Learning Predictive Substructures with Regularization for Network Data

Xuan Hong Dang, Hongyuan You
University of California Santa Barbara
Email: {xdang,hyou}@cs.ucsb.edu

Petko Bogdanov
University at Albany - SUNY
Email: pbogdanov@albany.edu

Ambuj K. Singh
University of California Santa Barbara
Email: ambuj@cs.ucsb.edu

***Abstract*—Learning a succinct set of substructures that predicts global network properties plays a key role in understanding complex network data. Existing approaches address this problem by sampling the exponential space of all possible subnetworks to find ones of high prediction accuracy. In this paper, we develop a novel framework that avoids sampling by formulating the problem of predictive subnetwork learning as node selection, subject to network-constrained regularization. Our framework involves two steps: (i) subspace learning, and (ii) predictive substructures discovery with network regularization. The framework is developed based upon two mathematically sound techniques of spectral graph learning and gradient descent optimization, and we show that their solutions converge to a global optimum solution—a desired property that cannot be guaranteed by sampling approaches. Through experimental analysis on a number of real world datasets, we demonstrate the performance of our framework against state-of-the-art algorithms, not only based on prediction accuracy but also in terms of domain relevance of the discovered substructures.**

I. Introduction

Network analysis plays a key role in understanding complex data. One important task in network analysis is learning local substructures that have impact on the global network properties. Compared to feature selection in high dimensional data, the analysis in a network context is more challenging due to the inherent interactions among features associated with nodes. Consider an example of Alzheimer's disease in neuroscience [1,2] where the task is to find markers that can predict the disease states. Due to the natural relationship between neural activity and neural connectivity, the loss of neurons/synapses in one brain region usually impacts the cognitive functions of other brain regions that are *physically* and *functionally* connected [3]. Hence, it is not sufficient to simply identify isolated neuronal populations (brain regions) whose activation is abnormal. Instead, it is crucial to consider the underlying network communication among neuronal populations that leads to different states of the disease [2].

Another example comes from biology, where studies have shown that biological outcomes (i.e. global properties) are determined by modules of proteins that interconnect within the context of protein-protein interaction (PPI) networks [4]. Such networks are essential in understanding genetic and regulatory diseases such as cancers and neurological disorders. Given the expression levels of genes for multiple patients, the task in this setting is to identify predictive substructures associated with the disease, while respecting the underlying interactions among implicated genes.

The success of machine learning and data mining algorithms depends on the representation of the underlying data. This is particularly true for network data where irrelevant nodes and edges may limit the understanding and hinder the discovery of local processes that impact and govern the global behaviour of the network. Understanding what these local subnetwork processes are, how they vary and what makes them different across network populations is critically important for a variety of domain applications. Network data (associated with network nodes) can naively be treated as an instance of high-dimensional data and common feature extraction methods such as PCA/SVD [5,6] can be adopted for its analysis. While such methods can optimize the prediction quality, the models they learn often lack domain relevance since they do not incorporate the underlying network interactions, and thus may fail to discover actual processes behind the observed data. A better suited approach to identifying discriminative substructures is to directly sample the space of subnetworks and find ones that lead to high prediction of the global network states [4,7]. While such approaches emphasize locality of the selected substructures, they need to explore an exponential number of connected subnetworks. Additionally, it is hard to ensure a unique and stable result across different runs due to the inherent drawback of sampling. Both stability and compactness of the predictive substructures are keys to unearth the complex relationships between local node values and global network properties.

In this paper, we propose a novel framework to learn a small set of subnetwork structures that are predictive of the global network properties. In contrast to existing approaches that either ignore the network topology or sample over the exponential space of subnetworks, we formulate the predictive subnetworks learning problem as explicit node selection subject to network-constrained regularization. Our framework is not only able to discover a succinct set of predictive subnetwork structures but also, in combination with a simple classifier such as the linear SVM, can accurately predict the global network behaviour within the space spanned by these substructures. Our learning task comprises of two steps. In the first step, we learn an optimal embedding subspace in which networks of different global states are maximally separated while those of the same state are clustered together. Each dimension of this low dimensional subspace essentially reflects a different aspect in separating networks labeled by different global states. In the second step, we perform substructures discovery through measuring the nodes' importance, regularized by their underlying local connectivity, along each intrinsic dimension of the embedding subspace. Specifically, we adopt L1-norm to explicitly remove irrelevant nodes that have little or no impact on the global network behaviour, and L2-norm to enforce connectivity among selected nodes. We solve the first step of subspace learning via matrix eigen-decomposition by taking the graph embedding approach, while for the second step, we employ the gradient descent technique to minimize fitting errors. The proposed framework possesses many appealing properties: (1) it makes no assumptions about the statistical distribution of network data; (2) it has a solid mathematical background from spectral graph learning and convex gradient optimization; (3) it is guaranteed to achieve the optimum solution in each step. Our experimental analysis using four real world datasets demonstrates the superior performance of the proposed algorithm not only in terms of prediction accuracy against state-of-the-art algorithms, but also in discovering predictive network substructures with domain relevance. To the best of our knowledge, the proposed framework is the first that combines both subspace learning and network structures discovery with explicit node-selection, providing better understanding of the complex relationships between local node values and global network behaviour.

II. Problem setting and preliminaries

A. Common Notation and Definitions

Definition 1: (Network sample) A *network sample* is a triple $S_i = (\mathcal{V}_i, E_i, \mathcal{F})$, where $\mathcal{V}_i = \{v_1, v_2, \ldots, v_{m_i}\}$ is a set of nodes, $E_i \subseteq \mathcal{V}_i \times \mathcal{V}_i$ is a set of undirected edges, and $\mathcal{F}$ is a function labeling each node with a real number.

Let $\mathcal{DS} = \{S_1, S_2, \ldots, S_n\}$ be a network dataset that consists of n network samples. Each network sample $S_i \in \mathcal{DS}$ is associated with a discrete *global* state or label ℓ_i. Unlike chemical compounds or XML web data where network samples may have considerably different network topologies [8,9], we focus on a family of networks whose topologies are relatively stable across network samples. Following this, we define an aggregate network S generalizing all samples S_i's, as follows:

Definition 2: (Aggregate network) Let $S = (\mathcal{V}, E, W)$ be a network summarizing the aggregate structure of all $S_i \in \mathcal{DS}$, where $\mathcal{V} = \mathcal{V}_1 \cup \mathcal{V}_2 \ldots \cup \mathcal{V}_n$, $E \subseteq \mathcal{V} \times \mathcal{V}$ and $E_i \subseteq E$ $\forall E_i$. Each edge $E(p,q) \in E$ is associated with a positive weight $W(p,q)$ defined as the fraction of network samples containing that edge in their structure, i.e., $W(p,q) = n^{-1} \times \sum_i E_i(p,q)$ with $E_i(p,q) = 1$ if v_p connects v_q in S_i. Since all S_i's are undirected networks, $W \in \mathbb{R}^{m \times m}$ defined for S is a symmetric matrix, with m as the total number of nodes in $\mathcal{V}$.

The setting defined above is general enough to accommodate a number of practical network applications including social networks, human brain networks or protein-protein interactions. For instance, S_i can model a snapshot captured at i-th timestamp of a social network, where v_p corresponds to a user whose local state presumably encodes her political viewpoint, while the global state ℓ_i indicates the overall political viewpoint of the entire community (at the i-th timestamp). Snapshots captured at different times vary slowly in terms of network structure. Likewise, S_i can also be used to encode a human brain network where the local value at an edge reflects the statistical correlation between a pair of brain regions and a global state ℓ_i encodes whether the subject is healthy or diseased. Different subjects may differ in their local node values but possess similar brain network structures [3].

