FASCINATE: Fast Cross-Layer Dependency Inference on Multi-layered Networks

Chen Chen[†], Hanghang Tong[†], Lei Xie[‡], Lei Ying[†], and Qing He[§]
[†]Arizona State University, {chen_chen, hanghang.tong, lei.ying.2}@asu.edu
[‡]City University of New York, lei.xie@hunter.cuny.edu
[§]University at Buffalo, qinghe@buffalo.edu

ABSTRACT

Multi-layered networks have recently emerged as a new network model, which naturally finds itself in many high-impact application domains, ranging from critical inter-dependent infrastructure networks, biological systems, organization-level collaborations, to cross-platform e-commerce, etc. Cross-layer dependency, which describes the dependencies or the associations between nodes across different layers/networks, often plays a central role in many data mining tasks on such multi-layered networks. Yet, it remains a daunting task to accurately know the cross-layer dependency a prior. In this paper, we address the problem of inferring the missing cross-layer dependencies on multi-layered networks. The key idea behind our method is to view it as a collective collaborative filtering problem. By formulating the problem into a regularized optimization model, we propose an effective algorithm to find the local optima with linear complexity. Furthermore, we derive an online algorithm to accommodate newly arrived nodes, whose complexity is just linear wrt the size of the neighborhood of the new node. We perform extensive empirical evaluations to demonstrate the effectiveness and the efficiency of the proposed methods.

1. INTRODUCTION

In an increasingly connected world, networks from many high-impact areas are often collected from multiple *inter-dependent* domains, leading to the emergence of *multi-layered networks* [4, 9, 26, 29, 30]. A typical example of multi-layered networks is inter-dependent critical infrastructure network. As illustrated in Figure 1, the full functioning of the telecom network, the transportation network and the gas pipeline 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 supply from the transportation network and the gas pipeline network. Moreover, to keep the whole complex system working in order, extensive communications are needed across the networks, which are in turn supported by the telecom network. Another example is *biological system*, where the protein-protein interaction network (PPI/gene network) is naturally linked to the disease similarity network by the known disease-gene associations,

KDD '16, August 13-17, 2016, San Francisco, CA, USA

DOI: http://dx.doi.org/10.1145/2939672.2939784

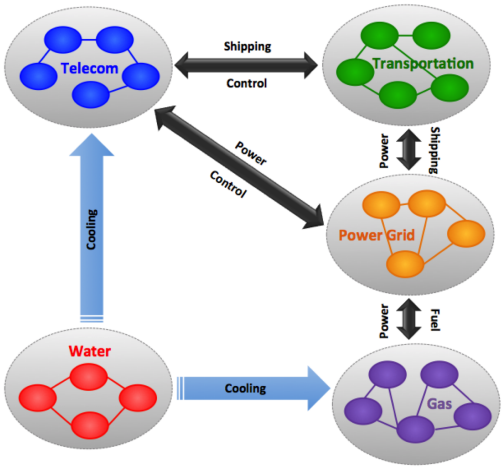


Figure 1: An illustrative example of multi-layered networks. Each gray ellipse is a critical infrastructure network (e.g., Telecom network, power grid, transportation network, etc). A thick arrow between two ellipses indicates a cross-layer dependency between the corresponding two networks (e.g., a router in the telecom network depends on one or more power plants in the power grid).

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 [5] and cross-platform e-commerce [6, 16, 21, 36].

Compared with single-layered networks, a unique topological characteristic of multi-layered networks lies in its *cross-layer dependency* structure. For example, in the critical infrastructure network, the full functioning of the telecom layer *depends* on the sufficient power supply from the power grid layer, which in turn relies on the functioning of the transportation layer (e.g., to deliver the sufficient fuel). While 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 existence of the cross-layer dependency is in general considered as a major factor of the vulnerability of the entire system. This is because a small disturbance on one supporting layer/network (e.g., power grid) might cause a ripple effect to all the dependent layers, leading to 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 (e.g., drug re-purposing).

Despite its key importance, it remains a daunting task to accurately know the cross-layer dependency in a multi-layered network,

due to a number of reasons, ranging from noise, incomplete data sources, 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 damage 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 wrt proteins and/or the diseases might be completely unknown due to clinical limitations.(i.e., the *zero-start* problem).

