Heterogeneous Graph Structure Learning through the Lens of Data-generating Processes

Keyue Jiang[1] **Bohan Tang**[2] **Xiaowen Dong**[2] **Laura Toni**[1]
[1]University College London, London, UK
[2]University of Oxford, Oxford, UK

Abstract

Inferring the graph structure from observed data is a key task in graph machine learning to capture the intrinsic relationship between data entities. While significant advancements have been made in learning the structure of homogeneous graphs, many real-world graphs exhibit heterogeneous patterns where nodes and edges have multiple types. This paper fills this gap by introducing the first approach for heterogeneous graph structure learning (HGSL). To this end, we first propose a novel statistical model for the data-generating process (DGP) of heterogeneous graph data, namely hidden Markov networks for heterogeneous graphs (H2MN). Then we formalize HGSL as a maximum a-posterior estimation problem parameterized by such DGP and derive an alternating optimization method to obtain a solution together with a theoretical justification of the optimization conditions. Finally, we conduct extensive experiments on both synthetic and real-world datasets to demonstrate that our proposed method excels in learning structure on heterogeneous graphs in terms of edge type identification and edge weight recovery.

1 INTRODUCTION

Graphs are a powerful and ubiquitous representation of relational data for real applications such as social networks (Salami et al., 2017), e-commerce (Wang et al., 2019) and financial transactions (Liu et al., 2018), in addition to various scientific and technological areas. However, a meaningful graph is not always readily available from the data (Dong et al., 2016), and sometimes the graph observed is not always clean (Chen et al., 2015). Learning or inferring the underlying graph structure from observed data living in the nodes is, therefore, an important problem in the field of both graph machine learning (Chami et al., 2022; Hamilton et al., 2017) and graph signal processing (Dong et al., 2019), (Dong et al., 2020; Ortega et al., 2018). In the former, e.g., graph neural networks, graph structure learning (GSL) is typically plugged in as an extra component to refine the graph topology, and empower the prediction ability of models (Zaripova et al., 2023), (Battiloro et al., 2023). In the latter, GSL assists in downstream algorithms, such as spectral clustering (Dong et al., 2016), cooperation game (Rossi et al., 2022), video prediction (Zhong et al., 2024), etc.

GSL algorithms (Dong et al., 2016; Kalofolias, 2016a; Pu et al., 2021b) assume that the features/signals on the graphs admit certain regularity or smoothness modeled by an underlying data-generating process (DGP). The common choices of the DGP for graph-structured data are Ising models (Ising, 1925), Gaussian graphical models (Yuan and Lin, 2007; Jia and Benson, 2022), and pair-wise exponential Markov random fields (PE-MRF) (Park et al., 2017), where the features are assumed to be emitted from a multivariate Gaussian distribution whose covariance matrix is uniquely determined by the graph structure. In such a scenario, the GSL problem is solved by the precision matrix estimation (Yuan and Lin, 2007; Friedman et al., 2008; Banerjee et al., 2008) from the graph features. Following this framework, various algorithms have proposed to inject structural prior (Pu et al., 2021a), guarantee graph connectivity (Dong et al., 2016; Kalofolias, 2016b), or ensure convergence through relaxed optimization (Egilmez et al., 2017).

The aforementioned methods are limited to homogeneous GSL tasks where nodes and edges in the graph belong to a single type, preventing the extension to heterogeneous graphs. Real-world graphs often exhibit heterogeneous patterns (Wang et al., 2020), with

Proceedings of the 28[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2025, Mai Khao, Thailand. PMLR: Volume 258.

multiple types of nodes and edges representing different kinds of entities and relationships. For instance, in a network representing a recommender system (Fu et al., 2020), nodes can have distinct types (e.g. users and items), and edges can represent different types of relations (e.g. like/dislike for user-item edges or following/being followed for user-user edges). Other examples include social networks, academic networks (Lv et al., 2021; Gao et al., 2009), and knowledge graphs (Bollacker et al., 2008; Dettmers et al., 2018). Effectively inferring the structure of heterogeneous graphs from node observations is an important challenge as for unsupervised discovery of complex relations within these social systems, but one that remains under-explored in the graph learning literature.

In this work, we aim to address this challenge and solve the problem of *heterogeneous* graph structure learning (HGSL), a new formulation that applies to a wide range of graphs, including bipartite, multi-relational, and knowledge graphs. Our contributions can be summarized as follows.

• **Problem formulation.** We formulate the HGSL problem as a maximum a-posterior (MAP) estimation of the adjacency tensor that captures the structure of heterogeneous graphs with different node/edge types.

• **Novel data-generating process.** Solving the HGSL problem requires a proper design of the DGP for heterogeneous graphs. We develop a novel DGP based on hidden Markov networks (Ghahramani, 2001) for the heterogeneous graph data, namely hidden Markov networks for heterogeneous graphs (H2MN).

• **Algorithm design.** Jointly estimating model parameters and graph structure is challenging as it is over-parameterized and leads to a non-convex optimization problem. Thus, we develop an effective algorithm to solve the HGSL problem and provide a theoretical justification for the optimization conditions.

• **Empirical study.** Extensive experiments on both synthetic and real-world datasets were conducted to test the efficacy of our algorithm. The experimental results demonstrate that our proposed method consistently excels in structure learning tasks in terms of both edge type identification and edge weight recovery. Furthermore, we give illustrative examples of learned heterogeneous graphs in real-world network systems.

2 PRELIMINARIES AND PROBLEM FORMULATION

2.1 Heterogeneous Graphs

We now introduce the main notation needed in the paper to formalize the HGSL problem. Let $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ denote an undirected graph with edges $\mathcal{E}$ and nodes $\mathcal{V}$. In the context of the heterogeneous graph (HG), each node $v \in \mathcal{V}$ is characterized by a node type $\phi(v) \in \mathcal{A}$ and each edge is assigned with a relation type $r \in \mathcal{R}$, where $\mathcal{A}$ and $\mathcal{R}$ are predefined node and edge type sets. Each edge in the HG is represented by a triplet $\{v, u, r\}$, which means nodes v and u are connected by relation type r. This formalizes a 3-D weighted tensor, $\mathbf{W} = \{w_{vur}\} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|\times|\mathcal{R}|}$, with $|\cdot|$ the set cardinality. For each node pair, $\mathbf{W}_{vu:}$ is a length-$|\mathcal{R}|$ vector with only one nonzero entry indicating the weight and type for the edge connecting u and v. We assume that the type of edge is determined by the types of connected nodes, i.e., if node u is type 'actor' and v is 'movie', the relation type r can only be in the subset of {'star in', 'support in'}, denoted as $\mathcal{R}_{\phi(v),\phi(u)}$. This is a conventional assumption in the literature (Lv et al., 2021; Fu et al., 2020; Guo et al., 2023; Hu et al., 2020a; Zhang and Chen, 2020) and it reduces the degree of freedom in determining relation types.

Furthermore, each node v is associated with a series of *node features* that can either be *signals* or *labels* for the classes. The observable features are different depending on the datasets. For example, in the movie recommendation datasets (Harper and Konstan, 2016), the node features only contain labels for movie genres. In the city traffic network (Barlacchi et al., 2015; Gallotti and Barthelemy, 2015), no node label is observed while the traffic volume is viewed as the signal.

The node signals. The node signals can be represented by a function $f : \mathcal{V} \to \mathbb{R}^K$, which assigns a vector $\boldsymbol{x}_v \in \mathbb{R}^K$ to node $v \in \mathcal{V}$. Some datasets (Lv et al., 2021; Traud et al., 2011; Guo et al., 2023) have a type-specific dimension for each node type. This study will focus on the cases with unified dimension sizes but the proposed method can be generalized to any heterogeneous graph by projecting the signals into a universal $\mathbb{R}^K$ with extra linear modules. Note that we do not consider edge signals encoding edge attributes and leave it for future work.

The node labels. Within each node type, a node could contain a class label. For instance, node v can have types "actor" and "movie" in a movie review graph, denoted by $\phi(v)$. Within each type, nodes are classified into genres "adventure" or "action", denoted by $\boldsymbol{y}_v$. The label is represented by a one-hot vector $\boldsymbol{y}_v \in \mathbb{R}^{C_{\phi(v)}}$ where $C_{\phi(v)}$ is the number of classes.

The relation-wise connectivity matrix. For each r, we define a probability matrix $\boldsymbol{B}_r \in \mathbb{R}^{|C_{\phi(v)}|\times|C_{\phi(u)}|}$, where each entry of the matrix encodes the probability of two nodes belonging to specific classes to be connected with r. Conceptually, $\boldsymbol{B}_r[p, q] = P(w_{uvr} = 1 \mid \boldsymbol{y}_{u,p} = \boldsymbol{y}_{v,q} = 1)$ with $\boldsymbol{y}_{u,p}$ the p-th entry of the vector

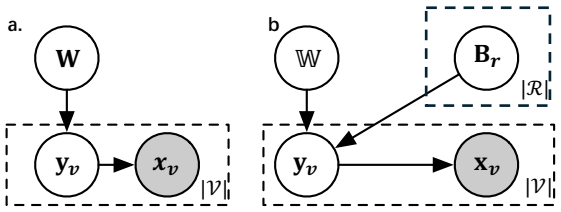


Figure 1: The graphical models for a) HMN and b) our H2MN. The shadowed variable is observable.

$\boldsymbol{y}_u$. Following the example of the movie review dataset, actors and movies in the same genre are more likely (with high probability) to be connected by a "star in" edge type. To ensure $\boldsymbol{B}_r$ formalizes a valid probability matrix, the sum of the entries is, $\|\boldsymbol{B}_r\|_1 = 1$.

2.2 Problem Formulation

This paper focuses on learning heterogeneous graph structure from features living on nodes, together with the optimal parameters as a bi-product. Denoting the extra parameters for modeling the DGP as Θ, we can formulate it as follows: Given nodes $\{v\}_{v\in\mathcal{V}}$ with corresponding type $\{\phi(v)\}_{v\in\mathcal{V}}$ and associated signals[1] $\{\boldsymbol{x}_v\}_{v\in\mathcal{V}}$, together with a potential relation type set $\mathcal{R}$, we aim to learn a weighted and undirected heterogeneous graph $\mathcal{G}$ represented by tensor $\mathbf{W}$ that encodes the weights and types of edges. Mathematically, this can be viewed as a MAP estimation problem,

$$\begin{aligned}\mathbf{W}^*,\Theta^* &= \arg\max_{\mathbf{W},\Theta} \log P(\mathbf{W},\Theta \mid \{\boldsymbol{x}_v\}) \\ &= \arg\min_{\mathbf{W},\Theta} -\log P(\{\boldsymbol{x}_v\} \mid \mathbf{W},\Theta) + \Omega(\mathbf{W}) + \Omega(\Theta),\end{aligned} \tag{1}$$

where $\Omega(\mathbf{W})$ and $\Omega(\Theta)$ are the negative-log priors.

3 CONNECTING GENERATING PROCESS AND GRAPH STRUCTURE LEARNING

3.1 Hidden Markov Networks of Graph Features

In this section, we introduce the usage of DGP, specifically hidden Markov networks (HMN) (Ghahramani, 2001), to model homogeneous graphs, where $|\mathcal{R}| = 1$ and $|\mathcal{A}| = 1$ so that $\mathbb{W}$ reduces to a matrix $\boldsymbol{W} = \{w_{uv}\}_{u,v\in\{1:|\mathcal{V}|\}} \in \mathbb{R}_+^{|\mathcal{V}|\times|\mathcal{V}|}$ and $\boldsymbol{Y}$ has a unified dimension C. The graph Laplacian matrix is further defined as $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$, with $\boldsymbol{D} = \mathrm{diag}(\boldsymbol{W}\mathbf{1})$ being the degree matrix and $d_v \equiv \boldsymbol{D}_{vv}$. It is possible to stack the feature vector on each node together and obtain $\boldsymbol{X} = \left[\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{|\mathcal{V}|}\right]^T \in \mathbb{R}^{|\mathcal{V}|\times K}$ and $\boldsymbol{Y} = \left[\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_{|\mathcal{V}|}\right]^T \in \mathbb{R}^{|\mathcal{V}|\times C}$.

A DGP on a graph models the underlying mechanism that generates data associated with the nodes and edges. Mathematically this is described by the joint probability density distribution (PDF) $P(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{W})$. Under the assumption of HMN, the hidden variable $\boldsymbol{Y}$[2] is generated from a multivariate Gaussian distribution whose covariance matrix is parameterized by the graph structure $\boldsymbol{W}$ and the signals $\boldsymbol{X}$ are emitted from the hidden variables through an emission process $P(\boldsymbol{X} \mid \boldsymbol{Y})$. This yields a graphical model as in fig. 1a and the joint PDF for the nodes and edges is decomposed as $P(\boldsymbol{X}, \boldsymbol{W}, \boldsymbol{Y}) = P(\boldsymbol{Y} \mid \boldsymbol{W})P(\boldsymbol{X} \mid \boldsymbol{Y})P(\boldsymbol{W})$.

Remark. The decomposition suggests that a properly defined DGP for a graph consists of 1) the potential function $P(\boldsymbol{Y} \mid \boldsymbol{W}) = \frac{1}{Z}\prod_{\xi\in\mathrm{cl}(G)} \varphi_\xi\left(\mathcal{V}_\xi\right)$ where $\mathrm{cl}(\mathcal{G})$ is the set of cliques of $\mathcal{G}$, $\mathcal{V}_\xi$ is the subset of nodes related to ξ, and Z is the partition function, 2) the emission process $P(\boldsymbol{X} \mid \boldsymbol{Y})$ that generate node features and 3) the structural prior $P(\boldsymbol{W})$ that encodes the graph structure, such as sparsity (Friedman et al., 2007) or connectivity (Kalofolias, 2016b).

We follow Zhu et al. (2003) and consider the cliques up to rank 2, i.e., the node-wise and pair-wise potential are selected to be $\varphi_1(v) = \exp(-(d_v + \nu)\|\boldsymbol{y}_v\|^2)$ with $\|\cdot\|$ the L^2 norm and $\varphi_2(u, v) = \exp(w_{uv}\boldsymbol{y}_u^\top \boldsymbol{y}_v)$ respectively. This leads to a multi-variate Gaussian on each column of the hidden variable $\boldsymbol{Y}_{:k}$ with zero mean and precision matrix $\Lambda = \boldsymbol{L} + \nu\boldsymbol{I}$ where $\boldsymbol{I}$ is the identity matrix. The emission function is considered as another multi-variate Gaussian (Liu et al., 2014) which has a linear relationship, specifically $\boldsymbol{x}_v \mid \boldsymbol{y}_v \sim \mathcal{N}(\boldsymbol{V}\boldsymbol{y}_v, \Sigma_{\boldsymbol{x}})$. We denote $\boldsymbol{V}$ as the linear transformation matrix and Σ_x the marginal covariance matrix for $\boldsymbol{x}_v$, which are shared across all nodes $v \in \mathcal{V}$.

Under the assumption that the marginal noise on $\boldsymbol{x}$ is not too strong to conceal the structure information, i.e., $\det(\Sigma_{\boldsymbol{x}}) \ll \det(\Lambda^{-1})$. we can marginalize out $\boldsymbol{Y}$ and obtain the likelihood function of $\boldsymbol{X}$ as

$$\begin{aligned} P_{\boldsymbol{V}}(\boldsymbol{X} \mid \boldsymbol{W}) &= \frac{1}{Z}\prod_v \varphi_1(v)\prod_{u,v}\varphi_2(u,v) \\ \text{with } \varphi_1(v) &= \exp(-(\nu + d_v)\|\boldsymbol{V}^\dagger \boldsymbol{x}_v\|^2) \\ \text{and } \varphi_2(u,v) &= \exp\left\{w_{uv}(\boldsymbol{x}_u^\top \boldsymbol{V}^{\dagger\top}\boldsymbol{V}^\dagger \boldsymbol{x}_v)\right\}, \end{aligned} \tag{2}$$

where $\boldsymbol{V}^\dagger$ is the Moore–Penrose pseudo inverse of ma-

[1] To be concise, we first focus on the case where only the node signals $\boldsymbol{x}_v$ are observable and leave the generalization to other scenarios in appendix D.1.

[2] Conceptually, the labels are different from the hidden variables. In the derivation, we simplify the concept by drawing the equivalence similar to Wei et al. (2022) and view the hidden variables as the continuous embedding of labels.

trix $\boldsymbol{V}$. We left the detailed derivation of the likelihood function in appendix A.1.