B. Relationships among Network Samples

In many practical applications, the discriminative structure behind the observed network data can be captured by only a small number of latent regularities (dimensions) since many irrelevant nodes/features may have little or no impact on the global network labels. Learning an informative set of predictive subnetworks that influence the global states depends on how the relationships among network samples are captured and represented. Toward this goal, we construct two graphs encoding the (dis)similarity among network samples. Our optimization framework employs these relationship graphs to learn a low dimensional subspace that captures both the neighboring and discriminative structures among network samples. The construction of the graphs of network samples follows a similar construction from [10]. We next summarize the main steps of this construction and introduce the necessary notation.

Let $\mathcal{G}^+$ be defined as the first graph in which each vertex represents a network sample $S_i \in \mathcal{DS}$. A link is placed between two vertices if the corresponding network samples S_i and S_j have the same global state, and S_i is among the k nearest neighbors (kNN) of S_j or vice versa. Each link is further associated with a non-negative value $\mathbf{K}^+_{ij}$ quantifying the degree of similarity between S_i and S_j. Since the network samples have a similar structure, only their node values are used to compute their neighborhood similarity. Furthermore, as each S_i is a subnetwork of S according to Def.2, we use an m-dimensional vector $\mathbf{v}_i$ (recall m is the total nodes in S) to store its local node values, with a null value being used to denote the value of a missing node. This makes any two network samples comparable in term of their local node values. As such, we adopt the cosine distance (though more complex measures could be used [11,12]) to compute $\mathbf{K}^+_{ij}$ as follows:

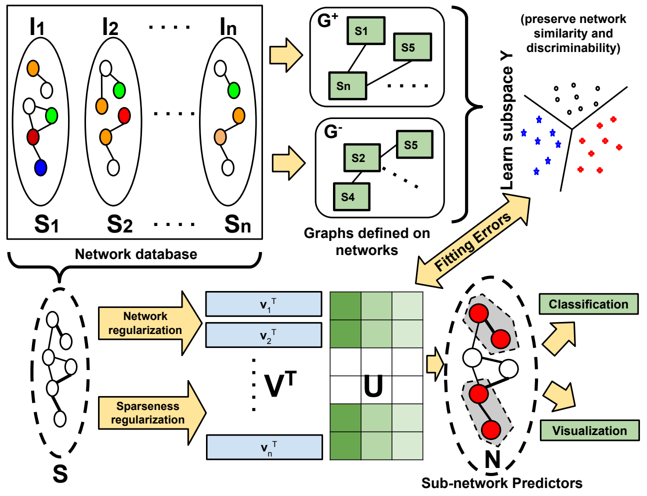


Figure 1: Overview of our framework in learning subnetwork structures for global network state prediction. An optimal subspace $\mathbf{Y}$ separating different network states is learnt first, then relevant subnetworks are mined through approximating $\mathbf{Y}$ with network and sparseness regularizations.

$$\mathbf{K}^{+}_{ij} = \begin{cases} \frac{\mathbf{v}_i \cdot \mathbf{v}_j}{\|\mathbf{v}_i\| \|\mathbf{v}_j\|}, & \text{if } \ell_i = \ell_j, S_j \in kNN(S_i) \text{ or } S_i \in kNN(S_j) \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Likewise, $\mathcal{G}^-$ is defined as a second graph with vertices also representing network samples and a link connecting two vertices, say S_i and S_j, if $S_i \in kNN(S_j)$ or $S_j \in kNN(S_i)$, yet their global states must be different. Likewise, a non-negative value $\mathbf{K}^{-}_{ij}$ is associated with a link to reflect how similar S_i and S_j are. Its computation is analogous to Eq.(1) except one difference: $\ell_i \neq \ell_j$.

It is important to note that while the development in [10] builds graphs among network instances in a similar fashion, its ultimate goal is classification of the global network states, while our goal here is feature selection in the form of subnetworks. As such, our feature selection framework can be combined with any classifier for global network states within subnetworks as features, including the classifier developed in [10]. Another way to view our current approach is as reducing the dimensions (features) in a controlled manner (i.e., enforcing a desired number of preserved features), while being aware of the network structure. Thus, it enables a more scalable and accurate training of classifiers, due to the reduced size of the network instances.

III. Our Proposed Framework

Given a set of network samples with global states, our goal is to uncover a succinct set of subnetwork structures whose local node values have the highest impact on global network states. Mining the optimal set of such predictive subnetworks is a non-trivial task since the number of possible subnetworks grows exponentially with the network size. As mentioned earlier, sampling is obviously one possible approach to overcome this challenge. However, without an effective indexing or pruning strategy, one might have to sample an impractically large number of times to ensure good quality substructures. Alternatively, methods operating on vector data and ignoring the network structures address only the goal of prediction accuracy while lacking domain relevance.

S_i, ℓ_i	a network sample and its global state/label
S, $\mathbf{C}_{m\times m}$	the aggregate network (Def. 2) and its Laplacian
n, m	#network samples and #nodes in S
$\mathcal{G}^+$, $\mathcal{G}^-$	similarity graphs among network samples
$\mathbf{K}^+$, $\mathbf{K}^-$	affinity matrices of $\mathcal{G}^+$ and $\mathcal{G}^-$
$\mathbf{V}_{m\times n}$	node local states (features) for all samples S_i's
$\mathbf{Y}_{n\times d}$	local states embedding in a low-dimensional subspace
$\mathbf{U}_{m\times d}$	output node-selection matrix
λ_1, λ_2	regularization parameters for sparsity and network impact
c	#nodes in selected substructures (controlled by λ_1)

Table I: Summary of notations used in the paper

Our proposed solution differs from the above two extremes by adopting the framework summarized in Fig.1 (with notation is summarized in Table I). Specifically, given $\mathcal{DS}$, we build two graphs $\mathcal{G}^+$ and $\mathcal{G}^-$. These two graphs serve our first objective of learning a low dimensional subspace $\mathbf{Y}$ in which network samples with different global states are well-distinguished while concurrently retaining the local similarities among network samples of the same global states. After obtaining $\mathbf{Y}$, we measure the importance of each node along each intrinsic dimension. This objective is achieved by minimizing the fitting errors between $\mathbf{Y}$ and the combination on the networks' local node values $\mathbf{V}$ by using a matrix $\mathbf{U}$. Such an error minimizing process is further complemented by L1-norm to remove irrelevant nodes, and L2-norm to enforce proximity of selected nodes in the general network structure S (Def.2). The predictive substructures are then selected based on $\mathbf{U}$'s sparse coefficients. The obtained substructures can be used for visualization and classification of new instances.

A. Subspace Learning

Given the two (dis)similarity graphs (defined in Sec.II-B), we construct our first objective function for learning a subspace in which the similarity relations among network samples of the same states are optimally preserved. This objective is formulated as follows:

$$\begin{aligned} \text{minimize} \sum_i \sum_j \|\mathbf{y}_i - \mathbf{y}_j\|_2^2 \mathbf{K}^+_{ij} \\ = \sum_i \sum_j (\|\mathbf{y}_i\|_2^2 + \|\mathbf{y}_j\|_2^2 - 2\mathbf{y}_i^T \mathbf{y}_j)\mathbf{K}^+_{ij} \\ = 2\mathbf{Y}^\mathbf{T}(\mathbf{D}^+ - \mathbf{K}^+)\mathbf{Y} = \mathbf{2tr}(\mathbf{Y}^\mathbf{T}\mathbf{L}^+\mathbf{Y}) \end{aligned} \tag{2}$$

in which $\mathbf{L}^+ = \mathbf{D}^+ - \mathbf{K}^+$ is the Laplacian matrix of $\mathcal{G}^+$; $\mathbf{D}^+$ is a diagonal matrix whose entries are summations over $\mathbf{K}^+$'s rows, i.e., $\mathbf{D}^+_{ii} = \sum_j \mathbf{K}^+_{ij}$, and $\mathbf{y}_i$, $\mathbf{y}_j$ (as $\mathbf{Y}$'s columns) are the maps of network samples S_i and S_j in our low dimensional subspace.

