complexity of computing $\tilde{\mathbf{R}}_{i,j}$ in $\mathbf{Y}$ is $O(m_{i,j}r)$. Computing $\mathbf{F}_i(\mathbf{F}'_j\mathbf{F}_j)$ requires $O((n_i + n_j)r^2)$ operations and computing $\alpha\mathbf{T}_i\mathbf{F}_i + \beta\mathbf{F}_i$ requires $O(n_i r)$ operations. So, it is of $O(\sum_{j:\ \mathbf{G}(i,j)=1}(m_{i,j}r + (n_i + n_j)r^2))$ complexity to get $\mathbf{Y}$ in line 7. Therefore, it takes $O(\sum_{j:\ \mathbf{G}(i,j)=1}(m_{i,j}r + (n_i + n_j)r^2) + m_i r)$ to update $\mathbf{F}_i$. Putting all together, the complexity of updating all low-rank matrices in each iteration is $O(\sum_{i=1}^{g}(\sum_{j:\ \mathbf{G}(i,j)=1}(m_{i,j}r + (n_i + n_j)r^2) + m_i r))$. Thus, the overall complexity of Algorithm 1 is $O([\sum_{i=1}^{g}(\sum_{\mathbf{G}(i,j)=1}(m_{i,j}r + (n_i + n_j)r^2) + m_i r)]t)$, where t is the maximum number of iterations in the algorithm. □

LEMMA 3.4. *The space complexity of Algorithm 1 is* $O(\sum_{i=1}^{g}(n_i r + m_i) + \sum_{i,j:\ \mathbf{G}(i,j)=1} m_{i,j})$
.

PROOF. It takes $O(\sum_{i=1}^{g} n_i r)$ to store all the low-rank matrices, and $O(\sum_{i=1}^{g} m_i + \sum_{i,j:\ \mathbf{G}(i,j)=1} m_{i,j})$ to store all the within-layer connectivity matrices and dependency matrices in the multi-layered network. To calculate $\mathbf{X}$ for $\mathbf{F}_i$, it costs $O(n_i r)$ to compute $\sum_{j:\ \mathbf{G}(i,j)=1} \mathbf{D}_{i,j}\mathbf{F}_j$ and $\alpha\mathbf{A}_i\mathbf{F}_i$. For $\mathbf{Y}$, the space cost of computing $\tilde{\mathbf{R}}_{i,j}$ and $\mathbf{F}_i(\mathbf{F}'_j\mathbf{F}_j)$ is $O(m_{i,j})$ and $O(n_i r)$ respectively. Therefore, the space complexity of calculating $\sum_{j:\ \mathbf{G}(i,j)=1}((1 - w^2)\tilde{\mathbf{R}}_{i,j} + w^2\mathbf{F}_i\mathbf{F}_j{}')\mathbf{F}_j$ is $O(\max_{j:\ \mathbf{G}(i,j)=1} m_{i,j} + n_i r)$. On the other hand, the space required to compute $\alpha\mathbf{T}_i\mathbf{F}_i + \beta\mathbf{F}_i$ is $O(n_i r)$. Putting all together, the space cost of updating all low-rank matrices in each iteration is of $O(\max_{i,j:\ \mathbf{G}(i,j)=1} m_{i,j} + \max_i n_i r)$. Thus, the overall space complexity of Algorithm 1 is $O(\sum_{i=1}^{g}(n_i r + m_i) + \sum_{i,j:\ \mathbf{G}(i,j)=1} m_{i,j})$. □

3.4. Variants

Here, we discuss some variants of the proposed FASCINATE algorithm.

3.4.1. Collective One Class Collaborative Filtering. By setting $w \in (0, 1)$, FASCINATE can be used to address one class collaborative filtering (OCCF) problem, where implicit dependencies extensively exist between nodes from different layers. Specifically, in two-layered networks, FASCINATE is reduced to *wiZAN-Dual*, a weighting-based, dual-regularized OCCF algorithm proposed in Yao et al. [2014].

3.4.2. Multi-Layered Network Clustering. By setting all the entries in the weight matrix $\mathbf{W}_{i,j}$ to 1 in Equation (1), we have the following objective function:

$$\min_{\mathbf{F}_i \geq \mathbf{0}(i=1,\ldots,g)} J = \sum_{i,j:\ \mathbf{G}(i,j)=1} \|\mathbf{D}_{i,j} - \mathbf{F}_i\mathbf{F}_j{}'\|_F^2 + \alpha\sum_{i=1}^{g}\mathrm{tr}(\mathbf{F}_i{}'(\mathbf{T}_i - \mathbf{A}_i)\mathbf{F}_i) + \beta\sum_{i=1}^{g}\|\mathbf{F}_i\|_F^2, \quad (28)$$

where $\mathbf{F}_i$ can be viewed as the cluster membership matrix for nodes in layer-i). By following similar procedure in Section 3.2, we can get the local optima of the above objective function with the following updating rule:

$$\mathbf{F}_i(u, v) \leftarrow \mathbf{F}_i(u, v)\sqrt{\frac{\mathbf{X}_c(u, v)}{\mathbf{Y}_c(u, v)}}, \quad (29)$$

where

$$\mathbf{X}_c = \sum_{j:\ \mathbf{G}(i,j)=1} \mathbf{D}_{i,j}\mathbf{F}_j + \alpha \mathbf{A}_i \mathbf{F}_i \tag{30}$$

$$\mathbf{Y}_c = \sum_{j:\ \mathbf{G}(i,j)=1} \mathbf{F}_i \mathbf{F}_j{}' \mathbf{F}_j + \alpha \mathbf{T}_i \mathbf{F}_i + \beta \mathbf{F}_i. \tag{31}$$

Although in the above updating rule, we do not need to calculate $\tilde{\mathbf{R}}_{i,j}$ for $\mathbf{Y}_c$ comparing to $\mathbf{Y}$ in Equation (7), the overall time complexity for the algorithm is still $O([\sum_{i=1}^{g}(\sum_{j:\ \mathbf{G}_{(i,j)}=1}(m_{i,j}r + (n_i + n_j)r^2) + m_i r)]t)$. If we restrict ourselves to two-layered networks (i.e., $g = 2$), the above variant for FASCINATE becomes a dual regularized co-clustering algorithm [Liu et al. 2015b].

3.4.3. Unconstrained FASCINATE. In FASCINATE, we place a non-negative constraint on the latent features $\{\mathbf{F}_i\}_{1=1\ldots g}$ in Equation (1) to pursue good interpretability and efficiency. By discarding the non-negative constraint, we have FASCINATE-UN, an unconstrained variant of FASCINATE, which can be solved with a gradient descent method as shown in the Algorithm 2. It first randomly initializes the low-rank matrices for each layer (line 1–line 3) and then begins the iterative update procedure. In each iteration (line 4–line 10), the algorithm alternatively updates $\{\mathbf{F}_i\}_{i=1,\ldots,g}$ with the gradient descent method one by one. Similar to FASCINATE, the two criteria we use to terminate the iteration are (1) either the difference of the objective function (J in Equation (1)) between two successive iterations is less than a threshold ξ, or (2) the maximum iteration number t is reached. The complexity of computing $\frac{\partial J_i}{\partial \mathbf{F}_i}$ is the same with the complexity of computing $\mathbf{X}$ and $\mathbf{Y}$ in Algoritm 1. However, in the backtracking line search procedure in step 7, calculating the value of the objective function J_i is required to find step size τ with complexity $O(\sum_{j:\ \mathbf{G}(i,j)=1} n_i n_j r + n_i^2 r)$. This quadratic complexity would increase the overall complexity of Algoritm 2 significantly in large systems.