3.2 Graph Structure Learning

The link between DGP under PE-MRF assumptions and GSL was introduced in Park et al. (2017). Here we extend the result to HMN and cast the GSL as a MAP estimation problem for the graph structure $\boldsymbol{W}$ and the linear transformation matrix $\boldsymbol{V}$[3], where,

$$\begin{aligned}\boldsymbol{W}^*, \boldsymbol{V}^* &= \arg\max_{\boldsymbol{W},\boldsymbol{V}} \log P_{\boldsymbol{V}}(\boldsymbol{W} \mid \boldsymbol{X}) \\ &= \arg\max_{\boldsymbol{W},\boldsymbol{V}} \log P_{\boldsymbol{V}}(\boldsymbol{X} \mid \boldsymbol{W}) + \log P(\boldsymbol{W}) + \log P(\boldsymbol{V}).\end{aligned} \tag{3}$$

Current graph structure learning (GSL) algorithms (Dong et al., 2016; Pu et al., 2021a; Kumar et al., 2019) can be considered as a special case of the above MAP problem when $\boldsymbol{V}^\dagger$ is rectangular orthogonal (semi-orthogonal) matrix. In such a scenario, the potential functions in eq. (2) reduces to $\varphi_1(v) = \exp(-(\nu + d_v)\|\boldsymbol{x}_v\|^2)$ and $\varphi_2(u,v) = \exp\left\{w_{uv}(\boldsymbol{x}_u^\top \boldsymbol{x}_v)\right\}$, which equates to a PE-MRF. Solving the MAP problem in eq. (3) leads to,

$$\arg\min_{\boldsymbol{W}} \underbrace{\sum_{uv} w_{uv}\|\boldsymbol{x}_u - \boldsymbol{x}_v\|^2}_{S(\boldsymbol{X},\boldsymbol{W})} \underbrace{- \mathbf{1}^\top \log(\boldsymbol{W}\cdot\mathbf{1}) - \log P(\boldsymbol{W})}_{\Omega(\boldsymbol{W})}. \tag{4}$$

The training objective consists of graph-signal fidelity term $S(\boldsymbol{X}, \boldsymbol{W})$ and a structural regularizer $\Omega(\boldsymbol{W})$. To conclude, learning the graph topology through MAP estimation parameterized by HMN is equivalent to the GSL algorithms if 1) $\boldsymbol{V}^\dagger$ is semi-orthogonal, 2) $\det(\Sigma_{\boldsymbol{x}}) \ll \det(\Lambda^{-1})$, 3) The log-determinant of the partition function is relaxed by the lower-bound in the optimization. We left the derivation in appendix A.2.

4 HETEROGENEOUS GRAPH STRUCTURE LEARNING

Our work focuses on generalizing HMN to facilitate the design of DGP, and solve the HGSL problem in eq. (1) via the link established in section 3. In this section, we will first propose a novel data-generating model in section 4.1 that assists in parameterizing $P(\{\boldsymbol{x}_v\} \mid \mathbf{W}, \Theta)$, and then dive into algorithm design to solve the problem in section 4.2. Finally we present analyses of the proposed algorithm in section 4.3.

[3] ν is considered as a hyperparameter.

4.1 Data-generating Process for Heterogeneous Graphs

We redefine the potential and emission functions in the DGP and obtain a novel model which we call hidden Markov networks for heterogeneous graphs (H2MN).

The model. We extend a graphical model for HG as in fig. 1b. Similar to the HMN introduced in section 3.1, the likelihood can be decomposed into the node-wise potential $\varphi_1(v)$, the pair-wise potential $\varphi_2(v, u \mid \boldsymbol{B}_r)$ and the signal emission process whose density function is $P(\boldsymbol{x}_v \mid \boldsymbol{y}_v)$. We first consider the joint PDF of node signals and labels that yields

$$\begin{aligned}&P(\{\boldsymbol{x}_v\}, \{\boldsymbol{y}_v\} \mid \mathbf{W}, \{\boldsymbol{B}_r\}) \\ &= P(\{\boldsymbol{y}_v\} \mid \mathbf{W}, \{\boldsymbol{B}_r\}) \prod_v P(\boldsymbol{x}_v \mid \boldsymbol{y}_v).\end{aligned} \tag{5}$$

Similar to the HMN for homogeneous graphs, the generation density of $\{\boldsymbol{y}_v\}$ in HGs is defined as,

$$P(\{\boldsymbol{y}_v\} \mid \mathbf{W}, \{\boldsymbol{B}_r\}) \propto \prod_v \varphi_1(v) \prod_{u,v,r} \varphi_r(u,v) \tag{6}$$

where the node-wise potential function is unchanged as $\varphi_1(v) = \exp(-(d_v + \nu)\|\boldsymbol{y}_v\|^2)$, and the edge-wise potential function is replaced by a bi-linear product dependent on the relation type $\varphi_r(u,v) = \exp(w_{uvr}\boldsymbol{y}_u^\top \boldsymbol{B}_r \boldsymbol{y}_v)$.

To link label variable $\boldsymbol{y}_v$ with the signal variable $\boldsymbol{x}_v$, we assume that $P(\boldsymbol{x}_v \mid \boldsymbol{y}_v)$ has a mean that is a linear function of $\boldsymbol{y}_v$, and a covariance $\sigma^2\boldsymbol{I}$ that is independent of $\boldsymbol{y}_v$, which gives,

$$\boldsymbol{x}_v \mid \boldsymbol{y}_v \sim \mathcal{N}\left(\boldsymbol{x}_v \mid \boldsymbol{V}_{\phi(v)}\boldsymbol{y}_v, \sigma^2\boldsymbol{I}\right). \tag{7}$$

The linear transformation matrix $\boldsymbol{V}_{\phi(v)} \in \mathbb{R}^{K \times |C_{\phi(u)}|}$ is determined by the node type $\phi(v)$. We then substitute eq. (6) and eq. (7) into eq. (5), marginalize out $\{\boldsymbol{y}_v\}$ and obtain the likelihood for the node signals,

$$\begin{aligned}&P(\{\boldsymbol{x}_v\} \mid \mathbf{W}, \{\boldsymbol{B}_r\}) \propto \prod_v \varphi_1(v) \prod_{u,v,r} \varphi_r(u,v) \\ &\text{with } \varphi_1(v) = \exp(-(\nu + d_v)\|\boldsymbol{U}_{\phi_v}\boldsymbol{x}_v\|^2) \\ &\text{and } \varphi_r(u,v) = \exp\left\{w_{uvr}(\boldsymbol{x}_u^\top \boldsymbol{U}_{\phi(u)}^\top \boldsymbol{B}_r \boldsymbol{U}_{\phi(v)}\boldsymbol{x}_v)\right\},\end{aligned} \tag{8}$$

where $\boldsymbol{U}_{\phi(u)} \in \mathbb{R}^{|C_{\phi(u)}| \times K}$ is the Moore–Penrose pseudo inverse of matrix $\boldsymbol{V}_{\phi(u)}$. The detailed derivations can be found in appendix B.1

Remark. Though we use a pair-wise HMN to approximate the overall joint probability distribution, this approximation preserves the interdependencies among edges with various relation types across the graph, even when they are not directly connected via a single node. The proof can be found in appendix C.

4.2 Algorithm Design

The optimization objective. To further simplify the estimation, we conduct a low-rank approximation by[4] $\boldsymbol{M}_r\boldsymbol{M}_r^\top \approx \boldsymbol{U}_{\phi(u)}^\top \boldsymbol{B}_r \boldsymbol{U}_{\phi(v)}$, where we specify a relation-wise matrix $\boldsymbol{M}_r \in \mathcal{R}^{K\times K'}$ that depends only on r since $\phi(u)$ and $\phi(v)$ can be directly determined when $r \in \mathcal{R}_{\phi(v),\phi(u)}$ is given. Our empirical study shows that a rank-1 relation embedding $\boldsymbol{e}_r \in \mathcal{R}^K$ is sufficient in the approximation. Thus, the training objective can be re-formalized as (derivation in appendix B.2),

$$\arg\min_{\mathbf{W},\boldsymbol{E}} \sum_{v\le u,r} w_{vur}\|\boldsymbol{e}_r^\top \cdot (\boldsymbol{x}_v - \boldsymbol{x}_u)\|^2 + \Omega(\mathbf{W}) + \Omega(\boldsymbol{E}) \tag{9}$$

where the relation-wise vectors $\boldsymbol{e}_r$ are stacked as $\boldsymbol{E} = [\boldsymbol{e}_1, \dots, \boldsymbol{e}_{|\mathcal{R}|}]$. One can see that the first term, $S(\boldsymbol{X}, \boldsymbol{E}, \mathbf{W}) = \sum_{u,v,r} w_{vur}\|\boldsymbol{e}_r^\top \cdot (\boldsymbol{x}_v - \boldsymbol{x}_u)\|^2$, promotes signal smoothness similar to that in the graph structure learning literature (Pu et al., 2021a; Dong et al., 2016) as in eq. (4).

Since the problem requires learning both the *graph structure* $\mathbf{W}$ and *relation embeddings* $\boldsymbol{E}$ that is not jointly convex, we adopt an alternating optimization scheme to solve it: $\mathbf{W}$ is optimized with $\boldsymbol{E}$ fixed, and $\boldsymbol{E}$ is optimized with $\mathbf{W}$ fixed. For notation consistency, we will use $\mathcal{E}'$ to denote the possible connections in a heterogeneous graph. Note that this is not the combination of all possible nodes and relation types that shapes a $\mathcal{V}\times\mathcal{V}\times\mathcal{R}$ space as node pairs with certain types can only be connected by specific edge types.

Graph structure learning step. The first sub-optimization problem is to find $\mathbf{W}$ that minimizes the training objective in eq. (9) with a fixed $\boldsymbol{E}$,

$$\arg\min_{\mathbf{W}} S(\boldsymbol{X}, \boldsymbol{E}, \mathbf{W}) + \Omega(\mathbf{W}). \tag{10}$$

Since we focus on undirected graphs (i.e., $\mathbf{W}$ symmetric), we only need to learn the upper triangle part of the tensor, i.e., the vectorized weights $\boldsymbol{w} \in \mathbb{R}_+^{|\mathcal{V}|\cdot(|\mathcal{V}|-1)\cdot|\mathcal{R}|/2}$. We reparameterize the training task as in Pu et al. (2021a) and obtain $\sum_{\{v,u,r\}\in\mathcal{E}'} w_{vur}\|\boldsymbol{e}_r \circ (\boldsymbol{x}_v - \boldsymbol{x}_u)\|^2 = \|\boldsymbol{w}\odot\boldsymbol{z}\|$, where $\boldsymbol{z} \in \mathbb{R}_+^{|\mathcal{V}|\cdot(|\mathcal{V}|-1)\cdot|\mathcal{R}|/2}$ is the half-vectorization of the tensor $\|(\boldsymbol{X}\otimes\mathbf{1} - \mathbf{1}\otimes\boldsymbol{X})\otimes\boldsymbol{E}\|_F^2$ with $\otimes$ the outer product. In addition, inspired by classical GSL methods (Kalofolias, 2016a; Dong et al., 2016), we choose $\Omega(\mathbf{W})$ to consist of a log barrier regularizer on node degrees and a L^1 norm regularizer on the weights to promote the connectivity and the sparsity of the graph

[4] $\boldsymbol{U}_{\phi(u)}^\top \boldsymbol{B}_r \boldsymbol{U}_{\phi(v)}$ is positive semi-definite if we consider the nodes for different types share the same signal space.

Algorithm 1: Heterogeneous Graph Structure Learning with Alternating Optimization.

Input: Node signals $\{\boldsymbol{x}_v\}_{v=1}^{|\mathcal{V}|}$ and types $\{\phi(v)\}_{v=1}^{|\mathcal{V}|}$; Relation type set $\mathcal{R}$; Maximum steps T.
Output: The graph weight $\boldsymbol{w}$; Embeddings $\boldsymbol{e}_r, \forall r \in \mathcal{R}$;
Init: Initialize $\boldsymbol{e}_r^0 = \mathbf{1^T}/K, \forall r \in \mathcal{R}$; Initialize $\boldsymbol{w}^0$ randomly; Initialize $t = 0$;
while $t < T$ **do**
  `/* Graph structure learning step */`
  Calculate the smoothness vector $\boldsymbol{z}$;
  Optimize $\boldsymbol{w}^{t+1}$ based on eq. (11);
  `/* Relation embedding update Step */`
  if Update Method == Gradient Descent **then**
    $\boldsymbol{e}_r^{t+1} = \mathrm{GD}(\boldsymbol{e}_r^t)$ according to eq. (12);
  else if Update Method == Iterative Reweighting **then**
    Update $\boldsymbol{e}_r^{t+1}$ based on eq. (13);
  end if;
  t = t+1;
end

respectively. The objective is reformulated as,

$$\arg\min_{\boldsymbol{w}} \|\boldsymbol{w}\odot\boldsymbol{z}\|^2 - \alpha\mathbf{1}^\top\log(\mathcal{T}\boldsymbol{w}) + \beta\|\boldsymbol{w}\|_1 + \mathcal{I}_{\boldsymbol{w}>0}, \tag{11}$$

where $\odot$ is the element-wise product and $\mathcal{T}$ is a linear operator that transforms $\boldsymbol{w}$ into the vector of node degrees such that $\mathcal{T}\boldsymbol{w} = (\sum_{r=1}^{|\mathcal{R}|}\mathbf{W}_{::r})\cdot\mathbf{1}$. We solve eq. (11) by alternating direction method of multipliers (ADMM) (Pu et al., 2021a) or primal-dual splitting algorithms (PDS) (Kalofolias, 2016a).

Relation embedding update step. The second sub-problem handles the optimization of relation embeddings $\boldsymbol{e}_r$. Considering $\Omega(\boldsymbol{E})$ as an elastic norm Zou and Hastie (2005), eq. (9) becomes

$$\arg\min_{\{\boldsymbol{e}_r\}} \quad S(\boldsymbol{X}, \boldsymbol{E}, \mathbf{W}) + \lambda_1\|\boldsymbol{E}\|^2 + \lambda_2\|\boldsymbol{E}\|_1. \tag{12}$$

λ_1 and λ_2 are hyper-parameters. Intuitively, $\boldsymbol{e}_r$ can be directly optimized by gradient descent, and we denote this approach as HGSL-GD. However, in our experiments, we found this unstable and easily stuck into a sub-optimum. Thus, a more efficient and stable solution is developed through analytically solving eq. (12) w.r.t. $\boldsymbol{e}_r$. The detailed derivation is left in appendix B.3. We denote this approach iterative reweighting (HGSL-IR): intuitively this solution assigns higher weights for relation r to dimensions that express higher dimension-wise similarity on the graph

learned at iteration t (i.e., w_{vur}^t):

$$\begin{aligned} \boldsymbol{e}_{r,k}^{t+1} &= \frac{\lambda_2}{2\lambda_1}\left(\frac{1}{\lambda_1}\sum_{\{v,u,r\}\in\mathcal{E}_r'} w_{vur}^t\ \boldsymbol{x}_{v,k}\cdot\boldsymbol{x}_{u,k} - 2\right) \\ &= \lambda_1' \sum_{\{v,u,r\}\in\mathcal{E}_r'} w_{vur}^t\ \boldsymbol{x}_{v,k}\cdot\boldsymbol{x}_{u,k} - \lambda_2' \end{aligned} \tag{13}$$

where $\mathcal{E}_r'$ is the set of possible edges of type r and the subscript k denotes the k-th dimension, $\lambda_1' = \frac{\lambda_2}{2\lambda_1^2}$ and $\lambda_2' = \frac{\lambda_2}{\lambda_1}$. It is noted that λ_1 cannot be chosen to be large to avoid a trivial solution (e.g. $\boldsymbol{e}_r = 0$). The overall process is described in algorithm 1.

4.3 Algorithm Analysis

The hierarchical nature of the DGP allows us to evaluate algorithm performance from two perspectives: the potential functions linking node label generation to the underlying graph structure, and the emission functions governing signal generation. We will demonstrate how the connectivity matrix within the potential function relates to homophily and how the signal generation process in the emission function affects edge type distinguishability, both of which influence model performance. Based on this understanding, we introduce two factors that impact algorithm performance.