According to Eq.(2), similar network instances S_i and S_j (i.e. high $\mathbf{K}^+_{ij}$) mapped to distant points $\mathbf{y}_i$ and $\mathbf{y}_j$ in the resulting subspace would incur a high cost. Minimizing this objective function is thus equivalent to optimally preserving the local similarities among network samples having the same global state.

Analogously, we construct the second objective function to maximize the separation of instances of different global states:

$$\begin{aligned}\text{maximize} \sum\nolimits_i \sum\nolimits_j \|\mathbf{y}_i - \mathbf{y}_j\|_2^2 \mathbf{K}^-_{ij} \\ = 2\mathbf{Y^T}(\mathbf{D}^- - \mathbf{K}^-)\mathbf{Y} = \mathbf{2tr}(\mathbf{Y^T L^- Y}),\end{aligned} \quad (3)$$

where $\mathbf{L}^- = \mathbf{D}^- - \mathbf{K}^-$ is the Laplacian of $\mathcal{G}^-$, $\mathbf{D}^-$ is a diagonal matrix, $\mathbf{D}^-_{ii} = \sum_j \mathbf{K}^-_{ij}$, and the embedding $\mathbf{Y}$ is similarly defined as in Eq.(2). Unlike Eq.(2), here we aim to minimize the similarity between network samples S_i and S_j with different global states in the induced subspace. Such network instances might reside close to the classification boundary and hence, by minimizing their similarity, we improve their separability in the induced subspace. Combining the two objectives, we obtain:

$$\begin{aligned}\mathbf{Y} = & \arg\max_{\mathbf{Y}} \left\{ \mathbf{tr}\left(\mathbf{Y^T}(\mathbf{L}^- - \beta\mathbf{L}^+)\mathbf{Y}\right)\right\} \\ & \text{subject to} \quad \mathbf{Y^T D^+ Y} = \mathbf{I}\end{aligned} \quad (4)$$

where $\beta \in (0, 1)$ is a trade-off parameter between the two objectives $\mathbf{L}^-$ and $\mathbf{L}^+$. It can be tuned via cross validation and based on empirical evaluation we find that $\beta \in [0.2, 0.4]$ often results in good performance. Additionally, the constraint $\mathbf{Y^T D^+ Y} = \mathbf{I}$ is used to ensure the uniqueness of the obtained subspace $\mathbf{Y}$. In solving this constrained trace optimization, we resort to the *Largrange multipliers* method [6]. Let us simplify the notation $\mathbf{L} = \mathbf{L}^- - \beta\mathbf{L}^+$ and $\mathbf{D}$ for $\mathbf{D}^+$. Our trace optimization in Eq.(4) can be recast as:

$$\mathcal{L}(\mathbf{Y}, \mathbf{\Lambda}) = \mathbf{Y^T L Y} - \mathbf{\Lambda}\left(\mathbf{Y^T D Y} - \mathbf{I}\right) \quad (5)$$

where $\mathbf{\Lambda}$ is a diagonal matrix with diagonal entries corresponding to the Lagrange multipliers. Taking the derivative of $\mathcal{L}$ with respect to $\mathbf{Y}$ and equating it to zero yields:

$$\mathbf{LY} = \mathbf{\Lambda D Y} \quad (6)$$

which gives us the optimal $\mathbf{Y}$ as the leading eigenvectors (corresponding to the largest eigenvalues) of the matrix $\mathbf{D^{-1}L}$. We choose the number of selected eigenvectors equal to d—the number of unique global network states, since a subspace with d dimensions is generally sufficient to capture the discriminative structure among d different classes.

B. Learning Sparse Subnetwork Structures

In the low-dimensional subspace $\mathbf{Y}$ (obtained according to Eq.(6)), each row represents the embedding of a network sample and each column reflects a different aspect in distinguishing global network states. We next proceed to quantifying the importance of each node in each dimension of $\mathbf{Y}$, or equivalently, the contribution of each node in differentiating global network states.

Let $\mathbf{V}$ be the matrix storing local node states for all network instances (see Sec.II-B) and $\mathbf{U}$ be a node-selection matrix that we aim to learn. Our objective is to minimize the fitting errors $\|\mathbf{Y} - \mathbf{V}^T\mathbf{U}\|_F^2$, while using the L2-norm to impose the network structure, and the L1-norm to remove irrelevant nodes.

In essence, each column $\mathbf{u} \in \mathbf{U}$ contains m coefficients for combining m $\mathbf{V}$'s rows in approximating one dimension of subspace $\mathbf{Y}$. Thus, by sparsifying and regularizing $\mathbf{u}$'s coefficients subject to the network structure, we learn the degree of importance of every node in discriminating global network states. Specifically, we formulate the network-constrained regularization via the Laplacian matrix $\mathbf{C}$ as follows:

$$\mathbf{C}_{pq} = \mathbf{C}_{qp} = \begin{cases} \sum_q W(p,q) & \text{if } v_p \equiv v_q \\ -W(p,q) & \text{if } v_p, v_q \text{ are connected} \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

The quadratic form of $\mathbf{C}$ is always non-negative, i.e., $\mathbf{u}^T\mathbf{C}\mathbf{u} = \frac{1}{2}\sum_{p=1}^m \sum_{q=1}^m W(p,q)(u_p - u_q)^2 \geq 0$ (where u_p and u_q are entries or components of $\mathbf{u}$. Recall from Def.2 that a large $W(p,q)$ indicates nodes v_p and v_q influence each other strongly, their corresponding coefficients, u_p and u_q, should be similar in order to minimize this quadratic equation. It is thus intuitive to say that, if node v_p impacts the global network state, its selection in $\mathbf{U}$ will increase the utility of also selecting its neighbor v_q, if the weight $W(p,q)$ on the edge connecting the two nodes is high. This network-regularized selection leads to connected subnetwork structures in our solution.

Another important regularization we impose on learning the node-selection matrix $\mathbf{U}$ is to explicitly exclude those nodes that have little or no impact on global state values. These nodes are either irrelevant or redundant for the classification of global network states and excluding them makes the final selected substructures concise and easy to interpret. One of the most effective ways to accomplish this task is to impose the L1-norm [6] on each vector $\mathbf{u}$. By its nature, L1-norm will shrink some of entries in the node-selection vector $\mathbf{u}$ to zero if a sufficiently large weight is placed on this sparsity regularization. In combination with the network regularization, our overall regularized subnetwork selection objective is formulated as follows:

$$\arg\min_{\mathbf{U}} \|\mathbf{Y} - \mathbf{V^T U}\|_F^2 + \lambda_2 \sum_{\mathbf{u}\in\mathbf{U}} \mathbf{u^T C u} + \lambda_1 \sum_{\mathbf{u}\in\mathbf{U}} |\mathbf{u}|, \quad (8)$$

where $|\mathbf{u}| = \sum_{t=1}^m |u_t|$. The λ_1 and λ_2 are the trade-off factors controlling respectively the shrinkage imposed on $\mathbf{u}$'s and the aggregate network topology (their impact will be demonstrated in Sec.IV-E). A larger value of λ_1 will result in a smaller number of selected nodes while a larger value of λ_2 emphasizes the network structure.

