Dissecting Cross-Layer Dependency Inference on Multi-Layered Inter-Dependent Networks

Yuchen Yan
yucheny5@illinois.edu
University of Illinois at Urbana-Champaign
urbana, IL, USA

Qinghai Zhou
qinghai2@illinois.edu
University of Illinois at Urbana-Champaign
urbana, IL, USA

Jinning Li
jinning4@illinois.edu
University of Illinois at Urbana-Champaign
urbana, IL, USA

Tarek Abdelzaher
zaher@illinois.edu
University of Illinois at Urbana-Champaign
urbana, IL, USA

Hanghang Tong
htong@illinois.edu
University of Illinois at Urbana-Champaign
urbana, IL, USA

ABSTRACT

Multi-layered inter-dependent networks have emerged in a wealth of high-impact application domains. Cross-layer dependency inference, which aims to predict the dependencies between nodes across different layers, plays a pivotal role in such multi-layered network systems. Most, if not all, of existing methods exclusively follow a *coupling principle* of design and can be categorized into the following two groups, including (1) heterogeneous network embedding based methods (*data coupling*), and (2) collaborative filtering based methods (*module coupling*). Despite the favorable achievement, methods of both types are faced with two intricate challenges, including (1) the *sparsity challenge* where very limited observations of cross-layer dependencies are available, resulting in a deteriorated prediction of missing dependencies, and (2) the *dynamic challenge* given that the multi-layered network system is constantly evolving over time.

In this paper, we first demonstrate that the inability of existing methods to resolve the *sparsity challenge* roots in the *coupling principle* from the perspectives of both *data coupling* and *module coupling*. Armed with such theoretical analysis, we pursue a new principle where the key idea is to *decouple* the within-layer connectivity from the observed cross-layer dependencies. Specifically, to tackle the *sparsity challenge* for static networks, we propose FITO-S, which incorporates a position embedding matrix generated by random walk with restart and the embedding space transformation function. More essentially, the *decoupling principle* ameliorates the *dynamic challenge*, which naturally leads to FITO-D, being capable of tracking the inference results in the dynamic setting through incrementally updating the position embedding matrix and fine-tuning the space transformation function. Extensive evaluations on real-world datasets demonstrate the superiority of the proposed framework FITO for cross-layer dependency inference.

CCS CONCEPTS

• **Information systems** → **Collaborative filtering**; **Data mining**.

KEYWORDS

Multi-Layered Inter-Dependent Networks, Cross-Layer Dependency, Sparsity Challenge, Dynamic Challenge, Random Walk with Restart.

ACM Reference Format:
Yuchen Yan, Qinghai Zhou, Jinning Li, Tarek Abdelzaher, and Hanghang Tong. 2022. Dissecting Cross-Layer Dependency Inference on Multi-Layered Inter-Dependent Networks. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM '22), October 17–21, 2022, Atlanta, GA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3511808.3557291

1 INTRODUCTI

In the era of big data, networks [49, 61] are often collected from multiple domains with *inter-dependent* relations, resulting in the emergence of *multi-layered inter-dependent networks* [3, 16, 18, 27, 39]. A typical example of such multi-layered networks is the inter-dependent infrastructure network. As illustrated in Figure 1, the router network relies on the internet center network, while the internet center network is dependent on the power grid for electricity supply. [1] In this example, accurately inferring the cross-layer dependency is critical for providing cost-effective internet center service for users and efficient power distribution. Besides the infrastructure network, cross-layer dependency inference also plays an essential role in a variety of high-impact

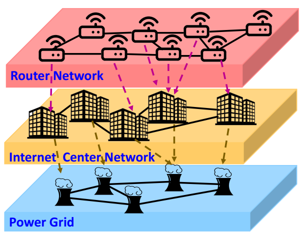


Figure 1: An inter-dependent infrastructure network. Dashed arrows represent dependencies.

[1] The dependency relationship between nodes can be either one-to-one or one-to-many. Our model can handle both cases without necessarily distinguishing them.

data mining applications, including new drug discovery in biological networks [33], recommendation in e-commerce networks [56], and team management in collaboration networks [5].

To predict unknown cross-layer dependencies, the vast majority of existing methods follow a *coupling principle* and can be categorized into two groups according to different *coupling* stages, including (1) *data coupling* for heterogeneous network embedding (HNE)-based methods in *data ingestion stage*, and (2) *module coupling* for collaborative filtering (CF)-based methods in *module design stage*. Concretely, first, heterogeneous network embedding (HNE)-based methods advocate the *data coupling* approach by viewing the multi-layered inter-dependent network as a world-view network, where all layers' networks are combined into one large heterogeneous network and the cross-layer dependency inference task is formulated as a link prediction problem on this heterogeneous network. In this category, existing methods propose to learn high-quality heterogeneous network representations through (1) generating metapaths [10, 17], (2) capturing node attribute information with graph neural network (GNN) [4, 29], and (3) leveraging knowledge graph embedding techniques to boost the link prediction performance [34]. Second, collaborative filtering (CF)-based methods, from data perspective, regard the input as a multiple-layered collection of homogeneous networks with cross-layer dependent interactions and pursue a *module coupling* approach. Specifically, CF-based methods aim to model the cross-layer dependency interaction and collectively exploit the within-layer connectivity as a regularization term in the final objective. To name a few, Singh et al. formulate this task as a collective matrix factorization (CMF) problem by equally treating the within-layer network adjacency matrix and the observed dependency matrix in factorization [43]. Homogeneous network embedding approaches such as DeepWalk [40] and LINE [44] are applied to capture the within-layer connectivity in [9, 62], where the key idea is to replace direct factorization in CMF with the corresponding node embedding method.

Nonetheless, most, if not all, of existing methods face two key challenges, including (1) the *sparsity challenge* and (2) the *dynamic challenge*. To be specific, *sparsity challenge* refers to the deteriorated prediction of missing dependencies when very limited observed cross-layer dependencies are available. For HNE-based methods, sparse cross-layer observations induce a remarkable distortion of the underlying connectivity distribution (e.g., insufficient generated high-quality metapaths) and consequently diminish the performance in inferring unobserved links between nodes of various types. In addition, the performance of CF-based approaches is exceptionally hindered by the dominating unlabeled data, which consists of both non-existing dependency and unobserved dependency. Furthermore, most existing methods assume that the multi-layered inter-dependent networks are static, despite the fact that the complex network system is topologically evolving over time in terms of both within-layer connectivity and newly observed cross-layer dependencies, which demands a novel method being capable of efficiently updating the inference results in the dynamic setting.

In this paper, we first tackle the sparsity challenge in static cross-layer dependency inference. We conduct a comprehensive theoretical analysis and demonstrate the deficiency of the aforementioned methods in the sparse setting roots in the *coupling principle*. Specifically, we uncover that the essence of HNE-based methods following *data coupling* is constructing a cross-layer *similarity* matrix for dependency inference. As the observed dependencies become sparse, conventional HNE-based methods cannot generate sufficient high-quality positive node pairs to construct the cross-layer *similarity* matrix, thereby bearing the defect in predicting cross-layer dependency. In addition, CF-based methods exclusively embrace the *module coupling* approach by regarding the within-layer connectivity as a regularization term in the final objective. Through detailed analysis, we demonstrate it is incapable of capturing the so-called *total connectivity information* (see details in Definition 2).

Based on the theoretical analysis, we pursue a *decoupling principle* in both *data ingestion stage* and *module design stage* and propose a novel algorithm FITO-S to address the *sparsity* challenge in the static setting. FITO-S first delves into each within-layer network and distills the *total connectivity information* into a position embedding matrix with random walk with restart (RWR) [45]. Then, FITO-S learns a space transformation function by exploiting the observed cross-layer dependency, which translates the obtained representations from various layers to a unified latent embedding space. Additionally, the *decoupling principle* naturally leads to a dynamic version of the algorithm (FITO-D) upon the static FITO-S. For dynamic systems, FITO-D follows a two-step procedure, including (1) incrementally updating the RWR position embedding matrix, and (2) fine-tuning the space transformation function by capitalizing on the newly observed cross-layer dependencies.

The main contributions of the paper are summarized as follows,

- **Analysis.** We conduct a theoretical analysis on the existing two main categories of methods for cross-layer dependency inference, centered around the *coupling principle* in different stages, which reveals some fundamental limitations of the existing methods in resolving the *sparsity challenge*.
- **Algorithms.** We propose a family of novel algorithms, FITO, based on the idea of decoupling the within-layer connectivity from cross-layer dependency, to effectively and efficiently infer cross-layer dependencies in both static and dynamic settings.
- **Evaluations.** We perform extensive empirical evaluations on real-world datasets from different domains. The evaluation results demonstrate (1) *effectiveness* for the sparsity challenge, where FITO-S outperforms all baseline methods in terms of inference accuracy; (2) *efficiency* for the dynamic challenge, where the proposed FITO-D achieves $29-39\times$ speed-up over retraining the FITO-S in dynamic steps.

