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.ying2}@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 priori. 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 (PP/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 [5] and cross-platform e-commerce [6, 16, 21, 36].

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).

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 net-
work, 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 FASCINATE algorithm can achieve up to $10^3$ 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., $A$), bold lower-case for vectors (e.g., $a$) and calligraphic for sets (e.g., $A$). $A'$ denotes the transpose of matrix $A$. We use the ‘$\hat{}$’ sign to denote the notations after a new node is accommodated to the system (e.g., $\hat{J}$, $\hat{A}_j$), 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 $G$, where $G(i, j) = 1$ if layer-$j$ depends on layer-$i$, and $G(i, j) = 0$ otherwise. Second, we need a set of $g$ within-layer connectivity matrices: $A = \{A_1, \ldots, A_g\}$ to describe the connectivities/similarities between nodes within the same layer. Third, we need a set of cross-layer dependency matrices $D = \{D_{i,j} \mid i, j = 1, \ldots, g\}$, where $D_{i,j}$ describes the dependencies between the nodes from layer-$i$ and the nodes from layer-$j$ if these two layers are directly dependent (i.e., $G(i, j) = 1$). When there is no direct dependencies between the two layers (i.e., $G(i, j) = 0$), the corresponding dependency matrix $D_{i,j}$ is absent. Taking the multi-layered network in Figure 2 for an example, the abstract layer-layer dependency network $G$ of this biological system can be viewed as a line graph. The four within-layer similarity matrices in $A$ are the chemical network ($A_1$), the drug network ($A_2$), the disease network ($A_3$) and the protein-protein interaction (PPI) network ($A_4$). Across those layers, we have three non-empty dependency matrices, including the chemical-drug dependency matrix ($D_{1,2}$), the drug-disease interaction matrix ($D_{2,3}$) and the disease-protein dependency matrix ($D_{3,4}$).

As mentioned earlier, it is often very hard to accurately know the cross-layer dependency matrices $\{D_{i,j} \mid i, j = 1, \ldots, g\}$. 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 $G$; (2) within-layer connectivity matrices $A = \{A_1, \ldots, A_g\}$; and (3) observed cross-layer dependency matrices $D = \{D_{i,j} \mid i, j = 1, \ldots, g\}$.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A, B$</td>
<td>the adjacency matrices (bold upper case)</td>
</tr>
<tr>
<td>$a, b$</td>
<td>column vectors (bold lower case)</td>
</tr>
<tr>
<td>$A, B$</td>
<td>sets (calligraphic)</td>
</tr>
<tr>
<td>$A(i, j)$</td>
<td>the element at $i^{th}$ row $j^{th}$ column in matrix $A$</td>
</tr>
<tr>
<td>$A(i,:)$</td>
<td>the $i^{th}$ row of matrix $A$</td>
</tr>
<tr>
<td>$A(:,j)$</td>
<td>the $j^{th}$ column of matrix $A$</td>
</tr>
<tr>
<td>$A'$</td>
<td>transpose of matrix $A$</td>
</tr>
<tr>
<td>$\hat{A}$</td>
<td>the adjacency matrix of $A$ with the newly added node</td>
</tr>
<tr>
<td>$G$</td>
<td>the layer-layer dependency matrix</td>
</tr>
<tr>
<td>$A$</td>
<td>within-layer connectivity matrices of the network $A = {A_1, \ldots, A_g}$</td>
</tr>
<tr>
<td>$D$</td>
<td>cross-layer dependency matrices $D = {D_{i,j} \mid i, j = 1, \ldots, g}$</td>
</tr>
<tr>
<td>$W_{i,j}$</td>
<td>weight matrix for $D_{i,j}$</td>
</tr>
<tr>
<td>$F_i$</td>
<td>low-rank representation for layer-i ($i = 1, \ldots, g$)</td>
</tr>
<tr>
<td>$m_i, n_i$</td>
<td>number of edges and nodes in graph $A_i$</td>
</tr>
<tr>
<td>$m_{i,j}$</td>
<td>number of dependencies in $D_{i,j}$</td>
</tr>
<tr>
<td>$g$</td>
<td>total number of layers</td>
</tr>
<tr>
<td>$r$</td>
<td>the rank for ${F_i}_{i=1}^{g}$</td>
</tr>
<tr>
<td>$t$</td>
<td>the maximal iteration number</td>
</tr>
<tr>
<td>$\xi$</td>
<td>the threshold to determine the iteration</td>
</tr>
</tbody>
</table>
different weights to different entries in the corresponding cross-layer dependency matrix $D_{i,j}$; and $F_i$ is the low-rank representation for layer $i$. For now, we set the weight matrices as follows: $W_{i,j}(u,v) = 1$ if $D_{i,j}(u,v)$ is observed, and $W_{i,j}(u,v) \in [0, 1]$ if $D_{i,j}(u,v) = 0$ (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 $F_i$ 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., $C_1$) is that we want to match all the cross-layer dependencies, calibrated by the weight matrix $W_{i,j}$. The second term (i.e., $C_2$) 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 $\|F_i(u,:) - F_i(v,:))\|_2$). If the within-layer connectivity between these two nodes is strong (i.e., large $A_i(u,v)$), the third term (i.e., $C_3$) is to regularize the norm of the low-rank matrices $\{F_j\}_{j=1,...,g}$ 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 $D_{i,j}(u,v) = F_i(u,:) \cdot F_j(v,:)'$