In minimizing this objective function, note that Eq.(8) is not strictly convex due to the absolute term $\sum_{\mathbf{u}\in\mathbf{U}} |\mathbf{u}|$. We thus optimize it through the gradient descent approach in which each $\mathbf{u}$ is optimized individually. We rewrite the function as follows:

$$\begin{aligned}\mathcal{R}(\mathbf{u}) &= \|\mathbf{y} - \mathbf{V^T u}\|_2^2 + \lambda_2 \mathbf{u^T C u} + \lambda_1 |\mathbf{u}| \\ &= \sum\nolimits_i (y_i - \mathbf{v_i^T u})^2 + \lambda_2 \mathbf{u^T C u} + \lambda_1 \sum\nolimits_t |u_t|\end{aligned} \quad (9)$$

The derivative of the first term of $\mathcal{R}(\mathbf{u})$, denoted $\mathcal{R}_1(\mathbf{u})$, w.r.t. a coordinate u_t is:

$$\begin{aligned}\frac{\partial \mathcal{R}_1(\mathbf{u})}{\partial u_t} &= 2\sum\nolimits_i (y_i - u_t v_{it} - \bar{\mathbf{u}}^T\bar{\mathbf{v}}_i)(-v_{it}) \\ &= 2\sum\nolimits_i v_{it}^2 u_t - 2\sum\nolimits_i (v_{it})(y_i - \bar{\mathbf{u}}^T\bar{\mathbf{v}}_i) \quad (10)\end{aligned}$$

where $\bar{\mathbf{u}}$ and $\bar{\mathbf{v}}_i$ denote the respective vectors excluding their t-th entry. Likewise, taking the derivative of the second term of $\mathcal{R}(\mathbf{u})$, denoted by $\mathcal{R}_2(\mathbf{u})$, w.r.t. u_t leads to:

$$\begin{aligned}\frac{\partial \mathcal{R}_2(\mathbf{u})}{\partial u_t} &= \frac{\partial}{\partial u_t}\lambda_2 \sum\nolimits_p \sum\nolimits_q u_p u_q \mathbf{C}_{pq} \\ &= \lambda_2(\sum_{p\neq t} \mathbf{C}_{pt}u_p + \sum_{q\neq t}\mathbf{C}_{qt}u_q + 2\mathbf{C}_{tt}u_t) \\ &= \lambda_2(\sum\nolimits_p \mathbf{C}_{pt}u_p + \sum\nolimits_q \mathbf{C}_{qt}u_q) = 2\lambda_2\mathbf{u}^T\mathbf{C}_{\star t} \\ &= 2\lambda_2\mathbf{C}_{tt}u_t + 2\lambda_2\bar{\mathbf{u}}^T\bar{\mathbf{C}}_{\star t} \quad (11)\end{aligned}$$

where $\mathbf{C}_{\star t}$ denotes the t-th column of matrix $\mathbf{C}$ and again $\bar{\mathbf{C}}_{\star t}$ is the respective vector excluding its t-th component.

Note that our last term in Eq.(9) is not a smooth function; so its derivative w.r.t. u_t is calculated as follows:

$$\frac{\partial \mathcal{R}_3(\mathbf{u})}{\partial u_t} = \frac{\partial \lambda_1|\mathbf{u}|}{\partial u_t} = \begin{cases}\{-\lambda_1\} & \text{if } u_t < 0 \\ \{+\lambda_1\} & \text{if } u_t > 0 \\ [-\lambda_1, +\lambda_1] & \text{if } u_t = 0\end{cases} \quad (12)$$

By combining the three derivatives, we get:

$$\frac{\partial \mathcal{R}(\mathbf{u})}{\partial u_t} = \begin{cases}\{a_1u_j - a_2 - \lambda_1\} & \text{if } u_t < 0 \\ \{a_1u_j - a_2 + \lambda_1\} & \text{if } u_t > 0 \\ [-a_2 - \lambda_1, -a_2 + \lambda_1] & \text{if } u_t = 0\end{cases} \quad (13)$$

where, for simplicity, we have introduced $a_1 = 2\sum_i v_{it}^2 + 2\lambda_2\mathbf{C}_{tt}$, which is always positive, and $a_2 = 2\sum_i(v_{it})(y_i - \bar{\mathbf{u}}^T\bar{\mathbf{v}}_i) - 2\lambda_2\bar{\mathbf{u}}^T\bar{\mathbf{C}}_{\star t}$. Setting this equations to zero and solving for u_t finally yields:

$$u_t = \begin{cases}(a_2 + \lambda_1)/a_1 & \text{if } a_2 < -\lambda_1 \\ (a_2 - \lambda_1)/a_1 & \text{if } a_2 > +\lambda_1 \\ 0 & \text{if } a_2 \in [-\lambda_1; +\lambda_1]\end{cases} \quad (14)$$

By the gradient descent optimization [6], we repeatedly update $\mathbf{u}$'s components until it converges. In what follows the convergence of our method is proven.

Theorem 1: Let $\mathbf{u}$ be a $m \times 1$ vector, and $\mathcal{R}_1(\mathbf{u}) = \|\mathbf{y} - \mathbf{V^T u}\|_2^2$ be a scalar function of n variables with second order derivatives defined on a convex domain $\mathcal{D}$. Then its second derivative is positive semi-definite and $\mathcal{R}_1(\mathbf{u})$ is convex.

Proof: Given $\mathcal{R}_1(\mathbf{u}) = \|\mathbf{y} - \mathbf{V^T u}\|_2^2$ where $\mathbf{y}$ is a column in $\mathbf{Y}$, it is to see that the second derivative $\partial^2\mathcal{R}_1(\mathbf{u})/\partial\mathbf{u}^2 = \mathbf{VV^T}$. Its quadratic form $\mathbf{b^T VV^T b}$ is non-negative given any $m\times 1$ vector $\mathbf{b}$ and thus $\partial^2\mathcal{R}_1(\mathbf{u})/\partial\mathbf{u}^2$ is positive semi-definite. Using Taylor approximation [13] of $\mathcal{R}_1(\mathbf{u})$ near $\mathbf{u}$, with a $\gamma \in (0,1)$ and $\mathbf{u}, \mathbf{u}+\mathbf{h} \in \mathcal{D}$, we can write:

$$\mathcal{R}_1(\mathbf{u}+\mathbf{h}) = \mathcal{R}_1(\mathbf{u}) + \frac{\partial \mathcal{R}_1(\mathbf{u})}{\partial \mathbf{u}}^T\mathbf{h} + \frac{1}{2}\mathbf{h}^T\frac{\partial^2\mathcal{R}_1(\mathbf{u}+\gamma\mathbf{h})}{\partial\mathbf{u}^2}\mathbf{h}$$

As shown above, the second derivative of $\mathcal{R}_1(\mathbf{u})$ is positive semi-definite, hence, the last term in this equation is non-negative, resulting in the following inequality:

$$\mathcal{R}_1(\mathbf{u}+\mathbf{h}) \geq \mathcal{R}_1(\mathbf{u}) + \frac{\partial \mathcal{R}_1(\mathbf{u})}{\partial \mathbf{u}}^T\mathbf{h} \quad (15)$$

Now, given any $\mathbf{u}'$ and $\mathbf{u}''$ in $\mathcal{D}$ and let $\zeta \in (0,1)$ be a scalar, we have $\mathbf{u} = \zeta\mathbf{u}' + (1-\zeta)\mathbf{u}'' \in \mathcal{D}$ due to the convexity of $\mathcal{D}$. By Eq.(15), it follows that:

$$\begin{cases}\mathcal{R}_1(\mathbf{u}') & \geq \mathcal{R}_1(\mathbf{u}) + \frac{\partial \mathcal{R}_1(\mathbf{u})}{\partial \mathbf{u}}^T(\mathbf{u}' - \mathbf{u}) \\ \mathcal{R}_1(\mathbf{u}'') & \geq \mathcal{R}_1(\mathbf{u}) + \frac{\partial \mathcal{R}_1(\mathbf{u})}{\partial \mathbf{u}}^T(\mathbf{u}'' - \mathbf{u})\end{cases} \quad (16)$$

Multiplying the first row in Eq.(16) by ζ, the second row by $(1-\zeta)$ and summing up yields:

$$\begin{aligned}&\zeta\mathcal{R}_1(\mathbf{u}') + (1-\zeta)\mathcal{R}_1(\mathbf{u}'') \\ &\quad \geq \mathcal{R}_1(\mathbf{u}) + \frac{\partial \mathcal{R}_1(\mathbf{u})}{\partial \mathbf{u}}^T(\zeta\mathbf{u}' + (1-\zeta)\mathbf{u}'' - \mathbf{u}) = \mathcal{R}_1(\mathbf{u})\end{aligned}$$

which shows the convexity of $\mathcal{R}_1(\mathbf{u})$. □

Theorem 2: Let $\mathbf{u}$ be $m \times 1$ vector and let $\mathcal{R}_2(\mathbf{u}) = \lambda_2\mathbf{u^T Cu}$ be a scalar function of n variables with second order derivatives defined on a convex domain $\mathcal{D}$. Then $\mathcal{R}_2(\mathbf{u})$ is a convex function.