2 PROBLEM DEFINITION

In this section, we first introduce the notations and then formally define the problem of cross-layer dependency inference.

The main symbols used in this paper are summarized in Table 1. We use bold uppercase letters for matrices (e.g., $\mathbf{A}$), bold lowercase letters for vectors (e.g., $\mathbf{r}$) and lowercase letters for scalars (e.g., α) . To index a matrix/vector, we use $\mathbf{A}(u,v)$ to represent the entry at the u-th row and the v-th column of matrix $\mathbf{A}$, $\mathbf{A}(u,:)$ to denote the u-th row of $\mathbf{A}$, $\mathbf{A}(:,v)$ to denote the v-th column of $\mathbf{A}$.

For a multi-layered inter-dependent network Γ composed of g layers of networks, there exist dependencies between some specific

Table 1: Symbols and Notations.

Symbol	Definition
Γ	the multi-layered network
G_i	the network of the i-th layer in Γ
$\mathbf{A}$	adjacency matrix
$\mathbf{L}^o_{i,j}$, $\mathbf{L}_{i,j}$	observed and predicted dependency matrices of (G_i, G_j)
$\mathbf{L}^h$	homophily-based *similarity* matrix in HNE-based methods
$\mathbf{D}$	the diagonal degree matrix of $\mathbf{A}$
$\Delta\mathbf{A}_i$	perturbation matrix of $\mathbf{A}_i$
$\Delta\mathbf{L}^o_{i,j}$	newly observed dependency matrix
$\mathbf{R}$	RWR matrix
$\mathbf{F}$	embedding matrix
$\mathbf{R}_{\mathcal{P}}$	RWR position embedding matrix
$\mathbf{A}^\top$	transpose of $\mathbf{A}$
$\hat{\mathbf{A}}$	row normalized matrix of $\mathbf{A}$
n_i	numbers of nodes in G_i
u_i	the u-th node of G_i
k	the second dimension of $\mathbf{R}_{\mathcal{P}_i}$
α, β, γ	parameters
$\mathrm{tr}(\mathbf{A})$	trace of $\mathbf{A}$
$\|\|\cdot\|\|_F^2$	Frobenius norm of matrix

layers (e.g., the i-th layer G_i and the j-th layer G_j).[2] Given the network connectivity for each layer's network (adjacency matrix $\mathbf{A}_i$) and some observed dependencies ($\mathbf{L}^o_{i,j}$), the task of static dependency inference is to find all dependencies across G_i and G_j, which can be defined similarly as in [7]:

Problem 1. ***Static Cross-Layer Dependency Inference***

Given: *a multi-layered network $\Gamma = (\mathcal{A}, \mathcal{L}^o)$, where (1) $\mathcal{A} = \{\mathbf{A}_1, \ldots, \mathbf{A}_g\}$ is the set of within-layer adjacency matrices; and (2) $\mathcal{L}^o = \{\mathbf{L}^o_{i,j} | i, j = 1, \ldots, g\}$ is the set of observed cross-layer dependency matrices.*

Output: *(1) embedding matrices $\{\mathbf{F}_1, \ldots, \mathbf{F}_g\}$; (2) the predicted cross-layer dependency matrices $\{\mathbf{L}_{i,j} | i, j = 1, \ldots, g\}$.*

Accordingly, the dynamic cross-layer dependency inference problem is defined as follows.

Problem 2. ***Dynamic Cross-Layer Dependency Inference***

Given: *(1) an original multi-layered network $\Gamma = \{\mathcal{A}, \mathcal{L}^o\}$; (2) the change of the multi-layer network $\Delta\Gamma = \{\Delta\mathcal{A}, \Delta\mathcal{L}^o\}$, where $\Delta\mathcal{A} = \{\Delta\mathbf{A}_1, \ldots, \Delta\mathbf{A}_g\}$ is the perturbation on connectivity occurring within layers and $\Delta\mathcal{L}^o = \{\Delta\mathbf{L}^o_{i,j} | i, j = 1, \ldots, g\}$ is newly observed dependency set.*[3]

Output: *(1) new embedding matrices $\{\mathbf{F}_1, \ldots, \mathbf{F}_g\}$; (2) new predicted cross-layer dependency matrices $\{\mathbf{L}_{i,j} | i, j = 1, \ldots, g\}$.*

The dynamic change of the multi-layered network (i.e., $\Delta\Gamma$) referrs to the following two perspectives. First, the topological perturbations may appear in each layer of network, i.e., $\Delta\mathcal{A}$. For example, in a given layer of network G_i, we might observe the vanishing of existing edges and new links to be established, which change the corresponding entries in $\mathbf{A}_i$. Besides, new nodes can be inserted in G_i, which extends the index in $\mathbf{A}_i$.[4]Second, additional cross-layer dependencies might emerge as the multi-layered network Γ evolves over time, i.e., $\Delta\mathcal{L}^o$.

Remarks. The cross-layer dependency inference (Problems 1 and 2) studied in this paper is quite general and it relates to several classic data mining tasks. For example, if $g = 2$ and both within-layer matrices are absent, the cross-layer dependency inference degenerates to the classic collaborative filtering problem [25, 37]; if $g = 2$ and only one of the two within-layer adjacency matrix is available, we can view cross-layer dependency inference as social recommendation problem [56]; if $g = 2$ and both layers are for the same type of nodes (e.g., the same populations from two different social platforms), cross-layer dependency inference becomes (soft) network alignment problem [59]; and if $g > 2$ and all the within-layer adjacency matrices are absent, cross-layer dependency inference can be viewed as collective collaborative filtering problem [43].

3 ANALYSIS

In this section, we theoretically demonstrate that under the sparsity challenge in static setting, both HNE-based and CF-based methods bear some fundamental limitations rooted in the *coupling principle*. Concretely, we reveal that the nature of *data coupling* HNE-based methods is to construct a cross-layer *similarity* matrix to predict cross-layer dependency link. When facing the sparsity challenge, they fall short in generating sufficient positive cross-layer node pairs to construct this matrix. For CF-based methods, viewing within-layer connectivity as regularization of cross-layer dependency and optimizing them in a coupling way result in the inability of capturing the *total connectivity information* of each layer.

3.1 Data Coupling HNE-Based Methods

In this subsection, we conduct a theoretical analysis about the limitations of HNE-based methods under the *sparsity challenge*. We demonstrate that HNE-based methods with the *homophily assumption* essentially construct a *homophily-based* cross-layer *similarity* matrix, which is used to predict cross-layer dependency link, thereby encountering a problem similar to *cold-start* in traditional collaborative filtering tasks. We start from the *homophily assumption* used in a variety of network embedding algorithms, including random walk (RW)-based, metapath-based, GNN-based and knowledge graph-based:

Definition 1. ***Homophily Assumption****: The homophily assumption in network embedding algorithm is that nodes topologically "close" to each other should have "similar" embedding vectors, while distant nodes should have disparate embedding representations.*

Here the term "close" does not limit to direct connectivity in networks like in LINE [44], other patterns also satisfy the definition of "close". To name a few, a node pair sampled from random walk is considered to be "close" in DeepWalk [40] and node2vec [21]. The "neighboring nodes" sampled from GNN with a fixed number of layers [22, 29] and nodes connected by a metapath [10, 17] can be interpreted as "close". The term "similar" refers to small distance

[2]Multi-layered inter-dependent networks are different from multiplex/multi-view network defined in [38, 58], where each layer has same set of nodes and represents different edge types. They can be viewed as a general form of heterogeneous network.
[3]In this paper, we only consider cumulative dependency. For example, once a product has been bought by a user, the dependency (having been bought) between the product and the user will stay.

[4]Deleting node is equivalent to removing all incident edges.

measure or large dot product among the node embeddings. In RW-based methods, the homophily loss can be formulated as follows:

$$\mathcal{J}_h = \sum_{u \in \mathcal{V}} \sum_{v,v' \in \mathcal{V}} -\log(\sigma(\mathbf{v}^T\mathbf{u})) - Q \cdot \mathbb{E}_{v' \sim P(v')} \log(\sigma(-\mathbf{v}'^T\mathbf{u})) \quad (1)$$

where (u, v) denotes a positive "close" node pair sampled by the model and (u, v') represents a negative "distant" pair. By minimizing Eq. (1), the model is driven to produce similar embeddings for "close" nodes (i.e., large dot-product) and distant embeddings otherwise. By putting the dot products of all node pairs together, we can find that the *homophily assumption* in RW-based embedding aims to construct a new *similarity* matrix. For graph auto-encoder (GAE) [30], the *similarity* matrix is exactly the adjacency matrix. For metapath-based methods, they have similar homophily loss functions as RW-based methods. For GNN-based and knowledge graph-based methods, they adopt embedding distance measurements, which can also be transformed into a *similarity* score between 0 and 1 (e.g.,$e^{-d(u,v)}$). Hence, the *similarity* matrix can be constructed.