Homophily on heterogeneous graphs: Homophily graphs (McPherson et al., 2001) suggest that connections between similar entities occur at a higher probability than among dissimilar entities. This translates to $P(\boldsymbol{W}_{uv} = 1 \mid \boldsymbol{y}_u = \boldsymbol{y}_v) > P(\boldsymbol{W}_{uv} = 1 \mid \boldsymbol{y}_u \neq \boldsymbol{y}_v)$, where $\boldsymbol{y}_u$ and $\boldsymbol{y}_v$ are the labels for nodes u and v, and the homophily ratio is linked to the ratio of the two values (Ma et al., 2022) with the relationship suggested in appendix E.3. We will generalize the concept of homophily in HGs through the connection probability matrix $\{\boldsymbol{B}_r\}$. To do so, a "representative" connection is defined between two classes p^* and q^* of nodes among relation type r if it gives the highest probability of connection $\boldsymbol{B}_r[p^*, q^*]$, i.e.,

$$\boldsymbol{B}_r[p^*, q^*] - \sum_{p\neq p^*\cap q\neq q^*} \boldsymbol{B}_r[p, q] > 0, \forall r. \tag{14}$$

The inequality yields the definition of homophily ratio,

$$\mathrm{HR}(\mathcal{G}, r) = \frac{\boldsymbol{B}_r[p^*, q^*]}{\sum_{p\neq p^*\wedge q\neq q^*} \boldsymbol{B}_r[p, q]}. \tag{15}$$

In appendix E.3 we prove that inequality in eq. (14) is a sufficient condition for guaranteeing algorithm 1 a meaningful solution[5]. However, in practice, the value in eq. (15) is difficult to compute, because labels are typically recorded for only one node type, rather than all node types while constructing graph datasets for node classification tasks. Therefore, we consider a new metric, named relaxed homophily ratio, in the following paragraph.

[Relaxed Homophily Ratio (RHR)] (Guo et al., 2023) is defined based on how the labels are similar along the meta-paths (Wang et al., 2020). A meta-path Φ is a path template following a specific sequence of node and relation types like $A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \cdots \xrightarrow{R_{L-1}} A4_L$ with node types $A_1, \cdots, A_L \in \mathcal{A}$ and edge types $R_1, \cdots, R_{L-1} \in \mathcal{R}$. When $A_1 = A_L$, it is possible to match the node and edge types along each path $\mathcal{P}$ in $\mathcal{G}$ with meta-path Φ and construct a new graph $\mathcal{G}_\Phi$ by connecting the source and end nodes for all $\mathcal{P}$. The RHR is defined on $\mathcal{G}_\Phi$ by measuring the label (y_u) similarity within the same node type:

$$\mathrm{RHR}\left(\mathcal{G}_\Phi\right) = \frac{\sum_{\{u,v\}\in\mathcal{E}_\Phi} \mathbb{I}\left(y_u = y_v\right)}{|\mathcal{E}_\Phi|}, \tag{16}$$

We show in section 6.3 that the RHR is positively related to the model performance.

Edge type distinguishability: We now examine the property of the signal generation function and its impact on the solution. Intuitively illustrated in fig. 5, the data-fidelity term in eq. (9) can be considered as a *reweighted smoothness* scheme that first measures dimension-wise smoothness, and then integrates it by emphasizing specific signal dimensions according to r. Using the movie review dataset again as an example, while determining whether a 'star in'-typed edge should be formed between two actor nodes, the model should put larger weights on the signal dimensions that represent the 'genres', but weigh less irrelevant ones such as 'company affiliation' or 'date'.

Thus, an important property that supports the learning is that the nodes connected with different relation types (inherently different node labels) would exhibit similarity in different dimensions of features. This can be measured by the following quantity, namely the smoothest-dimension overlapping ratio (SDOR).

[Smoothest-Dimension Overlapping Ratio (SDOR)] We first define the dimension-wise smoothness as $S(\boldsymbol{X}_{:k}, \mathbf{W}_{::r}) = \sum_{\{v,u,r\}\in\mathcal{E}} w_{vur}\|\mathbf{x}_{v,k} - \mathbf{x}_{u,k}\|_2^2$ for relation r and dimension k. The top-M smooth dimensions for r, $\mathcal{K}^M(r)$ are calculated by ranking dimension-wise smoothness across $k \in [K]$. For each relation pair (r, r'), the SDOR is obtained by counting the overlapping dimensions in $\mathcal{K}^M(r)$ and $\mathcal{K}^M(r')$:

$$\mathrm{SDOR}(r, r') = \frac{|\mathcal{K}^M(r) \cap \mathcal{K}^M(r')|}{|\mathcal{K}^M(r) \cup \mathcal{K}^M(r')|}. \tag{17}$$

[5] Avoiding non-convergence solution such as infinitely maximizing $\mathbf{W}$, or trivial solutions such that $\mathbf{W} = 0$

The SDOR reflects how different two relation types are in terms of exhibiting smoothness in signal generation. If relation pairs (r, r') exhibit a high SDOR, our algorithm will converge into a solution that has similar embeddings for r and r', leading to difficulties in distinguishing between the two relation types. In section 6.3, we will empirically demonstrate the correlation between SDOR and algorithm performance.

5 RELATED WORKS

Multiple Graphical Models Estimation. A line of research has extended precision matrix estimation methods such as the graphical Lasso to heterogeneous data, referred to as multiple graphical model estimation (MGME) (Gan et al., 2019; Ma and Michailidis, 2016; Hao et al., 2017). Examples include group graphical lasso (Jacob et al., 2009) and fused graphical Lasso (Danaher et al., 2014). The concept of heterogeneous data in MGME assumes that heterogeneity exists across multiple separate graphical models (termed as groups), while our work considers a heterogeneous graph in a different sense. In contrast to MGME methods, we assume there is a single underlying graph that contains multiple types of nodes and relationships, which motivates us to design HGSL algorithms to ensure the general property, such as connectivity and sparsity, of the entire graph.

Supervised HGSL. Another branch of research focuses on learning latent graph structures from supervision signals, such as existing links or downstream task objectives (Zaripova et al., 2023; Battiloro et al., 2024). Here, the goal is to construct a graph that benefits a specific downstream task. A handful of studies have also explored applying this supervised graph structure learning approach to heterogeneous graphs (Zhao et al., 2021, 2023). In contrast, our work addresses unsupervised GSL, where the aim is to model the interaction between the graph and the node features so that, if the graph is unavailable, it can be reconstructed from the node features in a way that preserves its key properties.

6 EXPERIMENTS

We evaluate the HGSL algorithm through quantitative and qualitative experiments. We first outline the experimental setup in section 6.1, followed by a quantitative assessment of model performance in section 6.2. Next, we analyze the impact of previously mentioned factors on HGSL performance in section 6.3 and demonstrate how our model reveals interesting relationships in real-world problems in section 6.4.

6.1 Experimental Setup

6.1.1 Datasets

Synthetic datasets. The synthetic dataset construction consists of 3 phases: 1) **Graph backbone generation.** We followed the process in Pu et al. (2021a) and used the stochastic block model and Watts–Strogatz model to generate the backbone with 20-100 nodes that encode the graph structure. 2) **Node type generation.** We traverse the graph by breadth-first search and generate the node types following the rule defined in a meta template (network schema (Shi, 2022)) that encodes how nodes/edges with different types are connected. **3) Feature generation.** We then generate node signals that satisfy the probability distribution as follows.

$$p(\boldsymbol{X}, \boldsymbol{E} \mid \mathbf{W}) \propto \exp\left(-\frac{1}{\sigma} \sum_{\{v,u,r\}\in\mathcal{E}'} w_{vur} \|\boldsymbol{e}_r \cdot (\boldsymbol{x}_v - \boldsymbol{x}_u)\|_2^2\right). \tag{18}$$

Real-world datasets: We consider IMDB (Guo et al., 2023) and ACM (Lv et al., 2021) datasets for quantitative evaluation. IMDB is a movie review dataset with node types including directors (D), actors (A), and movies (M) and with signals as 3066-D bag-of-words representation. ACM is an academic dataset that contains papers (P), authors (A), and subjects (S). Signals correspond to the 1902-D bag-of-words representation of the keywords in diverse research areas. Detailed statistics are presented in table 2. To augment more graph instances for a comprehensive evaluation, we subsample the dataset to generate smaller graphs with number of nodes ranging from 50-200 following the strategy in Hu et al. (2020b).

We also perform a qualitative experiment on Yahoo-Finance (Finance, 2024) financial dataset to uncover the relationships among S&P 100 companies with node types representing company sectors (health, finance, technology). Using daily stock returns as signals, the aim is to reveal connectivity patterns, represented by edge types, across different company sectors.

6.1.2 Metrics

The model performance is evaluated quantitatively on their ability to identify edge type and to recover edge weights. The edge type identification error is measured by the area under the curve (AUC) where we view "no edge" as a class and consider a classification problem on the edge types. The edge weight recovery error is measured by the mean squared error for graph recovery

Table 1: Quantitative experimental results on heterogeneous graph structure learning compared to covariance matrix recovery algorithms and graph structure learning algorithms for homogeneous graphs.

Model	Synthetic Dataset		IMDB (HR: 0.51)		ACM (HR: 0.64)	
	AUC	GMSE	AUC	GMSE	AUC	GMSE
GGL (Jacob et al., 2009)	0.63 ± 0.03	0.06 ± 0.02	0.60 ± 0.06	0.5 ± 0.02	0.68 ± 0.05	**0.02 ± 0.01**
FGL (Danaher et al., 2014)	0.62 ± 0.03	0.05 ± 0.01	0.73 ± 0.06	**0.03 ± 0.03**	0.59 ± 0.03	0.03 ± 0.00
GSL (Dong et al., 2016)	0.61 ± 0.04	0.04 ± 0.00	0.75 ± 0.06	0.30 ± 0.04	0.56 ± 0.06	0.09 ± 0.08
GSL (Kalofolias, 2016a)	0.58 ± 0.02	0.04 ± 0.02	0.74 ± 0.08	0.27 ± 0.10	0.57 ± 0.04	0.14 ± 0.03
GSL (Pu et al., 2021a)	0.65 ± 0.01	0.03 ± 0.00	0.74 ± 0.04	0.29 ± 0.08	0.65 ± 0.06	0.07 ± 0.10
HGSL-IR (Ours)	**0.83 ± 0.04**	**0.02 ± 0.01**	**0.81 ± 0.07**	0.07 ± 0.05	**0.73 ± 0.02**	0.12 ± 0.07

* Experimental results are evaluated over 30 trials and the mean/standard deviation is calculated.

(GMSE) introduced by Pu et al. (2021a),

$$\text{GMSE} = \frac{1}{|\mathcal{V}| \times |\mathcal{V}| \times |\mathcal{R}|} \sum_{\{u,v,r\}} \frac{\|\hat{w}_{uvr} - w_{uvr}\|^2}{\|w_{uvr}\|^2} \tag{19}$$

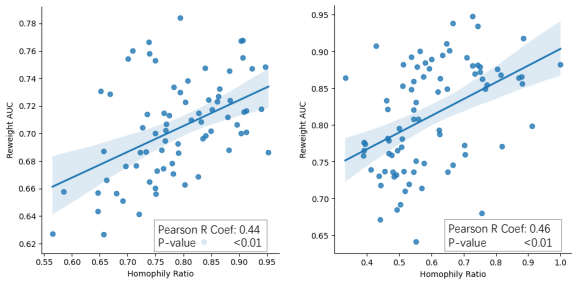
6.2 Quantitative Results

Due to the lack of appropriate baselines as no prior algorithm can identify the edge types, we compare against MGME methods, specifically group graphical lasso (GGL) (Jacob et al., 2009) and fused graphical Lasso (FGL) (Danaher et al., 2014), and conventional graph structure learning (GSL) algorithms[6] (Kalofolias, 2016a; Pu et al., 2021a) in terms of edge weights recovery and binary edge identification (omitting the types). The results are reported in table 1. With our HGSL algorithm with iterative reweighting (HGSL-IR), a consistent improvement is found in all the datasets in terms of both AUC and GMSE. This suggests the efficacy of our algorithm in both tasks. The improvement of AUC in synthetic datasets (avg +18.48%) is relatively larger than the real-world datasets (avg +10.88%), which suggests the more challenging nature of the real-world experiments.

We also test the HGSL with gradient descent (GD) to demonstrate the efficacy of our algorithm design in section 4.2, where the results are reported in table 3. Though a better AUC is found in most datasets, the improvement is rather marginal and the GMSE is much larger. This shows our HGSL-IR can lead to better performance.

6.3 Robustness Analysis

Relaxed homophily ratio. We are interested in how the algorithm performance is related to the homophily ratio. From our theoretical analysis in section 4.3, we hypothesize that a lower RHR hinders performance. Thus, we record the RHR of all the subgraphs and report the Pearson correlation coefficients between RHR and corresponding AUC as shown in fig. 2. According to the results, we can conclude that the homophily ratio is positively correlated with AUC, which validates our analysis.



(a) ACM (P-Coef: 0.44) (b) IMDB (P-Coef: 0.46)

Figure 2: The Pearson correlation test between relaxed homophily ratio and AUC.

Smoothest-dimension overlapping ratio. To understand the behavior of our algorithm against high **SDOR**, we fix the number of relation types $|\mathcal{R}| = 2$ in our synthetic experiment, and manually adjust the SDOR from 0 to 1. The higher SDOR is thought to hinder the distinguishability of relation types. We illustrate the AUC of the vanilla GSL and HGSL algorithms in fig. 3, and calculate its relative increase. It is clear that the performance of HGSL is significantly affected when the SDOR increases, which suggests that a higher SDOR would hinder the distinguishability of edge types. However, the HGSL algorithm is robust until the SDOR is increased to approximately 0.7, which suggests the applicability of our algorithm in most real-world datasets, as compared with the SDOR statistics in table 2.

6.4 Qualitative Results

We apply the HGSL algorithm to a financial dataset to recover different types of connections among S&P

[6] Following the implementation in Pu et al. (2021a).

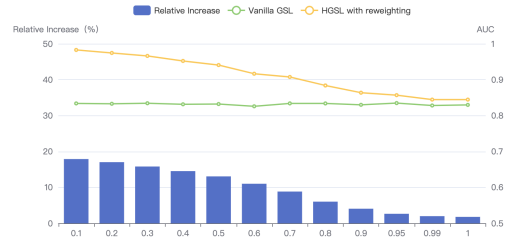
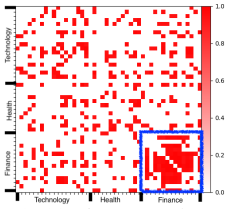
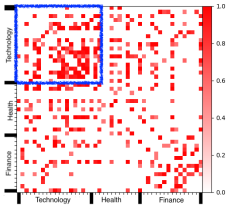


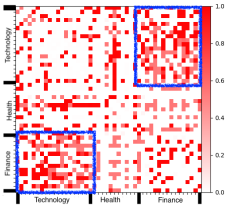
Figure 3: The SDOR and model performance.



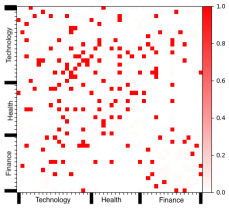
(a) Connections with type finance-to-finance.



(b) Connections with type technology-to-technology.



(c) Connections with type technology-to-finance.



(d) Connections with type health-to-technology.

Figure 4: The different types of connections recovered by the HGSL algorithm.

100 companies. We use the daily returns of stocks obtained from YahooFinance. Figure 4 visualized the estimated relation-wise graph adjacency matrix where sectors sort the rows and columns. The heatmap clearly shows that two stocks in the same sectors are likely to behave similarly. And most interestingly, the stock prices of companies coming from the technology and finance sectors exhibit a strong relationship. This suggests the ability of our HGSL algorithm to reveal different relationships in real-world network systems.

7 DISCUSSION AND FUTURE WORK

In this study, we propose an HGSL framework based on a data-generating process (DGP) for heterogeneous graphs. The method is grounded on assumptions such as signals for different node types existing in the same space and cliques being up to rank-2. However, it can easily be generalized to accommodate more complex cases with varying signal dimensions and higher-order cliques. Nonetheless, model-based design for DGP may struggle with real-world heterogeneous graphs that possess intricate structures deviating from the design. In such cases, leveraging deep learning-based generative models can be beneficial. Our future work will aim to enhance the DGP by utilizing neural networks to parameterize the potential and emission functions and developing optimization algorithms to learn the DGP. Furthermore, utilizing the DGP in scenarios with downstream tasks, such as node classification and link prediction, is a promising direction to explore. Examples along this line include understanding the performance of heterogeneous graph ML models through the compatibility between the DGP and the model architecture (Kaur et al., 2023), and informing the design of deep graph networks through understanding the DGP for graph-structured data (Wei et al., 2022).

Acknowledgements

K.J. was supported by the UKRI Engineering and Physical Sciences Research Council (EPSRC) [grant number EP/R513143/1]. X.D. acknowledges support from the Oxford-Man Institute of Quantitative Finance and the EPSRC (EP/T023333/1). The authors would like to thank the anynomous reviewers for their helpful comments and insights.

References

Banerjee, O., Ghaoui, L. E., and d'Aspremont, A. (2008). Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. *J. Mach. Learn. Res.*, 9:485–516.