### 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 $F_i$ naturally forms a ‘block’. To be specific, if we fix all other $F_j (j = 1, \ldots, g, j \neq i)$ and ignore the constant terms, Eq. (1) can be simplified as

$$\min_{F_i \geq 0} J = \sum_{i,j: G(i,j)=1} \|W_{i,j} \odot (D_{i,j} - F_i \odot F_j')\|_F^2$$  \hspace{1cm} (2)$$

The derivative of $J_i$ wrt $F_i$ is

$$\frac{\partial J_i}{\partial F_i} = 2 \sum_{j: G(i,j)=1} \left[-(W_{i,j} \odot W_{i,j} \odot D_{i,j})F_j \right] + \alpha \text{tr}(F_i'(T_i - A_i)F_i) + \beta \|F_i\|^2_F$$  \hspace{1cm} (3)$$

A fixed-point solution of Eq. (3) with non-negativity constraint on $F_i$ leads to the following multiplicative updating rule for $F_i$

$$F_i(u,v) \leftarrow F_i(u,v) \sqrt{X(u,v) \over Y(u,v)} \hspace{1cm} (4)$$

where

$$X = \sum_{j: G(i,j)=1} (W_{i,j} \odot W_{i,j} \odot D_{i,j})F_j + \alpha A_iF_i$$  \hspace{1cm} (5)$$

$$Y = \sum_{j: G(i,j)=1} (W_{i,j} \odot W_{i,j} \odot (F_iF_j'))F_j + \alpha T_iF_i + \beta F_i$$

Recall that we set $W_{i,j}(u,v) = 1$ when $D_{i,j}(u,v) > 0$, and $W_{i,j}(u,v) = w$ when $D_{i,j}(u,v) = 0$. Here, we define $I_{i,j}$ as an indicator matrix for the observed entries in $D_{i,j}$, that is, $I_{i,j}(u,v) = 1$ if $D_{i,j}(u,v) > 0$, and $I_{i,j}(u,v) = 0$ if $D_{i,j}(u,v) = 0$. Then, the estimated dependencies over the observed data can be represented as $R_{i,j} = I_{i,j} \odot (F_iF_j')$. With these notations, we can
further simplify the update rule in Eq. (5) as follows
\[
X = \sum_{j: G(i,j)=1} D_{i,j} F_j + \alpha A_i F_i \tag{6}
\]
\[
Y = \sum_{j: G(i,j)=1} ((1 - w^2)\bar{R}_{i,j} + w^2 F_{i,j}^\prime) F_j + \alpha T_i F_i + \beta F_i \tag{7}
\]

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 \( \{F_i\}_{i=1,\ldots,g} \) one by one. We use two criteria to terminate the iteration: (1) either the Frobenius norm between two successive iterations for all \( \{F_i\}_{i=1,\ldots,g} \) is less than a threshold \( \xi \), 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 \( G \), (b) within-layer connectivity matrices \( A = \{A_1, \ldots, A_n\} \), and (c) observed cross-layer node dependency matrices \( D = \{D_{i,j} i,j=1, \ldots, g\} \); (2) the rank size \( r \); (3) weight \( w \); (4) regularized parameters \( \alpha \) and \( \beta \).