With the underlying *homophily assumption*, HNE-based methods attempt to construct one integrated *similarity* matrix $\mathbf{L}^h$, and they essentially utilize the cross-layer (i.e., type) *similarity* matrices ($\mathbf{L}^h_{i,j}$) to predict the dependency link, which are sub-matrices of $\mathbf{L}^h$ with type-i nodes and type-j nodes as rows and columns in the *coupled* world-view network, respectively.

Therefore, we have the following claim about HNE-based methods under the sparsity challenge:

Claim 1. ***Sparsity challenge for HNE-based methods****: Under the sparsity challenge, HNE-based methods with finite length of metapaths or number of GNN layers cannot generate sufficient high-quality "similar" cross-layer (type) node pairs. The constructed similarity matrix $\mathbf{L}^h_{i,j}$ will become sparse when the number of observed dependencies decreases (equivalent to that matrix $\mathbf{L}^o_{i,j}$ becomes sparse.), thereby resulting in a similar cold-start problem in collaborative filtering.*

The correctness of this claim is shown as follows. Since the constructed cross-layer *similarity* matrix $\mathbf{L}^h_{i,j}$ measures the similarity between nodes from different layers (i.e., different types in the heterogeneous network), it has a *type transition* process. In metapath-based methods, this implies that at least one observed cross-layer dependency link is covered in the finite-length metapath. For GNN-based methods, it corresponds to the fact that nodes with different types reach each other in finite layers/hops, which passes at least one observed cross-layer dependency link. Therefore, when the observed $\mathbf{L}^o_{i,j}$ becomes extremely sparse, the number of cross-layer links and positive similar cross-layer node pairs will be reduced. Then, these positive "similar" node pairs might be falsely treated as negative "disparate" ones. Consequently, the constructed matrix $\mathbf{L}^h_{i,j}$ from HNE-based methods bears a similar cold-start problem as in CF task.

To summarize, we conclude that HNE-based methods will construct a cross-layer *similarity* matrix $\mathbf{L}^h_{i,j}$ for every network layer pair (i, j), as the result of *data coupling* approach, since the connectivity of each layer's network is fused in one matrix $\mathbf{L}^h$.

3.2 Module Coupling CF-Based Methods

Collaborative filtering (CF)-based methods do not follow the *coupling principle* in the *data ingestion stage*, where within-layer connectivity matrices are separated from cross-layer dependency matrices. Instead, they follow the *coupling principle* in the *module design stage*. In this subsection, we investigate the limitation of CF-based methods under the *sparsity challenge*. To better assist the analysis, we first give the definition of the concept, *total connectivity information*, as follows,

Definition 2. ***Total Connectivity Information.*** *Given the adjacency matrix $\mathbf{A}_i$ of a single layer network G_i, we say that the total connectivity information of G_i is captured if the learned node embeddings or some specific matrix to be factorized satisfy the following two conditions:*

- *High Order Connectivity: If two nodes in G_i are connected within an arbitrary number of steps, the high order connectivity information should be preserved by the embedding vectors or the specific matrix.*
- *Homophily Assumption: Nodes topologically "close" to each other should be discriminated with nodes topologically "distant" to each other by embedding vectors or some specific matrix.*

We now claim that under the *sparsity challenge*, the existing CF-based methods are unable to capture the *total connectivity information* of each layer as a consequence of the *module coupling* approach. In the executing process, CF-based methods exploit and optimize the within-layer connectivity and cross-layer dependency in a coupling fashion, and the within-layer connectivity information is formulated as a regularization term to facilitate the cross-layer dependency inference.

To demonstrate the correctness of the claim, we initiate the analysis on the objective function of FASCINATE [7], a typical CF-based method for cross-layer dependency inference:

$$\min_{\mathbf{F}_i \geq \mathbf{0}(i=1,\ldots,g)} \mathcal{J} = \underbrace{\sum_{i,j} \|\mathbf{W}_{i,j} \odot (\mathbf{L}^o_{i,j} - \mathbf{F}_i\mathbf{F}_j^\top)\|_F^2}_{\text{C1: Matching observed cross-layer dependencies}} + \underbrace{\alpha \sum_{i=1}^{g} \mathrm{tr}(\mathbf{F}_i^\top(\mathbf{D}_i - \mathbf{A}_i)\mathbf{F}_i) + \beta \sum_{i=1}^{g} \|\mathbf{F}_i\|_F^2}_{\text{C2: Within-layer connectivity regularization}} \quad (2)$$

where $\odot$ represents the element-wise product, $\mathbf{W}_{i,j}$ is particularly defined as a weight matrix to alleviate the *sparsity challenge*. The goal of the term C1 is to match the observed cross-layer dependencies, and C2 is the within-layer connectivity regularization, including a trace term and a matrix Frobenius norm of $\mathbf{F}_i$. We start with analyzing C2 in Eq. (2). $\mathrm{tr}(\mathbf{F}_i^\top(\mathbf{D}_i - \mathbf{A}_i)\mathbf{F}_i)$ can be further rewritten as $\sum \mathbf{A}_i(u_i, v_i)\|\mathbf{F}_i(u_i, :) - \mathbf{F}_i(v_i, :)\|^2$. By minimizing it, the learned node embeddings become similar for the connected nodes in network G_i. Combining with the Frobenius norm, i.e., $\|\mathbf{F}_i\|_F^2$, directly optimizing C2 will, in consequence, deliver a trivial solution for $\mathbf{F}_i$. Namely, the term C2 only satisfies the *high order connectivity* condition in *total connectivity information* but does not conform the *homophily assumption* since minimizing C2 is equivalent to factorizing a zero matrix.

FASCINATE [7] avoids the trivial solution issue by introducing C1, which can significantly improve the performance when the observed cross-layer dependency matrix (i.e., $\mathbf{L}^o_{i,j}$) is relatively dense. Nonetheless, under circumstance that very limited observed cross-layer dependencies are available, i.e., the *sparsity challenge*, the solution matrices $\mathbf{F}_i$ will be primarily determined by C2 and thus tend to become trivial.

Actually, the C2 term can be replaced with a variety of homogeneous network embedding approaches based on collaborative filtering. For example, in collective matrix factorization (CMF) [43], C2 $= \|\mathbf{F}_i\mathbf{F}_i^\top - \mathbf{A}_i\|_F^2$. For other methods [9, 62], C2 is modified according to different network embedding algorithms, e.g., Deepwalk [40] or LINE [44]. Table 2 summarizes the formats of C2 for different methods. It is straightforward to conduct a similar analysis on other CF-based methods and then we introduce Claim 2 as follows:

Claim 2. *In the existing CF-based methods, minimizing the within-layer connectivity term (i.e., C2 in Eq. (2)) w.r.t. the i-th layer of network is equivalent to directly factorizing some specific matrix, and it cannot capture* ***total connectivity information*** *of G_i, under the* sparsity challenge.

To be specific, for CMF, the claim obviously holds because minimizing $\|\mathbf{F}_i\mathbf{F}_i' - \mathbf{A}_i\|_F^2$ is equivalent to factorizing the within-layer adjacency matrix (i.e., $\mathbf{A}_i$), which does not satisfy the *high order connectivity* condition. For network embedding related methods, according to [41], a variety of network embedding algorithms can be unified in the format of matrix factorization. Given the network of the i-th layer (i.e., G_i), DeepWalk encodes the within-layer connectivity as, $\log(\text{vol}(\mathbf{A}_i)(\frac{1}{T}\sum_{s=1}^{T}(\mathbf{D}_i^{-1}\mathbf{A}_i)^s)\mathbf{D}_i^{-1})$-$\log b$, where T is the context window length, b is the number of negative samples in skip-gram [35] and $\text{vol}(\mathbf{A}_i) = \sum_{u_i,v_i}\mathbf{A}_i(u_i,v_i)$ is defined as the volume of network G_i. In addition, for LINE-based methods, the within layer connectivity can be written as, $\log(\text{vol}(G_i)\mathbf{D}_i^{-1}\mathbf{A}_i\mathbf{D}_i^{-1}) - \log b$. In Table 2, we summarize the corresponding matrices on which factorization is applied for different methods, given the network of the i-th layer (i.e., G_i).

Table 2: Matrices factorized by different choices of within-layer connectivity term for G_i.