Barata, J. and Hussein, M. (2011). The moore-penrose pseudoinverse. a tutorial review of the theory. *Brazilian Journal of Physics*, 42.

Barlacchi, G., De Nadai, M., Larcher, R., Casella, A., Chitic, C., Torrisi, G., Antonelli, F., Vespignani, A., Pentland, A., and Lepri, B. (2015). A multi-source dataset of urban life in the city of milan and the province of trentino. *Scientific Data*, 2:150055.

Battiloro, C., Spinelli, I., Telyatnikov, L., Bronstein, M. M., Scardapane, S., and Lorenzo, P. D. (2023). From latent graph to latent topology inference: Differentiable cell complex module. *CoRR*, abs/2305.16174.

Battiloro, C., Spinelli, I., Telyatnikov, L., Bronstein, M. M., Scardapane, S., and Lorenzo, P. D. (2024). From latent graph to latent topology inference: Differentiable cell complex module. In *ICLR*. OpenReview.net.

Bollacker, K., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. (2008). Freebase: a collaboratively cre-

ated graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.

Chami, I., Abu-El-Haija, S., Perozzi, B., Ré, C., and Murphy, K. (2022). Machine learning on graphs: A model and comprehensive taxonomy. *J. Mach. Learn. Res.*, 23:89:1–89:64.

Chen, S., Sandryhaila, A., Moura, J. M. F., and Kovacevic, J. (2015). Signal recovery on graphs: Variation minimization. *IEEE Trans. Signal Process.*, 63(17):4609–4624.

Danaher, P., Wang, P., and Witten, D. M. (2014). The joint graphical lasso for inverse covariance estimation across multiple classes. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(2):373–397.

Dettmers, T., Minervini, P., Stenetorp, P., and Riedel, S. (2018). Convolutional 2d knowledge graph embeddings. In *AAAI*, pages 1811–1818. AAAI Press.

Dong, X., Thanou, D., Frossard, P., and Vandergheynst, P. (2016). Learning laplacian matrix in smooth graph signal representations. *IEEE Trans. Signal Process.*, 64(23):6160–6173.

Dong, X., Thanou, D., Rabbat, M. G., and Frossard, P. (2019). Learning graphs from data: A signal representation perspective. *IEEE Signal Process. Mag.*, 36(3):44–63.

Dong, X., Thanou, D., Toni, L., Bronstein, M. M., and Frossard, P. (2020). Graph signal processing for machine learning: A review and new perspectives. *IEEE Signal Process. Mag.*, 37(6):117–127.

Egilmez, H. E., Pavez, E., and Ortega, A. (2017). Graph learning from data under laplacian and structural constraints. *IEEE J. Sel. Top. Signal Process.*, 11(6):825–841.

Finance, Y. (2024). Stock market data. Accessed: 2024-05-27.

Friedman, J., Hastie, T., and Tibshirani, R. (2007). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441.

Friedman, J., Hastie, T., and Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441.

Fu, X., Zhang, J., Meng, Z., and King, I. (2020). MAGNN: metapath aggregated graph neural network for heterogeneous graph embedding. In *WWW*, pages 2331–2341. ACM / IW3C2.

Gallotti, R. and Barthelemy, M. (2015). The multilayer temporal network of public transport in great britain. *Scientific Data*, 2(1):140056.

Gan, L., Yang, X., Narisetty, N. N., and Liang, F. (2019). Bayesian joint estimation of multiple graphical models. In *NeurIPS*, pages 9799–9809.

Gao, J., Liang, F., Fan, W., Sun, Y., and Han, J. (2009). Graph-based consensus maximization among multiple supervised and unsupervised models. In *NIPS*, pages 585–593. Curran Associates, Inc.

Ghahramani, Z. (2001). An introduction to hidden markov models and bayesian networks. *International journal of pattern recognition and artificial intelligence*, 15(01):9–42.

Guo, J., Du, L., Bi, W., Fu, Q., Ma, X., Chen, X., Han, S., Zhang, D., and Zhang, Y. (2023). Homophily-oriented heterogeneous graph rewiring. In *WWW*, pages 511–522. ACM.

Hamilton, W. L., Ying, R., and Leskovec, J. (2017). Representation learning on graphs: Methods and applications. *IEEE Data Eng. Bull.*, 40(3):52–74.

Hao, B., Sun, W. W., Liu, Y., and Cheng, G. (2017). Simultaneous clustering and estimation of heterogeneous graphical models. *J. Mach. Learn. Res.*, 18:217:1–217:58.

Harper, F. M. and Konstan, J. A. (2016). The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4):19:1–19:19.

Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M., and Leskovec, J. (2020a). Open graph benchmark: Datasets for machine learning on graphs. In *NeurIPS*.

Hu, Z., Dong, Y., Wang, K., and Sun, Y. (2020b). Heterogeneous graph transformer. In *WWW*, pages 2704–2710. ACM / IW3C2.

Ising, E. (1925). Beitrag zur theorie des ferromagnetismus. *Zeitschrift für Physik*, 31:253–258.

Jacob, L., Obozinski, G., and Vert, J. (2009). Group lasso with overlap and graph lasso. In *ICML*, volume 382 of *ACM International Conference Proceeding Series*, pages 433–440. ACM.

Jia, J. and Benson, A. R. (2022). A unifying generative model for graph learning algorithms: Label propagation, graph convolutions, and combinations. *SIAM J. Math. Data Sci.*, 4(1):100–125.

Kalofolias, V. (2016a). How to learn a graph from smooth signals. In *AISTATS*, volume 51 of *JMLR Workshop and Conference Proceedings*, pages 920–929. JMLR.org.

Kalofolias, V. (2016b). How to learn a graph from smooth signals. In Gretton, A. and Robert, C. C., editors, *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51 of *Proceedings of Machine Learning Research*, pages 920–929, Cadiz, Spain. PMLR.

Kaur, J. N., Kiciman, E., and Sharma, A. (2023). Modeling the data-generating process is necessary for out-of-distribution generalization. In *ICLR*. OpenReview.net.

Kumar, S., Ying, J., de Miranda Cardoso, J. V., and Palomar, D. P. (2019). Structured graph learning via laplacian spectral constraints. In *NeurIPS*, pages 11647–11658.

Liu, J., Zhang, C., Burnside, E., and Page, D. (2014). Learning Heterogeneous Hidden Markov Random Fields. In Kaski, S. and Corander, J., editors, *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, volume 33 of *Proceedings of Machine Learning Research*, pages 576–584, Reykjavik, Iceland. PMLR.

Liu, Z., Chen, C., Yang, X., Zhou, J., Li, X., and Song, L. (2018). Heterogeneous graph neural networks for malicious account detection. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 2077–2085.

Lv, Q., Ding, M., Liu, Q., Chen, Y., Feng, W., He, S., Zhou, C., Jiang, J., Dong, Y., and Tang, J. (2021). Are we really making much progress?: Revisiting, benchmarking and refining heterogeneous graph neural networks. In *KDD*, pages 1150–1160. ACM.

Ma, J. and Michailidis, G. (2016). Joint structural estimation of multiple graphical models. *J. Mach. Learn. Res.*, 17:166:1–166:48.

Ma, Y., Liu, X., Shah, N., and Tang, J. (2022). Is homophily a necessity for graph neural networks? In *ICLR*. OpenReview.net.

McPherson, M., Smith-Lovin, L., and Cook, J. M. (2001). Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, 27(1):415–444.

Ortega, A., Frossard, P., Kovačević, J., Moura, J. M. F., and Vandergheynst, P. (2018). Graph signal processing: Overview, challenges, and applications. *Proceedings of the IEEE*, 106(5):808–828.

Park, Y., Hallac, D., Boyd, S. P., and Leskovec, J. (2017). Learning the network structure of heterogeneous data via pairwise exponential markov random fields. In *AISTATS*, volume 54 of *Proceedings of Machine Learning Research*, pages 1302–1310. PMLR.

Pu, X., Cao, T., Zhang, X., Dong, X., and Chen, S. (2021a). Learning to learn graph topologies. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems*.

Pu, X., Chau, S. L., Dong, X., and Sejdinovic, D. (2021b). Kernel-based graph learning from smooth signals: A functional viewpoint. *IEEE Trans. Signal Inf. Process. over Networks*, 7:192–207.

Rossi, E., Monti, F., Leng, Y., Bronstein, M. M., and Dong, X. (2022). Learning to infer structures of network games. In *ICML*, volume 162 of *Proceedings of Machine Learning Research*, pages 18809–18827. PMLR.

Salami, H., Ying, B., and Sayed, A. H. (2017). Social learning over weakly connected graphs. *IEEE Trans. Signal Inf. Process. over Networks*, 3(2):222–238.

Shi, C. (2022). *Heterogeneous Graph Neural Networks*, pages 351–369. Springer Nature Singapore, Singapore.

Traud, A. L., Mucha, P. J., and Porter, M. A. (2011). Social structure of facebook networks. *CoRR*, abs/1102.2166.

Wang, X., Bo, D., Shi, C., Fan, S., Ye, Y., and Yu, P. S. (2020). A survey on heterogeneous graph embedding: Methods, techniques, applications and sources. *CoRR*, abs/2011.14867.

Wang, X., He, X., Wang, M., Feng, F., and Chua, T. (2019). Neural graph collaborative filtering. In Piwowarski, B., Chevalier, M., Gaussier, É., Maarek, Y., Nie, J., and Scholer, F., editors, *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, pages 165–174. ACM.

Wei, R., Yin, H., Jia, J., Benson, A. R., and Li, P. (2022). Understanding non-linearity in graph neural networks from the bayesian-inference perspective. In *NeurIPS*.

Yuan, M. and Lin, Y. (2007). Model selection and estimation in the gaussian graphical model. *Biometrika*, 94(1):19–35.

Zaripova, K., Cosmo, L., Kazi, A., Ahmadi, S., Bronstein, M. M., and Navab, N. (2023). Graph-in-graph (gig): Learning interpretable latent graphs in non-euclidean domain for biological and healthcare applications. *Medical Image Anal.*, 88:102839.

Zhang, M. and Chen, Y. (2020). Inductive matrix completion based on graph neural networks. In *ICLR*. OpenReview.net.

Zhao, J., Wang, X., Shi, C., Hu, B., Song, G., and Ye, Y. (2021). Heterogeneous graph structure learning for graph neural networks. In *AAAI*, pages 4697–4705. AAAI Press.

Zhao, W., Wu, Q., Yang, C., and Yan, J. (2023). Graphglow: Universal and generalizable structure learning for graph neural networks. In *KDD*, pages 3525–3536. ACM.

Zhong, Y., Liang, L., Tang, B., Zharkov, I., and Neumann, U. (2024). Motion graph unleashed: A novel approach to video prediction. In *The Thirty-eighth*

Annual Conference on Neural Information Processing Systems.

Zhou, D. and Schölkopf, B. (2004). A regularization framework for learning from graph data. In *ICML 2004 Workshop on Statistical Relational Learning and Its Connections to Other Fields (SRL 2004)*, pages 132–137.

Zhu, X., Lafferty, J., and Ghahramani, Z. (2003). *Semi-supervised learning: From Gaussian fields to Gaussian processes*. School of Computer Science, Carnegie Mellon University.

Zou, H. and Hastie, T. (2005). Regularization and Variable Selection Via the Elastic Net. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 67(2):301–320.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes. The mathematical settings and assumptions are described in Sec. 2. The algorithm and model design are in Sec.4.]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes. We include the algorithm analysis in Appendix E.]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [No. This is a rather theoretical paper and the code for toy experiments can be provided upon request.]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes. The assumptions for theoretical results are provided together with the statements.]
   (b) Complete proofs of all theoretical results. [Yes. For every claim we made in the theoretical part, we provide the corresponding derivations in appendix.]
   (c) Clear explanations of any assumptions. [Yes. We provide more explanations for the theories in the appendix.]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [No. The code for toy experiments can be provided upon request.]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes. We provide such experiment settings in Sec. 6.]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes. We reported the standard deviations together with the mean to illustrate statistical significance.]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [No. The experiments are reproducible through any CPUs.]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes. The datasets used in the paper is well-cited.]
   (b) The license information of the assets, if applicable. [Not Applicable.]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Yes. We provide the citations to the codes that we used to produce the baselines.]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Heterogeneous Graph Structure Learning through the Lens of Data-generating Processes: Supplementary Materials

A THEORETICAL RESULTS ON HOMOGENEOUS GRAPH STRUCTURE LEARNING

A.1 The Data-generating Process for Homogeneous Graphs

We wish to derive the probability density function (PDF) for the data-generating process (DGP) on homogeneous graphs. To recall, the DGP for a homogeneous graph can be decomposed as, $P(\boldsymbol{X}, \boldsymbol{W}, \boldsymbol{Y}) = P(\boldsymbol{Y} \mid \boldsymbol{W})P(\boldsymbol{X} \mid \boldsymbol{Y})P(\boldsymbol{W})$. We are interested in deriving the joint probability of $\boldsymbol{X}$ and $\boldsymbol{Y}$, denoted as $P(\boldsymbol{X}, \boldsymbol{Y} \mid \boldsymbol{W}) = P(\boldsymbol{Y} \mid \boldsymbol{W})P(\boldsymbol{X} \mid \boldsymbol{Y})$, and then obtain the marginal distribution of $P(\boldsymbol{X} \mid \boldsymbol{W})$ by,

$$P(\boldsymbol{X} \mid \boldsymbol{W}) = \int P(\boldsymbol{Y} \mid \boldsymbol{W})P(\boldsymbol{X} \mid \boldsymbol{Y})d\boldsymbol{Y}, \tag{20}$$

which we use as the likelihood term to formalize the maximum a-posteriori (MAP) estimation objective on $\boldsymbol{W}$.

We first show the form of $P(\boldsymbol{Y} \mid \boldsymbol{W})$. Similar to Zhu et al. (2003), we choose the node-wise and pair-wise potential as $\varphi_1(v) = \exp(-(d_v + \nu)\|\boldsymbol{y}_v\|^2)$ with $\|\cdot\|$ the L^2 norm and $\varphi_2(u, v) = \exp(w_{uv}\boldsymbol{y}_u^\top \boldsymbol{y}_v)$ respectively. This leads to a multi-variate Gaussian on each column of the hidden variable $\boldsymbol{Y}_{:,k} \in \mathcal{R}^{|\mathcal{V}|}$ with zero mean and precision matrix $\Lambda = \boldsymbol{L} + \nu\boldsymbol{I}$ where $\boldsymbol{I}$ is the identity matrix. The overall likelihood function for $\boldsymbol{Y}$ is then,

$$\begin{aligned}
P(\boldsymbol{Y} \mid \boldsymbol{W}) &= \frac{1}{Z}\prod_{v\in\mathcal{V}}\varphi_1(\boldsymbol{y}_v)\prod_{\{v,u\}\in\mathcal{E}}\varphi_2(\boldsymbol{y}_v, \boldsymbol{y}_u) \\
&= \frac{1}{Z}\prod_{v\in\mathcal{V}}\exp(-(d_v+\nu)\|\boldsymbol{y}_v\|^2)\prod_{\{v,u\}\in\mathcal{E}}\exp(w_{uv}\boldsymbol{y}_u^\top\boldsymbol{y}_v) \\
&= \frac{1}{Z}\exp\left\{-\sum_{v\in\mathcal{V}}(d_v+\nu)\|\boldsymbol{y}_v\|^2 + \sum_{\{v,u\}\in\mathcal{E}} w_{uv}\boldsymbol{y}_u^\top\boldsymbol{y}_v\right\} \\
&= \frac{1}{Z}\exp\left\{-\frac{1}{2}\sum_{k=1}^{C}(\boldsymbol{Y}_{:,k}^\top\Lambda\boldsymbol{Y}_{:,k})\right\} \\
&= \frac{1}{Z}\exp\left\{-\frac{1}{2}\operatorname{tr}(\boldsymbol{Y}^\top\Lambda\boldsymbol{Y})\right\},
\end{aligned} \tag{21}$$

where tr is the trace operator and we simplify the partition function $Z(\boldsymbol{W})$ as Z. If each dimension of the hidden variable, $\boldsymbol{Y}_{:,k}$ is independent to each other, it has a PDF $P(\boldsymbol{Y}_{:,k} \mid \boldsymbol{W}) \propto \exp\left\{-\boldsymbol{Y}_{:,k}^\top\Lambda\boldsymbol{Y}_{:,k}\right\}$, which is a multivariate Gaussian with precision matrix Λ. Please note that the overall likelihood function differs from a simple multivariate Gaussian as $\boldsymbol{Y} \in \mathbb{R}^{N\times C}$ is a matrix instead of a vector. But we can model the joint probability distribution of two hidden variables $\boldsymbol{y}_u$ and $\boldsymbol{y}_v$ instead, which is introduced as follows.