ALGORITHM 2: The FASCINATE-UN algorithm

Input: (1) a multi-layered network with (a) layer–layer dependency matrix $\mathbf{G}$, (b) within-layer connectivity matrices $\mathcal{A} = \{\mathbf{A}_1, \ldots, \mathbf{A}_g\}$, and (c) observed cross-layer node dependency matrices $\mathcal{D} = \{\mathbf{D}_{i,j}\ i, j = 1, \ldots, g\}$; (2) the rank size r; (3) weight w; (4) regularized parameters α and β; (5) parameters $a \in (0, 0.5)$, $b \in (0, 1)$

Output: low-rank representations for each layer $\{\mathbf{F}_i\}_{i=1,\ldots,g}$

```
1: for $i = 1$ to $g$ do
2:    initialized $\mathbf{F}_i$ as $n_i \times r$ random matrix
3: end for
4: while not converge do
5:    for $i = 1$ to $g$ do
6:       compute $\frac{\partial J_i}{\partial \mathbf{F}_i}$ with Equation (3)
7:       $\tau$ ←step size from backtracking line search
8:       $\mathbf{F}_i \leftarrow \mathbf{F}_i - \tau \frac{\partial J_i}{\partial \mathbf{F}_i}$
9:    end for
10: end while
11: return $\{\mathbf{F}_i\}_{i=1,\ldots,g}$
```

3.4.4. Collective Matrix Factorization. Instead of exploiting node homophily effect from each layers, we can view the within-layer networks as additional constraints for matrix factorization problem as modeled in the following objective function:

$$\min_{\mathbf{F}_i \geq \mathbf{0}(i=1,\ldots,g)} \sum_{i,j:\ \mathbf{G}(i,j)=1} \|\mathbf{W}_{i,j} \odot (\mathbf{D}_{i,j} - \mathbf{F}_i \mathbf{F}_j{}')\|_F^2 + \alpha \sum_{i=1}^{g} \|\mathbf{A}_i - \mathbf{F}_i \mathbf{F}_i{}'\|_F^2 + \beta \sum_{i=1}^{g} \|\mathbf{F}_i\|_F^2, \quad (32)$$

where $\mathbf{F}_i$ is the latent features for nodes in layer-*i*.

Again, the above problem can be solved with similar procedure in Fascinate. The updating rules are as follows:

$$\mathbf{F}_i(u, v) \leftarrow \mathbf{F}_i(u, v) \sqrt{\frac{\mathbf{X}_{col}(u, v)}{\mathbf{Y}_{col}(u, v)}}, \quad (33)$$

where $\mathbf{X}_{col}$ and $\mathbf{Y}_{col}$ are defined as follows:

$$\mathbf{X}_{col} = \sum_{j:\ \mathbf{G}(i,j)=1} \mathbf{D}_{i,j} \mathbf{F}_j + 2\alpha \mathbf{A}_i \mathbf{F}_i \quad (34)$$

$$\mathbf{Y}_{col} = \sum_{j:\ \mathbf{G}(i,j)=1} ((1 - w^2)\tilde{\mathbf{R}}_{i,j} + w^2 \mathbf{F}_i \mathbf{F}_j{}')\mathbf{F}_j + 2\alpha \mathbf{F}_i \mathbf{F}_i{}' \mathbf{F}_i + \beta \mathbf{F}_i. \quad (35)$$

The complexity of the above method is of the same order with Fascinate. In particular, when the within-layer connectivity matrices $\mathcal{A} = \{\mathbf{A}_1, \ldots, \mathbf{A}_g\}$ are absent, the proposed Fascinate can be viewed as a collective matrix factorization method in Singh and Gordon [2008].

While the proposed Fascinate includes these existing methods as its special cases, its major advantage lies in its ability to collectively leverage all the available information (e.g., the within-layer connectivity and the observed cross-layer dependency) for dependency inference. As we will demonstrate in the experimental section, such a methodical strategy leads to a substantial and consistent inference performance boosting. Nevertheless, a largely unanswered question for these methods (including Fascinate) is how to handle *zero-start* nodes. That is, when a new node arrives with no observed cross-layer dependencies, how can we effectively and efficiently infer its dependencies without rerunning the algorithm from scratch. In the next section, we present a *sub-linear* algorithm to solve this problem (i.e., Problem 2).

4. FASCINATE-ZERO FOR PROBLEM 2

A multi-layered network often exhibits high dynamics, e.g., the arrival of new nodes. For example, for a newly identified chemical in the biological system, we might know how it interacts with some existing chemicals (i.e., the within-layer connectivity). However, its cross-layer dependencies w.r.t. proteins and/or diseases might be completely unknown. This section addresses such *zero-start* problems (i.e., Problem 2). Without loss of generality, we assume that the newly added node resides in layer-*1*, indexed as its $(n_1 + 1)$th node. The within-layer connectivity between the newly added node and the existing n_1 nodes is represented by a $1 \times n_1$ row vector $\mathbf{s}$, where $\mathbf{s}(u)$ $(u = 1, \ldots, n_1)$ denotes the (within-layer) connectivity between the newly added node and the uth existing node in layer-*1*.

We could just rerun our Fascinate algorithm on the entire multi-layered network with the newly added node to get its low-rank representation (i.e., a $1 \times r$ row vector $\mathbf{f}$), based on which its cross-layer dependencies can be estimated. However, the running time of this strategy is linear w.r.t. the size of the *entire* multi-layered network. For example, on a three-layered infrastructure network whose size is in the order of 14 million, it would take Fascinate 2, 500+ seconds to update the low-rank matrices $\{\mathbf{F}_i\}$ for a *zero-start* node with rank $r = 200$, which might be too costly in online settings.

In contrast, our upcoming algorithm is *sub-linear*, and it only takes less than 0.001 seconds on the same network without jeopardizing the accuracy.

There are two key ideas behind our online algorithm. The first is to view the newly added node as a perturbation to the original network. In detail, the updated within-layer connectivity matrix $\hat{\mathbf{A}}_1$ for layer-*1* can be expressed as

$$\hat{\mathbf{A}}_1 = \begin{bmatrix} \mathbf{A}_1 & \mathbf{s}' \\ \mathbf{s} & 0 \end{bmatrix}, \tag{36}$$

where $\mathbf{A}_1$ is the within-layer connectivity matrix for layer-*1* before the arrival of the new node.

Correspondingly, the updated low-rank representation matrix for layer-*1* can be expressed as $\hat{\mathbf{F}}_1 = [\hat{\mathbf{F}}'_{1(n_1\times r)}\ \mathbf{f}']'$, where $\hat{\mathbf{F}}_{1(n_1\times r)}$ is the updated low-rank representation for the existing n_1 nodes in layer-*1*. Then, the new objective function $\hat{J}$ in Equation (1) can be reformatted as

$$\begin{aligned}\hat{J} = &\sum_{\substack{i,j:\ \mathbf{G}(i,j)=1\\ i,j\neq 1}} \|\mathbf{W}_{i,j}\odot(\mathbf{D}_{i,j}-\hat{\mathbf{F}}_i\hat{\mathbf{F}}'_j)\|_F^2 + \sum_{j:\ \mathbf{G}(1,j)=1} \|\hat{\mathbf{W}}_{1,j}\odot(\hat{\mathbf{D}}_{1,j}-\hat{\mathbf{F}}_1\hat{\mathbf{F}}'_j)\|_F^2 \\ &+ \sum_{i=2}^{g}\frac{\alpha}{2}\sum_{u=1}^{n_i}\sum_{v=1}^{n_i}\mathbf{A}_i(u,v)\|\hat{\mathbf{F}}_i(u,:)-\hat{\mathbf{F}}_i(v,:)\|_2^2 + \frac{\alpha}{2}\sum_{u=1}^{n_1}\sum_{v=1}^{n_1}\mathbf{A}_1(u,v)\|\hat{\mathbf{F}}_1(u,:)-\hat{\mathbf{F}}_1(v,:)\|_2^2 \\ &+ \beta\sum_{i=2}^{g}\|\hat{\mathbf{F}}_i\|_F^2 + \beta\|\hat{\mathbf{F}}'_{1(n_1\times r)}\|_F^2 + \alpha\sum_{v=1}^{n_1}\mathbf{s}(v)\|\mathbf{f}-\hat{\mathbf{F}}_1(v,:)\|_2^2 + \beta\|\mathbf{f}\|_2^2.\end{aligned} \tag{37}$$