**Output:** low-rank representations for each layer \( \{F_i\}_{i=1,\ldots,g} \)

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

3.3 Proof and Analysis

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

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
\[
L_i = \sum_{j: G(i,j)=1} \|W_{i,j} \odot (D_{i,j} - F_i F_j^\prime)\|_F^2 + \alpha \text{tr}(F_i^\prime T_i F_i) - \alpha \text{tr}(F_i^\prime A_i F_i) + \beta \|F_i\|_F^2 - \text{tr}(A^\prime F_i)
\]

where \( \Lambda \) is the Lagrange multiplier. Setting the derivative of \( L_i \) wrt \( F_i \) to 0, we get
\[
2\sum_{j: G(i,j)=1} \left[ -(W_{i,j} \odot W_{i,j} \odot D_{i,j}) F_j \right] + \left( W_{i,j} \odot W_{i,j} \odot (F_j F_j^\prime) \right) F_j + \alpha T_i F_i - \alpha A_i F_i + \beta F_i = \Lambda
\]

By the KKT complementary slackness condition, we have
\[
\sum_{j: G(i,j)=1} \left( W_{i,j} \odot W_{i,j} \odot (F_j F_j^\prime) \right) F_j + \alpha T_i F_i + \beta F_i
\]

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

**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 \( F_i \).

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 \( O(\sum_{i=1}^g (\sum_{j: G(i,j)=1} (m_{i,j} r + (n_i + n_j) r^2) + m_i r)) t \).

**Proof.** In each iteration Alg. 1 for updating \( F_i \), the complexity of calculating \( X \) by Eq. (6) is \( O(\sum_{j: G(i,j)=1} m_{i,j} r + m_i r) \) due to the sparsity of \( D_{i,j} \) and \( A_i \). The complexity of computing \( R_{i,j} \) in \( Y \) is \( O(m_{i,j} r) \). Computing \( F_i (F_i^\prime F_j) \) requires \( O((n_i + n_j) r^2) \) operations and computing \( \alpha T_i F_i + \beta F_i \) requires \( O(n_i r) \) operations. So, it is of \( O(\sum_{j: G(i,j)=1} (m_{i,j} r + (n_i + n_j) r^2) + m_i r) \) complexity to get \( Y \) in line 7. Therefore, it takes \( O(\sum_{j: G(i,j)=1} (m_{i,j} r + (n_i + n_j) r^2) + m_i r) \) to update \( F_i \). Putting all together, the complexity of updating all low-rank matrices in each iteration is \( O(\sum_{i=1}^g (\sum_{j: G(i,j)=1} (m_{i,j} r + (n_i + n_j) r^2) + m_i r)) t \), thus, the overall complexity of Alg. 1 is \( O(\sum_{i=1}^g (\sum_{j: G(i,j)=1} (m_{i,j} r + (n_i + n_j) r^2) + m_i r)) t \), where \( t \) is the maximum number of iterations in the algorithm.

**Lemma 3.** The space complexity of Alg. 1 is \( O(\sum_{i=1}^g (n_i r + m_i r)) \).

**Proof.** It takes \( O(\sum_{i=1}^g n_i r) \) to store all the low-rank matrices, and \( O(\sum_{i=1}^g m_i r + \sum_{j: G(i,j)=1} m_{i,j} r) \) to store all the within-layer connectivity matrices and dependency matrices in the multi-layered network. To compute \( X \) for \( F_i \), it costs \( O(n_i r) \) to compute \( \sum_{j: G(i,j)=1} D_{i,j} F_j + \alpha A_i F_i \). For \( Y \), the space cost of computing \( R_{i,j} \) and \( F_i (F_j F_j^\prime) \) is \( O(m_{i,j}) \) and \( O(n_i r) \) respectively.
Therefore, the space complexity of calculating $\sum_{j:G(i,j)=1} ((1 - w^5)R_{i,j} + w^2F_iF_j^\top)F_j$ is $O(\max_{j:G(i,j)=1} m_{i,j} + n_i r)$. On the other hand, the space required to compute $\alpha T_i F_i + \beta F_i$ is $O(n_i r)$. Putting all together, the space cost of updating all low-rank matrices in each iteration is of $O(\max_{i,j:G(i,j)=1} m_{i,j} + max_i n_i r)$. Thus, the overall space complexity of Alg. 1 is $O(\sum_i (n_i r + m_i) + \sum_{i,j:G(i,j)=1} m_{i,j})$.

