Mining Discriminative Subgraphs from Global-state Networks

Sayan Ranu
IBM Research
Manyata Tech Park
Bangalore, India.
sayanranu@in.ibm.com

Minh Hoang
University of California
Dept. of Computer Science
Santa Barbara, CA, USA.
mhoang@cs.ucsb.edu

Ambuj Singh
University of California
Dept. of Computer Science
Santa Barbara, CA, USA.
ambuj@cs.ucsb.edu

ABSTRACT

Global-state networks provide a powerful mechanism to model the increasing heterogeneity in data generated by current systems. Such a network comprises of a series of network snapshots with dynamic local states at nodes, and a global network state indicating the occurrence of an event. Mining discriminative subgraphs from global-state networks allows us to identify the influential subnetworks that have maximum impact on the global state and unearth the complex relationships between the local entities of a network and their collective behavior. In this paper, we explore this problem and design a technique called *MINDS* to mine *minimally discriminative* subgraphs from large global-state networks. To combat the exponential subgraph search space, we derive the concept of an *edit map* and perform Metropolis Hastings sampling on it to compute the answer set. Furthermore, we formulate the idea of *network-constrained decision trees* to learn prediction models that adhere to the underlying network structure. Extensive experiments on real datasets demonstrate excellent accuracy in terms of prediction quality. Additionally, MINDS achieves a speed-up of at least four orders of magnitude over baseline techniques.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*Data Mining*

General Terms

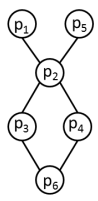
Algorithms

Keywords

discriminative subgraphs, network-constrained decision trees

1. INTRODUCTION

The ability to capture multiple snapshots of a network leads to a "global-state" network in which the snapshots share the same structure but have different values on nodes and/or edges. Furthermore, the network-guided evolution of the local states jointly determines the global network state in each snapshot. Such global-state networks (GS-network) can model a myriad of domain specific features such as traffic congestion in transportation networks [3], evolution of opinions and sentiments on social networks [12], gene expression levels on protein-protein interaction networks [6] and scaffolds in molecular libraries [15]. For example, in protein-protein interaction networks, the expression levels of individual proteins encode logical functions that determine the presence or absence of a disease. In social networks, opinion expressed on a movie by a certain user affects the opinions of his/her friends which in turn sets off a word-of-the-mouth cascade that ultimately decides the global consensus. *How do local node labels govern the evolution of the global network state? Can we save cost by monitoring only a discriminative subgraph and still be able to predict the global network state accurately?* In this paper, we investigate these questions.



Human ID	p_1	p_2	p_3	p_4	p_5	p_6	Cancer
1	1	0	1	0	0	1	No
2	1	1	0	0	1	0	Yes
3	1	1	0	1	1	0	Yes
4	1	1	1	0	0	0	No
5	1	1	1	1	1	1	Yes
6	0	0	0	0	0	0	No
7	0	1	0	0	1	0	No
8	0	1	0	1	1	0	No

Network Structure Local and Global states of the network

Figure 1: A GS-network based modeling of protein-protein interaction data on eight different humans (snapshots). The GS-network models the occurrence of cancer. A protein expression level of 1 denotes abnormal activity, whereas 0 indicates normal expression levels.

Consider the problem of inferring biological outcomes from the human protein-protein interaction network (*PPI*). A hypothetical example is shown in Fig.1. In a PPI, each node corresponds to a protein and two proteins are connected by an edge if they are known to interact while regulating a common biological process. As a result, abnormality in the expression level of a certain protein directly impacts only its neighbors. As evident in Fig.1, the expression level of a protein varies from individual to individual. Research in systems biology has shown that clinical outcomes, such as susceptibility to cancer, depend not only on the expression level of a single protein, but on pathways or network modules [6]. Modeling this phenomenon, therefore, requires us to have a network with dynamic node labels and a global dynamic state; the node labels indicate the protein expression levels in a human and the global state indicates the presence or absence of the disease. To predict the biological outcome, we need to find the sub-networks whose local states accurately predict the global network state.

KDD'13, August 11–14, 2013, Chicago, Illinois, USA.

As illustrated above, GS-networks can model aspects of data that are beyond the scope of static networks. A line of work that closely resembles GS-networks is the idea of time-evolving dynamic networks. A dynamic network consists of a series of networks whose properties change with time. However, dynamic networks lack a global network state and existing techniques on analyzing dynamic networks primarily focus on studying time-evolving recurrent patterns [2, 3, 18]. In contrast, the goal of our problem is to learn the network-encoded logic functions from the local states, and then predict the global network state. For that purpose, we develop a technique called *MINDS (MINing Discriminative Subgraphs)* to mine discriminative subgraphs from large GS-networks.

Learning discriminative subgraphs from GS-networks is key to understanding the complex relationship that exists between the local and the global states. Consider the problem of monitoring environmental sensor networks and learning regression models to predict the intensity of climatic conditions. In an environmental network, each sensor represents a node and measures environmental properties such as pressure and temperature. Two sensors are connected if changes in environmental factors directly influence each other. Now, research in meteorological science has established that climatic conditions in a region depend not only on local factors, but also on environmental conditions across the globe. For example, the intensity of Indian monsoon is linked to El Niño [11]. While limited success has been achieved in making short-term forecasts based on local environmental factors, long-term forecasts based on global factors remain a challenge. Mining discriminative subgraphs from environmental sensor networks would help us identify such global factors and forecast onsets of extreme conditions to minimize the resulting damage.