Since the newly added node has no dependencies, we can set

$$\hat{\mathbf{W}}_{1,j} = \begin{bmatrix} \mathbf{W}_{1,j} \\ \mathbf{0}_{(1\times n_j)} \end{bmatrix},\quad \hat{\mathbf{D}}_{1,j} = \begin{bmatrix} \mathbf{D}_{1,j} \\ \mathbf{0}_{(1\times n_j)} \end{bmatrix}.$$

Therefore, the second term in $\hat{J}$ can be simplified as

$$\sum_{j:\ \mathbf{G}(1,j)=1} \|\mathbf{W}_{1,j}\odot(\mathbf{D}_{1,j}-\hat{\mathbf{F}}_{1(n_1\times r)}\hat{\mathbf{F}}'_j)\|_F^2. \tag{38}$$

Combining Equation (37), Equation (38), and J in Equation (1) together, $\hat{J}$ can be expressed as

$$\hat{J} = J + J^1, \tag{39}$$

where $J^1 = \alpha\sum_{v=1}^{n_1}\mathbf{s}(v)\|\mathbf{f}-\hat{\mathbf{F}}_1(v,:)\|_2^2 + \beta\|\mathbf{f}\|_2^2$, and J is the objective function without the newly arrived node.

The second key idea of our online algorithm is that in Equation (39), J is often orders of magnitude larger than J^1. For example, in the *BIO* dataset used in Section 5.2.2, J is in the order of 10^3, while J^1 is in the order of 10^{-1}. This naturally leads to the following approximation strategy, that is, we (1) fix J with $\{\mathbf{F}_i^*\}_{i=1,\ldots,g}$ (i.e., the previous local optimal solution to Equation (1) without the newly arrived node), and (2) optimize J^1 to find out the low-rank representation $\mathbf{f}$ for the newly arrived node. That is, we seek to solve the following optimization problem:

$$\mathbf{f} = \arg\min_{\mathbf{f}\geq\mathbf{0}} J^1 \quad \text{subject to: } \hat{\mathbf{F}}_{1(n_1\times r)} = \mathbf{F}_1^* \tag{40}$$

with which, we can get an approximate solution $\{\hat{\mathbf{F}}_i\}_{i=1,\ldots,g}$ to $\hat{J}$.

To solve $\mathbf{f}$, we take the derivative of J^1 w.r.t. $\mathbf{f}$ and get

$$\begin{aligned}\frac{1}{2}\frac{\partial J^1}{\partial \mathbf{f}} &= \beta\mathbf{f} + \alpha\sum_{v=1}^{n_1}\mathbf{s}(v)(\mathbf{f} - \mathbf{F}_1^*(v,:)) \\ &= (\beta + \alpha\sum_{v=1}^{n_1}\mathbf{s}(v))\mathbf{f} - \alpha\mathbf{s}\mathbf{F}_1^*.\end{aligned} \tag{41}$$

Since α and β are positive, the Hessian matrix of J^1 is a positive diagonal matrix. Therefore, the global minimum of J^1 can be obtained by setting its derivative to zero. Then, the optimal solution to J^1 can be expressed as

$$\mathbf{f} = \frac{\alpha\mathbf{s}\mathbf{F}_1^*}{\beta + \alpha\sum_{v=1}^{n_1}\mathbf{s}(v)}. \tag{42}$$

For the newly added node, $\mathbf{f}$ can be viewed as the weighted average of its neighbors' low-rank representations. Notice that in Equation (42), the non-negativity constraint on $\mathbf{f}$ naturally holds. Therefore, we refer to this solution (i.e., Equation (42)) as FASCINATE-ZERO. In this way, we can successfully decouple the cross-layer dependency inference problem for *zero-start* node from the entire multi-layered network and localize it only among its neighbors in layer-*1*. The localization significantly reduces the time complexity, as summarized in Lemma 4.1, which is linear w.r.t. the number of neighbors of the new node (and therefore is *sub-linear* w.r.t. the size of the entire network).

LEMMA 4.1. *Let nnz*($\mathbf{s}$) *denotes the total number of within-layer links between the newly added node and the original nodes in layer-1 (i.e., nnz*($\mathbf{s}$) *is the degree for the newly added node). Then, the time complexity of* FASCINATE-ZERO *is* $O(nnz(\mathbf{s})r)$.

PROOF. Since the links between the newly added node and the original nodes in layer-*1* are often very sparse, the number of non-zero elements in $\mathbf{s}$ (nnz($\mathbf{s}$)) is much smaller than n_1. Therefore, the complexity of computing $\mathbf{s}\mathbf{F}_1^*$ can be reduced to $O(\text{nnz}(\mathbf{s})r)$. The multiplication between α and $\mathbf{s}\mathbf{F}_1^*$ takes $O(r)$. Computing $\sum_{v=1}^{n_1}\mathbf{s}(v)$ takes $O(\text{nnz}(\mathbf{s}))$. Thus, the overall complexity of computing $\mathbf{f}$ is $O(\text{nnz}(\mathbf{s})r)$. □

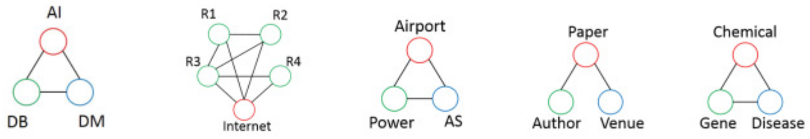
Remarks. Following the similar procedure in FASCINATE-ZERO, it is easy to extend the zero-start problem to the scenario where a new within-layer edge is added to two existing nodes. Suppose in layer-*1*, a new edge $< u, v >$ is added between node u and node v. To find out the updated low-rank matrices $\{\hat{\mathbf{F}}_i\}$ efficiently after the perturbation, we can partition the nodes in the multi-layered network into following two parts: (1) nodes that can be affected by either node u or node v (denoted as $\mathcal{N}^{\{u,v\}}$) and (2) nodes that are irrelevant to both node u and node v (denoted as $\mathcal{N}^{\backslash\{u,v\}}$). Specifically, we define that node w can be affected by node u if and only if there exists a path from u to w, and the links in the path can be either within-layer edges or cross-layer dependencies; otherwise, node w is viewed as irrelevant to u. By this definition, we have $\mathcal{N}^{\{u,v\}} \cap \mathcal{N}^{\backslash\{u,v\}} = \Phi$ and the new objective function $\hat{J}$ can be decomposed into two parts as

$$\hat{J} = \hat{J}^{\{u,v\}} + \hat{J}^{\backslash\{u,v\}}, \tag{43}$$

where $\hat{J}^{\{u,v\}}$ only contains the optimization terms for the latent features of the affected nodes ($\{\hat{\mathbf{F}}_i\}^{\{u,v\}}$), while $\hat{J}^{\backslash\{u,v\}}$ contains the terms for latent features of irrelevant nodes ($\{\hat{\mathbf{F}}_i\}^{\backslash\{u,v\}}$). As the newly added edge $< u, v >$ in layer-*1* would not cause any changes in $\hat{J}^{\backslash\{u,v\}}$, $\{\hat{\mathbf{F}}_i\}^{\backslash\{u,v\}}$ would remain the same with the previous local optima solution $\{\mathbf{F}_i^*\}^{\backslash\{u,v\}}$.