In this paper, we aim to tackle the above challenges and develop effective and efficient methods to infer cross-layer dependency on multi-layered networks. The main contributions of the paper can be summarized as

- *Problem Formulations*. We formally formulate the cross-layer dependency inference problem as a regularized optimization problem. The key idea of our formulation is to collectively leverage the within-layer topology as well as the observed cross-layer dependency to infer a latent, low-rank representation for each layer, based on which the missing cross-layer dependencies can be inferred.
- *Algorithms and Analysis*. We propose an effective algorithm (FASCINATE) for cross-layer dependency inference 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.
- *Evaluations*. We perform extensive experiments on real datasets to validate the effectiveness, efficiency and scalability of the proposed algorithms. Specially, 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. Specifically, the proposed FASCINATE-ZERO algorithm can achieve up to $10^7\times$ speedup with barely no compromise on accuracy.

The rest of the paper 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 paper.

2. PROBLEM DEFINITION

In this section, we give the formal definitions of the cross-layer dependency inference problems. The main symbols used throughout the paper are listed in Table 1. 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 $\hat{}$ sign to denote the notations after a new node is accommodated to the system (e.g., $\hat{J}$, $\hat{\mathbf{A}}_1$), and the ones without the $\hat{}$ 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 [5], due to its flexibility to model more complicated cross-layer dependency structure. We refer the readers to [5] for its full details. For the purpose of this paper, 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

Table 1: 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 i^{th} row j^{th} column in matrix $\mathbf{A}$
$\mathbf{A}(i,:)$	the i^{th} row of matrix $\mathbf{A}$
$\mathbf{A}(:,j)$	the j^{th} 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 $\mathcal{D} = \{\mathbf{D}_{i,j}\ i,j = 1, ..., g\}$
$\mathbf{W}_{i,j}$	weight matrix for $\mathbf{D}_{i,j}$
$\mathbf{F}_i$	low-rank representation for layer-*i* $(i = 1, ..., g)$
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

$\mathbf{G}(i,j) = 1$ if layer-*j* depends on layer-*i*, and $\mathbf{G}(i,j) = 0$ otherwise. Second, we need a set of g within-layer connectivity matrices: $\mathcal{A} = \{\mathbf{A}_1, ..., \mathbf{A}_g\}$ to describe the connectivities/similarities between nodes within the same layer. Third, we need a set of cross-layer dependency matrices $\mathcal{D} = \{\mathbf{D}_{i,j}\ i,j = 1, ..., g\}$, 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 multi-layered 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 *protein-protein interaction (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 ($\mathbf{D}_{3,4}$).

As mentioned earlier, it is often very hard to accurately know the cross-layer dependency matrices $\{\mathbf{D}_{i,j}\ i,j = 1, ..., g\}$. 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 $\mathcal{A} = \{\mathbf{A}_1, ..., \mathbf{A}_g\}$; and (3) observed cross-layer dependency matrices $\mathcal{D} = \{\mathbf{D}_{i,j}\ i,j = 1, ..., g\}$;*

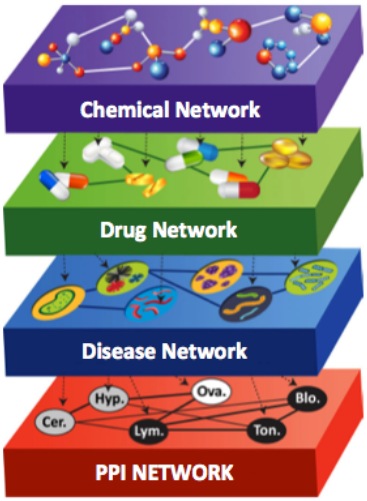


Figure 2: A simplified 4-layered network for biological systems.

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

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.,* $\tilde{\mathbf{D}}_{l,j}(p,:)$ $(j = 1, ..., g,\ \mathbf{G}(l,j) = 1)$.

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, items) across different domains. Having this analogy in mind, we propose to formulate Problem 1 as the following regularized optimization problem

$$\min_{\mathbf{F}_i \geq \mathbf{0}(i=1,...,g)} J = \underbrace{\sum_{i,j:\ \mathbf{G}(i,j)=1} \|\mathbf{W}_{i,j} \odot (\mathbf{D}_{i,j} - \mathbf{F}_i \mathbf{F}_j{}')\|_F^2}_{\text{C1: Matching Observed Cross-Layer Dependencies}} \quad (1)$$

$$+ \underbrace{\alpha \sum_{i=1}^{g} \text{tr}(\mathbf{F}_i{}'(\mathbf{T}_i - \mathbf{A}_i)\mathbf{F}_i)}_{\text{C2: Node Homophily}} + \underbrace{\beta \sum_{i=1}^{g} \|\mathbf{F}_i\|_F^2}_{\text{C3: Regularization}}$$

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 (Eq. (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 $\{\mathbf{F}_i\}_{i=1,...,g}$ to prevent over-fitting.