Without loss of interpretability, we denote the concatenation of two hidden variable vectors as $\boldsymbol{y} \coloneqq \text{cat}(\boldsymbol{y}_v, \boldsymbol{y}_u) \in \mathbb{R}^{2C}$, two signal vectors as $\boldsymbol{x} \coloneqq \text{cat}(\boldsymbol{x}_v, \boldsymbol{x}_u) \in \mathbb{R}^{2K}$, and consider Λ_{uv} as an entry of the precision matrix. The joint distribution for the hidden variable vectors on a pair of nodes, $\boldsymbol{y}_v$ and $\boldsymbol{y}_u$ becomes,

$$\begin{aligned} P(\boldsymbol{y} \mid \Lambda) &= \frac{1}{Z} \exp\left\{-\frac{1}{2}(\boldsymbol{y}^\top \begin{bmatrix} (d_v+\nu)\boldsymbol{I} & \Lambda_{uv}\boldsymbol{I} \\ \Lambda_{vu}\boldsymbol{I} & (d_v+\nu)\boldsymbol{I} \end{bmatrix} \boldsymbol{y})\right\} \\ &= \frac{1}{Z} \exp\left\{-\frac{1}{2}(\boldsymbol{y}^\top \begin{bmatrix} (d_v+\nu)\boldsymbol{I} & -\boldsymbol{W}_{uv}\boldsymbol{I} \\ -\boldsymbol{W}_{vu}\boldsymbol{I} & (d_v+\nu)\boldsymbol{I} \end{bmatrix} \boldsymbol{y})\right\}. \end{aligned} \tag{22}$$

Then, the emission function $P(\boldsymbol{X} \mid \boldsymbol{Y})$ is considered as another multi-variate Gaussian (Liu et al., 2014) which has a linear relationship, specifically $\boldsymbol{x}_v \mid \boldsymbol{y}_v \sim \mathcal{N}(\boldsymbol{V}\boldsymbol{y}_v, \Sigma_{\boldsymbol{x}})$. We denote $\boldsymbol{V}$ as the linear transformation matrix and Σ_x the marginal covariance matrix for $\boldsymbol{x}_v$, which are shared across all nodes $v \in \mathcal{V}$. Then, PDF of $P(\boldsymbol{X} \mid \boldsymbol{Y})$ can be decomposed to nose-wise, namely,

$$P_{\boldsymbol{V}}(\boldsymbol{X} \mid \boldsymbol{Y}) = \prod_{v \in \mathcal{V}} P_{\boldsymbol{V}}\left(\boldsymbol{x}_v \mid \boldsymbol{y}_v\right). \tag{23}$$

Note that the linear transformation matrix $\boldsymbol{V}$ is shared across all nodes $v \in \mathcal{V}$. Thus, by marginalizing out $\boldsymbol{y}_v$, we have the likelihood function for $\boldsymbol{x} \equiv \text{cat}(\boldsymbol{x}_v, \boldsymbol{x}_u)$ as

$$\begin{aligned} P(\boldsymbol{x} \mid \boldsymbol{W}) &= \int P_{\boldsymbol{V}}(\boldsymbol{x} \mid \boldsymbol{y}) P(\boldsymbol{y} \mid \boldsymbol{W}) d\boldsymbol{y} \\ \text{and } \boldsymbol{x} \mid \boldsymbol{W} &\sim \mathcal{N}(0, \Sigma_x + \boldsymbol{V}' \begin{bmatrix} (d_v+\nu)\boldsymbol{I} & \Lambda_{uv}\boldsymbol{I} \\ \Lambda_{vu}\boldsymbol{I} & (d_u+\nu)\boldsymbol{I} \end{bmatrix}^{-1} \boldsymbol{V}'^\top). \end{aligned} \tag{24}$$

$\boldsymbol{V}'$ is the stack of two matrix $\boldsymbol{V}$, i.e. $\boldsymbol{V}' = [\boldsymbol{V}^\top, \boldsymbol{V}^\top]^\top$.

Proof: Finding the marginal distribution, $P(\boldsymbol{x} \mid \boldsymbol{W})$ given analytical form of $P(\boldsymbol{y} \mid \boldsymbol{W})$ and $P(\boldsymbol{x} \mid \boldsymbol{y})$ is a problem that arises frequently in Bayesian theorem for Gaussian variables. We are going to use it a few more times so it would be convenient to derive the general results here. To do so, we introduced the following lemma A.1.

Lemma A.1 ***Linear Gaussian Model***. *(Ghahramani, 2001)*

Given two vectorized random variables, $\boldsymbol{b}$ following a Marginal Gaussian distribution, and $\boldsymbol{a}$ following a Gaussian distribution conditioned on $\boldsymbol{b}$, which has the following form,

$$\begin{aligned} P(\boldsymbol{b}) &= \mathcal{N}\left(\boldsymbol{b} \mid \boldsymbol{\mu}_1, \Omega_1^{-1}\right) \\ P(\boldsymbol{a} \mid \boldsymbol{b}) &= \mathcal{N}\left(\boldsymbol{a} \mid \boldsymbol{T}\boldsymbol{b} + \boldsymbol{\mu}_2, \Omega_2^{-1}\right) \end{aligned}$$

where $\boldsymbol{T}$ is the linear transformation matrix, Ω_1 and Ω_2 are two precision matrices. The random variable $\boldsymbol{a}$ has a marginal distribution,

$$P(\boldsymbol{a}) = \mathcal{N}\left(\boldsymbol{a} \mid \boldsymbol{T}\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2, \Omega_2^{-1} + \boldsymbol{T}\Omega_1^{-1}\boldsymbol{T}^{\mathrm{T}}\right) \tag{25}$$

We can simply apply the lemma to eq. (22) and $\boldsymbol{x}_v \mid \boldsymbol{y}_v \sim \mathcal{N}(\boldsymbol{V}\boldsymbol{y}_v, \Sigma_{\boldsymbol{x}})$, which gives the result in eq. (24). This ends the proof.

To further simplify the notation, we make the assumption that the marginal noise on $\boldsymbol{x}$ is not too strong to conceal the structure information, i.e., $\det(\Sigma_{\boldsymbol{x}}) \ll \det(\Lambda^{-1})$, the PDF function can be obtained,

$$\begin{aligned} P_{\boldsymbol{V}}(\boldsymbol{X} \mid \boldsymbol{W}) &= \frac{1}{Z(\boldsymbol{W})} \prod_{v \in \mathcal{V}} \varphi_1(v) \prod_{\{v,u\} \in \mathcal{E}} \varphi_2(u, v) \\ \text{with } \varphi_1(v) &= \exp(-(\nu + d_v)\|\boldsymbol{V}^\dagger \boldsymbol{x}_v\|^2) \\ \text{and } \varphi_2(u, v) &= \exp\left\{w_{uv}(\boldsymbol{x}_u^\top \boldsymbol{V}^{\dagger\top} \boldsymbol{V}^\dagger \boldsymbol{x}_v)\right\}, \end{aligned} \tag{26}$$

where $\boldsymbol{V}^\dagger$ is the Moore–Penrose pseudo inverse of matrix $\boldsymbol{V}$. This is exactly eq. (2) thus ends the derivation.

A.2 The Optimization Objective on Homogeneous Graph Structure Learning

Here we derive the optimization objective for GSL on homogeneous grahs, which starts from eq. (3) and targets to eq. (4). According to eq. (2), the parameters contain the weighting matrix $\boldsymbol{W}$ (or Laplacian matrix equivalently), transformation matrix $\boldsymbol{V}^\dagger$ and we view the scalar ν considered as hyper-paramter. This yields a MAP problem as introduced in eq. (3), we restated as follows,

$$\begin{aligned}\boldsymbol{W}^*, \boldsymbol{V}^* &= \arg\max_{\boldsymbol{W},\boldsymbol{V}} \log P_{\boldsymbol{V}}(\boldsymbol{W} \mid \boldsymbol{X}) \\ &= \arg\max_{\boldsymbol{W},\boldsymbol{V}} \log P_{\boldsymbol{V}}(\boldsymbol{X} \mid \boldsymbol{W}) + \log P(\boldsymbol{W}) + \log P(\boldsymbol{V}).\end{aligned} \tag{27}$$

Current graph structure learning (GSL) algorithms (Dong et al., 2016; Pu et al., 2021a; Kumar et al., 2019) can be considered as a special case of the above MAP problem when $\boldsymbol{V}^\dagger$ is rectangular orthogonal (semi-orthogonal) Matrix, so that $\boldsymbol{V}^{\dagger\top}\boldsymbol{V}^\dagger = \boldsymbol{I}$. In such a scenario, the likelihood function of eq. (2) reduces to the same form as the Gaussian Graphical Model with,

$$P(\boldsymbol{X} \mid \boldsymbol{W}) = \frac{1}{Z(\boldsymbol{W})} \prod_{v\in\mathcal{V}} \exp(-(\nu + d_v)\|\boldsymbol{x}_v\|^2) \cdot \prod_{\{v,u\}\in\mathcal{E}} \exp\left\{w_{uv}(\boldsymbol{x}_u^\top \boldsymbol{x}_v)\right\}. \tag{28}$$

The only parameter left is the weight matrix $\boldsymbol{W} = \{w_{ij}\}_{i,j\in\{1:N\}}$ representing the graph structure. Then the problem becomes, $\boldsymbol{W}^* = \arg\max_{\boldsymbol{W}} \log P(\boldsymbol{X} \mid \boldsymbol{W}) + \log P(\boldsymbol{W})$. One would care about the log-likelihood of the signal $\boldsymbol{X}$ for convenient optimization,

$$\begin{aligned}\log\left[P\left(\boldsymbol{X} \mid \boldsymbol{W}\right)\right] &= \sum_{v\in\mathcal{V}} -(d_v + \nu)\|\boldsymbol{x}_v\|^2 + \sum_{u\neq v} w_{uv}\boldsymbol{x}_u^\top \boldsymbol{x}_v - \log Z(\boldsymbol{W}) \\ &= -\nu\sum_{u=1}^{N}\|\boldsymbol{x}_u\|^2 - \sum_{u,v}\boldsymbol{L}_{uv}\boldsymbol{x}_u^\top\boldsymbol{x}_v - \log Z(\boldsymbol{W}),\end{aligned} \tag{29}$$

The step comes from the fact that $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$. Please note that the summation changed from $u \neq v$ to all the possible $\{u, v\}$ to transit to the Laplacian matrix. Then we obtain the log-likelihood in matrix form,

$$\begin{aligned}\log\left[P\left(\boldsymbol{X} \mid \boldsymbol{W}\right)\right] &= -\nu\sum_{u=1}^{N}\|\boldsymbol{x}_u\|^2 - \sum_{u,v}\boldsymbol{L}_{uj}\boldsymbol{x}_u^\top\boldsymbol{x}_v - \log Z(\boldsymbol{W}) \\ &= -\nu\,\mathrm{tr}(\boldsymbol{X}^\top\boldsymbol{X}) - \mathrm{tr}(\boldsymbol{X}^\top\boldsymbol{L}\boldsymbol{X}) + \log\det(\boldsymbol{L} + \nu\boldsymbol{I}) + \text{Constant}.\end{aligned} \tag{30}$$

The last step is derived given the only term related to the optimization goal in $Z(\boldsymbol{W})$ is the log-determinant of the precision matrix (negative log-determinant of the covariance matrix). ν works as the factor balancing the node-wise and pair-wise effects. The pair-wise potentials are scaled by parameter $\boldsymbol{L}_{uv}$ which encodes the connection strength. In graph signal processing, the second term, in the quadratic form of the graph Laplacian, is called the "smoothness" of the signal on the graph. As shown in Zhou and Schölkopf (2004), it can be measured in terms of pair-wise difference of node signals,

$$\mathrm{tr}(\boldsymbol{X}^\top\boldsymbol{L}\boldsymbol{X}) = \frac{1}{2}\sum_{\{u,v\}} w_{uv}\|\boldsymbol{x}_u - \boldsymbol{x}_v\|^2, \tag{31}$$

given that $\sum_v w_{uv} = \boldsymbol{D}_{uu}$. Intuitively this term suggests, if two signal vectors x_u and x_v from a smooth set reside on two well-connected nodes with large w_{uv}, they are expected to have a small dissimilarity.

Plug eq. (30) into the training objective will give the same form as Graphical Lasso (Banerjee et al., 2008). However, the computationally demanding log-determinant term makes the problem difficult to solve (Kalofolias, 2016a). Thus, Egilmez et al. (2017) consider a relaxed optimization on the lower bound instead.

Lemma A.2 ***The Log-determinant Lower Bound.***

The optimization objective,

$$\arg\min_{\Lambda} -\sum_{v}\log\left(\det(\Lambda)_v\right),$$

can be relaxed through

$$\arg\min_{\Lambda} -\sum_{v} \log\left(\operatorname{diag}(\Lambda)_v\right).$$

Proof. By using Hadamard's inequality, we get,$\det(\Lambda) \leq \prod_v \Lambda_{vv}$. So that,

$$\log(\det(\Lambda)) \leq \log\left(\prod_v \Lambda_{vv}\right) = \sum_v \log\left(\Lambda_{vv}\right). \tag{32}$$

After rearranging the terms, we derive the following lower bound that $-\sum_v \log\left(\operatorname{diag}(\Lambda)_v\right) \leq -\log\det(\Lambda)$, which ends our proof.

Using lemma A.2 and substituting the log-determinant with eq. (30), we can further derive the optimization objective as,

$$\begin{aligned}
&\arg\max_{\boldsymbol{W}} \log P(\boldsymbol{X} \mid \boldsymbol{W}) + \log P(\boldsymbol{W}) \\
\rightarrow &\arg\max_{\boldsymbol{L}} -\nu \operatorname{tr}((\boldsymbol{X}^\top \boldsymbol{X}) - \operatorname{tr}(\boldsymbol{X}^\top \boldsymbol{L}\boldsymbol{X}) + \mathbf{1}^\top \log\left(\operatorname{diag}(\boldsymbol{L} + \nu\boldsymbol{I})_v\right) + \log P(\boldsymbol{W}) \\
= &\arg\min_{\boldsymbol{W}} \underbrace{\sum_{ij} w_{ij}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}_{S(\boldsymbol{X},\boldsymbol{W})} \underbrace{-\mathbf{1}^\top \log\left(\boldsymbol{W}\cdot\mathbf{1}\right) - \log P(\boldsymbol{W}) + \nu\|\boldsymbol{X}\|^2|}_{\Omega(\boldsymbol{W})} \\
\equiv &\arg\min_{\boldsymbol{W}} S(\boldsymbol{X}, \boldsymbol{W}) + \Omega(\boldsymbol{W}),
\end{aligned} \tag{33}$$

The training objective consists of graph-signal fidelity term $S(\boldsymbol{X}, \boldsymbol{W})$ and a structural regularizer $\Omega(\boldsymbol{W})$, which is exactly our training objective in eq. (4) if we absorb the L^2 regularizer into the negative log-prior.

Remark. To transit from the precision matrix MAP estimation through the parameterization of hidden Markov Networks Ghahramani (2001); Yuan and Lin (2007) to graph structure learning problem, three important assumptions are made: 1) Each feature dimension in $\boldsymbol{X}$ contributes equally to the structure estimation, i.e,$\boldsymbol{V}^\dagger$ is semi-orthogonal; 2) the variance of $\boldsymbol{x}$, Σ_x is considered to be relatively small to the feature generation covariance across all the nodes, so the graph structure dominates the conditional covariance matrix of $\boldsymbol{X}$, i.e. $\det(\Sigma_{\boldsymbol{x}}) \ll \det(\Lambda^{-1})$, and 3) The log-determinant of the partition function is relaxed by the lower-bound in the optimization.