Table II. Statistics of Datasets

Dataset	# of Layers	# of Nodes	# of Links	# of CrossLinks
CITATION	3	33,249	27,017	4,589
INFRA-5	5	349	379	565
INFRA-3	3	15,126	29,861	28,023,500
SOCIAL	3	125,344	214,181	188,844
BIO	3	35,631	253,827	75,456



(a) CITATION (b) INFRA-5 (c) INFRA-3 (d) SOCIAL (e) BIO

Fig. 3. The abstract dependency structure of each dataset.

Therefore, the only terms, we need to optimize is $\hat{J}^{\{u,v\}}$ w.r.t. the affected latent features $\{\hat{\mathbf{F}}_i\}^{\{u,v\}}$.

5. EVALUATIONS

In this section, we evaluate the proposed Fascinate algorithms. All experiments are designed to answer the following questions:

—*Effectiveness*. How effective are the proposed Fascinate algorithms in inferring the missing cross-layer dependencies?

—*Efficiency*. How fast and scalable are the proposed algorithms?

5.1. Experimental Setup

5.1.1. Datasets Description. We perform our evaluations on five different datasets, including (1) a three-layer cross-domain paper citation network in the academic research domain (CITATION); (2) a five-layer Italy network in the critical infrastructure domain (INFRA-5); (3) a three-layer network in the critical infrastructure domain (INFRA-3); (4) a three-layer Comparative Toxicogenomics Database (CTD) network in the biological domain (BIO); and (5) a three-layer Aminer academic network in the social collaboration domain (SOCIAL). The statistics of these datasets are shown in Table II, and the abstract layer–layer dependency graphs of these four datasets are summarized in Figure 3 In all these four datasets, the cross-layer dependencies are binary and undirected (i.e., $\mathbf{D}_{i,j}(u, v) = \mathbf{D}_{j,i}(v, u)$).

CITATION. The construction of this publication network is based on the work in Li et al. [2015]. It contains three layers, which correspond to the paper citation networks in Artificial Intelligence (AI), Database (DB), and Data Mining (DM) domains. The cross-domain citations naturally form the cross-layer dependencies in the system. For example, the cross-layer dependency between AI layer and DM layer indicates the citations between AI papers and DM papers. The papers in the system are from the top conferences in the corresponding areas as shown in Table III. The number of nodes in each layer varies from 5,158 to 18,243, and the number of within-layer links ranges from 20,611 to 40,885. The number of cross-layer dependencies ranges from 536 to 2,250. The structure of the entire system is shown in Figure 3(a).

INFRA-5. The construction of this critical infrastructure network is based on the data implicated from an electrical blackout in Italy in Sept 2003 [Rosato et al. 2008]. It contains five layers, including four layers of regional power grids and one Internet

Table III. List of Conferences in Each Domain

Domain	AI	DM	DB
Conferences	IJCAI	KDD	SIGMOD
	AAAI	ICDM	VLDB
	ICML	SDM	ICDM
	NIPS	PKDD	PODS

network [Rosato et al. 2008]. The regional power grids are partitioned by macroregions.[2] To make the regional networks more balanced, we merge the Southern Italy power grid and the Island power grid together. The power transfer lines between the four regions are viewed as cross-layer dependencies. For the Italy Internet network, it is assumed that each Internet center is supported by the power stations within a radius of 70km. Its abstract dependency graph is shown in Figure 3(b). The smallest layer in the network has 39 nodes and 50 links, while the largest network contains 151 nodes and 158 links. The number of dependencies is up to 307.

INFRA-3. This dataset contains the following three critical infrastructure networks: an airport network,[3] an AS network,[4] and a power grid [Watts and Strogatz 1998]. We construct a three-layered network in the same way as Chen et al. [2015]. The three infrastructure networks are functionally dependent on each other. Therefore, they form a triangle-shaped multi-layered network as shown in Figure 3(c). The construction of the cross-layer dependencies is based on geographic proximity.

SOCIAL. This dataset contains three layers, including a collaboration network among authors, a citation network between papers, and a venue network [Tang et al. 2008]. The number of nodes in each layer ranges from 899 to 62,602, and the number of within-layer links ranges from 2,407 to 201,037. The abstract layer–layer dependency graph of SOCIAL is shown in Figure 3(d). The collaboration layer is connected to the paper layer with the authorship dependency, while the venue layer is connected to the paper layer with publishing dependency. For the *Paper-Author* dependency, we have 126,242 links cross the two layers; for the *Paper-Venue* dependency, we have 62,602 links.

BIO. The construction of CTD network is based on the works in Davis et al. [2015], Razick et al. [2008], and Van Driel et al. [2006]. It contains three layers, which are chemical, disease, and gene similarity networks. The number of nodes in these networks ranges from 4,256 to 25,349, and the number of within-layer links ranges from 30,551 to 154,167. The interactions between chemicals, genes, and diseases form the cross-layer dependency network as shown in Figure 3(e). For *Chemical–Gene* dependency, we have 53,735 links cross the two layers; for *Chemical–Disease* dependency, we have 19,771 links; and for *Gene–Disease* dependency, we have 1,950 links.

For all datasets, we randomly select 50% cross-layer dependencies as the training set and use the remaining 50% as the test set.

5.1.2. Comparing Methods. We compare FASCINATE with the following methods, including (1) FASCINATE-CLUST—a variant of the proposed method for the purpose of dependency clustering, (2) FASCINATE-UN—a variant of FASCINATE without non-negative constraint, (3) *MulCol*—a collective matrix factorization method [Singh and Gordon 2008], (4) *PairSid*—a pairwise OCCF method proposed in Yao et al. [2014], (5) *PairCol*—a pairwise collective matrix factorization method degenerated from *MulCol*, (6) *PairNMF*—a pairwise non-negative matrix factorization (*NMF*)-based method [Lin 2007], (7) *Pair-*

[2] https://en.wikipedia.org/wiki/First-level_NUTS_of_the_European_Union.
[3] http://www.levmuchnik.net/Content/Networks/NetworkData.html.
[4] http://snap.stanford.edu/data/.

Rec—a pairwise matrix factorization-based algorithm introduced in Koren et al. [2009], (8) *FlatNMF*—an NMF based method that treats the input multi-layered network as a flat-structured single network (i.e., by putting the within-layer connectivity matrices in the diagonal blocks, and the cross-layer dependency matrices in the off-diagonal blocks), and (9) *FlatRec*—a matrix factorization-based method using the same techniques as *PairRec* but treating the input multi-layered network as a single network as in *FlatNMF*.

For the experimental results reported in this article, we set rank $r = 100$, maximum iteration $t = 100$, termination threshold $\xi = 10^{-8}$, weight $w^2 = 0.1$, regularization parameters $\alpha = 0.1$, $\beta = 0.1$, and backtracking line search parameters $a = 0.1$, $b = 0.8$ unless otherwise stated.

5.1.3. Evaluation Metrics. We use the following metrics for the effectiveness evaluations.

—*MAP*. It measures the mean average precision over all entities in the cross-layer dependency matrices [Li et al. 2010]. A larger *MAP* indicates better inference performance.

—*R-MPR*. It is a variant of Mean Percentage Ranking for OCCF [Hu et al. 2008]. *MPR* is originally used to measure the user's satisfaction of items in a ranked list. In our case, we can view the nodes from one layer as users, and the nodes of the dependent layer(s) as items. The ranked list therefore can be viewed as ordered dependencies by their importance. Smaller *MPR* indicates better inference performance. Specifically, for a randomly produced list, its *MPR* is expected to be 50%. Here, we define R-MPR = 0.5 − MPR so that larger *R-MRP* indicates better inference performance.