Once we solve Eq. (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 Eq. (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, Eq. (1) can be simplified as

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

The derivative of J_i wrt $\mathbf{F}_i$ is

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

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

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

where

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

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

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 Eq. (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 Alg. 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 $\{\mathbf{F}_i\}_{i=1,...,g}$ one by one. We use two criteria to terminate the iteration: (1) either the Frobenius norm between two successive iterations for all $\{\mathbf{F}_i\}_{i=1,...,g}$ 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 $\mathcal{A} = \{\mathbf{A}_1, ..., \mathbf{A}_g\}$, and (c) observed cross-layer node dependency matrices $\mathcal{D} = \{\mathbf{D}_{i,j}\ i, j = 1, ..., g\}$; (2) the rank size r; (3) weight w; (4) regularized parameters α and β;
Output: low-rank representations for each layer $\{\mathbf{F}_i\}_{i=1,...,g}$

```
1: for i = 1 to g do
2:     initialized F_i as n_i × r non-negative random matrix
3: end for
4: while not converge do
5:     for i = 1 to g do
6:         compute X as Eq. (6)
7:         compute Y as Eq. (7)
8:         update F_i as Eq. (4)
9:     end for
10: end while
11: return {F_i}_{i=1,...,g}
```

3.3 Proof and Analysis

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

A - Effectiveness Analysis.

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

THEOREM 1. *The fixed point solution of Eq.* (4) *satisfies the KKT condition.*

PROOF. The Lagrangian function of Eq. (2) can be written as

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

where $\mathbf{\Lambda}$ is the Lagrange multiplier. Setting the derivative of L_i wrt $\mathbf{F}_i$ to 0, we get

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

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 Eq. (4) satisfies the above equation. □

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

LEMMA 1. *Under the updating rule in Eq.* (4)*, the objective function in Eq.* (2) *decreases monotonically.*

PROOF. Omitted for brevity. □

According to Theorem 1 and Lemma 1, we conclude that Alg. 1 converges to a local minimum solution for Eq. 2 wrt each individual $\mathbf{F}_i$.

B - Efficiency Analysis.

In terms of efficiency, we analyze both the time complexity as well as the space complexity of the proposed FASCINATE algorithm, which are summarized in Lemma 2 and Lemma 3. We can see that FASCINATE scales linearly wrt the size of the entire multi-layered network.

LEMMA 2. *The time complexity of Alg. 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 Alg. 1 for updating $\mathbf{F}_i$, the complexity of calculating $\mathbf{X}$ by Eq. (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 Alg. 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. *The space complexity of Alg. 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 Alg. 1 is $O(\sum_{i=1}^{g}(n_i r+m_i)+\sum_{i,j:\ \mathbf{G}(i,j)=1} m_{i,j})$. □

C - Variants.

Here, we discuss some variants of the proposed FASCINATE algorithm. First, by setting all the entries in the weight matrix $\mathbf{W}_{i,j}$ to 1, FASCINATE becomes a clustering algorithm (i.e., $\mathbf{F}_i$ can be viewed as the cluster membership matrix for nodes in layer-*i*). Furthermore, if we restrict ourselves to two-layered networks (i.e., $g=2$), FASCINATE becomes a dual regularized co-clustering algorithm [20]. Second, by setting $w\in(0,1)$, FASCINATE can be used to address one class collaborative filtering problem, where implicit dependencies extensively exist between nodes from different layers. Specifically, on two-layered networks, FASCINATE is reduced to a weighting-based, dual-regularized one class collaborative filtering algorithm [37]. Third, 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 [32].

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, 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 wrt 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 $(n_1+1)^{\text{th}}$ 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 $\mathbf{s}(u)$ $(u=1,...,n_1)$ denotes the (within-layer) connectivity between the newly added node and the u^{th} 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 wrt 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

$$\hat{\mathbf{A}}_1=\begin{bmatrix}\mathbf{A}_1 & \mathbf{s}'\\ \mathbf{s} & 0\end{bmatrix} \tag{11}$$