B THEORETICAL RESULTS ON HETEROGENEOUS GRAPH STRUCTURE LEARNING

B.1 The Data-generating Process for Heterogeneous Graphs

Similarly, we want to derive the likelihood term $P(\{\boldsymbol{x}_v\} \mid \mathbf{W}, \{\boldsymbol{B}_r\})$ so that we can formalize the MAP estimation objective. We first model the joint PDF of $\{\boldsymbol{x}_v\}$ and $\{\boldsymbol{y}_v\}$ through the following form,

$$\begin{aligned}
&P(\{\boldsymbol{x}_v\}, \{\boldsymbol{y}_v\} \mid \mathbf{W}, \{\boldsymbol{B}_r\}) \\
=&P(\{\boldsymbol{y}_v\} \mid \mathbf{W}, \{\boldsymbol{B}_r\}) \prod_v P\left(\boldsymbol{x}_v \mid \boldsymbol{y}_v\right) \\
=&\frac{1}{Z}\prod_v \varphi_1\left(v\right) \prod_{u\neq v, r} \varphi_r\left(u, v\right) \prod_v P\left(\boldsymbol{x}_v \mid \boldsymbol{y}_v\right)
\end{aligned} \tag{34}$$

where the node-wise and edge-wise potential functions are respectively $\varphi_1(v) = \exp(-(d_v + \nu)\|\boldsymbol{y}_v\|^2)$ and $\varphi_r(u, v) = \exp(w_{uvr}\boldsymbol{y}_u^\top \boldsymbol{B}_r \boldsymbol{y}_v)$. Z is the partition function and d_v is the degree of node v, $d_v = \sum_{ru} w_{uvr}$. Multiple distribution types belonging to the exponential family satisfy the form in eq. (5). For simplicity, we view the hidden variables as the continuous embedding of labels and assume $P(\boldsymbol{y}_u, \boldsymbol{y}_v)$ is a joint Gaussian that has,

$$\operatorname{cat}([\boldsymbol{y}_u, \boldsymbol{y}_v]) \sim \mathcal{N}\left(\begin{bmatrix} \mu_{\boldsymbol{y}_v} \\ \mu_{\boldsymbol{y}_u} \end{bmatrix}, \begin{bmatrix} (d_u + \nu)\boldsymbol{I} & -w_{vur}\boldsymbol{B}_r \\ -w_{vur}\boldsymbol{B}_r^\top & (d_v + \nu)\boldsymbol{I} \end{bmatrix}^{-1}\right), \tag{35}$$

where cat is the concatenation operator. It is noted that the off-diagonal entries in the precision matrix are no longer a diagonal matrix but instead controlled by $\boldsymbol{B}_r$, which suggests the intertwined nature between different dimensions of the hidden variable vector $\{\boldsymbol{y}_v\}$. Given all the above necessary notations, we can now get the generation density of $\{\boldsymbol{y}_v\}$ in heterogeneous graphs as follows,

$$P(\{\boldsymbol{y}_v\} \mid \mathbf{W}, \{\boldsymbol{B}_r\}) \propto \exp\left(-\sum_v (d_v+\nu)\|\boldsymbol{y}_v\|^2 + \sum_{u\neq v, r} w_{vur}\boldsymbol{y}_u^\top \boldsymbol{B}_r \boldsymbol{y}_v\right), \tag{36}$$

For now we use "proportional to" and ignore the partition function to simplify the notation. The partition function will be handled with a lower bound in the optimization objective using lemma A.2.

From our assumption, the conditional distribution of $P(\boldsymbol{x}_v \mid \boldsymbol{y}_v)$ has a mean that is linearly transformed from $\boldsymbol{y}_v$, and a covariance that is independent of $\boldsymbol{y}_v$, which gives[7],

$$P(\boldsymbol{x}_v \mid \boldsymbol{y}_v) = \mathcal{N}\left(\boldsymbol{x}_v \mid \boldsymbol{V}_{\phi(v)}(\boldsymbol{y}_v - \mu_{\boldsymbol{y}_v}), \Sigma_{\boldsymbol{x}_v}\right) \tag{37}$$

where we assume that the linear transformation matrix $\boldsymbol{V}_{\phi(v)} \in \mathbb{R}^{K\times|C_{\phi(u)}|}$ is determined by the node type $\phi(v)$ and simply consider the covariance matrix $\Sigma_{\boldsymbol{x}_v} = \sigma^2\boldsymbol{I}$, which gives eq. (7). Thus, the joint distribution $\boldsymbol{y}_v$ and $\boldsymbol{x}_v$ also follow a Gaussian distribution, which gives,

$$\begin{bmatrix} \boldsymbol{x}_v \\ \boldsymbol{y}_v \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ \mu_{\boldsymbol{y}_v} \end{bmatrix}, \begin{bmatrix} \Sigma_{\boldsymbol{x}_v} + \boldsymbol{V}_{\phi(v)}\Sigma_{\boldsymbol{y}_v}\boldsymbol{V}_{\phi(v)}^\top & -\Sigma_{\boldsymbol{y}_v}\boldsymbol{V}_{\phi(v)}^\top \\ -\boldsymbol{V}_{\phi(v)}\Sigma_{\boldsymbol{y}_v} & \Sigma_{\boldsymbol{y}_v} \end{bmatrix}\right). \tag{38}$$

Until now we have derive the $P(\boldsymbol{x}_v \mid \boldsymbol{y}_v)$ and $P(\boldsymbol{y}_u, \boldsymbol{y}_v \mid \lambda_{uvr})$ where for simplicity the parameters are denoted as $\Theta = \{\mathbf{W}, \{\boldsymbol{B}_r\}\}$. Recall that $\boldsymbol{x} \coloneqq \text{cat}(\boldsymbol{x}_v, \boldsymbol{x}_u)$ and $\boldsymbol{y} \coloneqq \text{cat}(\boldsymbol{y}_v, \boldsymbol{y}_u)$. We can apply lemma A.1 and obtain the conditional distribution of $P(\boldsymbol{x} \mid \Theta)$ also being a Gaussian with the following joint likelihood function,

$$P(\boldsymbol{x} \mid \Theta) \propto \exp\left\{-\frac{1}{2}(\boldsymbol{x}^\top \Lambda_{\boldsymbol{x}|\Theta}\boldsymbol{x})\right\}, \tag{39}$$

where the mean $\mu_{x|\Theta} = 0$ and the precision matrix,

$$\Lambda_{\boldsymbol{x}|\Theta} = (\Sigma_{\boldsymbol{x}} + \boldsymbol{V}\Sigma_{\boldsymbol{y}|\Theta}\boldsymbol{V}^\top)^{-1} \tag{40}$$

with $\Sigma_{\boldsymbol{y}|\Theta} = \Lambda^{-1}_{\boldsymbol{x}|\Theta}$ the covariance matrix. $\boldsymbol{V} = \left[\boldsymbol{V}_{\phi(v)}^\top, \boldsymbol{V}_{\phi(u)}^\top\right]^\top$. If we consider the variance of $\boldsymbol{x}$ is relatively small compared to the graph emission matrix $\sigma_{\boldsymbol{y}}$, i.e. $\det(\Sigma_{\boldsymbol{x}}) \ll \det(\boldsymbol{V}\Sigma_{\boldsymbol{y}|\Theta}\boldsymbol{V}^\top)$, we obtain,

$$\begin{aligned} &\Lambda_{\boldsymbol{x}|\Theta} = (\Sigma_{\boldsymbol{x}} + \boldsymbol{V}\Sigma_{\boldsymbol{y}|\Theta}\boldsymbol{V}^\top)^{-1} \approx \\ &\left(\begin{bmatrix} \boldsymbol{U}_{\phi(u)} \\ \boldsymbol{U}_{\phi(v)} \end{bmatrix}^\top \begin{bmatrix} (d_u+\nu)\boldsymbol{I} & -w_{vur}\boldsymbol{B}_r \\ -w_{vur}\boldsymbol{B}_r^\top & (d_v+\nu)\boldsymbol{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{U}_{\phi(u)} \\ \boldsymbol{U}_{\phi(v)} \end{bmatrix}\right) \end{aligned} \tag{41}$$

where $\boldsymbol{U}_{\phi(u)} \in \mathbb{R}^{|C_{\phi(u)}|\times K}$ is the Moore–Penrose Pseudo inverse of matrix $\boldsymbol{V}_{\phi(u)}$ which satisfies $\boldsymbol{V}_{\phi(u)}\boldsymbol{U}_{\phi(u)}\boldsymbol{V}_{\phi(u)} = \boldsymbol{V}_{\phi(u)}$ and $\boldsymbol{U}_{\phi(u)}\boldsymbol{V}_{\phi(u)}\boldsymbol{U}_{\phi(u)} = \boldsymbol{U}_{\phi(u)}$. For the details of such a pseduo inverse method, we refer to Barata and Hussein (2011) for a tutorial. With such a derivation, we can get

$$-\Lambda_{\boldsymbol{x}|\Theta} = -(d_u+\nu)\boldsymbol{U}_{\phi(u)}^\top\boldsymbol{U}_{\phi(u)} - (d_v+\nu)\boldsymbol{U}_{\phi(v)}^\top\boldsymbol{U}_{\phi(v)} + 2w_{uvr}\boldsymbol{U}_{\phi(u)}^\top\boldsymbol{B}_r\boldsymbol{U}_{\phi(v)}. \tag{42}$$

We then combine eq. (42) and eq. (39) to obtain the overall likelihood for the whole graph signals,

$$\begin{aligned} P(\{\boldsymbol{x}_v\} \mid \mathbf{W}, \{\boldsymbol{B}_r\}) &\propto \prod \exp\left\{-\frac{1}{2}(\boldsymbol{x}^\top\Sigma^{-1}_{\boldsymbol{x}|\Theta}\boldsymbol{x})\right\} \\ &= \prod_v \exp(-(\nu+d_v)\|\boldsymbol{U}_{\phi_v}\boldsymbol{x}_v\|_{\text{F}}^2) \cdot \prod_{u,v,r} \exp\left\{w_{uvr}(\boldsymbol{x}_u^\top\boldsymbol{U}_{\phi(u)}^\top\boldsymbol{B}_r\boldsymbol{U}_{\phi(v)}\boldsymbol{x}_v)\right\} \end{aligned} \tag{43}$$

It is worth pointing out that the matrix $\boldsymbol{U}_{\phi(u)}^\top\boldsymbol{B}_r\boldsymbol{U}_{\phi(v)}$ is positive semi-definite if the signals on u and v lies in the same space. This is not always true but in this paper we make this assumption.

[7] From now on, all the precision matrices are termed as Λ and covariance matrix as Σ.

B.2 The Optimization Objective for Heterogeneous Graph Structure Learning

We first consider the likelihood function:

$$\begin{aligned} &P(\{\boldsymbol{x}_v\} \mid \mathbf{W}, \{\boldsymbol{B}_r\}) \propto \prod_v \varphi_1(v) \prod_{u \neq v, r} \varphi_r(u, v) \\ &\text{with } \varphi_1(v) = \exp(-(\nu + d_v)\|\boldsymbol{U}_{\phi_{(v)}}\boldsymbol{x}_v\|^2) \text{ and } \varphi_r(u,v) = \exp\left\{w_{uvr}(\boldsymbol{x}_u^\top \boldsymbol{U}_{\phi(u)}^\top \boldsymbol{B}_r \boldsymbol{U}_{\phi(v)}\boldsymbol{x}_v)\right\}. \end{aligned} \tag{44}$$

Lemma B.1 *If $\boldsymbol{P} \in \mathcal{R}^{n\times n}$ is positive-semidefinite, there always exits a matrix $\boldsymbol{Q}$ that has the same rank as $\boldsymbol{P}$, so that $\boldsymbol{Q}^\top\boldsymbol{Q} = \boldsymbol{P}$.*

Recall that the type of edge can help determine the node types connected (reversely is not possible). With the lemma, we can use a relation-specific matrix $\boldsymbol{M}_r \in \mathcal{R}^{K\times K'}$ to replace $\boldsymbol{x}_u^\top \boldsymbol{M}_r \boldsymbol{M}_r^\top \boldsymbol{x}_v = \boldsymbol{x}_u^\top \boldsymbol{U}_{\phi(u)}^\top \boldsymbol{B}_r \boldsymbol{U}_{\phi(v)}\boldsymbol{x}_v$ for each type of relation M_r. So the log-potential becomes,

$$\sum_{u\neq v, r} w_{uvr}(\boldsymbol{x}_u^\top \boldsymbol{M}_r \boldsymbol{M}_r^\top \boldsymbol{x}_v) - \sum_v (\nu + d_v)\|\boldsymbol{U}_{\phi(v)}\boldsymbol{x}_v\|^2 \tag{45}$$

It is noted that the intra-type node connection (nodes connected within the same type) can be viewed as a special relation type. In such a scenario, the connectivity matrix $\boldsymbol{B}_r$ is a positive symmetric matrix, so it is possible to find a matrix $\boldsymbol{M}_r$ such that $\sum_v \|\boldsymbol{x}_v^\top \boldsymbol{U}_{\phi(v)}^\top \boldsymbol{B}_r \boldsymbol{U}_{\phi(v)}\boldsymbol{x}_v\|^2 = \sum_v \|\boldsymbol{M}_r\boldsymbol{x}_v\|^2$. It is also easier for us to unify the notation. This leads to the following simplification,

$$\begin{aligned} &\sum_{u\neq v, r} w_{uvr}(\boldsymbol{x}_u^\top \boldsymbol{M}_r \boldsymbol{M}_r^\top \boldsymbol{x}_v) - \sum_v (\nu + d_v)\|\boldsymbol{U}_{\phi(v)}\boldsymbol{x}_v\|^2 \\ =& \sum_{u\neq v, r} w_{uvr}(\boldsymbol{x}_u^\top \boldsymbol{M}_r \boldsymbol{M}_r^\top \boldsymbol{x}_v) - \sum_r \sum_v (\nu + d_v)\|\boldsymbol{M}_r^\top \boldsymbol{x}_v\|^2 \\ =& -\frac{1}{2}\sum_{u,v,r} w_{uvr}\|\boldsymbol{M}_r^\top \boldsymbol{x}_u - \boldsymbol{M}_r^\top \boldsymbol{x}_v\|^2 - \nu \sum_r \sum_v \|\boldsymbol{M}_r^\top \boldsymbol{x}_v\|^2 \end{aligned} \tag{46}$$

The last step comes from the fact that $\sum_{r,v} w_{uvr} = d_v$ and the submission subscript changes from $u \neq v$ to all possible combinations u, v. If we take $\nu \to 0$, the only term left is

$$-\frac{1}{2}\sum_{u,v,r} w_{uvr}\|\boldsymbol{M}_r^\top \boldsymbol{x}_u - \boldsymbol{M}_r^\top \boldsymbol{x}_v\|^2.$$

We next consider a low-rank approximation such that,

$$\boldsymbol{M}_r\boldsymbol{M}_r^\top \approx \boldsymbol{U}_{\phi(u)}^\top \boldsymbol{B}_r \boldsymbol{U}_{\phi(v)},$$

which would further simplify the parameter setting. Our empirical study shows that a rank-1 relation embedding $\boldsymbol{e}_r \in \mathcal{R}^K$ is sufficient in the approximation. Thus, the training objective leads to,

$$\arg\min_{\mathbf{W}, \boldsymbol{E}} \sum_{v\leq u, r} w_{vur}\|\boldsymbol{e}_r^\top \cdot (\boldsymbol{x}_v - \boldsymbol{x}_u)\|^2 + \Omega(\mathbf{W}) + \Omega(\boldsymbol{E}) \tag{47}$$

which is exactly eq. (9). It is noted that we absorb the partition function into the regularizer. And such a term can be relaxed by the log-barrier on the degree matrix, similar to what we did in eq. (33) using lemma A.2.