Methods	Matrices
FASCINATE	$\mathbf{0}$
CMF	adjacency matrix $\mathbf{A}_i$
DeepWalk	$\log(\text{vol}(\mathbf{A}_i)(\frac{1}{T}\sum_{s=1}^{T}(\mathbf{D}_i^{-1}\mathbf{A}_i)^s)\mathbf{D}_i^{-1})$-$\log b$
LINE	$\log(\text{vol}(\mathbf{A}_i)\mathbf{D}_i^{-1}\mathbf{A}_i\mathbf{D}_i^{-1}) - \log b$

For methods following LINE to capture within-layer connectivity, they are similar to CMF and can only extract the information from one-hop neighboring nodes, which is not in accordance with the *high order connectivity* condition in Claim 2.

For DeepWalk related methods, we give a brief discussion from the perspective of the walk length, i.e., T. When the context window length $T = 1$, the matrix is exactly the same as that in LINE. When T is small, the *high order connectivity* condition is not satisfied. For a large walk length, we have the following lemma,

Lemma 1. *For an undirected connected network G_i, if $T \to \infty$, $(\frac{1}{T}\sum_{s=1}^{T}(\mathbf{D}_i^{-1}\mathbf{A}_i)^s) \to \mathbf{M}_i$, where $\mathbf{M}_i$ is a matrix where all rows are the identical eigenvectors corresponding to the eigenvalue equaling* 1 *associated with the matrix $(\mathbf{D}_i^{-1}\mathbf{A}_i)^\top$.*

The proof of Lemma 1 is attached in Appendix due to the page limit. It indicates that DeepWalk-based methods will hamper the homophily assumption with a large T, because for each column of $\mathbf{M}_i$, i.e., $\mathbf{M}_i(:, u_j)$, the entries are identical values. Therefore, DeepWalk-based methods cannot strike a balance between capturing the high order connectivity (i.e., large walk length T) and satisfying the homophily assumption (i.e., small T).

Remarks. The violation of the homophily assumption appears to be similar to the over-smoothing issue [31] discovered in GNN [29], however, they are actually separate problems. To be specific, the over-smoothing issue in GNN is designed to directly learn the final node embeddings, whereas our analysis focuses on the constructed matrix $\mathbf{M}_i$, which will be further factorized to obtain the final node representations. One typical characteristic of the over-smoothing issue in GNN, is that the final embeddings tend to become identical for all nodes. For DeepWalk-based methods, when T approaches infinity, the entry values in a specific column of $\mathbf{M}_i$ are the same, nevertheless, the column-wise values are different. Therefore, the obtained node embeddings after factorization remain distinct.

To summarize, following the *module coupling* approach, existing CF-based methods optimize the within-layer connectivity and the cross-layer dependency in a coupling fashion while regarding the within-layer connectivity information as a regularization term. However, when $\mathbf{L}^o_{i,j}$ degenerates into a sparse matrix (i.e., the *sparsity challenge*), the regularization term representing within-layer connectivity becomes the dominant term in the objective function. Therefore, it cannot capture the *total connectivity information* of G_i, falling short of striking a good balance between propagating critical information (i.e., *high order connectivity*) and differentiating adjacent nodes from distant ones (i.e., *homophily assumption*).

4 METHOD

In this section, we introduce our proposed method, including *First-Inner-Then-Outer-Static* (FITO-S) to resolve the *sparsity challenge* in the static setting and FITO-D to tackle the *dynamic challenge*. Armed with the theoretical analysis in Section 3, the key idea of our proposed model is to pursue a *decoupling principle* in both the *data ingestion stage* and the *module design stage*. To be specific, in the *data ingestion stage*, following a similar procedure in CF-based methods, we separate the within-layer connectivity from the cross-layer dependency. In the *module design* stage, instead of simultaneously exploiting the within-layer connectivity and the cross-layer dependency, we first focus on the within-layer connectivity and design a random walk with restart (RWR) position embedding module to capture the *total connectivity information* (i.e., First-Inner). Then, we leverage a space transformation function to translate the node embeddings from different layers to the same latent space (i.e., Then-Outer). An add-on benefit of this *decoupling* design is that it naturally leads to an efficient and effective model FITO-D in the dynamic setting, which can separately update the RWR position embedding matrix and fine-tune the space transformation functions.

4.1 FITO-S

In this subsection, we present our proposed model FITO-S to resolve the *sparsity challenge* in the static setting. According to the analysis on the CF-based algorithms, we first introduce the proposed within-layer RWR position embedding module whose goal is to capture the *total connectivity information* (i.e., Definition 2).

A - RWR Position Embedding Module. Regarding within-layer connectivity, we have demonstrated that for existing CF-based methods, matrices associated with C2 in Eq. (2) are unable to capture the *total connectivity information* due to the *coupling principle*. Alternatively, *random walk with restart* (RWR) [45], a popular algorithm to measure the node proximity, manages to preserve the connectivity information, and we have the following claim:

Claim 3. *Random walk with restart (RWR) matrix is capable of capturing the total connectivity information.*

To prove the correctness of the claim, we first present the mathematical details of the information propagation mechanism of RWR in the following equation,

$$\mathbf{r}_{u_i} = c\hat{\mathbf{A}}_i^\top \mathbf{r}_{u_i} + (1-c)\mathbf{e}_{u_i} \tag{3}$$

where $\hat{\mathbf{A}}_i = \mathbf{D}_i^{-1}\mathbf{A}_i$ is the row-normalized adjacency matrix of G_i, $\mathbf{r}_{u_i}$ denotes the node proximity vector for node u_i, $1-c$ is the restart probability and $\mathbf{e}_{u_i}$ represents a one-hot starting vector with the u_i-th element equal to 1.

From Eq. 3, we can derive the closed-form solution of the RWR matrix, $\mathbf{R}_i = (1-c)(\mathbf{I} - c\hat{\mathbf{A}}_i^\top)^{-1}$ where $\mathbf{I}$ is the identity matrix. The corresponding Taylor series can be calculated as $\mathbf{R}_i = (1-c)\sum_{s=0}^{\infty}(c\hat{\mathbf{A}}_i^\top)^s\mathbf{I}$. The column of $\mathbf{R}_i$, (e.g., $\mathbf{R}_i(:,u_i)$) represents the RWR vector $\mathbf{r}_{u_i}$ of node u_i, $\mathbf{r}_{u_i} = (1-c)\sum_{s=0}^{\infty}(c\hat{\mathbf{A}}_i^\top)^s\mathbf{e}_{u_i}$.

Comparing $\mathbf{R}_i$ with the matrices in Table 2, we can see that the RWR matrix $\mathbf{R}_i$ can indeed capture the *total connectivity information*. For the *high order connectivity* condition, $\mathbf{R}_i$ contains the terms when s is large and approaches infinity, such that the high-order information can be captured. For the *homophily assumption* condition, the decaying factor c along with the walk length s ensures that a close node pair has a larger value than a distant node pair in the position matrix $\mathbf{R}_i$. Therefore, Claim 3 holds.

Actually, the u_i-th row of $\mathbf{R}_i$ (i.e., $\mathbf{R}_i(u_i,:)$) measures the relevant position of node u_i from all nodes in network G_i. n_i is the number of nodes in network G_i, which implies that the size of matrix $\mathbf{R}_i$ may be too large thereby decreasing the model efficiency. To address this issue, we identify k nodes with the top-k highest PageRank scores [36] in G_i as the node set $\mathcal{P}_i$.[5] Note that other sampling methods to construct $\mathcal{P}_i$ (e.g., random selection) will also be evaluated in the experimental section. The RWR vectors of all selected nodes in $\mathcal{P}_i$ form the RWR position embedding matrix $\mathbf{R}_{\mathcal{P}_i} \in \mathbb{R}^{n_i \times k}$, where k is the dimension of RWR position embedding vector (i.e., $\mathbf{R}_{\mathcal{P}_i}(u_i,:)$) for node u_i. After the *First-Inner* step, we get RWR position embedding matrices for networks of all layers-$\{\mathbf{R}_{\mathcal{P}_1},\ldots,\mathbf{R}_{\mathcal{P}_g}\}$ from the multi-layered networks.

B - Space Transformation. The RWR position embedding matrix $\mathbf{R}_{\mathcal{P}_i}$ represents a relevant position embedding space w.r.t. $\mathcal{P}_i$ of network G_i, it is crucial to transform the embedding matrices of different layers to one unified space as follows:

$$\mathbf{F}_i = \text{transform}_i(\mathbf{R}_{\mathcal{P}_i}) \tag{4}$$

The space transformation can be achieved with various functions, e.g., a linear layer or a multi-layer perceptron (MLP) [19].