Proof: With similar arguments as those in Theorem 1, it is straightforward to prove $\mathcal{R}_2(\mathbf{u})$ is a convex function given that $\mathbf{C}$ is a positive semi-definite matrix as defined in Eq(7). □

Theorem 3: Given $\mathcal{R}(\mathbf{u}) = \mathcal{R}_1(\mathbf{u}) + \lambda_2\mathcal{R}_2(\mathbf{u}) + \lambda_1\mathcal{R}_3(\mathbf{u})$ where λ_1, λ_2 are non-negative, our optimization function $\mathcal{R}(\mathbf{u})$ formulated in Eq.(9) is convex.

Proof: For any $\mathbf{u}', \mathbf{u}'' \in \mathcal{D}$, $\zeta \in (0,1)$, where $\mathbf{u} = \zeta\mathbf{u}' + (1-\zeta)\mathbf{u}''$, the function $\mathcal{R}_3(\mathbf{u})$ is also convex since:

$$\begin{aligned}&\zeta\mathcal{R}_3(\mathbf{u}') + (1-\zeta)\mathcal{R}_3(\mathbf{u}'') = \zeta|\mathbf{u}'| + (1-\zeta)|\mathbf{u}''| \\ &\quad = |\zeta\mathbf{u}'| + |(1-\zeta)\mathbf{u}''| \geq |\zeta\mathbf{u}' + (1-\zeta)\mathbf{u}''| = \mathcal{R}_3(\mathbf{u})\end{aligned}$$

in which we have used the triangle inequality in the second row. Based on (i) Theorems 1, 2, (ii) the fact that the summation of convex functions over a convex domain $\mathcal{D}$ is also convex [13] and (iii) given that both $\lambda_1, \lambda_2 \geq 0$, it follows that $\mathcal{R}(\mathbf{u})$ is a convex function. □

Corollary 1: Theorem 3 guarantees the convergence of our gradient descent method and the solution it finds is the unique optimum.

C. Subnetwork Structures

Due to the L1-regularization, the number of non-zero coefficients in each $\mathbf{u} \in \mathbf{U}$ is usually small according to the setting of λ_1. Let us further index $\{\mathbf{u}_i\}_{i=1}^d$ for column vectors in $\mathbf{U}$ and assume that, for a specific λ_1 value, c is the number of non-zero coefficients from each vector $\mathbf{u}_i$. In a general case, these coefficients might be slightly different across $\{\mathbf{u}_i\}_{i=1}^d$. Thus, we first rank the largest absolute non-zero entries across $\{\mathbf{u}_i\}_{i=1}^d$ and aggregate them into a single final vector $\mathbf{u}$ as follows:

$$\mathbf{u} = (u_1, \ldots, u_m)^T \text{ with } u_p = \max_i |\mathbf{u}_{i,p}| \quad (17)$$

where $\mathbf{u}_{i,p}$ is the p-th entry of eigenvector $\mathbf{u}_i$. We then select nodes according to the top c ranking entries in $\mathbf{u}$. The subnetworks induced by these nodes are obtained by matching with the nodes of S defined in Def.2 along with connecting edges encoded in E.

D. Algorithm Complexity

We name our framework DIPS—DIscovering Predictive subnetwork Structures, and analyze its complexity in what follows. We first build the two similarity graphs $\mathcal{G}^+$ and $\mathcal{G}^-$ which takes $O(mn^2)$ in the worst case and can reduce to $O(mn \log n)$ with the use of a kd-Tree structure [14]. Both $\mathbf{D}$ and $\mathbf{L}$ in Eq.(6) are of size $n \times n$ and the eigen-decomposition for our first step takes $O(n^2 \log n)$ with the Lanczos technique. In the second step, only a_2 is impacted by the change of $\mathbf{u}$'s and its computation takes $O(m)$ assuming that the sizes of $\mathbf{u}$, $\mathbf{v}_i$ and $\bar{\mathbf{C}}_{\star j}$ are all $m \times 1$, while the time for $\mathbf{u}$'s updates in Eq.(14) can be considered as a constant. We need to update every component of $\mathbf{u}$, the computation thus amounts to $O(dMm^2)$ assuming M is the maximal number of iterations until convergence of all d vectors $\mathbf{u}$'s. The overall complexity is thus $O(mn \log n + n^2 \log n + dMm^2)$.

E. Discussion

It is worthwhile to mention that both the sparseness and network-constrained regularizations can be directly integrated into our first step formulated in Eq.(4). Nevertheless, that combination will result in a harder non-convex optimization problem, rendering standard convex optimization theory not applicable. In order to address this difficulty, we can either apply the minorization-maximization algorithm as developed in [15,16] or introduce another variable to convert the problem into the bi-convex optimization [17]. However, in both cases, it is not certain whether stable optimum solutions can be achieved due to the original non-convexity optimization. We leave these issues for future studies.

IV. Experiments

A. Methodology

We compare the performance of DIPS against the following representative methods: (1) MINDS [7]—among the first subnetwork sampling approaches for global-state network classification, and (2) NGF [4]—a random forest based approach incorporating the structure of PPI networks. Additionally, we also compare DIPS against techniques that do not incorporate a network topology: (3) DIPSw/o, a variant of DIPS without network constraints, (4) SVM by first performing SVD for dimensionality reduction (retaining 95% of the singular values). For each of our datasets, we evaluate the performance of all algorithms via 5-fold stratified cross validation (CV). The parameters of DIPS are chosen based on the estimated prediction accuracy within every 4 training folds and tested on the left-out fold, following the protocol in [18,19]. Unless otherwise specified, its parameters are tuned within the ranges: $k \in [30-80]$ with a step of 10; $\lambda_1 \in [0.15-0.005]$ and $\lambda_2 \in [0.1-1000]$ (in log-scale). For MINDS, the minimum discriminative potential for each network constrained decision tree is set to 0.8 and $K = 200$ [7]. For NGF, we use the Gini index for tree building and the threshold to stop growing a tree $\epsilon = 0.02$ [4]. In both MINDS and NGF we sample/grow $10,000$ tree substructures.

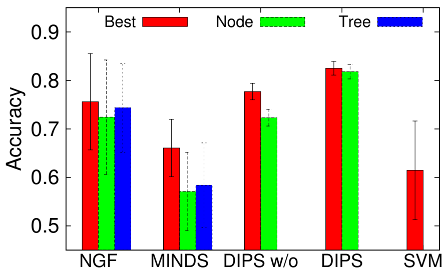


Figure 2: Prediction in image networks. Red bars show the best accuracies; Green bars show accuracies with substructures having 80 nodes; Blue bars show accuracies with 80 trees used as features in NGF and MINDS.