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 Eq. (1) can be reformatted as

$$\begin{aligned}\hat{J}=&\sum_{\substack{i,j:\ \mathbf{G}(i,j)=1\\ i,j\neq 1}}\|\mathbf{W}_{i,j}\odot(\mathbf{D}_{i,j}-\hat{\mathbf{F}}_i\hat{\mathbf{F}}_j')\|_F^2\\ &+\sum_{j:\ \mathbf{G}(1,j)=1}\|\hat{\mathbf{W}}_{1,j}\odot(\hat{\mathbf{D}}_{1,j}-\hat{\mathbf{F}}_1\hat{\mathbf{F}}_j')\|_F^2\\ &+\sum_{i=2}^{g}\frac{\alpha}{2}\sum_{u=1}^{n_i}\sum_{v=1}^{n_i}\mathbf{A}_i(u,v)\|\hat{\mathbf{F}}_i(u,:)-\hat{\mathbf{F}}_i(v,:)\|_2^2\\ &+\frac{\alpha}{2}\sum_{u=1}^{n_1}\sum_{v=1}^{n_1}\mathbf{A}_1(u,v)\|\hat{\mathbf{F}}_1(u,:)-\hat{\mathbf{F}}_1(v,:)\|_2^2\\ &+\beta\sum_{i=2}^{g}\|\hat{\mathbf{F}}_i\|_F^2+\beta\|\hat{\mathbf{F}}'_{1(n_1\times r)}\|_F^2\\ &+\alpha\sum_{v=1}^{n_1}\mathbf{s}(v)\|\mathbf{f}-\hat{\mathbf{F}}_1(v,:)\|_2^2+\beta\|\mathbf{f}\|_2^2\end{aligned} \tag{12}$$

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

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

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

$$\sum_{j:\ \mathbf{G}(1,j)=1}\|\mathbf{W}_{1,j}\odot(\mathbf{D}_{1,j}-\hat{\mathbf{F}}_{1(n_1\times r)}\hat{\mathbf{F}}_j')\|_F^2 \tag{13}$$

Combining Eq. (12), Eq. (13) and J in Eq. (1) together, $\hat{J}$ can be expressed as

$$\hat{J}=J+J^1 \tag{14}$$

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 Eq. (14), 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 Eq. (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

$$\mathbf{f}=\arg\min_{\mathbf{f}\geq\mathbf{0}} J^1\quad\text{subject to: }\hat{\mathbf{F}}_{1(n_1\times r)}=\mathbf{F}_1^* \tag{15}$$

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 wrt $\mathbf{f}$ and get

$$\frac{1}{2}\frac{\partial J^1}{\partial \mathbf{f}} = \beta\mathbf{f} + \alpha\sum_{v=1}^{n_1}\mathbf{s}(v)(\mathbf{f} - \mathbf{F}_1^*(v,:)) \tag{16}$$

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

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

$$\mathbf{f} = \frac{\alpha\mathbf{s}\mathbf{F}_1^*}{\beta + \alpha\sum_{v=1}^{n_1}\mathbf{s}(v)} \tag{17}$$

For the newly added node, $\mathbf{f}$ can be viewed as the weighted average of its neighbors' low-rank representations. Notice that in Eq. (17), the non-negativity constraint on $\mathbf{f}$ naturally holds. Therefore, we refer to this solution (i.e., Eq. (17)) 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, which is linear wrt the number of neighbors of the new node (and therefore is *sub-linear* wrt the size of the entire network).

LEMMA 4. *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 $O(\text{nnz}(\mathbf{s})r)$. □

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?

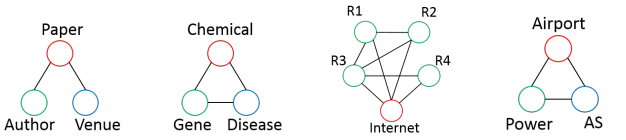
5.1 Experimental Setup

5.1.1 Datasets Description

We perform our evaluations on four different datasets, including (1) a three-layer Aminer academic network in the social collaboration domain (SOCIAL); (2) a three-layer CTD (Comparative Toxicogenomics Database) network in the biological domain (BIO); (3) a five-layer Italy network in the critical infrastructure domain (INFRA-5); and (4) a three-layer network in the critical infrastructure domain (INFRA-3). The statistics of these datasets are shown in Table 2, 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., $\mathbf{D}_{i,j}(u,v) = \mathbf{D}_{j,i}(v,u)$).

Table 2: Statistics of Datasets.

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



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

Figure 3: The abstract dependency structure of each dataset.

SOCIAL. This dataset contains three layers, including a collaboration network among authors, a citation network between papers and a venue network [33]. 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(a). 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 [7, 27, 34]. 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(b). 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.