To guide the model training, we adopt a ranking loss which is similar to Bayesian personalized ranking (BPR) [42]:

$$\mathcal{J}_s = \sum_{i=1}^{g}\sum_{j=1}^{g}\frac{1}{|\mathcal{N}_{\{\mathbf{L}^o_{i,j}(u_i,v_j)=1\}}|}\max\{0, \gamma + d(u_i,v_j) - d(u_i',v_j')\} \tag{5}$$

where $d(u_i,v_j) = \|\mathbf{F}_i(u_i,:) - \mathbf{F}_j(v_j,:)\|_1$ denotes the $L1$ norm distance between node embeddings $\mathbf{F}_i(u_i,:)$ and $\mathbf{F}_j(v_j,:)$, γ is the margin parameter and $\mathcal{N}_{\{\mathbf{L}^o_{i,j}(u_i,v_j)=1\}}$ represents the set of negative samples for the observed dependency (i.e., $\mathbf{L}^o_{i,j}(u_i,v_j) = 1$). For an observed cross-layer dependency (u_i,v_j), the negative sample (u_i',v_j') is obtained by randomly selecting a node from either G_i or G_j to replace either u_i or v_j. The cross layer dependency inference matrix $\mathbf{L}_{i,j}$ can be constructed as follows,

$$\mathbf{L}_{i,j}(u_i,v_j) = e^{-d(u_i,v_j)} \tag{6}$$

To summarize, FITO-S effectively resolves the *sparsity challenge* from the following aspects, including (1) *inner part*: capturing the total connectivity information with RWR position embedding matrix; and (2) *outer part*: unifying the obtained embeddings from all layers with a space transformation function with a ranking loss.

4.2 FITO-D

The *decoupling* idea to resolve the *sparsity challenge* further motivates predicting cross-layer dependencies by tracking the complex and evolving multi-layered network systems. In this subsection, we leverage the *First-Inner-Then-Outer* approach to tackle the *dynamic challenge.* Starting from the inner layer, after the set of RWR position embedding matrices (i.e., $\{\mathbf{R}^{new}_{\mathcal{P}_1},\ldots,\mathbf{R}^{new}_{\mathcal{P}_g}\}$) of the new multi-layered network (i.e., $\Gamma + \Delta\Gamma$) is updated, we fine-tune outer space transformation functions of different layers with the newly observed dependencies.

For the i-th layer network G_i, its dynamic change includes edge/node deletion and insertion, which are represented in the perturbation matrix (i.e., $\Delta\mathbf{A}_i$). Note that if the number of nodes in the network changes due to node deletion/insertion, we can zero out the corresponding row/column or insert row/column, respectively.

We refer to Offset Score Propagation (OSP) algorithm [57] for solving the dynamic random walk with restart in a single graph. Assuming node u_i in network G_i is a selected node with high PageRank score in $\mathcal{P}_i$, we use $\mathbf{r}_{\text{old}}$ and $\mathbf{r}_{\text{new}}$ to denote the RWR vectors of node u_i in $\mathbf{A}_i$ and $\mathbf{B}_i = \mathbf{A}_i + \Delta\mathbf{A}_i$ for brevity. We represent $\Delta\hat{\mathbf{A}}_i = \hat{\mathbf{B}}_i - \hat{\mathbf{A}}_i$ as the difference between the new row-normalized adjacency matrix $\hat{\mathbf{B}}_i$ and the old row-normalized adjacency matrix $\hat{\mathbf{A}}_i$. Note that $\Delta\hat{\mathbf{A}}_i$ is not the row-normalized matrix of $\Delta\mathbf{A}_i$. The new RWR vector $\mathbf{r}_{\text{new}}$ can be calculated as,

$$\mathbf{q}_{\text{offset}} = c\Delta\hat{\mathbf{A}}_i^\top\mathbf{r}_{\text{old}}, \quad \mathbf{x}^{(s)}_{\text{offset}} = (c\hat{\mathbf{B}}_i^\top)^s\mathbf{q}_{\text{offset}} \tag{7}$$

$$\mathbf{r}_{\text{offset}} = \sum_{s=0}^{\infty}\mathbf{x}^{(s)}_{\text{offset}}, \quad \mathbf{r}_{\text{new}} = \mathbf{r}_{\text{old}} + \mathbf{r}_{\text{offset}} \tag{8}$$

[5] if $k > n_i$, we set $k = n_i$.

Theorem 1. *(EXACTNESS) [57]* $\mathbf{r}_{\text{new}}$ *is exact the RWR score vector of node* u_i *in the updated network* $\mathbf{B}_i = \mathbf{A}_i + \Delta\mathbf{A}_i$.

Proof. See the original paper [57]. □

Here, we give a proposition about the necessary iterations for convergence by re-running the RWR algorithm on the updated network $\mathbf{B}_i$ and the computation of $\mathbf{r}_{\text{new}}$.

Proposition 1. *Given the convergence tolerance* ϵ *in L1 norm, the number of iterations that is necessary for* $\mathbf{r}_{u_i}$ *and* $\mathbf{r}_{\text{offset}}$ *to converge are* $\log_c(\frac{\epsilon}{1-c})$ *and* $\log_c(\frac{\epsilon}{\|\mathbf{q}_{\text{offset}}\|_1})$*, respectively.*

Proof. Since $\hat{\mathbf{A}}_i$ and $\hat{\mathbf{B}}_i$ are both row normalized stochas matrices, $\|\hat{\mathbf{A}}_i^\top\|_1 = 1$ and $\|\hat{\mathbf{B}}_i^\top\|_1 = 1$. So, the convergence it ation number for $\mathbf{r}_{u_i}$ can be calculated by $\|(c\hat{\mathbf{A}}_i^\top)^s(1-c)\mathbf{e}_{u_i}\|_1$ $c^s\|(1-c)\mathbf{e}_{u_i}\|_1 \le \epsilon$. We can get the number of iterations (i.e., for $\mathbf{r}_{u_i}$ is $\log_c(\frac{\epsilon}{1-c})$. Similarly, the number of iterations (i.e., s) f $\mathbf{r}_{\text{offset}}$ is $\log_c(\frac{\epsilon}{\|\mathbf{q}_{\text{offset}}\|_1})$.

Having completed the update of the inner part, we further inve gate the newly observed cross-layer dependencies in the outer pa According to the space transformation function trained in FITO we adapt the transformation function to the dynamic systems v fine-tuning with few newly observed cross-layer dependencies. T fine-tuning can be efficiently achieved because the newly observ dependency matrix (i.e., $\Delta\mathbf{L}_{i,j}^o$) is very sparse considering that nu ber of perturbations is limited within a short period of time. V fine-tune the parameters using the following loss function with a few epochs,

$$\mathcal{J}_d = \sum_{i=1}^{g}\sum_{j=1}^{g}\frac{1}{|\mathcal{N}_{\{\Delta\mathbf{L}_{i,j}^o(u_i,v_j)=1\}}|}\max\{0, \gamma + d(u_i,v_j) - d(u_i',v_j')\} \tag{9}$$

Note that the difference between $\mathcal{J}_d$ and $\mathcal{J}_s$ in Eq. (5) is the training set, i.e., $\Delta\mathbf{L}_{i,j}^o$ and $\mathbf{L}_{i,j}^o$, respectively.

4.3 Complexity Analysis

For FITO-S, we adopt one linear layer as the space transformation function in our experiment. The time complexity of FITO-S with vanilla implementation is composed of three parts: (1) $\mathcal{P}_i$ generated by PageRank for each layer network G_i. The corresponding time complexity is $O(s\text{vol}(\mathbf{A}_i) + n_i\log(k))$, where $\text{vol}(\mathbf{A}_i)$ is the number of edges in G_i, s is the number of iterations in PageRank and $O(n_i\log(k))$ is the time for picking top-k nodes with largest PageRank scores; (2) RWR for each node in $\mathcal{P}_i$ for each layer network G_i, the time complexity is $O(ks\cdot\text{vol}(\mathbf{A}_i))$; and (3) the space transformation function training for each dependency matrix $\mathbf{L}_{i,j}^o$. For this part, time is mainly consumed in negative sampling. It is related to the size of training set (i.e., $\text{vol}(\mathbf{L}_{i,j}^o)$), multiplied by the number of negative samples n_{neg} for each observed dependency.

For FITO-D, it can achieve better performance in terms of efficiency than FITO-S from the following three aspects, including: (1) time saved in updating RWR position embedding matrix, as stated in Proposition 1, (2) a small number of epochs for fine-tuning and (3) the sparse newly observed dependency matrix (i.e., $\Delta\mathbf{L}_{i,j}^o$ is even sparser than $\mathbf{L}_{i,j}^o$), which requires less time for obtaining negative samples.

5 EXPERIMENT

We evaluate the proposed FITO in the following aspects:

- How effective is our proposed FITO-S handling the *sparsity challenge* compared with the existing methods?
- How sensitive is the proposed FITO-S w.r.t. model components and parameters?
- When the multi-layered network Γ evolves, how can FITO-D resolve the *dynamic challenge* effectively and efficiently?

5.1 Experimental Setup

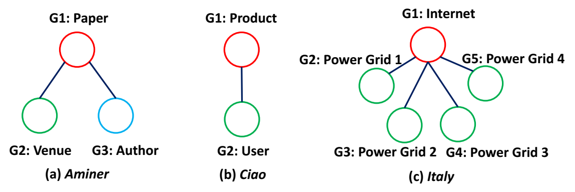


Figure 2: The abstract dependency structures.