C - Variants.

Here, we discuss some variants of the proposed FASCINATE algorithm. First, by setting all the entries in the weight matrix $W_{i,j}$ to 1, FASCINATE becomes a clustering algorithm (i.e., $F_i$ 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 $A = \{A_1, \ldots, A_J\}$ 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 methodological 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 without any 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-$l$, 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 $s$, where $s(u)$ ($u = 1, \ldots, n_1$) denotes the (within-layer) connectivity between the newly added node and the $u^{th}$ existing node in layer-$l$.

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 $F_i$), 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 $\{F_i\}$ 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 $A_1$ for layer-$l$ can be expressed as

$$A_1 = \begin{bmatrix} A_1 & s' \\ s & 0 \end{bmatrix}$$

(11)

where $A_1$ is the within-layer connectivity matrix for layer-$l$ before the arrival of the new node.

Correspondingly, the updated low-rank representation matrix for layer-$l$ can be expressed as $F_1 = [\hat{F}_1, F_1^{(n_1 \times r)}]$, where $\hat{F}_1$ is the updated low-rank representation for the existing $n_1$ nodes in layer-$l$. Then the new objective function $\hat{J}$ in Eq. (1) can be reformatted as

$$\hat{J} = \sum_{i,j:G(i,j)=1} \|W_{i,j} \odot (D_{i,j} - \hat{F}_1 \hat{F}_j^\top)\|_F^2$$

(12)

$$+ \sum_{j:G(\cdot,j)=1} \|W_{1,j} \odot (D_{1,j} - \hat{F}_1)\|_F^2$$

$$+ \alpha \sum_{u=1}^{n_1} \sum_{v=1}^{n_1} A_1(u,v)\|\hat{F}_i(u,:) - \hat{F}_1(v,:))\|_2^2$$

$$+ \alpha \sum_{u=1}^{n_1} \sum_{v=1}^{n_1} A_1(u,v)\|\hat{F}_i(u,:) - \hat{F}_1(v,:)\|_2^2$$

$$+ \beta \sum_{i=2}^{g} \|\hat{F}_i\|_F^2 + \beta \|\hat{F}_1^{(n_1 \times r)}\|_F^2$$

(13)

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

$$\hat{W}_{1,j} = \begin{bmatrix} W_{1,j} \\ 0_{1 \times n_1} \end{bmatrix}, \hat{D}_{1,j} = \begin{bmatrix} D_{(1,j)} \\ 0_{1 \times n_1} \end{bmatrix}$$

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

$$\sum_{j:G(\cdot,j)=1} \|W_{1,j} \odot (D_{1,j} - \hat{F}_1^{(n_1 \times r)} \hat{F}_j)\|_F^2$$

(14)

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

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

where $J^1 = \alpha \sum_{u=1}^{n_1} s(u)\|\hat{f} - \hat{F}_1(v,:))\|_2^2 + \beta \|\hat{f}\|_2^2$, 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 $10^4$, while $J^1$ is in the order of $10^{-3}$. This naturally leads to the following approximation strategy, that is, we (1) fix $J$ with $\{\hat{F}_i^{(n_1 \times r)}\}_{i=1, \ldots, g}$ (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 $\hat{f}$ for the newly arrived node. That is, we seek to solve the following optimization problem

$$f = \arg \min_{f \geq 0} J^1 \text{ subject to: } F_1^{(n_1 \times r)} = F_1^*$$

(15)

with which, we can get an approximate solution $\{\hat{F}_i\}_{i=1, \ldots, g}$ to $\hat{J}$. 
To solve \( f \), we take the derivative of \( J^1 \) wrt \( f \) and get
\[
\frac{1}{2} \frac{\partial J^1}{\partial f} = \beta f + \alpha \sum_{v=1}^{n_1} s(v) (f - F^1_v (v, \cdot)) \tag{16}
\]
\[
= (\beta + \alpha \sum_{v=1}^{n_1} s(v)) f - \alpha sF^1_v
\]
Since \( \alpha \) and \( \beta \) 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
\[
f = \frac{\alpha sF^1_v}{\beta + \alpha \sum_{v=1}^{n_1} s(v)} \tag{17}
\]
For the newly added node, \( f \) can be viewed as the weighted average of its neighbors’ low-rank representations. Notice that in Eq. (17), the non-negativity constraint on \( \alpha \) 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 \( \text{nnz}(s) \) denotes the total number of within-layer links between the newly added node and the original nodes in layer-1 (i.e., \( \text{nnz}(s) \) is the degree for the newly added node). Then the time complexity of FASCINATE-ZERO is \( O(\text{nnz}(s)r) \).