INFRA-5. The construction of this critical infrastructure network is based on the data implicated from an electrical blackout in Italy in Sept 2003 [28]. It contains five layers, including four layers of regional power grids and one Internet network [28]. The regional power grids are partitioned by macroregions[1]. 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(c). 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[2], an autonomous system network[3] and a power grid [35]. We construct a three-layered network in the same way as [5]. The three infrastructure networks are functionally dependent on each other. Therefore, they form a triangle-shaped multi-layered network as shown in Figure 3(d). The construction of the cross-layer dependencies is based on geographic proximity. The number of nodes in each layer varies from

[1]https://en.wikipedia.org/wiki/First-level_NUTS_of_the_European_Union

[2]http://www.levmuchnik.net/Content/Networks/NetworkData.html

[3]http://snap.stanford.edu/data/

$2,833$ to $7,325$, and the number of within-layer links ranges from $6,594$ to $15,665$. The number of cross-layer dependencies ranges from $4,434,116$ to $14,921,765$.

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) *MulCol* - a collective matrix factorization method [32], (3) *PairSid* - a pairwise one-class collaborative filtering method proposed in [37], (4) *PairCol* - a pairwise collective matrix factorization method degenerated from *MulCol* (5) *PairNMF* - a pairwise non-negative matrix factorization (*NMF*) based method [19], (6) *PairRec* - a pairwise matrix factorization based algorithm introduced in [15], (7) *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 (8) *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 paper, we set rank $r = 100$, maximum iteration $t = 100$, termination threshold $\xi = 10^{-8}$, weight $w^2 = 0.1$ and regularization parameters $\alpha = 0.1$, $\beta = 0.1$ 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 [18]. A larger *MAP* indicates better inference performance.
- **R-MPR**. It is a variant of Mean Percentage Ranking for one-class collaborative filtering [12]. *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 one-class collaborative filtering. 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 [25]. 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 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 2 processors Intel Xeon 3.5GHz with 256GB of RAM. The algorithms are programmed with MATLAB using single thread. We will release the code and the non-proprietary datasets after the paper is published.

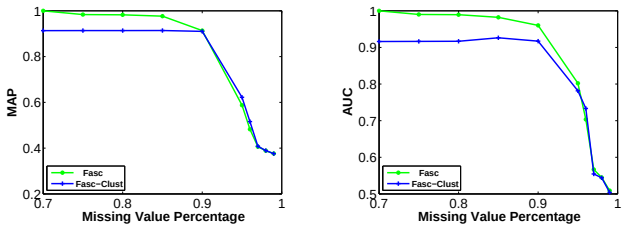
5.2 Effectiveness

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 wrt the model parameters?

5.2.1 Effectiveness of FASCINATE

We compare the proposed algorithms and the existing methods on all the four datasets. The results are shown in Table 3 through Table 6. There are several interesting observations. First is that our proposed FASCINATE algorithm and its variant (FASCINATE-CLUST) consistently outperform all other methods in terms of all the five evaluation metrics. Second, by exploiting the structure of multi-layered network, FASCINATE, FASCINATE-CLUST and *MulCol* have significantly better performance than the pairwise methods. 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.

We also test the sensitivity of the proposed algorithms wrt 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.



(a) MAP (b) AUC

Figure 4: Performance of FASCINATE and FASCINATE-CLUST on INFRA-3 dataset under different missing value percentages.

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 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 homophi-

Table 3: 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
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 4: Cross-Layer Dependency Inference on BIO.

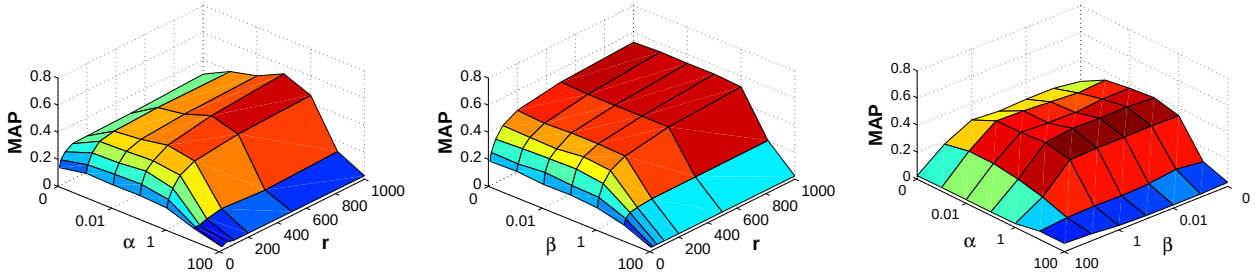
Methods	MAP	R-MPR	HLU	AUC	Prec@10
FASCINATE	**0.3979**	**0.4066**	**45.1001**	**0.9369**	**0.1039**
FASCINATE-CLUST	0.3189	0.3898	37.4089	0.9176	0.0857
MulCol	0.3676	0.3954	42.8687	0.9286	0.0986
PairSid	0.3623	0.3403	40.4048	0.8682	0.0941
PairCol	0.3493	0.3153	38.4364	0.8462	0.0889
PairNMF	0.1154	0.1963	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

Table 5: 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
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 6: 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
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



(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 $r = 100$).