5.1.1 *Datasets.* Our method is evaluated mainly on three datasets. The dataset statistics are summarized in Table 3 and the abstract layer-layer dependency graphs are shown in Figure 2. The detailed descriptions of datasets are presented in Appendix.

5.1.2 *Scenarios.* We have two settings in our experiments, including the static setting and the dynamic setting. In the static setting, to reflect the *sparsity challenge*, we set the observed dependency matrices, $\mathbf{L}_{i,j}^o$, to be extremely sparse (i.e., with a very small training ratio). Taking the ***Aminer*** dataset as an example, if we set the training ratio as 0.01, it means that we only have 1% observed dependencies for both *Paper-Venue* (G_1: G_2) and *Paper-Author* (G_1: G_3). For G_1: G_2, we randomly select 79 dependencies from 7,925 groundtruth dependencies. For G_1: G_3, the prior dependency number is 93 out of 9,299 groundtruth dependencies. We have five scenarios for the static task, including (1) ***Aminer-0.01*** (2) ***Aminer-0.05*** (3) ***Ciao-0.005*** (4) ***Ciao-0.01*** and (5) ***Italy-0.1***, where {0.01, 0.05, 0.005, 0.01, 0.1} are different training ratios. In the dynamic setting, we set five steps for the ***Aminer*** dataset. In step 0, we randomly reserve 90% within-layer edges in each layer and set the training ratio as 0.025. Then, at each step, we add 2% within-layer edges in each layer network and 0.5% of the groundtruth cross-layer dependencies. For example, at step 2, we have 94% within-layer edges with a 0.035 training ratio and the test set has the remaining 96.5% groundtruth cross-layer dependencies. Notice that at step 5, the setting is same as ***Aminer-0.05***. FITO-S is run at step 0 as the starting model for FITO-D at step 1. Then, FITO-D at each step fine-tunes model parameters based on previous step's FITO-D.

5.1.3 *Baselines & Metrics.* For the static task, we select seven representative baselines from two main categories of existing methods: (1) hin2vec [17], GATNE [4] and RHINE [34] are metapath-based, GNN-based and knowledge graph embedding based HNE methods respectively; and (2) CMF [43], wpZAN [56] and FASINATE [7] are CF-based methods. node2vec [21] functions as homogeneous

Table 4: Performance on Cross-Layer Dependency Inference for *Aminer* and *Ciao* (Higher is better).

Dataset	***Aminer-0.01***				***Aminer-0.05***				***Ciao-0.005***		***Ciao-0.01***	
Layer pair	G_1: G_2		G_1: G_3		G_1: G_2		G_1: G_3		G_1: G_2		G_1: G_2	
Metrics	Hit@5	Hit@10	Hit@50	Hit@100	Hit@5	Hit@10	Hit@50	Hit@100	Hit@50	Hit@100	Hit@50	Hit@100
RHINE	0.36%	0.70%	0.90%	1.64%	0.31%	0.69%	0.61%	1.23%	1.00%	2.02%	1.12%	2.31%
GATNE	0.61%	1.55%	0.32%	0.76%	0.60%	0.96%	0.42%	0.94%	0.94%	1.67%	3.05%	3.80%
hin2vec	9.14%	14.49%	0.34%	0.63%	18.36%	28.14%	2.89%	3.83%	0.61%	1.31%	0.89%	1.70%
node2vec	12.15%	17.58%	3.97%	5.98%	20.22%	28.57%	**14.79%**	20.15%	2.94%	5.20%	4.10%	6.45%
wpZAN	5.33%	11.45%	4.51%	7.65%	16.07%	25.21%	14.11%	19.87%	1.48%	3.32%	3.29%	5.94%
CMF	7.18%	11.93%	5.54%	7.76%	17.63%	24.52%	12.64%	17.15%	3.99%	6.74%	5.09%	8.45%
FASINATE	6.67%	12.73%	4.84%	7.73%	17.33%	26.60%	14.17%	19.77%	1.42%	3.21%	3.31%	5.85%
FITO-S	**17.05%**	**24.40%**	**7.57%**	**11.90%**	**30.90%**	**40.44%**	13.63%	**20.37%**	**5.52%**	**9.79%**	**6.07%**	**11.25%**

Table 5: Performance for *Italy* (Higher is better).

Dataset	***Italy-0.1***							
Layer pair	G_1:G_2		G_1: G_3		G_1: G_4		G_1: G_5	
Metrics	Hit@1	Hit@5	Hit@1	Hit@5	Hit@1	Hit@5	Hit@1	Hit@5
RHINE	0.00%	**17.69%**	0.00%	0.00%	**20.00%**	34.67%	0.00%	0.00%
GATNE	0.00%	4.69%	0.00%	18.39%	0.00%	5.33%	0.00%	5.45%
hin2vec	1.44%	7.94%	3.45%	12.64%	2.67%	16.00%	3.64%	7.27%
node2vec	2.17%	11.55%	4.60%	24.14%	4.00%	36.00%	3.64%	20.00%
wpZAN	0.00%	4.33%	1.15%	19.54%	4.00%	36.00%	5.45%	12.73%
CMF	0.00%	6.50%	1.15%	21.84%	5.33%	36.00%	5.45%	18.18%
FASINATE	0.00%	3.97%	2.30%	19.54%	4.00%	36.00%	5.45%	20.00%
FITO-S	**2.52%**	13.72%	**17.24%**	**41.37%**	8.00%	**38.66%**	**16.36%**	**30.90%**

network embedding baseline for comparison. The metrics is Hit@K. We select different Ks according to sizes of different layers.

5.1.4 Implementation Details. See the Appendix due to the page limit.

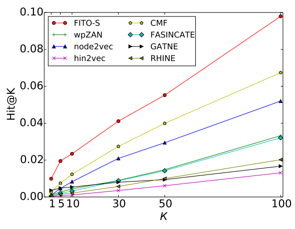
5.2 Effectiveness of FITO-S in Static Setting

The results of FITO-S and different methods are presented in Tables 4 and 5. We can see that FITO-S achieves the best performance in almost all scenarios. For example, in ***Aminer-0.01***, FITO-S has a 5% advantage in Hit@5 on G_1: G_2 than the best baseline (i.e., node2vec). In the second dataset ***Ciao***, CMF has the best performance among all baselines. FITO-S has a 3% improvement over CMF in Hit@100 in both scenarios. For the smallest dataset ***Italy***, we use the stricter Hit@1 and Hit@5. FITO-S's performance is at least 3 times better than the best baseline in Hit@1 on both G_1: G_3 and G_1: G_5. Among baselines, RHINE achieves good performance in Hit@5 on G_1: G_2 and Hit@1 on G_1: G_4. This is because RHINE uses all information from G_1 to G_5 to predict the cross-layer dependency for these two graph pairs (G_1: G_2 and G_1: G_4), which sacrifices the performance for other graph pairs (0 Hit@5 for G_1: G_3 and G_1: G_5).

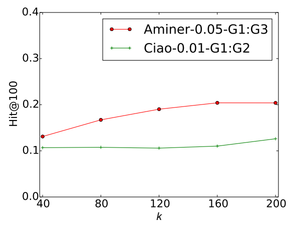
These results are consistent with the theoretical analysis in Section 3. For HNE-based methods (i.e., RHINE, GATNE and hin2vec), the experiments show that this category of methods have the lowest performance compared with other baselines, including the homogeneous network embedding baseline node2vec. The reason is that these methods embrace the *coupling principle* in the *data ingestion* stage, which leads to the inability of generating sufficient cross-layer similar node pairs under the sparsity challenge. For CF-based methods, when the observed cross-layer dependency matrix becomes sparse, the within-layer connectivity dominates the coupled objective function, which in turn hurts their overall performance.

5.3 Parameter Study and Ablation Study

5.3.1 Hit@K for different Ks. We focus on one scenario (i.e., ***Ciao-0.005***-G_1: G_2) with different Ks in {1, 5, 10, 30, 50, 100}. The results



(a) Hit@K for *Ciao-0.005*-G_1: G_2.

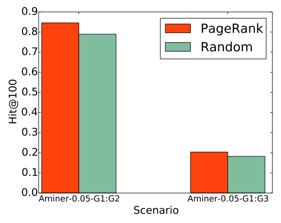


(b) Size of RWR seeds set $\mathbf{R}_{\mathcal{P}_i}$.

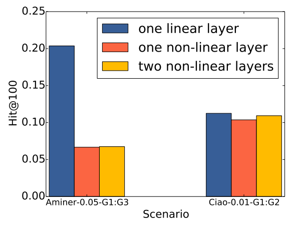
Figure 3: Parameter studies.

are shown in Figure 3 (a). From the figure we can see that for all Ks, FITO-S consistently obtains the best performance.