—*HLU*. Half-Life Utility is also a metric from OCCF. By assuming that the user will view each consecutive items in the list with exponential decay of possibility, it estimates how likely a user will choose an item from a ranked list [Pan et al. 2008]. In our case, it measures how likely a node will establish dependencies with the nodes in the ranked list. A larger *HLU* indicates better inference performance.

—*AUC*. Area under ROC (Receiver Operating Characteristic) curve is a metric that measures the classification accuracy. A larger *AUC* indicates better inference performance.

—*Prec@K*. Precision at *K* is defined by the proportion of true dependencies among the top *K* inferred dependencies. A larger *Prec@K* indicates better inference performance.

5.1.4. Machine and Repeatability. All the experiments are performed on a machine with two processors of Intel Xeon 3.5GHz with 256GB of RAM. The algorithms are programmed with MATLAB using single thread. We will release the code and all the non-proprietary datasets after the paper is published.

5.2. Effectiveness

In this section, we aim to answer the following three questions, (1) how effective is FASCINATE for Problem 1 (i.e., CODE)? (2) how effective is FASCINATE-ZERO for Problem 2 (i.e., CODE-ZERO)? and (3) how sensitive are the proposed algorithms w.r.t. the model parameters?

5.2.1. Effectiveness of FASCINATE. We compare the proposed algorithms and the existing methods on all the five datasets. The results are shown in Table IV through Table VIII. As FASCINATE-UN is not scalable to large networks, we only evaluate its performance on two small datasets—CITATION and INFRA-5 (Table IV and Table V). There are several interesting observations. First is that our proposed FASCINATE algorithm and its variants (FASCINATE-CLUST and FASCINATE-UN) consistently outperform all other methods in terms of all the five evaluation metrics. We perform a t-test between FASCINATE algorithms and other comparing methods w.r.t. the MAP metric. The results show that

Table IV. Cross-Layer Dependency Inference on CITATION

Methods	MAP	R-MPR	HLU	AUC	Prec@10
Fascinate	0.1389	**0.3907**	19.1264	**0.8523**	0.0428
Fascinate-Clust	0.1347	0.3882	19.8367	0.8487	0.0407
Fascinate-UN	**0.1873**	0.2685	**25.1961**	0.7423	**0.0532**
MulCol	0.1347	0.3882	19.8367	0.8487	0.0459
PairSid	0.1623	0.3868	21.8641	0.8438	0.0480
PairCol	0.1311	0.3838	19.1697	0.8388	0.0446
PairNMF	0.0338	0.1842	4.4397	0.6009	0.0103
PairRec	0.0351	0.2582	5.3407	0.6527	0.0129
FlatNMF	0.0811	0.3539	12.1835	0.8084	0.0284
FlatRec	0.0032	0.3398	0.0608	0.8113	0.0001

Table V. Cross-Layer Dependency Inference on INFRA-5

Methods	MAP	R-MPR	HLU	AUC	Prec@10
Fascinate	**0.5040**	**0.3777**	**67.2231**	**0.8916**	**0.2500**
Fascinate-Clust	0.4297	0.3220	56.8215	0.8159	0.2340
Fascinate-UN	0.4354	0.3631	60.2393	0.8575	0.2412
MulCol	0.4523	0.3239	59.8115	0.8329	0.2413
PairSid	0.3948	0.2392	49.5484	0.7413	0.2225
PairCol	0.3682	0.2489	48.5966	0.7406	0.2309
PairNMF	0.1315	0.0464	15.7148	0.5385	0.0711
PairRec	0.0970	0.0099	9.4853	0.5184	0.0399
FlatNMF	0.3212	0.2697	44.4654	0.7622	0.1999
FlatRec	0.1020	0.0778	11.5598	0.5740	0.0488

Table VI. Cross-Layer Dependency Inference on INFRA-3

Methods	MAP	R-MPR	HLU	AUC	Prec@10
Fascinate	0.4780	0.0788	**55.7289**	0.6970	**0.5560**
Fascinate-Clust	**0.5030**	**0.0850**	49.1223	**0.7122**	0.4917
Fascinate-UN	–	–	–	–	–
MulCol	0.4606	0.0641	49.3585	0.6706	0.4930
PairSid	0.4253	0.0526	47.7284	0.5980	0.4773
PairCol	0.4279	0.0528	48.1314	0.5880	0.4816
PairNMF	0.4275	0.0511	48.8478	0.5579	0.4882
PairRec	0.3823	0.0191	38.9226	0.5756	0.3895
FlatNMF	0.4326	0.0594	45.0090	0.6333	0.4498
FlatRec	0.3804	0.0175	38.0550	0.5740	0.3805

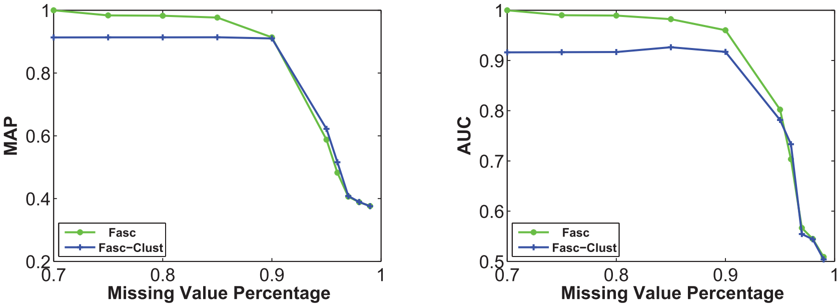
Fascinate is significantly better with a 0.01 significance level. Second, by exploiting the structure of multi-layered network, Fascinate, Fascinate-Clust, Fascinate-UN, and *MulCol* can achieve significantly better performance than the pairwise methods in most datasets. Third, among the pairwise baselines, *PairSid* and *PairCol* are better than *PairNMF* and *PairRec*. The main reason is that the first two algorithms utilize both within-layer connectivity matrices and cross-layer dependency matrix for matrix factorization, while the latter two only use the observed dependency matrix. Finally, the relatively poor performance of *FlatNMF* and *FlatRec* implies that simply flattening the multi-layered network into a single network is insufficient to capture the intrinsic correlations across different layers.

Table VII. Cross-Layer Dependency Inference on SOCIAL

Methods	MAP	R-MPR	HLU	AUC	Prec@10
Fascinate	0.0660	**0.2651**	**8.4556**	**0.7529**	**0.0118**
Fascinate-Clust	**0.0667**	0.2462	8.2160	0.7351	0.0108
Fascinate-UN	–	–	–	–	–
MulCol	0.0465	0.2450	6.0024	0.7336	0.0087
PairSid	0.0308	0.1729	3.8950	0.6520	0.0062
PairCol	0.0303	0.1586	3.7857	0.6406	0.0056
PairNMF	0.0053	0.0290	0.5541	0.4998	0.0007
PairRec	0.0056	0.0435	0.5775	0.5179	0.0007
FlatNMF	0.0050	0.0125	0.4807	0.5007	0.0007
FlatRec	0.0063	0.1009	0.6276	0.5829	0.0009

Table VIII. Cross-Layer Dependency Inference on Bio

Methods	MAP	R-MPR	HLU	AUC	Prec@10
Fascinate	**0.3979**	**0.4066**	**45.1001**	**0.9369**	**0.1039**
Fascinate-Clust	0.3189	0.3898	37.4089	0.9176	0.0857
Fascinate-UN	–	–	–	–	–
MulCol	0.3676	0.3954	42.8687	0.9286	0.0986
PairSid	0.3623	0.3403	40.4048	0.8682	0.0941
PairCol	0.3493	0.3153	38.4364	0.8462	0.0889
PairNMF	0.1154	0.1963	15.8486	0.6865	0.0393
PairRec	0.0290	0.2330	3.6179	0.7105	0.0118
FlatNMF	0.2245	0.2900	26.1010	0.8475	0.0615
FlatRec	0.0613	0.3112	8.4858	0.8759	0.0254



(a) MAP (b) AUC

Fig. 4. Performance of Fascinate and Fascinate-Clust on INFRA-3 dataset under different missing value percentages.