Figure 6: The parameter studies of the BIO dataset.

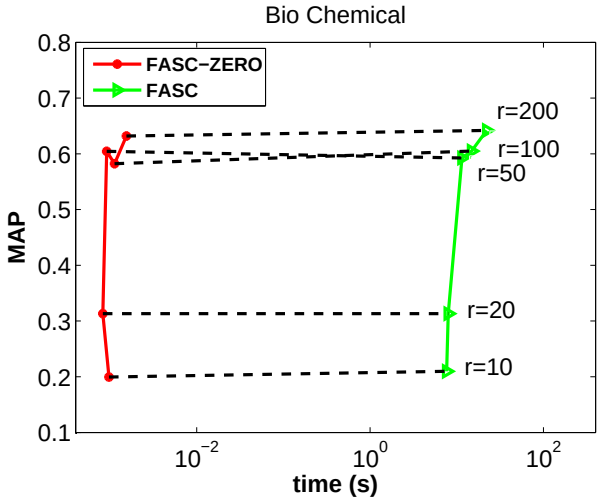


Figure 5: Effectiveness of FASCINATE-ZERO in BIO network wrt different rank r.

ly, 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. Specifically, a relatively high *MAP* can be achieved when α is between 0.1 to 1 and β is less than 1. As for the third parameter r, the inference performance quickly increases wrt 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.

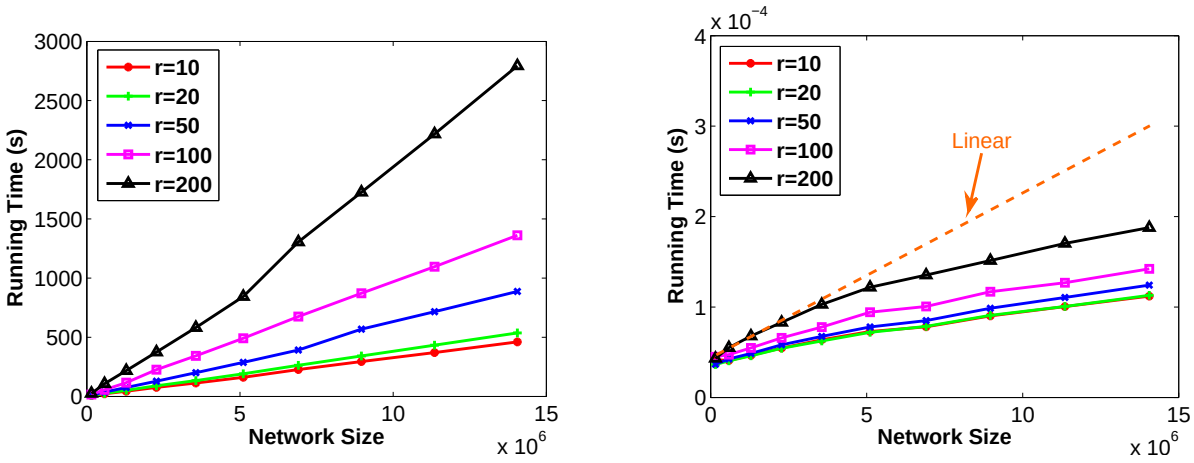
5.3 Efficiency

The scalability results of FASCINATE and FASCINATE-ZERO are presented in Figure 7. As we can see in Figure 7(a), FASCINATE scales linearly wrt 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 2. As for FASCINATE-ZERO, it scales *sub-linearly* wrt the entire network size. This is because, by Lemma 4, 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)

6. RELATED WORK

In this section, we review the related literature, which can be classified into 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



(a) FASCINATE (b)FASCINATE-ZERO

Figure 7: Wall-clock time vs. the size of the network.

research attentions in recent years. In [13], Kivela et al. provide a comprehensive survey about different types of multi-layered networks, including multi-modal networks [11], multi-dimensional networks [2], multiplex networks [1] and inter-dependent networks [4]. The network studied in our paper belongs to the category of inter-dependent networks. One of the mostly studied problems in inter-dependent networks is network robustness [8]. Most of the previous researches are based on *two-layered* networks [4, 9, 26, 31], with a few exceptions that focus on arbitrarily structured multi-layered networks [5]. Other remotely related studies in the context of multi-layered networks include cross-network ranking [23] and clustering [24]. 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. From this perspective, our proposed algorithms in this paper might benefit these works by providing a more accurate input network.