5.3.2 Size of RWR seed set k. Since we select nodes with top-k PageRank score as RWR seeds, we only keep k columns of the RWR matrix $\mathbf{R}_i$. Here, we study the effect of k on the performance of FITO-S on two scenarios: (1) ***Aminer-0.05***-G_1: G_3 and (2) ***Ciao-0.01***-G_1: G_2. k ranges from 40 to 200. The results (Hit@100) are shown in Figure 3 (b). We can see that Hit@100 largely remains stable with a slight increase with k.



(a) RWR seeds generation methods.



(b) Space transformation functions.

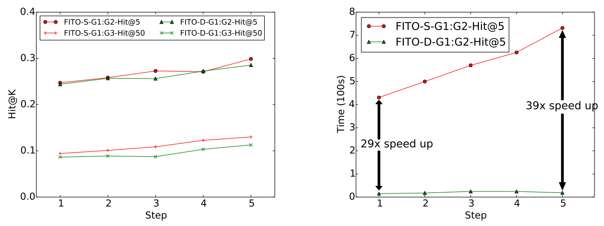
Figure 4: Ablation study for RWR seeds generation and space transformation functions in FITO-S.

5.3.3 RWR seeds generation methods. We use PageRank to generate RWR seeds for FITO-S with the intuition that nodes with highest PageRank scores dominate the entire network. In this subsection, we replace it with random selection and Figure 4 (a) demonstrates the difference between them. FITO-S with PageRank consistently has better performance than that with random selection, which shows the effectiveness of PageRank component.

5.3.4 Space transformation function. The space transformation function in FITO is one linear layer. Here we study three different forms of space transformation functions, including (1) one linear layer, (2) one non-linear layer (one linear layer plus relu activation function [20]), and (3) two non-linear layers. From Figure 4 (b) we can notice that for both datasets (i.e., ***Aminer-0.05***-G_1: G_3 and ***Ciao-0.01***-G_1: G_2), linear transformation outperforms non-linear transformation, which suggests that embedding spaces of different layers are linearly related with each other.

5.4 Effectiveness and Efficiency of FITO-D

In this subsection, we present the results of FITO-D compared with FITO-S in the dynamic setting. As shown in Figure 5 (a), FITO-D's performance is very close to that of retraining FITO-S, with less than 1% Hit@K loss at each step. However, in Figure 5 (b), FITO-S's running time increases linearly from 430s to 731s as multi-layered network evolves, due to the linearly increasing training set in each epoch. On the contrary, it takes only about 20s for FITO-D to update node embeddings and dependency inferences, which is $29-39\times$ faster than FITO-S. As analyzed in Subsection 4.3, the efficiency comes from three aspects: (1) the time saved in RWR updating process; (2) a smaller size of newly observed dependencies-$\Delta\mathbf{L}^o_{i,j}$ used in fine-tuning compared with the whole training set for FITO-S at each step. (3) a smaller training epoch number for FITO-D (200) than FITO-S (1500). To summarize, FITO-D enjoys both effectiveness and efficiency in the dynamic setting.



(a) Performances of FITO-S and FITO-D. (b) Running time of FITO-S and FITO-D.

Figure 5: FITO-D's effectiveness and efficiency.

6 RELATED WORK

Multi-layered network mining. Multi-layered networks [13] naturally exist in real-world applications and can be in a variety of types, including multi-modal networks [24], multi-dimensional networks [2], multiplex networks [1, 26, 28] and inter-dependent networks [6, 8, 39]. Enormous research efforts have been directed towards investigating the problem of multi-layered network mining. For example, [53, 65] studies the network alignment problem on multi-layered networks and [59] proposes to transform one layer network to another in the multi-layered network. [63, 64] investigates the adversarial problem in the multi-network setting. For dynamic multi-layered networks that evolve over time, [54] achieves tracking the evolution of entity linking on multiple knowledge graphs. Our work focuses on inferring the cross-layer dependencies and hence is related to multi-layered network mining.
Collaborative filtering. The cross-layer dependency inference task[11, 12] is related to one-class collaborative filtering (OCCF) problem. [25, 37] adopt matrix factorization (MF) based methods on the observed rating matrix. [32] further incorporates the user information to improve the performance in inference. In wpZAN [56], the authors regularize the OCCF with the social network and the product similarity network. Recently, neural networks are leveraged to replace the dot product in MF [23]. The relations of items are captured in RCF [52] to boost the performance in recommendation task. NGCF [50] utilizes GNN [29] to propagate the collaborative signal along the interaction in a bipartite graph. FASCINATE [7] generalizes collaborative filtering to cross-layer dependency inference on multi-layered networks.
Network representation learning. Network representation learning, which aims to effectively embed the network elements in a low-dimensional latent space, is an essential topic in network mining. For homogeneous networks, DeepWalk [40] inherits ideas from word2vec [35] to preserve local context of nodes. Node2vec [21] exploits an interpolation strategy in random walk sampling. LINE [44] incorporates both the first order and the second order proximity to learn high-quality node-level representations. More recently, neural network based approaches for graph structured data have been widely explored, including graph convolution network (GCN) [29], graph attention network (GAT) [46], etc. In addition, [14, 15, 47, 48, 60] are algorithms specially designed for on dynamic temporal graph representation learning. Regarding heterogeneous networks, metapath2vec [10] and hin2vec [17] are proposed based on metapath generation. GATNE [4] and HANE [51] are graph neural network based methods. Yang et al. summarize the recent development in heterogeneous network representation learning in a detailed survey [55]. Our work learns the node representation of multi-layered inter-dependent network.

7 CONCLUSION

In this paper, we reveal that the *coupling principle* underlying the existing cross-layer dependence inference methods bears both the *sparsity challenge* and the *dynamic challenge* in cross-layer dependency inference. In response, we start from the key insight of *decoupling* and propose a family of novel algorithms FITO. FITO-S leverages random walk with restart to capture the within-layer *total connectivity information*, followed by the space transformation functions. FITO-D further updates the RWR position embedding matrix and fine-tunes space transformation function. We perform extensive experiments to validate and verify the effectiveness and the efficiency of our models.

8 ACKNOWLEDGEMENT

This work is supported by the NSF Program on Fairness in AI in collaboration with Amazon (1939725), DARPA (HR001121C0165), NIFA (2020-67021-32799), ARO (W911NF2110088), NSF (1947135, and 2134079). The content of the information in this document does not necessarily reflect the position or the policy of the Government or Amazon, and no official endorsement should be inferred. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

REFERENCES

[1] Federico Battiston, Vincenzo Nicosia, and Vito Latora. 2014. Structural measures for multiplex networks. *Physical Review E* 89, 3 (2014), 032804.
[2] Michele Berlingerio, Michele Coscia, Fosca Giannotti, Anna Monreale, and Dino Pedreschi. 2011. Foundations of multidimensional network analysis. In *2011 international conference on advances in social networks analysis and mining*. IEEE, 485–489.

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

[4] Yukuo Cen, Xu Zou, Jianwei Zhang, Hongxia Yang, Jingren Zhou, and Jie Tang. 2019. Representation Learning for Attributed Multiplex Heterogeneous Network. In *KDD'2019*. ACM, 1358–1368.

[5] Chen Chen, Jingrui He, Nadya Bliss, and Hanghang Tong. 2015. On the connectivity of multi-layered networks: Models, measures and optimal control. In *2015 IEEE International Conference on Data Mining*. IEEE, 715–720.

[6] Chen Chen, Jingrui He, Nadya Bliss, and Hanghang Tong. 2017. Towards optimal connectivity on multi-layered networks. *IEEE transactions on knowledge and data engineering* 29, 10 (2017), 2332–2346.

[7] Chen Chen, Hanghang Tong, Lei Xie, Lei Ying, and Qing He. 2016. FASCINATE: fast cross-layer dependency inference on multi-layered networks. In *KDD'2016*. 765–774.

[8] Chen Chen, Hanghang Tong, Lei Xie, Lei Ying, and Qing He. 2017. Cross-dependency inference in multi-layered networks: A collaborative filtering perspective. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 11, 4 (2017), 1–26.

[9] Xiaokai Chu, Xinxin Fan, Di Yao, Zhihua Zhu, Jianhui Huang, and Jingping Bi. 2019. Cross-network embedding for multi-network alignment. In *The world wide web conference*. 273–284.

[10] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *KDD'2017*. 135–144.

[11] Boxin Du, Lihui Liu, and Hanghang Tong. 2021. Sylvester Tensor Equation for Multi-Way Association. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 311–321.

[12] Boxin Du and Hanghang Tong. 2019. Mrmine: Multi-resolution multi-network embedding. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 479–488.

[13] Boxin Du, Si Zhang, Yuchen Yan, and Hanghang Tong. 2021. New Frontiers of Multi-Network Mining: Recent Developments and Future Trend. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 4038–4039.