Methods	10	30	50	70	90
NGF-node	0.58(.10)	0.66(.10)	0.68(.06)	0.72(.08)	0.72(.09)
NGF-tree	0.68(.07)	0.72(.10)	0.74(.09)	0.72(.10)	0.74(.10)
MINDS-node	0.56(.05)	0.57(.10)	0.58(.08)	0.57(.10)	0.58(.09)
MINDS-tree	0.58(.07)	0.59(.10)	0.58(.10)	0.57(.07)	0.59(.07)
DIPSw/o	0.76(.02)	0.77(.02)	0.75(.02)	0.72(.02)	0.71(.02)
DIPS	0.76(.02)	0.79(.02)	0.82(.02)	0.82(.01)	0.81(.01)

Table II: Prediction accuracy on image networks for varying number of features in predictive substructures between 10 and 90 (std. dev. in brackets)

B. Image Network Dataset

Since most network datasets (presented next) lack ground truth for the best features, we conduct an experiment on image data, namely the CMUFace [20], in order to evaluate the relevance of uncovered predictive subnetworks *via visualization*. Though images do not originally involve explicit network structures, studying them as graphs has been deemed advantageous [21]. In particular, graphs enable the discovery of local image properties, especially in image denoising since many pixels may be missing or tampered. Following [21], we first down-sample the number of pixels to 50% and construct a common network topology relying on the remaining pixels. Within each image, a pixel corresponds to a node and has edges connected to the 5 nearest pixels. The value associated with an edge is inversely proportional to its length, and a value associated with a node is the grey level of the corresponding pixel. For the sake of visualization, we work with all straight pose images where open eyes and sunglasses are selected as the global network states, resulting in 156 network samples, each containing $1,920$ nodes and $11,172$ edges. The subnetwork structures corresponding to the "eye" areas are therefore the ground truth features.

Prediction Performance: In Fig.2, we report the prediction accuracy of all algorithms averaged from 5-fold CV for 2 cases: (i) the best performance (red bars) not constrained by the number of selected nodes, and (ii) the best performance when the node number (i.e., value of c in Sec.III-C) in the predictive subnetworks is 80 (equal to the number of ground

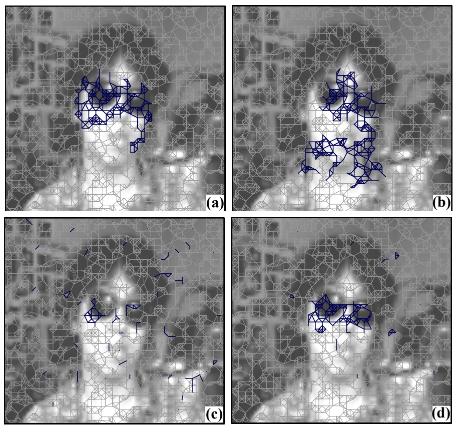
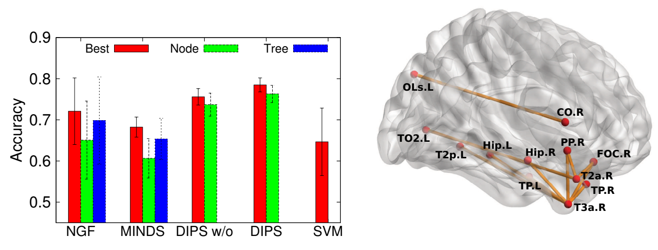


Figure 3: Subnetworks selected by each method in image data. Overall network is shown in grey while selected subnetworks are shown in blue (a background image is selected from sunglass class). (a) NGF (b) MINDS (c) DIPSw/o (d) DIPS.

truth nodes). Since NGF and MINDS are ensemble methods, we further report when their selected trees are counted as the number of features (blue bars). As seen from Fig.2, both DIPS and DIPSw/o usually outperform the sampling methods and SVM. Only DIPSw/o is inferior to NGF when the selected nodes are constrained to 80. In fact, DIPSw/o achieves the best accuracy when its substructures contains only 30 nodes while DIPS achieves the best accuracy of 82.1% when selecting 71 nodes to form subnetworks. Both NGF and MINDS need a much higher number of nodes, respectively 355 and 202, to achieve their best accuracy. When the number of nodes is limited to 80, NGF achieves only 71.9% of accuracy, which is considerably lower than DIPS's 81.1%.

Table II provides more insights into the performance of all methods for varying the number of selected nodes (trees in NGF- and MINDS-tree) from 10 to 90. The classification rates of NGF and MINDS show similar trends of achieving higher accuracies for larger numbers of selected nodes/trees, while generally stabilizing when more than 70 nodes/trees are selected. DIPSw/o outperforms DIPS for a small number of selected nodes but its classification rate quickly deteriorates as the number of selected nodes increases. As seen from the last row of Table II, DIPS consistently maintains the high accuracy rate and its performance only drops when the number of nodes exceeds the size 80 of the ground truth nodes.

Substructure Visualization: We also explore the subnetworks selected by each algorithm. In Fig.3, we plot the selected substructures aggregated from 5-fold CV for all algorithms by limiting the selected nodes to 80. Though several subnetworks are found in the area of the "eyes", NGF shows many irrelevant substructures selected across the space. Most subnetworks found by MINDS are well connected and less fragmented, yet some irrelevant regions are included. On



(a) (b)

Figure 4: Performance on brain networks. (a) Prediction accuracy of all methods (b) Common predictive substructures found by DIPS.

Methods	10	30	50	70	90	100
NGF-node	0.57(.08)	0.64(.03)	0.65(.02)	0.66(.01)	0.65(.01)	0.65(.01)
NGF-tree	0.64(.02)	0.65(.01)	0.65(.01)	0.65(.02)	0.65(.03)	0.60(.03)
MINDS-node	0.58(.08)	0.61(.05)	0.64(.02)	0.61(.05)	0.61(.03)	0.61(.03)
MINDS-tree	0.64(.02)	0.61(.03)	0.65(.01)	0.66(.02)	0.68(.03)	0.67(.02)
DIPSw/o	0.63(.02)	0.71(.02)	0.74(.03)	0.76(.03)	0.76(.03)	0.74(.03)
DIPS	0.63(.03)	0.68(.03)	0.71(.02)	0.75(.02)	0.76(.02)	0.75(.04)

Table III: Accuracy rates on brain networks with varying numbers of selected features (std. dev. in brackets)

the other hand, DIPSw/o (Fig.3c) discovers a fragment of ground truth nodes but many irrelevant isolated nodes are also seen throughout the entire network. This is justified by the lack of network regularization in this model. Overall, none of the methods discover better features than DIPS as visualized in Fig.3d. DIPS's substructures are consistent across all folds and highly overlapping with the ground truth region.

C. Brain Network Dataset

The second dataset we examine is collected from the ADNI project[1] focusing on the Alzheimer's disease. It consists of resting state fMRI scans captured from normal control (NC) and Alzheimer's disease (AD) subjects. For each fMRI scan, we employ the FSL toolbox [22] to extract sequences of responses from 112 anatomical volumes of interest (AVOI), each representing a brain region. We focus on the resulting functional brain network [1] by building an edge-dual network: a node in the edge-dual network is defined if the temporal correlation between the two underlying brain regions is ≥ 0.5. The node value is the amount of correlation degree, and an edge connects two nodes if they share one of the underlying brain regions. This setting ensures that any predictive substructure involves at least two brain regions, which is consistent with the task of analyzing *functional* brain connectivity [1,3]. After preprocessing, the dataset consists of 173 network samples, each having on average of $2,948$ nodes and $191,750$ edges.
Prediction Performance: We plot in Fig.4(a) the best prediction accuracy (red bars) of all algorithms, and the best prediction accuracy in which the predictive substructures involve 100 nodes (blue), and 100 trees (green) for NGF- and MINDS-tree variants. Performance with smaller settings of selected nodes is reported in Table III. As observed, while NGF and MINDS are designed to find trees with low training

[1]http://www.adni-info.org/

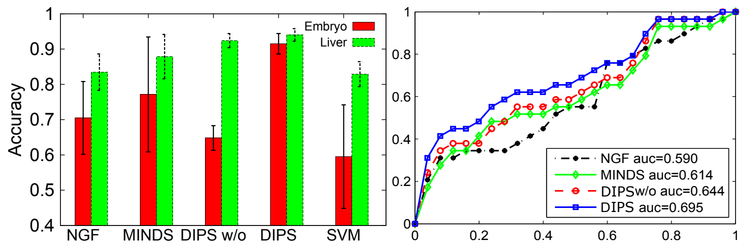


Figure 5: (a) Prediction accuracy in embryonic development and liver metastasis datasets combined with PPI networks. (b) ROC performance w.r.t. ground truth genes provided in [23] for liver metastasis.

errors, they both do not perform noticeably better than the baseline SVM. In the case of NGF, a root node selection in a densely connected area has limited impact. Both NGF and MINDS suffer from an exponential number of possible subnetworks within the densely connected brain graphs. In contrast, DIPS avoids this exponential complexity through subspace learning and network-constrained regularization. Examining the detailed performance in Table III provides further evidence of the accuracy gap between the two approaches. While DIPS achieves 76% with 90 nodes in its predictive substructures, NGF and MINDS can only get between 65% and 68% in their best performance.