Collaborative Filtering. As mentioned earlier, the cross-layer dependency inference problem is conceptually related to collaborative filtering [10]. Commonly used collaborative filtering methods can be roughly classified into two basic models: neighborhood models [3] and latent factor models [15]. 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 [12, 22], one class collaborative filtering (OCCF) problems [25], feature selection problems [17], etc. Instead of only using the user-item rating matrix for preference inference, Li et al. propose a method that can effectively incorporate user information into OCCF to improve the performance [18]. To further exploit more data resources for preference inference, Yao et al. propose *wiZAN-Dual* to take both user similarity network and item similarity network as side information for OCCF [37]. In [38], multiple similarity networks of users and items are integrated together for drug-target interaction prediction. In [16, 36], user and item features are incorporated into the traditional collaborative filtering algorithms for cross-domain recommendation. To deal with domains with multiple dependencies, Singh et al. propose a collective matrix factorization model to learn the dependencies across any two inter-dependent domains [32]. A less studied scenario in collaborative filtering is to handle user/item dynamics [14] (e.g., the arrival a new user or item, a new rating between an user and an item, etc).

7. CONCLUSIONS

In this paper, we address the cross-layer dependency inference problem (CODE) and the corresponding *zero-start* problem (CODE-ZERO) in multi-layered networks. By formulating CODE as a collective collaborative filtering problem, we propose an effective algorithm (FASCINATE), and prove its optimality, correctness and scalability. Moreover, by viewing the *zero-start* node as a perturbation to the multi-layered network system, we derive an effective and efficient algorithm (FASCINATE-ZERO) to approximate the dependencies of newly added nodes, whose complexity is *sub-linear* wrt the overall network size. We experiment on four real-world datasets from three different domains, which demonstrates the effectiveness and efficiency of FASCINATE and FASCINATE-ZERO.

Acknowledgements

We thank V. Rosato for providing the Italy 2003 blackout data. This work is partially supported by the National Science Foundation under Grant No. IIS1017415, by DTRA under the grant number HDTRA1-16-0017, by Army Research Office under the contract number W911NF-16-1-0168, by 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.

8. REFERENCES

[1] F. Battiston, V. Nicosia, and V. Latora. Structural measures for multiplex networks. *Physical Review E*, 89(3):032804, 2014.

[2] M. Berlingerio, M. Coscia, F. Giannotti, A. Monreale, and D. Pedreschi. Foundations of multidimensional network analysis. In *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*, pages 485–489. IEEE, 2011.

[3] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pages 43–52. Morgan Kaufmann Publishers Inc., 1998.

[4] S. V. Buldyrev, R. Parshani, G. Paul, H. E. Stanley, and S. Havlin. Catastrophic cascade of failures in interdependent networks. *Nature*, 464(7291):1025–1028, 2010.

[5] C. Chen, J. He, N. Bliss, and H. Tong. On the connectivity of multi-layered networks: Models, measures and optimal control. In *Data Mining (ICDM), 2015 IEEE 15th International Conference on*, pages 715–720. IEEE, 2015.

[6] W. Chen, W. Hsu, and M. L. Lee. Making recommendations from multiple domains. In *Proceedings of the 19th ACM*

SIGKDD international conference on Knowledge discovery and data mining, pages 892–900. ACM, 2013.

[7] A. P. Davis, C. J. Grondin, K. Lennon-Hopkins, C. Saraceni-Richards, D. Sciaky, B. L. King, T. C. Wiegers, and C. J. Mattingly. The comparative toxicogenomics database's 10th year anniversary: update 2015. *Nucleic acids research*, page gku935, 2014.

[8] J. Gao, S. V. Buldyrev, S. Havlin, and H. E. Stanley. Robustness of a network of networks. *Physical Review Letters*, 107(19):195701, 2011.

[9] J. Gao, S. V. Buldyrev, H. E. Stanley, and S. Havlin. Networks formed from interdependent networks. *Nature physics*, 8(1):40–48, 2012.

[10] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.