We also test the sensitivity of the proposed algorithms w.r.t. the sparsity of the observed cross-layer dependency matrices (i.e., the ratio of the missing values) on INFRA-3. The results in Figure 4 demonstrate that both Fascinate and Fascinate-Clust perform well even when 90%+ entries in the dependency matrices are missing.

5.2.2. Effectiveness of Fascinate-ZERO. To evaluate the effectiveness of Fascinate-ZERO, we randomly select one node from the *Chemical* layer in the BIO dataset as the newly arrived node and compare the inference performance between Fascinate-ZERO and Fascinate. The average results over multiple runs are presented in Figure 5. We can see that Fascinate-ZERO bears a very similar inference power as Fascinate, but it is

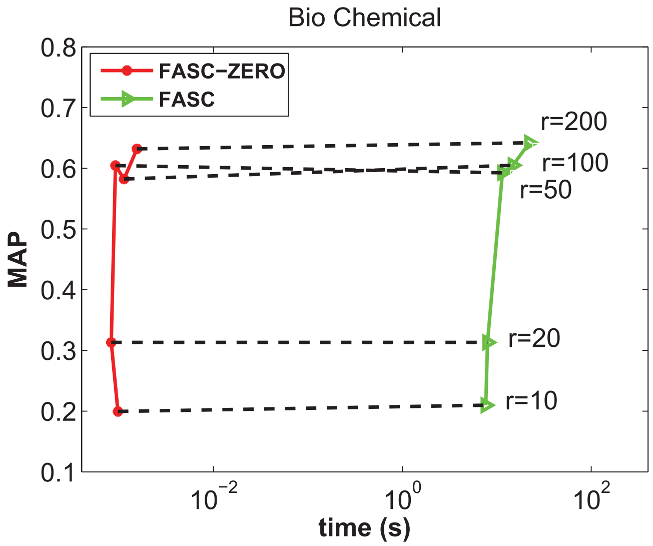
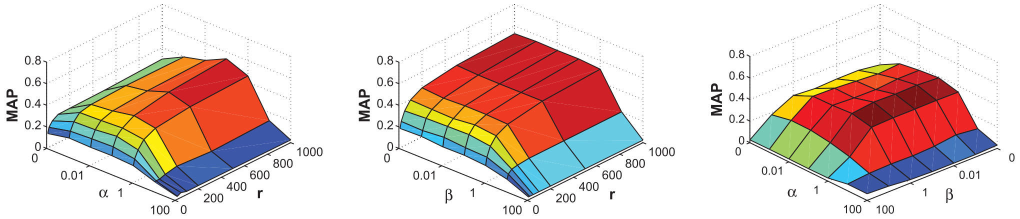


Fig. 5. Effectiveness of FASCINATE-ZERO in BIO network w.r.t. different rank r.



(a) Impact of α and r. (fixing $\beta = 0.1$)

(b) Impact of β and r. (fixing $\alpha = 0.1$)

(c) Impact of α and β . (fixing $r = 100$)

Fig. 6. The parameter studies of the BIO dataset.

orders of magnitude faster. We observe similar performance when the *zero-start* nodes are selected from the other two layers (i.e., *Gene* and *Disease*).

5.2.3. Parameter Studies. There are three parameters α, β, and r in the proposed FASCINATE algorithm. α is used to control the impact of node homophily, β is used to avoid over-fitting, and r is the number of columns of the low-rank matrices $\{\mathbf{F}_i\}$. We fix one of these parameters, and study the impact of the remaining two on the inference results. From Figure 6, we can see that *MAP* is stable over a wide range of both α and β. As for the third parameter r, the inference performance quickly increases w.r.t. r until it hits 200, after which the *MAP* is almost flat. This suggests that a relatively small size of the low-rank matrices might be sufficient to achieve a satisfactory inference performance.

For FASCINATE-UN, we study the impact of the backtracking line search parameters on its performance. By fixing α, β, and rank r to 0.1, 0.1, and 100, respectively, we examine a wide range of a and b within their domains as shown in Figure 7. We can see that the inference performance is sensitive to the combination of a and b because subtle parameter changes may affect the convergence speed in Algoritm 2 greatly, which would have impact on the inference performance within limited iterations consequently.

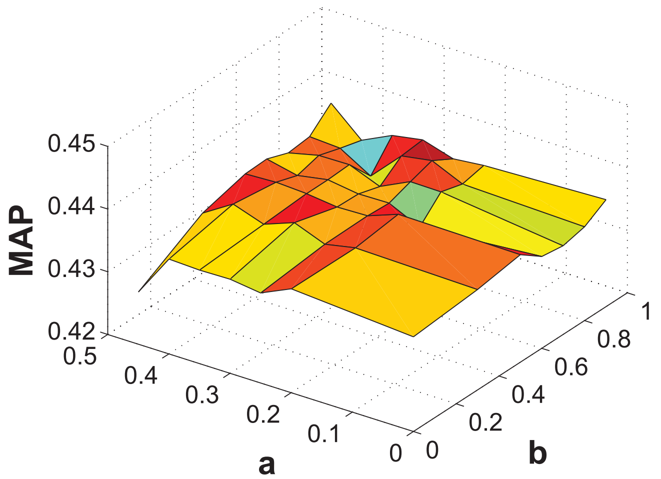
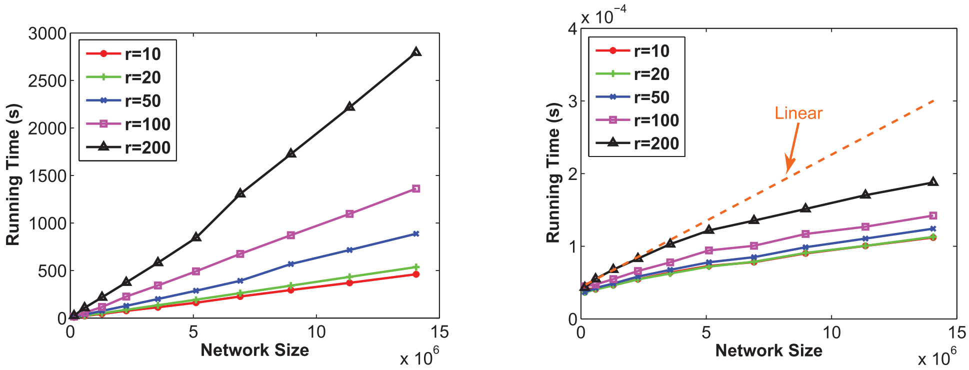


Fig. 7. The backtracking line search parameter study of the INFRA-5 dataset.



(a) Fascinate (b)Fascinate-ZERO

Fig. 8. Wall-clock time vs. the size of the network.

5.3. Efficiency

The scalability results of Fascinate and Fascinate-ZERO are presented in Figure 8. As we can see in Figure 8(a), Fascinate scales linearly w.r.t. the overall network size (i.e., $\sum_i (n_i + m_i) + \sum_{i,j} m_{i,j}$), which is consistent with our previous analysis in Lemma 3.3. As for Fascinate-ZERO, it scales *sub-linearly* w.r.t. the entire network size. This is because, by Lemma 4.1, the running time of Fascinate-ZERO is only dependent on the neighborhood size of the newly added node, rather than that of the entire network. Finally, we can see that Fascinate-ZERO is much more efficient than Fascinate. To be specific, on the entire INFRA-3 dataset, Fascinate-ZERO is 10,000,000+ faster than Fascinate (i.e., 1.878×10^{-4} seconds vs. 2.794×10^{3} seconds).