**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 \( s \) (\( \text{nnz}(s) \)) is much smaller than \( n_1 \). Therefore, the complexity of computing \( sF^1_v \) can be reduced to \( O(\text{nnz}(s)r) \). The multiplication between \( \alpha \) and \( sF^1_v \) takes \( O(r) \). Computing \( \sum_{v=1}^{n_1} s(v) \) takes \( O(\text{nnz}(s)) \). Thus, the overall complexity of computing \( f \) is \( O(\text{nnz}(s)r) \). □

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., \( D_{x,y}(u, v) = D_{y,x}(v, u) \)).

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

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 proposed in [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 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-MPR 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.

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 \( \alpha \), \( \beta \) and \( r \) in the proposed FASCINATE algorithm. \( \alpha \) is used to control the impact of node homophily, \( \beta \) is used to avoid over-fitting, and \( r \) is the number of columns of the low-rank matrices \( \{ F_i \} \). 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 \( \alpha \) and \( \beta \). Specifically, a relatively high MAP can be achieved when \( \alpha \) is between 0.1 to 1 and \( \beta \) 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), FASCI-
Table 3: Cross-Layer Dependency Inference on SOCIAL

<table>
<thead>
<tr>
<th>Methods</th>
<th>MAP</th>
<th>R-MPR</th>
<th>HLU</th>
<th>AUC</th>
<th>Prec@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>FASCINATE</td>
<td>0.0660</td>
<td>0.2651</td>
<td>8.4556</td>
<td>0.7529</td>
<td>0.0118</td>
</tr>
<tr>
<td>FASCINATE-CLUST</td>
<td>0.0667</td>
<td>0.2462</td>
<td>8.2160</td>
<td>0.7351</td>
<td>0.0108</td>
</tr>
</tbody>
</table>

Table 4: Cross-Layer Dependency Inference on B10.

<table>
<thead>
<tr>
<th>Methods</th>
<th>MAP</th>
<th>R-MPR</th>
<th>HLU</th>
<th>AUC</th>
<th>Prec@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>FASCINATE</td>
<td>0.3979</td>
<td>0.4066</td>
<td>45.1001</td>
<td>0.9369</td>
<td>0.1039</td>
</tr>
<tr>
<td>FASCINATE-CLUST</td>
<td>0.3189</td>
<td>0.3898</td>
<td>37.4089</td>
<td>0.9176</td>
<td>0.0857</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Methods</th>
<th>MAP</th>
<th>R-MPR</th>
<th>HLU</th>
<th>AUC</th>
<th>Prec@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>FASCINATE</td>
<td>0.5040</td>
<td>0.3777</td>
<td>67.2231</td>
<td>0.8916</td>
<td>0.2500</td>
</tr>
<tr>
<td>FASCINATE-CLUST</td>
<td>0.4297</td>
<td>0.3220</td>
<td>56.8215</td>
<td>0.8159</td>
<td>0.2340</td>
</tr>
</tbody>
</table>

Table 6: Cross-Layer Dependency Inference on INFRA-3.

<table>
<thead>
<tr>
<th>Methods</th>
<th>MAP</th>
<th>R-MPR</th>
<th>HLU</th>
<th>AUC</th>
<th>Prec@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>FASCINATE</td>
<td>0.4780</td>
<td>0.0788</td>
<td>55.7289</td>
<td>0.6970</td>
<td>0.5560</td>
</tr>
<tr>
<td>FASCINATE-CLUST</td>
<td>0.5030</td>
<td>0.0850</td>
<td>49.1223</td>
<td>0.7122</td>
<td>0.4917</td>
</tr>
</tbody>
</table>

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

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

FASCINATE scales linearly wrt the overall network size (i.e., \( \sum (n_i + m_j) + \sum m_{i,j} \)), 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., \( 1.878 \times 10^{-4} \) seconds vs. \( 2.794 \times 10^{-3} \) 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 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 priori. 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 (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


![Figure 7: Wall-clock time vs. the size of the network.](image-url)