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

[12] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 263–272. Ieee, 2008.

[13] M. Kivelä, A. Arenas, M. Barthelemy, J. P. Gleeson, Y. Moreno, and M. A. Porter. Multilayer networks. *Journal of Complex Networks*, 2(3):203–271, 2014.

[14] Y. Koren. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 447–456. ACM, 2009.

[15] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, (8):30–37, 2009.

[16] B. Li, Q. Yang, and X. Xue. Can movies and books collaborate? cross-domain collaborative filtering for sparsity reduction. In *IJCAI*, volume 9, pages 2052–2057, 2009.

[17] J. Li, X. Hu, L. Wu, and H. Liu. Robust unsupervised feature selection on networked data. In *Proceedings of SIAM International Conference on Data Mining*. SIAM, 2016.

[18] Y. Li, J. Hu, C. Zhai, and Y. Chen. Improving one-class collaborative filtering by incorporating rich user information. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 959–968. ACM, 2010.

[19] C.-b. Lin. Projected gradient methods for nonnegative matrix factorization. *Neural computation*, 19(10):2756–2779, 2007.

[20] J. Liu, C. Wang, J. Gao, Q. Gu, C. C. Aggarwal, L. M. Kaplan, and J. Han. Gin: A clustering model for capturing dual heterogeneity in networked data. In *SDM*, pages 388–396. SIAM, 2015.

[21] Z. Lu, W. Pan, E. W. Xiang, Q. Yang, L. Zhao, and E. Zhong. Selective transfer learning for cross domain recommendation. In *SDM*, pages 641–649. SIAM, 2013.

[22] H. Ma. An experimental study on implicit social recommendation. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pages 73–82. ACM, 2013.

[23] J. Ni, H. Tong, W. Fan, and X. Zhang. Inside the atoms: ranking on a network of networks. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1356–1365. ACM, 2014.

[24] J. Ni, H. Tong, W. Fan, and X. Zhang. Flexible and robust multi-network clustering. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 835–844. ACM, 2015.

[25] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, and Q. Yang. One-class collaborative filtering. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 502–511. IEEE, 2008.

[26] R. Parshani, S. V. Buldyrev, and S. Havlin. Interdependent networks: reducing the coupling strength leads to a change from a first to second order percolation transition. *Physical review letters*, 105(4):048701, 2010.

[27] S. Razick, G. Magklaras, and I. M. Donaldson. irefindex: a consolidated protein interaction database with provenance. *BMC bioinformatics*, 9(1):1, 2008.

[28] V. Rosato, L. Issacharoff, F. Tiriticco, S. Meloni, S. Porcellinis, and R. Setola. Modelling interdependent infrastructures using interacting dynamical models. *International Journal of Critical Infrastructures*, 4(1-2):63–79, 2008.

[29] A. Sen, A. Mazumder, J. Banerjee, A. Das, and R. Compton. Multi-layered network using a new model of interdependency. *arXiv preprint arXiv:1401.1783*, 2014.

[30] J. Shao, S. V. Buldyrev, S. Havlin, and H. E. Stanley. Cascade of failures in coupled network systems with multiple support-dependent relations. *arXiv preprint arXiv:1011.0234*, 2010.

[31] J. Shao, S. V. Buldyrev, S. Havlin, and H. E. Stanley. Cascade of failures in coupled network systems with multiple support-dependence relations. *Physical Review E*, 83(3):036116, 2011.

[32] A. P. Singh and G. J. Gordon. Relational learning via collective matrix factorization. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 650–658. ACM, 2008.

[33] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 990–998. ACM, 2008.

[34] M. A. Van Driel, J. Bruggeman, G. Vriend, H. G. Brunner, and J. A. Leunissen. A text-mining analysis of the human phenome. *European journal of human genetics*, 14(5):535–542, 2006.

[35] D. J. Watts and S. H. Strogatz. Collective dynamics of small-world networks. *nature*, 393(6684):440–442, 1998.

[36] D. Yang, J. He, H. Qin, Y. Xiao, and W. Wang. A graph-based recommendation across heterogeneous domains. In *Proceedings of the 24rd ACM International Conference on Conference on Information and Knowledge Management*, pages 463–472. ACM, 2015.

[37] Y. Yao, H. Tong, G. Yan, F. Xu, X. Zhang, B. K. Szymanski, and J. Lu. Dual-regularized one-class collaborative filtering. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 759–768. ACM, 2014.

[38] X. Zheng, H. Ding, H. Mamitsuka, and S. Zhu. 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, 2013.