In addition, we compare the running time of Fascinate and Fascinate-UN on CITATION and INFRA-5 networks. The results are as shown in Figure 9. As we can see, Fascinate-UN is orders of magnitude slower than Fascinate to achieve similar inference results, which is consistent to our complexity analysis in Section 3.4.

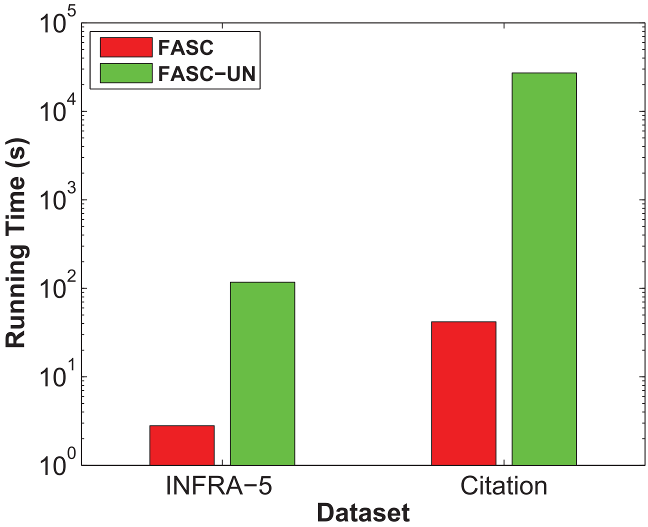


Fig. 9. Wall-clock running time of FASCINATE and FASCINATE-UN.

6. RELATED WORK

In this section, we review the related literature, which can be classified into following two categories: (1) multi-layered network, and (2) collaborative filtering.

Multi-Layered Network. Multi-layered networks (also referred as network of networks in some scenarios) have attracted a lot research attentions in recent years. In Kivelä et al. [2014], the authors provide a comprehensive survey about different types of multi-layered networks, including multi-modal networks [Heath and Sioson 2009], multi-dimensional networks [Berlingerio et al. 2011], multiplex networks [Battiston et al. 2014], and inter-dependent networks [Buldyrev et al. 2010]. The network studied in our article belongs to the category of inter-dependent networks. One of the mostly studied problems in inter-dependent networks is network robustness [Gao et al. 2011]. Most of the previous researches are based on *two-layered* networks [Buldyrev et al. 2010; Gao et al. 2012; Parshani et al. 2010; Shao et al. 2011], with a few exceptions that focus on arbitrarily structured multi-layered networks [Chen et al. 2015]. Notice that all these existing works assume that the network structure (including both the within-layer connectivity and the cross-layer dependency) is *given* a prior, which is not the case in real-world applications due to noise, limited accessibility, and so forth. In Chen et al. [2016], a collaborative filtering-based method is proposed to infer the missing cross-layer dependencies in multi-layered network. Other remotely related studies include cross-network ranking [Ni et al. 2014] and clustering [Ni et al. 2015; Liu et al. 2015a] in the context of multi-layered networks and [Xu et al. 2013; Zhou et al. 2015; Zhang et al. 2016] in multi-view data analysis.

Collaborative Filtering. As mentioned earlier, the cross-layer dependency inference problem is conceptually related to collaborative filtering [Goldberg et al. 1992]. Commonly used collaborative filtering methods can be roughly classified into two basic models: neighborhood models [Breese et al. 1998] and latent factor models [Koren et al. 2009]. As the latent factor model is more effective in capturing the implicit dependencies between users and items, many variants have been proposed to address implicit feedback problems [Hu et al. 2008; Ma 2013], OCCF problems [Pan et al. 2008], feature selection problems [Li et al. 2016a, 2016b], and even crowdsourcing problems in high dimensional settings [Zhou and He 2016]. Instead of only using the user-item rating matrix for preference inference, Li et al. [2010] propose a method that can effectively incorporate user information into OCCF to improve the performance. To further exploit more data resources for preference inference, Yao et al. [2014] propose *wiZAN-Dual* to take both user similarity network and item similarity network as side information for OCCF. In Zheng et al. [2013], multiple similarity networks of users and items are integrated together for drug-target interaction prediction. In Li et al. [2009] and Yang et al. [2015], user and item features are incorporated into the traditional collaborative filtering algorithms for cross-domain recommendation. To deal with domains with multiple dependencies, Singh and Gordon [2008] propose a collective matrix factorization model to learn the dependencies across any two inter-dependent domains. Some less studied scenarios in collaborative filtering include handling cold-start problems [Xu et al. 2015] and user/item dynamics [Koren 2009; He et al. 2016] (e.g., the arrival a new user or item and a new rating between an user and an item).

7. CONCLUSIONS

In this article, we formally define the cross-layer dependency inference problem (CODE) and its *zero-start* problem (CODE-ZERO) in the context of multi-layered networks. To address these problems, we propose to formulate the inference problem as a collective collaborative filtering problem and introduce FASCINATE, an algorithm that can effectively infer the missing dependencies with provable optimality and scalability. In particular, by modeling the impact of *zero-start* node as a perturbation in the multi-layered network, we derive FASCINATE-ZERO, an online variant of FASCINATE that can approximate the dependencies of the newly added node with *sub-linear* complexity w.r.t. the overall system size. The experimental results on five real-world datasets demonstrate the superiority of our proposed algorithm both by its effectiveness and efficiency.

REFERENCES

Federico Battiston, Vincenzo Nicosia, and Vito Latora. 2014. Structural measures for multiplex networks. *Physical Review E* 89, 3 (2014), 032804.

Michele Berlingerio, Michele Coscia, Fosca Giannotti, Anna Monreale, and Dino Pedreschi. 2011. Foundations of multidimensional network analysis. In *Proceedings of the 2011 International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE, 485–489.

John S. Breese, David Heckerman, and Carl Kadie. 1998. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann Publishers Inc., 43–52.

Sergey V. Buldyrev, Roni Parshani, Gerald Paul, H. Eugene Stanley, and Shlomo Havlin. 2010. Catastrophic cascade of failures in interdependent networks. *Nature* 464, 7291 (2010), 1025–1028.

Chen Chen, Jingrui He, Nadya Bliss, and Hanghang Tong. 2015. On the connectivity of multi-layered networks: Models, measures and optimal control. In *Proceedings of the 2015 IEEE 15th International Conference on Data Mining (ICDM)*. IEEE, 715–720.

Chen Chen, Hanghang Tong, Lei Xie, Lei Ying, and Qing He. 2016. FASCINATE: Fast cross-layer dependency inference on multi-layered networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 765–774. DOI:http://dx.doi.org/10.1145/2939672.2939784

Wei Chen, Wynne Hsu, and Mong Li Lee. 2013. Making recommendations from multiple domains. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 892–900.

Allan Peter Davis, Cynthia J. Grondin, Kelley Lennon-Hopkins, Cynthia Saraceni-Richards, Daniela Sciaky, Benjamin L. King, Thomas C. Wiegers, and Carolyn J. Mattingly. 2015. The comparative toxicogenomics database's 10th year anniversary: Update 2015. *Nucleic Acids Research* 43, D1 (2015), D914–D920.

Chris Ding, Tao Li, Wei Peng, and Haesun Park. 2006. Orthogonal nonnegative matrix t-factorizations for clustering. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 126–135.