Predictive Substructures: Unlike image data where ground truth substructures can be marked via visualization, defining predictive features for Alzheimer's disease remains an ongoing challenge [2]. Nevertheless, one way to evaluate the quality of uncovered substructures from each technique is through their consistency in cross validation. Substructures that are consistently found across training folds are likely the disease-related biomarkers and they should be the first candidates for further investigation.

We report the ratio between the intersection and union of substructures selected across training folds of each algorithm. Specifically, we observe that both sampling techniques show less overlapping substructures for all sizes of selected nodes varied from 10 to 100, with NGF starts reporting substructures of limited overlap (ratio of 0.017) only when its tree ensembles contain at least 70 unique nodes. DIPS consistently shows overlap ratios between 0.034 and 0.055 for the number of predictive nodes between 30 and 100. These results reveal that our method selects more stable predictive substructures across all training sets. Looking into the obtained structures, we observe that DIPS uncovers 11 nodes, out of 70 nodes, in common across all folds and these nodes form two large subnetworks as depicted in Fig.4(b). By further investigation, we observe that these substructures largely reside in the temporal lobe (TP.L, TP.R, T2a.R, T2p.L, T3a.R) and the hippocampus (Hip.L/Hip.R), which are the two important brain regions strongly impacted by Alzheimer's disease [24,25].

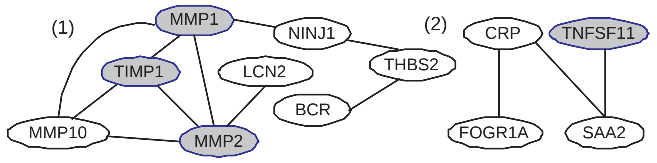


Figure 6: Typical predictive subnetworks uncovered by DIPS from liver metastasis. Shaded nodes correspond to domain-relevant genes.

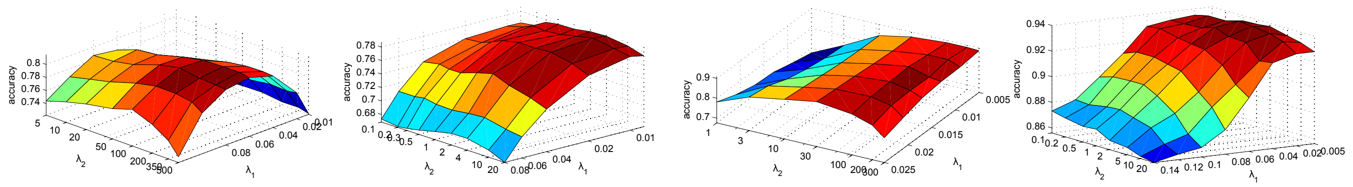
D. Gene Network Datasets

The last two datasets we use contain microarray co-expression data from studies on embryonic development [4] and liver metastasis in human [23], combined with the corresponding functional PPI network [26]. The former dataset contains $1,321$ genes, $5,227$ edges and 35 subjects divided into two global labels based on the developmental tissue layers [4], while the latter comprises $7,383$ genes, $251,916$ edges from 123 subjects labeled as disease/non-disease states.

Prediction Performance: Fig.5(a) compares the prediction accuracy in both datasets. Due to space constraints, we only report the best performance with fine-tuned parameters for each technique. Other than SVM, all techniques perform competitively on the embryonic dataset, achieving more than 70% of prediction accuracy averaged from all folds. MINDS performs better than NGF on these two microarray datasets, and also better than DIPSw/o on the embryonic one. Nevertheless, the performance of DIPS is by far the best, dominating both the baseline SVM and the sampling methods, with a significant accuracy advantage on the liver metastasis dataset, which has a larger number of nodes. Moreover, DIPS's relatively small standard deviation in accuracy in both network datasets demonstrates its stability in predicting global network states across all folds.

Predictive Substructures: As in the case of brain networks, there are no exact ground truths for microarray data. Nevertheless, we choose the more challenging liver metastasis dataset, along 39 cancer-related genes identified in [23] to serve as a ground truth set, and further investigate the overlap of subnetworks selected by DIPS in relation to the ground truth. At its best classification rate, DIPS's substructures involve 65 nodes from each training set and cumulatively 194 nodes from all training sets. Out of these, two subnetworks (shown in Fig.6) are consistently observed across all folds. The genes MMP1, TIMP1, MMP2 and TNFSF11 (shaded in Fig.6) are particularly interesting since they are in agreement with the findings in [23] that identify these genes as the main targets related to the liver metastasis.

As a comparison against other techniques, we evaluate the selected subnetworks in terms of ROC performance in identifying genes in the ground truth. Fig.5(b) plots the ROC curves of competing methods when they obtain their best prediction rate. For node ranking in DIPS and DIPSw/o, we rely on Eq.(17) while in NGF and MINDS, we rank



(a) Imaging networks (b) Brain networks (c) Embryonic develpoment (d) Liver metastasis

Figure 7: Impact of regularization factors λ_1 and λ_2 on the prediction accuracy of DIPS in all datasets.

nodes based on their frequency of occurrence in sampled trees. The ROC curves demonstrate that DIPS selects more ground truth nodes than NGF and MINDS for small false positive rates. As this rate increases, the behavior of all methods become quite similar since none of them are able to discover all ground truth genes. However, the higher rate of true positive makes DIPS a better candidate for identifying new target genes associated with the disease. This property is very important in practice since validating a new target gene is costly and a method with better ROC performance can provide better candidate genes to test experimentally.

E. Impact of regularization factors

In order to provide more insights into the performance of DIPS, we conduct a series of experiments to examine the impact of the two regularization factors on the prediction accuracy. Recall that λ_1 controls the number of selected nodes, while λ_2 enforces the network connectivity within selected nodes. In Fig.7, we show the relationships between the two factors and the prediction accuracy on the 4 datasets. From the plots, a trend is observed that as λ_1 decreases, more nodes are selected, and the prediction accuracy keeps increasing. However, when λ_1 is too small, the prediction rate deteriorates in all datasets, especially on the image data. This can be explained by the fact that a very small λ_1 leads to an overwhelming number of selected substructures, which can potentially contain many irrelevant nodes and lead to over-fitting. A similar trend is also observed by varying λ_2. A small λ_2 leads to a low accuracy as the connectivity information is almost disregarded, while a large λ_2 favors tightly connected subnetworks whose local node values may not strongly influence the global network properties. The most important observation from the four plots is that all surfaces have convex-shapes which suggests that the optimal model can be selected at the peak point at which it has a small number of selected nodes to favor a simple model, while the network connectivity in the predictive substructures is maximized.