B.3 The relation embedding update step in HGSL optimization

Here we gave a detailed analysis of algorithm 1 when updating relation embeddings, especially when we transit from eq. (12) to eq. (13). We first repeat the training objective here,

$$\arg\min_{\{\boldsymbol{e}_r\}} \quad \sum_{vur} w_{vur}\|\boldsymbol{e}_r^\top \cdot (\boldsymbol{x}_v - \boldsymbol{x}_u)\|^2 + \lambda_1\|\boldsymbol{E}\|^2 + \lambda_2\|\boldsymbol{E}\|_1. \tag{48}$$

As we show in appendix B.2, the training objective is equivalent to the following:

$$\underset{\{\boldsymbol{e}_r\}}{\arg\min} \quad -\sum_{vur} w_{vur}(\boldsymbol{x}_u^\top \boldsymbol{e}_r \boldsymbol{e}_r^\top \boldsymbol{x}_v) + \lambda_1 \sum_r \boldsymbol{e}_r^\top \cdot \boldsymbol{e}_r + \lambda_2 \sum_r \|\boldsymbol{e}_r\|_1. \tag{49}$$

It is possible to disentangle through the relation types and get the sub-problem as,

$$\underset{\boldsymbol{e}_r}{\arg\min} \quad -\sum_{vu} w_{vur}(\boldsymbol{x}_u^\top \boldsymbol{e}_r \boldsymbol{e}_r^\top \boldsymbol{x}_v) + \lambda_1 \boldsymbol{e}_r^\top \cdot \boldsymbol{e}_r + \lambda_2 \|\boldsymbol{e}_r\|_1. \tag{50}$$

Denote $\boldsymbol{P}_r = \sum_{u,v} w_{uvr} \boldsymbol{x}_u \boldsymbol{x}_v^\top$, and notice that $\boldsymbol{x}_v^\top \boldsymbol{e}_r \cdot \boldsymbol{e}_r^\top \boldsymbol{x}_u = \boldsymbol{e}_r^\top \boldsymbol{x}_u \boldsymbol{x}_v^\top \boldsymbol{e}_r$. We get $-\sum_{u,v} w_{uvr} \left(\boldsymbol{x}_v^\top \boldsymbol{e}_r\right) \left(\boldsymbol{e}_r^\top \boldsymbol{x}_u\right) = -\boldsymbol{e}_r^\top \boldsymbol{P}_r \boldsymbol{e}_r$ and the objective function simplifies to:

$$\begin{aligned} \mathcal{O}(\boldsymbol{e}_r) &= -\boldsymbol{e}_r^\top \boldsymbol{P}_r \boldsymbol{e}_r + \lambda_1 \boldsymbol{e}_r^\top \boldsymbol{e}_r + \lambda_2 \|\boldsymbol{e}_r\|_1 \\ &= \boldsymbol{e}_r^\top (\lambda_1 \boldsymbol{I} - \boldsymbol{P}_r) \boldsymbol{e}_r + \lambda_2 \|\boldsymbol{e}_r\|_1 \end{aligned} \tag{51}$$

We wish to ensure the $\boldsymbol{e}_r \geq 0$ to be non-negative so that we set up the Lagrangian with Non-negativity Constraints using Lagrange multipliers $\gamma \geq 0$, and obtain,

$$L(\boldsymbol{e}_r, \gamma) = \boldsymbol{e}_r^\top (\lambda_1 \boldsymbol{I} - \boldsymbol{P}_r) \boldsymbol{e}_r + \lambda_2 \boldsymbol{e}_r^\top 1 - \gamma^\top \boldsymbol{e}_r \tag{52}$$

Setting the derivative to zero, we get $\frac{\partial L}{\partial \boldsymbol{e}_r} = 2(\lambda_1 \boldsymbol{I} - \boldsymbol{P}_r)\boldsymbol{e}_r + \lambda_2 - \gamma = 0$ and Complementary Slackness: $\gamma_i \boldsymbol{e}_{ri} = 0, \forall i$. Solve for $e_r > 0$, we get $\lambda_i = 0$, so the stationarity condition becomes:

$$\frac{\partial L}{\partial \boldsymbol{e}_r} = 2(\lambda_1 \boldsymbol{I} - \boldsymbol{P}_r)\boldsymbol{e}_r + \lambda_2 \mathbf{1} = 0 \tag{53}$$

And this leads to the analytical solution, with

$$\boldsymbol{e}_r = \frac{\lambda_2}{2}(\boldsymbol{P}_r - \lambda_1 \boldsymbol{I})^{-1} \mathbf{1} \tag{54}$$

We assume $\|\frac{1}{\lambda_1}\boldsymbol{P}_r - \boldsymbol{I}\| < 1$, thus we can use first-order Neumann Series Expansion and yield:

$$\boldsymbol{e}_r \approx \frac{\lambda_2}{2\lambda_1}(\frac{1}{\lambda_1}\boldsymbol{P}_r - 2\boldsymbol{I})\mathbf{1} \tag{55}$$

replace $\boldsymbol{P}_r = \sum_{u,v} w_{uvr} \boldsymbol{x}_u \boldsymbol{x}_v^\top$, we obtain:

$$\boldsymbol{e}_r = \frac{\lambda_2}{2\lambda_1}(\frac{1}{\lambda_1}\sum_{u,v} w_{uvr} \boldsymbol{x}_u \boldsymbol{x}_v^\top - 2\boldsymbol{I})\mathbf{1} \tag{56}$$

Looking into each dimension will yield eq. (13), which has an entry-wise solution as:

$$\boldsymbol{e}_{r,k}^{t+1} = \frac{\lambda_2}{2\lambda_1}(\frac{1}{\lambda_1}\sum_{\{v,u,r\}\in\mathcal{E}_r'} w_{vur}^t \ \boldsymbol{x}_{v,k} \cdot \boldsymbol{x}_{u,k} - 2) \tag{57}$$

In practice, we rewrite $\alpha = \frac{\lambda_2}{2\lambda_1^2}$ and $\beta = \frac{\lambda_2}{\lambda_1}$, and the solution becomes

$$\boldsymbol{e}_{r,k}^{t+1} = \alpha \sum_{\{v,u,r\}\in\mathcal{E}_r'} w_{vur}^t \ \boldsymbol{x}_{v,k} \cdot \boldsymbol{x}_{u,k} - \beta \tag{58}$$

We do a hyperparameter search for the α and β.

C THE EXPRESSIVENESS OF H2MN

Designing a statistical model that effectively captures the heterogeneity of node and edge types is a central challenge in extending Graph Structure Learning (GSL) algorithms to Heterogeneous Graph Structure Learning (HGSL). This challenge stems from three key aspects. First, the complex dependencies among various components, including graph structures with multiple relation types, node features, and node labels, necessitate a model that can capture these multifaceted dependencies. Second, there exists an intertwined dependency among edges with different relation types, and modeling this interdependency is crucial. Finally, the node feature generation needs to consider the influence on different node labels and relation types.

We first answer the question about why the dependencies exist and how our model design captures the dependency among edges with different relation types.

Propensity C.1 *The probabilities of edges in the same graph having specific relation types are not independent under the assumption of H2MN.*

To illustrate, we consider a simple path in the graph $\mathcal{P}: v_1 \overset{e_1}{\longleftrightarrow} v_2 \overset{e_2}{\longleftrightarrow} v_3$, where e_i denote edges, $e_i = r$ indicates the edge has relation type r. We consider the probability of e_2 having type r given e_1 having r', $P(e_2 = r \mid e_1 = r')$, with r and r' two different relation types. Considering the variables follow the Markov property, we can derive that $P(e_2 = r \mid e_1 = r') = \sum_{v_2} P(e_2 = r \mid v_2) \cdot P(v_2 \mid e_1 = r')$. There is no way we can further eliminate the dependency on e_1, which means $P(e_2 = r \mid e_1 = r') \neq P(e_2 = r)$. So the random variables e_1 and e_2 are not independent. This necessitates joint statistical modeling of the edges with various relation types rather than treating them independently.

The correlation modeling of H2MN We wish to emphasize that, even though we consider the pair-wise HMN to approximate the overall joint probability distribution function, the approximation preserves the dependency among edges with diverse relation types. To illustrate, we consider the example of a path in the graph $\mathcal{P}: v_1 \overset{e_1}{\longleftrightarrow} v_2 \overset{e_2}{\longleftrightarrow} v_3$ again, and factorize the joint PDF as (omitting the node-wise potential $\varphi(v)$ for simplicity),

$$P(\mathcal{P}) = \frac{1}{Z}\varphi_r(v_1, v_2)\varphi_{r'}(v_2, v_3)$$

where r is the type for e_1 and r' is the type for e_2, and Z is the partition function. And then we can obtain

$$P(e_2 \mid e_1) = \frac{P(e_2, e_1)}{P(e_1)} = \frac{\sum_{v_1, v_2, v_3} \varphi_r(v_1, v_2) \cdot \varphi_{r'}(v_2, v_3)}{\sum_{v_1, v_2, v_3, e_2} \varphi_r(v_1, v_2) \cdot \varphi_{r'}(v_2, v_3)}$$

In this expression, the influence of e_1 is retained in the numerator of the fractional above, demonstrating that our model inherently captures the interdependencies among edges with different relation types. The correlation is implicitly modeled within the parameterization of the HMN.

D EXTENSION OF THE ALGORITHM

D.1 Learning the heterogeneous graphs when the node labels are observable.

When labels are observable, such as in IMDB and ACM dataset Fu et al. (2020), we wish to learn $\mathbf{W}$ and $\{\boldsymbol{B}_r\}$ directly from the labels $\{\boldsymbol{y}_v\}_{v \in \mathcal{V}}$. To this end, we consider the HGSL problem as a MAP estimation problem parameterized by eq. (6) and define $\Omega(\cdot)$ as the negative log prior, which gives,

$$\begin{aligned} &\arg\max_{\mathbf{W}, \{\boldsymbol{B}_r\}} \log P(\mathbf{W}, \{\boldsymbol{B}_r\} \mid \{\boldsymbol{y}_v\}) \\ =&\arg\max_{\mathbf{W}, \{\boldsymbol{B}_r\}} \log P(\{\boldsymbol{y}_v\} \mid \mathbf{W}, \{\boldsymbol{B}_r\}) - \Omega(\mathbf{W}) - \Omega(\{\boldsymbol{B}_r\}). \end{aligned} \tag{59}$$

The log-likelihood follows eq. (6), thus we derive the training objective as,

$$\arg\max_{\mathbf{W}, \{\boldsymbol{B}_r\}} \sum_{u \neq v, r} (\boldsymbol{y}_u^\top \boldsymbol{B}_r \boldsymbol{y}_v) \cdot w_{uvr} - \sum_v (d_v + \nu) \|\boldsymbol{y}_v\|_{\mathrm{F}}^2 - \Omega(\mathbf{W}) - \Omega(\{\boldsymbol{B}_r\}). \tag{60}$$

Given that $d_v = \sum_{r,u} w_{vur}$, we can further derive first two terms as,

$$\begin{aligned}
&-\sum_v (d_v+\nu)\|\boldsymbol{y}_v\|^2 + \sum_{\{u\neq v,r\}\in\mathcal{E}} w_{vur}\boldsymbol{y}_u^\top \boldsymbol{B}_r \boldsymbol{y}_v \\
=&-\nu\sum_v \|\boldsymbol{y}_v\|^2 + \sum_{u\neq v,r} w_{vur}\boldsymbol{y}_u^\top \boldsymbol{B}_r \boldsymbol{y}_v - \sum_v d_v\|\boldsymbol{y}_v\|^2 \\
=&-\frac{1}{2}\sum_{u,v,r} w_{uvr}\|\operatorname{vec}(\boldsymbol{B}_r)\odot(\boldsymbol{y}_u\otimes\mathbf{1}_v - \boldsymbol{y}_v\otimes\mathbf{1}_u)\|^2 - \nu\sum_v\|\boldsymbol{y}_v\|^2
\end{aligned} \tag{61}$$

Here vec is the vectorization function that flattens the matrix into a long vector, $\otimes$ is the Kronecker product, $\odot$ is the element-wise product, and $\mathbf{1}_v$ is the all-1 vector that has the same size as y_v. The node-wise effects solely related to $\boldsymbol{y}_v$ do not affect the optimization over $\mathbf{W}$ and $\{\boldsymbol{B}_r\}$ so they can be omitted. Thus, we get,

$$\arg\min_{\mathbf{W},\{\boldsymbol{B}_r\}} \sum_{u,v,r} \operatorname{vec}(\boldsymbol{B}_r)\|\boldsymbol{y}_u\otimes\mathbf{1}_v - \boldsymbol{y}_v\otimes\mathbf{1}_u\|_2^2\cdot w_{uvr} + \Omega(\mathbf{W}) + \Omega(\{\boldsymbol{B}_r\}). \tag{62}$$

This is the training objective we are going to use if we want to learn the graph structure from node labels $\{\boldsymbol{y}_v\}$.

E ALGORITHM ANALYSIS: INTUITION AND THEORY

In the vanilla GSL problem, it is widely acknowledged that "on a homophily graph, minimizing the smoothness could lead to a meaningful graph structure", just like what we did in section 3. In appendix E.2, we gave the first attempt to understand how the homophily concepts and smoothness assumption would impact the GSL algorithms from the perspective of the data-generating process. In more complex graphs, e.g. heterogeneous graphs, the definition of smoothness and homophily is never properly stated. We mange to link the smoothness in heterogeneous graphs with the DGP proposed in appendix E.1. Next, in appendix E.3, We move to heterogeneous cases to see how we can interpret the homophily in heterogeneous graphs through the H2MN, and derive the condition that our algorithm, proposed in algorithm 1, will converge to a meaningful solution.

E.1 Understanding the Graph-data Fidelity Term through Generalized Smoothness

In the HGSL training objective as in eq. (9), the first term in the training objective is the graph-data fidelity term,

$$\begin{aligned}
S(\boldsymbol{X},\boldsymbol{E},\mathbf{W}) &= \sum_{\{v,u,r\}\in\mathcal{E}} w_{vur}\|\boldsymbol{e}_r\circ(\boldsymbol{x}_v-\boldsymbol{x}_u)\|^2 \\
&= \langle \mathbf{W}, \|(\boldsymbol{X}\otimes\mathbf{1}-\mathbf{1}\otimes\boldsymbol{X})\otimes\boldsymbol{E}\|_F^2\rangle,
\end{aligned} \tag{63}$$

where $\circ$ is the element-wise product and $\|\cdot\|_F$ is the Frobenius norm along the last dimension which collapses the tensor from dimension 4 to 3. $\boldsymbol{X} = [\boldsymbol{x}_1,...,\boldsymbol{x}_{|\mathcal{V}|}]^{\boldsymbol{T}} \in \mathbb{R}^{|\mathcal{V}|\times K}$, $\boldsymbol{E} = [\boldsymbol{e}_1,\ldots,\boldsymbol{e}_{|\mathcal{R}|}]^{\boldsymbol{T}} \in \mathbb{R}^{|\mathcal{R}|\times K}$, and $\mathbf{1}\in\mathbb{R}^{|\mathcal{V}|\times K}$ is an all-one matrix. $\langle\cdot,\cdot\rangle$ and $\otimes$ are the tensor inner product and outer product, respectively.

The graph-data fidelity term in eq. (4) measures dimension-wise smoothness via $\sum_{u,v}\boldsymbol{W}_{u,v}\|\boldsymbol{x}_v-\boldsymbol{x}_u\|^2$, and sum over different dimensions without any weighting. Similarly in HGSL, if the relation embedding $\boldsymbol{e}_r$ is normalized and constrained to be positive, the graph-data fidelity term can be interpreted as a weighted smoothness, as illustrated in fig. 5. The term can be considered as a *reweighted smoothness* scheme that first measures dimension-wise smoothness, and then integrates it by emphasizing specific signal dimensions according to r. Using the movie review dataset again as an example, while determining whether a 'star in'-typed edge should be formed between two actor nodes, the model should put larger weights on the signal dimensions that represent the 'genres', but weigh less irrelevant ones such as 'company affiliation' or 'date'.

Different from the original smoothness, the contribution of each signal dimension to the overall smoothness is reweighted by $\boldsymbol{e}_r$. Intuitively, a heterogeneous graph is thought to be smooth if strongly connected nodes (with larger w_{vur}) have similar signal values in the dimensions emphasized by $\boldsymbol{e}_r$ for the relation r.

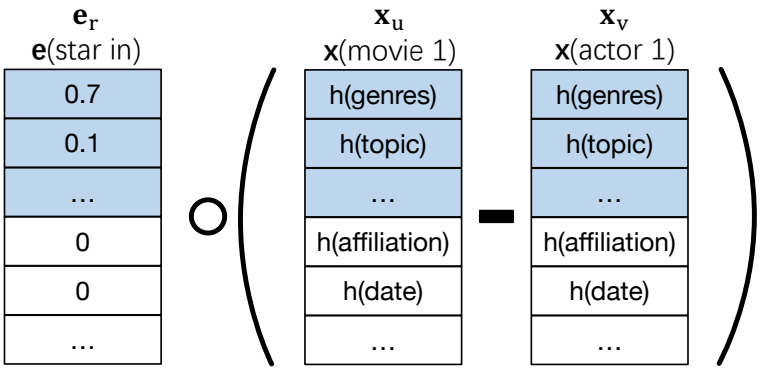


Figure 5: Visualization of the generalized smoothness.

E.2 The Optimization Conditions for Graph Structure Learning in Homogeneous Graphs

We will first derive the optimization condition based on understanding the homophily property in homogeneous graphs, and then we slowly move toward the heterogeneous graph cases.

We start by introducing the homophily ratio. The concept of homophily in networks was first proposed in McPherson et al. (2001), suggesting that a connection between similar entities occurs at a higher probability than among dissimilar entities. Statistically, this suggests that we have a higher probability of observing an edge between two nodes belonging to the same communities. Such a concept gives,

$$P(\boldsymbol{W}_{uv} = 1 \mid y_u = y_v) > P(\boldsymbol{W}_{uv} = 1 \mid y_u \neq y_v), \tag{64}$$

where y_u and y_v are the labels for nodes u and v. For simplicity, we denote $P(\boldsymbol{W}_{uv} = 1 \mid y_u = y_v) = p$ and $P(\boldsymbol{W}_{uv} = 1 \mid y_u \neq y_v) = q$. And this definition gives the homophily ratio in homogeneous graphs, as follows.