Jianxi Gao, Sergey V. Buldyrev, Shlomo Havlin, and H. Eugene Stanley. 2011. Robustness of a network of networks. *Physical Review Letters* 107, 19 (2011), 195701.

Jianxi Gao, Sergey V. Buldyrev, H. Eugene Stanley, and Shlomo Havlin. 2012. Networks formed from interdependent networks. *Nature Physics* 8, 1 (2012), 40–48.

David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. 1992. Using collaborative filtering to weave an information tapestry. *Communications of the ACM* 35, 12 (1992), 61–70.

Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 2016. Fast matrix factorization for online recommendation with implicit feedback. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'16)*. 549–558. DOI:http://dx.doi.org/10.1145/2911451.2911489

Lenwood S. Heath and Allan A. Sioson. 2009. Multimodal networks: Structure and operations. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 6, 2 (2009), 321–332.

Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM'08)*. IEEE, 263–272.

Mikko Kivelä, Alex Arenas, Marc Barthelemy, James P. Gleeson, Yamir Moreno, and Mason A. Porter. 2014. Multilayer networks. *Journal of Complex Networks* 2, 3 (2014), 203–271.

Yehuda Koren. 2009. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 447–456.

Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 8 (2009), 30–37.

Daniel D. Lee and H. Sebastian Seung. 2001. Algorithms for non-negative matrix factorization. In *Proceedings of the Advances in Neural Information Processing Systems*. 556–562.

Bin Li, Qiang Yang, and Xiangyang Xue. 2009. Can movies and books collaborate? Cross-domain collaborative filtering for sparsity reduction. In *Proceedings of the 21st International Jont Conference on Artifical Intelligence*, Vol. 9. 2052–2057.

Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Trevino Robert, Jiliang Tang, and Huan Liu. 2016a. Feature selection: A data perspective. (2016). arXiv:1601.07996.

Jundong Li, Xia Hu, Liang Wu, and Huan Liu. 2016b. Robust unsupervised feature selection on networked data. In *Proceedings of the 2016 SIAM International Conference on Data Mining*. SIAM, 387–395.

Liangyue Li, Hanghang Tong, Nan Cao, Kate Ehrlich, Yu-Ru Lin, and Norbou Buchler. 2015. Replacing the irreplaceable: Fast algorithms for team member recommendation. In *Proceedings of the 24th International Conference on World Wide Web*. ACM, 636–646.

Yanen Li, Jia Hu, ChengXiang Zhai, and Ye Chen. 2010. Improving one-class collaborative filtering by incorporating rich user information. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*. ACM, 959–968.

Chuan-bi Lin. 2007. Projected gradient methods for nonnegative matrix factorization. *Neural Computation* 19, 10 (2007), 2756–2779.

Jialu Liu, Chi Wang, Jing Gao, Quanquan Gu, Charu C. Aggarwal, Lance M. Kaplan, and Jiawei Han. 2015b. GIN: A clustering model for capturing dual heterogeneity in networked data. In *Proceedings of the 2015 SIAM International Conference on Data Mining*. SIAM, 388–396.

Rui Liu, Wei Cheng, Hanghang Tong, Wei Wang, and Xiang Zhang. 2015a. Robust multi-network clustering via joint cross-domain cluster alignment. In *Proceedings of the 2015 IEEE International Conference on Data Mining (ICDM'15)*. IEEE Computer Society, Washington, DC, 291–300. DOI:http://dx.doi.org/10.1109/ICDM.2015.13

Zhongqi Lu, Weike Pan, Evan Wei Xiang, Qiang Yang, Lili Zhao, and ErHeng Zhong. 2013. Selective transfer learning for cross domain recommendation. In *Proceedings of the SIAM International Conference on Data Mining*. SIAM, 641–649.

Hao Ma. 2013. An experimental study on implicit social recommendation. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 73–82.

Jingchao Ni, Hanghang Tong, Wei Fan, and Xiang Zhang. 2014. Inside the atoms: Ranking on a network of networks. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1356–1365.

Jingchao Ni, Hanghang Tong, Wei Fan, and Xiang Zhang. 2015. Flexible and robust multi-network clustering. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 835–844.

Rong Pan, Yunhong Zhou, Bin Cao, Nathan N. Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. 2008. One-class collaborative filtering. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM'08)*.IEEE, 502–511.

Roni Parshani, Sergey V. Buldyrev, and Shlomo Havlin. 2010. Interdependent networks: Reducing the coupling strength leads to a change from a first to second order percolation transition. *Physical Review Letters* 105, 4 (2010), 048701.

Sabry Razick, George Magklaras, and Ian M. Donaldson. 2008. iRefIndex: A consolidated protein interaction database with provenance. *BMC Bioinformatics* 9, 1 (2008), 1.

Vittorio Rosato, L. Issacharoff, F. Tiriticco, Sandro Meloni, S. Porcellinis, and Roberto Setola. 2008. Modelling interdependent infrastructures using interacting dynamical models. *International Journal of Critical Infrastructures* 4, 1–2 (2008), 63–79.

Arunabha Sen, Anisha Mazumder, Joydeep Banerjee, Arun Das, and Randy Compton. 2014. Identification of k most vulnerable nodes in multi-layered network using a new model of interdependency. In *Proceedings of the 2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS'14)*. IEEE, 831–836.

Jia Shao, Sergey V. Buldyrev, Shlomo Havlin, and H. Eugene Stanley. 2011. Cascade of failures in coupled network systems with multiple support-dependence relations. *Physical Review E* 83, 3 (2011), 036116.

Ajit P. Singh and Geoffrey J. Gordon. 2008. Relational learning via collective matrix factorization. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 650–658.

Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. Arnetminer: Extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 990–998.

Marc A. Van Driel, Jorn Bruggeman, Gert Vriend, Han G. Brunner, and Jack A. M. Leunissen. 2006. A text-mining analysis of the human phenome. *European Journal of Human Genetics* 14, 5 (2006), 535–542.

Duncan J. Watts and Steven H. Strogatz. 1998. Collective dynamics of small-world networks. *Nature* 393, 6684 (1998), 440–442.

Chang Xu, Dacheng Tao, and Chao Xu. 2013. A survey on multi-view learning. (2013). arXiv preprint arXiv:1304.5634 .

Jingwei Xu, Yuan Yao, Hanghang Tong, XianPing Tao, and Jian Lu. 2015. Ice-breaking: Mitigating cold-start recommendation problem by rating comparison. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI'15)*. 3981–3987. http://ijcai.org/Abstract/15/559

Deqing Yang, Jingrui He, Huazheng Qin, Yanghua Xiao, and Wei Wang. 2015. A graph-based recommendation across heterogeneous domains. In *Proceedings of the 24rd ACM International Conference on Conference on Information and Knowledge Management*. ACM, 463–472.

Yuan Yao, Hanghang Tong, Guo Yan, Feng Xu, Xiang Zhang, Boleslaw K. Szymanski, and Jian Lu. 2014. Dual-regularized one-class collaborative filtering. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. ACM, 759–768.

Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 353–362. DOI:http://dx.doi.org/10.1145/2939672.2939673

Xiaodong Zheng, Hao Ding, Hiroshi Mamitsuka, and Shanfeng Zhu. 2013. Collaborative matrix factorization with multiple similarities for predicting drug-target interactions. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM.

Dawei Zhou, Jingrui He, K. Seluk Candan, and Hasan Davulcu. 2015. MUVIR: Multi-view rare category detection. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI'15)*. 4098–4104.

Yao Zhou and Jingrui He. 2016. Crowdsourcing via tensor augmentation and completion. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI'16)*. 2435–2441. http://www.ijcai.org/Abstract/16/347

Received November 2016; revised February 2017; accepted February 2017