F. Scalability

We also evaluate the scalability of our proposed algorithms. Our implementation employs the Lanczos method [27] to compute the leading eigenvectors. To provide an idea about typical running times: DIPS takes 9s and 140s to complete a round of cross validation on the image and brain networks, whereas for the embryonic and liver cancer data, it takes 182s and 17s respectively. These computations are twice as fast as DIPSw/o since the network regularization term allows DIPS to converge much faster. NGF and MINDS take similar time on the image network data at 147s and 165s, and embryonic at 109s and 110s, respectively. However, both methods require much more time on the denser networks of liver and brain. In particular, they both take more than 1900s on the brain networks, due to training and examining a huge number of decision trees. DIPS thus shows much better scalability against the subnetwork sampling methods due to its efficient and effective subspace learning approach with network regularization.

V. Related Work

Recent research in bioinformatics and systems biology has demonstrated the untility of network structure for both classification [4,7] and regression [28,29] tasks. NGF [4] is one representative classification approach based on a random forest classifier that also incorporates biological constraints encoded by a protein-protein interaction (PPI) network. The random forest is iteratively built over features within a connected subgraph of the PPI. Another recent classification scheme, MINDS [7], builds decision trees constrained by the network topology and employs Markov Chain Monte Carlo to sample the space of connected structures. Both MINDS and NGF are sampling ensemble methods that classify network instances based on majority voting. In contrast, our work avoids searching in the exponential space of subnetworks by learning a discriminative subspace and selecting predictive feature substructures within that subspace. In addition, our approach directly optimizes the compactness (feature selection) and discriminative power (accuracy) of features.

Our work also fundamentally differs from other subspace learning approaches [5,10] that either focus on classification in a subspace instead of feature selection [10], or do not consider the network structure to obtain a subspace [5]. Our method selects a controlled-size subset of the features in a network-aware manner, which can then be used for classification or other types of analysis.

Graph classification based on frequent/discriminative substructures has also been considered for chemical and program call graph databases [8,9]. Nodes in this setting are labeled by discrete labels (e.g. chemical element or function name) and the task is to find substructures that are *frequent* in one class of database graphs and *infrequent* in another. The original database graphs are represented and classified as binary feature vectors encoding the inclusion/exclusion of discriminative subgraphs. Our setting is different from frequent substructure classification as we work with continuous labels on nodes whose values are directly used for classification of the global states. The learnt substructures by our method are structurally contained in all network samples and selected feature node values collectively discriminate between global network states.

Dynamic network mining approaches also aim to discover subnetworks of interest in multiple discrete snapshots of an evolving network or in a multi-layer network [30]–[33]. The goal in this line of work, however, is not classification of global network states, but the discovery of abnormal substructures that persist in time or across network layers.

VI. Conclusions

We proposed DIPS, a novel framework for mining predictive substructures in global-state networks. Unlike existing techniques, DIPS avoids the exponential complexity of subnetwork sampling through a two-step scheme: (i) subspace learning, and (ii) predictive substructures selection with network regularization. We utilize two mathematically appealing approaches of spectral graph learning and gradient descent optimization, and we show that DIPS converges to a single optimum solution—a desired property that is hard to achieve by sampling approaches. Through experimental analysis over four real-world datasets, we demonstrated the advantageous performance of DIPS not only based on classification rate but also in terms of domain relevance of the discovered predictive subnetworks.

Acknowledgements: The research work is supported in part by the NSF (IIS-1219254), the NIH (R21-GM094649), and the Army Research Laboratory under Cooperative Agreement Number W911NF-09-2-0053 (NSCTA). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

[1] I. N. Davidson *et al.*, "Network discovery via constrained tensor analysis of fmri data," in *KDD*, 2013.

[2] M. Weiner *et al.*, "The Alzheimer's disease neuroimaging initiative: A review of papers published since its inception," *Alzheimer's & Dementia*, vol. 9, no. 1, pp. 111–194, 2013.

[3] O. Sporns, "Networks analysis, complexity, and brain function," *Complex.*, vol. 8, pp. 56–60, Sept. 2002.

[4] J. Dutkowski and T. Ideker, "Protein networks as logic functions in development and cancer," *PLoS*, 2011.

[5] M. Wall, A. Rechtsteiner, and L. Rocha, "Singular value decomposition and principal component analysis," *A Practical Approach to Microarray Data Analysis*, 2003.

[6] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer-Verlag, 2006.

[7] S. Ranu, M. Hoang, and A. K. Singh, "Mining discriminative subgraphs from global-state networks," in *KDD*, 2013.

[8] S. Ranu and A. K. Singh, "Graphsig: A scalable approach to mining significant subgraphs in large graph databases," in *ICDE*, 2009.

[9] X. Yan, H. Cheng, J. Han, and P. S. Yu, "Mining significant graph patterns by leap search," in *SIGMOD*, 2008.

[10] X. H. Dang *et al.*, "Discriminative subnetworks with regularized spectral learning for global-state network data," in *ECML*, 2014.

[11] S. Soundarajan, T. Eliassi-Rad, and B. Gallagher, "A guide to selecting a network similarity method.," in *SDM*, 2014.

[12] X. H. Dang *et al.*, "Local outlier detection with interpretation," in *PKDD*, pp. 304–320, 2013.

[13] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[14] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Communications ACM*, 1975.

[15] D. R. Hunter and K. Lange, "A tutorial on MM algorithms," *The American Statistician*, 2004.

[16] K. Lange, D. R. Hunter, and I. Yang, "Optimization transfer using surrogate objective functions," *Journal of Computational and Graphical Statistics*, 2000.

[17] J. Gorski, F. Pfeuffer, and K. Klamroth, "Biconvex sets and optimization with biconvex functions: a survey and extensions," *Math. Meth. of OR*, vol. 66, no. 3, pp. 373–407, 2007.

[18] T. Hastie *et al.*, *The Elements of Statistical Learning. Data Mining, Inference, and Prediction*. Springer, 2009.

[19] D. Krstajic *et al.*, "Cross-validation pitfalls when selecting and assessing regression and classification models," *J. Cheminformatics*, 2014.

[20] K. Bache and M. Lichman, "UCI repository," 2013.

[21] D. I. Shuman *et al.*, "The emerging field of signal processing on graphs," *IEEE Signal Process. Mag.*, 2013.

[22] S. Smith *et al.*, "Advances in functional and structural MR image analysis and implementation as FSL," *Neuroimage*, vol. 23, pp. 208–219, 2004.

[23] D. H. Ki *et al.*, "Whole genome analysis for liver metastasis gene signatures in colorectal cancer," *Int J Cancer*, 2007.

[24] B. E. Leonard, *Alzheimer's disease and stroke: possible biochemical causes and treatment strategies*. John Wiley and Sons, 1992.

[25] A. R. Crossman and D. Neary, *Neuroanatomy. An illustrated colour text*. Churchill Livingstone Elsevier, 2010.

[26] D. Ruth *et al.*, "Genes2fans: connecting genes through functional association networks," *BMC bioinformatics*, 2012.

[27] R. B. Lehoucq, D. C. Sorensen, and C. Yang, *ARPACK Users Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*. SIAM, 1998.

[28] C. Li and H. Li, "Variable selection and regression analysis for graph-structured covariates with an application to genomics," *The Annals of Applied Statistics*, 2010.

[29] J. Noirel *et al.*, "Identifying differentially expressed subnetworks with mmg," *Bioinformatics*, 2008.

[30] M. Mongiovì, P. Bogdanov, and A. K. Singh, "Mining evolving network processes," in *ICDM*, 2013.

[31] M. Kivelä, Arenas, *et al.*, "Multilayer networks," *Journal of Complex Networks*, vol. 2, no. 3, pp. 203–271, 2014.

[32] P. Bogdanov, M. Mongiovì, and A. K. Singh, "Mining heavy subgraphs in time-evolving networks," in *ICDM*, 2011.

[33] P. Holme and J. Saramäki, "Temporal networks," *Physics Reports*, vol. 519, no. 3, pp. 97–125, 2012.