Expected Homophily Ratio.

$$\text{HR in homogeneous graphs} = \frac{P(\boldsymbol{W}_{uv} = 1 \mid y_u = y_v)}{P(\boldsymbol{W}_{uv} = 1 \mid y_u = y_v) + P(\boldsymbol{W}_{uv} = 1 \mid y_u \neq y_v)} = \frac{p}{p+q}.$$

Empirical Homophily Ratio. While a more practical measurement would be the empirical homophily ratio (EHR), defined as,

$$\text{EHR}\left(\mathcal{G}\right) = \frac{\sum_{\{u,v\}\in\mathcal{E}} \mathbb{I}\left(y_u = y_v\right)}{|\mathcal{E}|}. \tag{65}$$

Which in expectation gives expected homophily ratio.

We are interested in how the homophily parameters p and q are related to the GSL optimization solution. To this end, we first consider the case when $\boldsymbol{B}$ is a 2×2 matrix encoding only the probabilities of having an edge and no edge among nodes with the same label, p, and q, where,

$$\boldsymbol{B} = \begin{bmatrix} p & q \\ q & p \end{bmatrix}. \tag{66}$$

Usually, the parameterization considers the average degrees $\bar{d}$ and the total number of nodes N, so that $p = \bar{d}\hat{p}/N$ and $q = \bar{d}\hat{q}/N$ with $d \ll N$, $\hat{p} + \hat{q} = 1$. We consider the GSL problem when labels are given for simplicity and a

L^1 regularizer on $\boldsymbol{W}$, which yields the following optimization objective:

$$\begin{aligned}
&\arg\max_{\boldsymbol{W}} \log P(\boldsymbol{Y} \mid \boldsymbol{W}) + \log P(\boldsymbol{W}) \\
&= \arg\max_{\boldsymbol{W}} \sum_{i \le j} \boldsymbol{y}_i^\top \mathbf{B} \boldsymbol{y}_j \cdot \boldsymbol{W}_{ij} - \beta \|\boldsymbol{W}\|_1 \\
&= \arg\max_{\boldsymbol{W}} \sum_{\{i,j\}:y(i)=y(j)} p\boldsymbol{W}_{ij} + \sum_{\{i,j\}:y(i)\neq y(j)} q\boldsymbol{W}_{ij} - \beta \|\boldsymbol{W}\|_1 \\
&= \arg\max_{\boldsymbol{W}} \sum_{\{i,j\}:y(i)=y(j)} (p-q)\boldsymbol{W}_{ij} + \sum_{\{i,j\}:y(i)=y(j)} q\boldsymbol{W}_{ij} + \sum_{\{i,j\}:y(i)\neq y(j)} q\boldsymbol{W}_{ij} - \beta \|\boldsymbol{W}\|_1 \\
&= \arg\max_{\boldsymbol{W}} \quad (p-q) \sum_{\{i,j\}:y(i)=y(j)} \boldsymbol{W}_{ij} + (q-\beta)\|\boldsymbol{W}\|_1.
\end{aligned} \tag{67}$$

In such a situation, the training objective can be decomposed into two parts, and the optimization solution becomes a trade-off game between the two components. The first part suggests that the algorithm should put edges between nodes with the same label, and the second encourages the sparsity of the solution. To avoid trivial solution (i.e. simply maximizing $\|\mathbf{W}\|_1$ or setting $\|\mathbf{W}\|_1 = 0$) for the problem, we need to have $p - q > 0$ and $q - \beta < 0$, i.e the graph is homophily with $p > q$.

This suggests that, only on graphs with high homophily edge generation probability p, the GSL algorithm can achieve a meaningful solution. Thus, to conduct the algorithm analysis in the HGSL algorithm, we can similarly define the homophily and derive the conditions.

E.3 Algorithm Analysis for Heterogenous Graph Structure Learning

Similar to the homogeneous graphs, we wish to understand what setting of $\mathbf{B}_r, r \in \mathcal{R}$ could give a guarantee on the optimization problem of heterogeneous graph structure learning. We again consider the optimization problem when labels $\{\boldsymbol{y}_v\}$ are observable, as introduced in eq. (62) but ignore the regularizer for now. Considering all possible relation types as $r \in \mathcal{R}_{\phi(u)\phi(v)}$ and denoting an element of possible node label combinations within two node types as $(p, q) \in \mathcal{Y}_{\phi(u)} \times \mathcal{Y}_{\phi(v)}$, we can derive the optimization as follows,

$$\begin{aligned}
\mathbf{W}^* &= \arg\max_{\mathbf{W}} \sum_{u,v,r} \mathbf{y}_u^\top \mathbf{B}_r \mathbf{y}_v \mathbb{W}_{uvr} - \beta \|\mathbf{W}\|_1 \\
&= \arg\max_{\mathbf{W}} \sum_{r,p,q} \boldsymbol{B}_r[p,q] \sum_{\{u,v\}:(y(u)=p)\wedge(y(v)=q)} \mathbf{W}_{uvr} - \beta \|\mathbf{W}\|_1
\end{aligned} \tag{68}$$

where $\boldsymbol{B}_r[p,q] = P(w_{uvr} = 1 \mid \boldsymbol{y}_{u,p} = \boldsymbol{y}_{v,q} = 1)$ with $\boldsymbol{y}_{u,p}$ the p-th entry of the vector $\boldsymbol{y}_u$ and y_u is the label index of node u. The sum is over two parts: 1) over all relation types within one node type combination (in many scenarios this is only one) and 2) over all possible node labels combination.

Then we define a "representative" connection between two nodes within a combination of node types, that gives the highest probability of connection $\boldsymbol{B}_r[p^*, q^*]$, we can derive the summation as,

$$\begin{aligned}
&\sum_r \left\{ (\boldsymbol{B}_r[p^*,q^*] - \sum_{(p,q)\neq(p^*,q^*)} \boldsymbol{B}_r[p,q]) \sum_{(y_u,y_v)=(p^*,q^*)} \mathbf{W}_{uvr} \right\} \\
&+ \sum_r \left\{ \sum_{(p,q)\neq(p^*,q^*)} \boldsymbol{B}_r[p,q] (\sum_{(y_u,y_v)=(p^*,q^*)} \mathbf{W}_{uvr} + \sum_{(y_u,y_v)\neq(p^*,q^*)} \mathbf{W}_{uvr}) \right\} - \beta\|\mathbf{W}\|_1 \\
&= \sum_r \left\{ \underbrace{(\boldsymbol{B}_r[p^*,q^*] - \sum_{(p,q)\neq(p^*,q^*)} \boldsymbol{B}_r[p,q]) \sum_{(y_u,y_v)=(p^*,q^*)} \mathbf{W}_{uvr}}_{\text{Homophily Term}} \right\} + \underbrace{(\sum_r \sum_{(p,q)\neq(p^*,q^*)} \boldsymbol{B}_r[p,q] - \beta)\|\mathbf{W}\|_1}_{L^1 \text{ Regularizer}}
\end{aligned} \tag{69}$$

The algorithm aims at maximizing the above objective. The first part is to assign larger weights on the edges connecting homophily nodes, and the second part is a L^1 regularizer enforcing the sparsity on $\mathbf{W}$. The second

regularizer has to have a negative coefficient and the first term need to have a positive coefficient. Thus, the sufficient (not necessary) condition for the optimization problem to have meaningful solutions is stated as follows.

Optimization Condition: $\forall r \in \mathcal{R}$, there exists a pair (p^*, q^*) that satisfies,

$$\boldsymbol{B}_r[p^*, q^*] - \sum_{(p,q)\neq(p^*,q^*)} \boldsymbol{B}_r[p, q] > 0 \tag{70}$$

Homophily Ratio on Heterogeneous graphs. From the above optimization condition, the homophily ratio is consequently defined as,

$$\mathrm{HR}(\mathcal{G}, r) = \frac{\boldsymbol{B}_r[p^*, q^*]}{\sum_{(p,q)\neq(p^*,q^*)} \boldsymbol{B}_r[p, q])} \tag{71}$$

We then show that the homophily ratio is linked to the relaxed homophily ratio defined in eq. (16).

$$\mathrm{RHR}\left(\mathcal{G}, \Phi\right) = \frac{\sum_{\{u,v\}\in\mathcal{E}_\Phi} \mathbb{I}\left(y_u = y_v\right)}{|\mathcal{E}_\Phi|}, \tag{72}$$

Propensity E.1 *The HR and RHR are positively related.*

Proof. Consider a meta-path Φ consisting of relations $R_1, R_2, \ldots, R_{L-1}$.

We first compute the probability of having two nodes, u and v connected by a path, share the same label, i.e., $y_u = y_v = p$. We denote two nodes connected via a meta-path Φ as $u \leftrightarrow v$ via Φ. So that, $P\left(u \leftrightarrow v \text{ via } \Phi \mid y_u = y_v = p_0\right)$ is:

$$P_\Phi(p_0) = P\left(u \leftrightarrow v \text{ via } \Phi \mid y_u = y_v = p_0\right) = \sum_{p_1,\ldots,p_{L-1}} \prod_{l=0}^{L-1} \boldsymbol{B}_{R_l}\left[p_l, p_{l-1}\right]$$

where we assume the labels along the path are $p_0, p_1, \ldots, p_{L-1}$ and $p_L = p_0$.

Thus, the expected value of the relaxed homophily ratio is the expected ratio of same-label connections over all the possible paths. This is derived by the probability of having the starting node and targeting node with the same labels:

$$\begin{aligned}\mathbb{E}(\mathrm{RHR}) &= \sum_p P\left(y_u = p\right) \cdot P\left(y_v = p\right) \cdot P\left(u \leftrightarrow v \text{ via } \Phi \mid y_u = y_v = p\right) \\ &= \sum_p P\left(y_u = p\right) \cdot P\left(y_v = p\right) \cdot \sum_{p_1,\ldots,p_{L-1}} \prod_{l=0}^{L-1} \boldsymbol{B}_{R_l}\left[p_l, p_{l-1}\right]\end{aligned} \tag{73}$$

We consider the marginal distribution $P(y_u = p)$ is a uniform distribution , we have

$$\mathrm{RHR} \propto \sum_p \sum_{p_1,\ldots,p_{L-1}} \prod_{l=0}^{L-1} \boldsymbol{B}_{R_l}\left[p_l, p_{l-1}\right] \tag{74}$$

We focus on the dominant path, which is the path along the "representative relationship" $\boldsymbol{B}_{R_l}\left[p_l^*, p_{l-1}^*\right]$ for all l. This decomposes the RHR as,

$$\mathrm{RHR} \propto \sum_p \left\{ \underbrace{\prod_{l=0}^{L-1} \boldsymbol{B}_{R_l}\left[p_l^*, p_{l-1}^*\right]}_{\text{Dominant Term}} + \underbrace{\sum_{p_1,\ldots,p_{L-1}\neq p_1^*\ldots p_{L-1}^*} \prod_{l=0}^{L-1} \boldsymbol{B}_{R_l}\left[p_l, p_{l-1}\right]}_{\text{Other Terms}} \right\}$$

If we assume $\sum_{p,q} \boldsymbol{B}_r[p, q] = \text{Constant}$ and $\boldsymbol{B}_r[p^*, q^*] \geq \sum_{(p,q)\neq(p^*,q^*)} \boldsymbol{B}_r[p, q]$ for each r, the RHR increases if any element in the dominant term, $\boldsymbol{B}_{R_l}[p_l^*, p_{l-1}^*]$, increases. It is easy to see that the HR is positively related to $\boldsymbol{B}_r[p^*, q^*]$. So this can show that the homophily ratio (HR) we defined, is positively related to the relaxed homophily ratio (RHR) (Guo et al., 2023).

F ADDITIONAL EXPERIMENTAL RESULTS

F.1 The dataset statistics

The dataset statistics are shown in table 2. With $|\mathcal{V}|$ and $|\mathcal{E}|$ the number of nodes and edges, $\mathcal{R}$ the relation types, HR the homophily ratio and SDOR the Smoothest-dimension Overlapping ratio.

Table 2: Real-world datasets statistics

Dataset	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	$\mathcal{R}$	RHR	SDOR
IMDB	11k	550k	Movie-Actor (MA)	0.510	MA-MD: 0.65
			Movie-Director (MD)	0.900	
ACM	21k	87k	Paper-cite-Paper (PP)	0.64	PP-PA: 0.66
			Author-write-Paper (PA)	0.925	PP-PS: 0.67
			Paper-has-subject (PS)	0.804	PA-PS: 0.94
Yahoo	100	/	Finance-Health	/	/
			Finance-Tech	/	/
			Tech-Health	/	/

F.2 Additional Quantitative Results

We conduct analysis studies according to the embedding size, the homophily ratio and the optimization methods and the results can be found in table 3. The results illustrated that:

1. The algorithm is more robust when the homophily ratio is higher, as we discussed in section 6.3.
2. The iterative refining optimization is more stable in ensuring convergence and thus yields better performance, as we suggested in section 6.2.

Table 3: Quantitative experimental results on Heterogeneous Graph Structure Learning.

Dataset	**RHR**	**K**	**Vanilla GSL**		**HGSL-GD**		**HGSL-IR**	
			AUC	GMSE	AUC	GMSE	AUC	GMSE
Synthetic	0.95+	60	0.66 ± 0.05	0.03 ± 0.00	0.70 ± 0.04	0.27 ± 0.05	**0.75 ± 0.02**	**0.02 ± 0.00**
		300	0.65 ± 0.01	0.03 ± 0.00	0.75 ± 0.05	0.30 ± 0.04	**0.83 ± 0.04**	**0.02 ± 0.01**
		600	0.78 ± 0.03	**0.04 ± 0.02**	0.77 ± 0.04	0.26 ± 0.04	**0.89 ± 0.04**	0.06 ± 0.01
IMDB	0.51	3066	0.74 ± 0.04	0.29 ± 0.08	0.75 ± 0.06	0.21 ± 0.05	**0.81 ± 0.07**	**0.07 ± 0.05**
ACM	0.64	1902	0.65 ± 0.06	**0.07± 0.10**	0.62 ± 0.06	0.09 ± 0.15	**0.73 ± 0.02**	0.12 ± 0.07

* Experimental results are evaluated over 30 trials and the mean/standard deviation is calculated. 0.00 means the value < 0.01.

F.3 Analysis on the relation-wise embedding

In synthetic dataset, we conduct the analysis of $\boldsymbol{E}$ by measuring the normalized root mean squared error (NRMSE) of the estimated value and the ground truth by

$$\frac{1}{|\mathcal{R}|}\sum_{r}\frac{\sqrt{\sum_{k}\left(\boldsymbol{e}_{r,k}-\hat{\boldsymbol{e}}_{r,k}\right)^{2}}/K}{\max\left(\boldsymbol{e}_{r}\right)-\min\left(\boldsymbol{e}_{r}\right)}$$

where r is the relation type, $k=[1,\ldots,K]$ is the entry on each signal dimension. We report the results below:

Feature Dimension	K = 60	K = 100	K=300
NRMSE	0.0320 ± 0.0006	0.0189 ± 0.0003	0.0063 ± 0.0001

The error is indeed small, which demonstrates the efficacy of our algorithm in learning a proper $\boldsymbol{e}_r$.

To better understand the property of $\boldsymbol{e}_r$, we can set $\alpha=\frac{\lambda_2}{2\lambda_1^2}$ and $\beta=\frac{\lambda_2}{\lambda_1}$ and rewrite eq. (13) as,

$$\boldsymbol{e}_{r,k}^{t+1}=\frac{\lambda_2}{2\lambda_1}\left(\frac{1}{\lambda_1}\sum_{v,u,r\in\mathcal{E}_r'}w_{vur}^{t}\boldsymbol{x}_{v,k}\cdot\boldsymbol{x}_{u,k}-2\right)=\alpha\sum_{v,u,r\in\mathcal{E}_r'}w_{vur}^{t}\boldsymbol{x}_{v,k}\cdot\boldsymbol{x}_{u,k}-\beta$$

An intuition for this solution, as illustrated in fig. 5, is that: If we consider $\beta\to 0$ and the features for each node $\boldsymbol{X}_v$ is normalized to 1 , the solution of $\boldsymbol{e}_r$ will put larger weights on the dimensions that exhibit higher similarity, and those dimensions are considered to be most informative in recovering the ground truth $\mathbf{W}$ under our assumption. This phenomenon is observed in synthetic data, where the algorithm puts larger weights on the dimensions exhibiting higher smoothness.