[14] Dongqi Fu, Liri Fang, Ross Maciejewski, Vetle I. Torvik, and Jingrui He. 2022. Meta-Learned Metrics over Multi-Evolution Temporal Graphs. In *KDD 2022*.

[15] Dongqi Fu and Jingrui He. 2021. SDG: A Simplified and Dynamic Graph Neural Network. In *SIGIR*.

[16] Dongqi Fu, Zhe Xu, Bo Li, Hanghang Tong, and Jingrui He. 2020. A View-Adversarial Framework for Multi-View Network Embedding. In *CIKM*.

[17] Tao-yang Fu, Wang-Chien Lee, and Zhen Lei. 2017. Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 1797–1806.

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

[19] Matt W Gardner and SR Dorling. 1998. Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric environment* 32, 14-15 (1998), 2627–2636.

[20] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 315–323.

[21] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *KDD'2016*. 855–864.

[22] William L Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 1025–1035.

[23] Xiangnan He and Tat-Seng Chua. 2017. Neural factorization machines for sparse predictive analytics. In *SIGIR*. 355–364.

[24] Lenwood S. Heath and Allan A. Sioson. 2009. Multimodal Networks: Structure and Operations. *IEEE/ACM Trans. Comput. Biol. Bioinformatics* 6, 2 (April 2009), 321–332. https://doi.org/10.1109/TCBB.2007.70243

[25] Y. Hu, Y. Koren, and C. Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *2008 Eighth IEEE International Conference on Data Mining*. 263–272. https://doi.org/10.1109/ICDM.2008.22

[26] Baoyu Jing, Chanyoung Park, and Hanghang Tong. 2021. Hdmi: High-order deep multiplex infomax. In *Proceedings of the Web Conference 2021*. 2414–2424.

[27] Baoyu Jing, Yuejia Xiang, Xi Chen, Yu Chen, and Hanghang Tong. 2021. Graph-MVP: Multi-View Prototypical Contrastive Learning for Multiplex Graphs. *arXiv preprint arXiv:2109.03560* (2021).

[28] Baoyu Jing, Zeyu You, Tao Yang, Wei Fan, and Hanghang Tong. 2021. Multiplex Graph Neural Network for Extractive Text Summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 133–139.

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

[30] Thomas N. Kipf and Max Welling. 2016. Variational Graph Auto-Encoders. arXiv:stat.ML/1611.07308

[31] Qimai Li, Zhichao Han, and Xiao-Ming Wu. 2018. Deeper insights into graph convolutional networks for semi-supervised learning. In *Thirty-Second AAAI conference on artificial intelligence*.

[32] Yanen Li, Jia Hu, ChengXiang Zhai, and Ye Chen. 2010. Improving One-Class Collaborative Filtering by Incorporating Rich User Information. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM '10)*. Association for Computing Machinery, New York, NY, USA, 959–968. https://doi.org/10.1145/1871437.1871559

[33] Qiao Liu, Chen Chen, Annie Gao, Hang Hang Tong, and Lei Xie. 2017. VariFunNet, an integrated multiscale modeling framework to study the effects of rare non-coding variants in genome-wide association studies: Applied to Alzheimer's disease. In *2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, 2177–2182.

[34] Yuanfu Lu, Chuan Shi, Linmei Hu, and Zhiyuan Liu. 2019. Relation structure-aware heterogeneous information network embedding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 4456–4463.

[35] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[36] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The PageRank citation ranking: Bringing order to the web.* Technical Report. Stanford InfoLab.

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

[38] Chanyoung Park, Donghyun Kim, Jiawei Han, and Hwanjo Yu. 2020. Unsupervised attributed multiplex network embedding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 5371–5378.

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

[40] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *KDD'2014*. 701–710.

[41] Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang. 2018. Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In *WSDM*. 459–467.

[42] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618* (2012).

[43] Ajit P Singh and Geoffrey J Gordon. 2008. Relational learning via collective matrix factorization. In *KDD'2008*. 650–658.

[44] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *WWW*. 1067–1077.

[45] Hanghang Tong, Christos Faloutsos, and Jia-Yu Pan. 2006. Fast random walk with restart and its applications. In *Sixth international conference on data mining (ICDM'06)*. IEEE, 613–622.

[46] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).

[47] Ruijie Wang, Zijie Huang, Shengzhong Liu, Huajie Shao, Dongxin Liu, Jinyang Li, Tianshi Wang, Dachun Sun, Shuochao Yao, and Tarek Abdelzaher. 2021. Dydiff-vae: A dynamic variational framework for information diffusion prediction. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 163–172.

[48] Ruijie Wang, Zheng Li, Danqing Zhang, Qingyu Yin, Tong Zhao, Bing Yin, and Tarek Abdelzaher. 2022. RETE: Retrieval-Enhanced Temporal Event Forecasting on Unified Query Product Evolutionary Graph. In *Proceedings of the ACM Web Conference 2022*. 462–472.

[49] Ruijie Wang, Yuchen Yan, Jialu Wang, Yuting Jia, Ye Zhang, Weinan Zhang, and Xinbing Wang. 2018. Acekg: A large-scale knowledge graph for academic data mining. In *Proceedings of the 27th ACM international conference on information and knowledge management*. 1487–1490.

[50] Xiang Wang, Tinglin Huang, Dingxian Wang, Yancheng Yuan, Zhenguang Liu, Xiangnan He, and Tat-Seng Chua. 2021. Learning Intents behind Interactions with Knowledge Graph for Recommendation. In *Proceedings of the Web Conference 2021*. 878–887.

[51] Yueyang Wang, Ziheng Duan, Binbing Liao, Fei Wu, and Yueting Zhuang. 2019. Heterogeneous attributed network embedding with graph convolutional networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 10061–10062.

[52] Xin Xin, Xiangnan He, Yongfeng Zhang, Yongdong Zhang, and Joemon Jose. 2019. Relational collaborative filtering: Modeling multiple item relations for recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 125–134.

[53] Hao Xiong, Junchi Yan, and Li Pan. 2021. Contrastive Multi-View Multiplex Network Embedding with Applications to Robust Network Alignment. In *KDD'2021*.

1913–1923.

[54] Yuchen Yan, Lihui Liu, Yikun Ban, Baoyu Jing, and Hanghang Tong. 2021. Dynamic knowledge graph alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 4564–4572.

[55] Carl Yang, Yuxin Xiao, Yu Zhang, Yizhou Sun, and Jiawei Han. 2020. Heterogeneous network representation learning: A unified framework with survey and benchmark. *IEEE Transactions on Knowledge and Data Engineering* (2020).

[56] Yuan Yao, Hanghang Tong, Guo Yan, Feng Xu, Xiang Zhang, Boleslaw K Szymanski, and Jian Lu. 2014. Dual-regularized one-class collaborative filtering. In *CIKM*. 759–768.

[57] Minji Yoon, Woojeong Jin, and U Kang. 2018. Fast and accurate random walk with restart on dynamic graphs with guarantees. In *Proceedings of the 2018 World Wide Web Conference*. 409–418.

[58] Hongming Zhang, Liwei Qiu, Lingling Yi, and Yangqiu Song. 2018. Scalable Multiplex Network Embedding.. In *IJCAI*, Vol. 18. 3082–3088.

[59] Si Zhang, Hanghang Tong, Yinglong Xia, Liang Xiong, and Jiejun Xu. 2020. Nettrans: Neural cross-network transformation. In *KDD'2020*. 986–996.

[60] Dawei Zhou, Lecheng Zheng, Jiawei Han, and Jingrui He. 2020. A data-driven graph generative model for temporal interaction networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 401–411.

[61] Dawei Zhou, Lecheng Zheng, Jiejun Xu, and Jingrui He. 2019. Misc-GAN: A multi-scale generative model for graphs. *Frontiers in big Data* 2 (2019), 3.

[62] Fan Zhou, Lei Liu, Kunpeng Zhang, Goce Trajcevski, Jin Wu, and Ting Zhong. 2018. Deeplink: A deep learning approach for user identity linkage. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. IEEE, 1313–1321.

[63] Qinghai Zhou, Liangyue Li, Nan Cao, Lei Ying, and Hanghang Tong. 2019. ADMIRING: Adversarial multi-network mining. In *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE, 1522–1527.

[64] Qinghai Zhou, Liangyue Li, Nan Cao, Lei Ying, and Hanghang Tong. 2021. Adversarial Attacks on Multi-Network Mining: Problem Definition and Fast Solutions. *IEEE Transactions on Knowledge and Data Engineering* (2021).

[65] Qinghai Zhou, Liangyue Li, Xintao Wu, Nan Cao, Lei Ying, and Hanghang Tong. 2021. Attent: Active attributed network alignment. In *Proceedings of the Web Conference 2021*. 3896–3906.