While optimizing prediction quality is important, it is also essential to learn local models that are consistent with the network structure. Additionally, the mined sub-networks should regularize the GS-network by applying a bias of network constraint towards shrinking the hypotheses space. Compactness of discriminative sub-networks is key to both network regularization as well as real time monitoring. Consider the scenario in sentiment analysis on social networks to predict stock market momentum [12]. The global behavior of users in the network is shaped through their individual opinions and the resulting cascading effects within their social circles. Given the scale of social networks such as Facebook or Twitter, monitoring the entire user base to provide real-time updates on the global consensus is not feasible. Mining the most compact discriminative subgraphs promises to penetrate this scalability bottleneck by identifying smaller groups of influential users that maximally predict the global behavior.

Clearly, mining discriminative subgraphs from GS-networks is a powerful mechanism for identifying network components that are influential in determining the global state. However, given the fact that decades of research has already been performed on learning classification models, an obvious question arises: *How is the problem of mining discriminative subgraphs different from training classifiers?* To answer this question, we highlight the key aspects of our problem that are beyond the scope of a traditional classifier.

1. Semantics: Learn local prediction models that are sensitive to the underlying network structure. In our problem, each feature (or node) is constrained within a structure and the network event being modeled evolves through that structure. On the contrary, a traditional classifier operates on unstructured data where each feature represents an axis in a high-dimensional space. Consequently, any model learned lacks semantic meaning.

2. Level of abstraction: Mine discriminative subgraphs, each of which is self-sufficient in explaining the evolution of the global network state and modeling a coherent event. For example, in PPI, such a subgraph corresponds to a biological process, whereas in environmental networks, a subnetwork represents a region. The local models can further be combined to design ensemble learning algorithms. On the other hand, a traditional classifier is only capable of mining discriminative nodes with the sole focus on prediction quality. Consequently, the learned patterns are of a low-level and do not capture the higher-level structure.

3. Beyond Classification: Mine discriminative subgraphs that not only provide the platform for learning classification models, but also network regularization, regression and monitoring.

To achieve the properties highlighted above, we design a technique called *MINDS* to mine *minimally discriminative* subgraphs from large GS-networks without compromising the underlying network structure. To summarize:

- We formulate the problem of mining minimally discriminative subgraphs from large GS-networks. To learn local prediction models and quantify the discriminative potential of a subgraph, we introduce the concept of *network-constrained decision tree* that learns *network-encoded logic functions* to predict the global network state.
- To tackle the exponential subgraph search space, we formulate the idea of an *Edit Map*, on which we perform Metropolis-Hastings sampling algorithm to drastically reduce the computational cost.
- We perform extensive experiments on real GS-networks to evaluate the efficiency and effectiveness of MINDS. Our results show that the proposed algorithm achieves an accurate approximation of the optimal answer set. Furthermore, MINDS outperforms the current state-of-the-art classifiers developed for PPIs.

2. PROBLEM FORMULATION

A network/graph $G = (V, E)$ is composed of a set of nodes $V = \{v_1, v_2, \cdots, v_n\}$ modeling the entities of the network and a set of edges $E = \{(v_i, v_j) \mid v_i, v_j \in V\}$ modeling the relationships between these entities. A *network snapshot* $N = (V, E, L, S)$ contains two additional parameters: a labeling function $L : V \rightarrow \mathbb{R}$ and the global network state S. While L operates on the node IDs and models the local states, the global state function S quantifies the success of the event being modeled. For simplicity, we assume edges to be *undirected* and $S \in \{-1, 1\}$. However, all of the theory developed in this paper is generalizable to variants such as edge-weighted graphs, directed edges, multi-class states, or continuous valued states.

DEFINITION 1. GLOBAL-STATE NETWORK: *A GS-network is a set of network snapshots* $\mathbb{N} = \{N_1, \cdots, N_n \mid N_i = (V_i, E_i, L_i, S_i)\}$. *We alternatively use the notation* $\mathbb{N} = (V_N, E_N, L_i, S_i)$ *to denote a GS-network where* $V_N = \bigcup_{\forall N_i \in \mathbb{N}} V_i$ *and* $E_N = \bigcup_{\forall N_i \in \mathbb{N}} E_i$.

EXAMPLE 1. *Fig.1 demonstrates a hypothesized GS-network modeling the occurrence of cancer. The global state encodes the presence or absence of cancer and the local states indicate the protein expression levels. All snapshots in this GS-network share the same structure. For snapshots with different structures, the null value is used to denote the state of a missing node. As a result, an edge exists in a snapshot only if it connects to non-null nodes.*

A graph $G = (V, E)$ is a subgraph of a GS-network $\mathbb{N} = (V_N, E_N, L_i, S_i)$, denoted by $G \subseteq \mathbb{N}$, if $V \subseteq V_N$ and $E \subseteq E_N$. A stronger constraint is enforced by the relationship of *induced* subgraphs.

DEFINITION 2. INDUCED SUBGRAPH: $G = (V_G, E_G)$ *is an induced subgraph of GS-network* $\mathbb{N} = (V_N, E_N, L_i, S_i)$, *denoted as* $G \subseteq \mathbb{N}$, *if and only if* $V_G \subseteq V_N$, $E_G \subseteq E_N$, *and* $\forall (u, v) \in E_N$ *where* $u \in V_G$ *and* $v \in V_G$, $(u, v) \in E_G$.

A *supergraph* is defined analogously.

In this paper, we focus on mining only connected induced discriminative subgraphs of a GS-network. Consequently, any reference to a subgraph is assumed to be a connected induced subgraph.

Given a training dataset, our goal is to mine subgraphs that accurately predict the global state S of any snapshot $N \in \mathbb{N}$. Furthermore, the mined subgraphs should be as compact as possible to ensure network regularization. Towards that goal, we first define the notion of *discriminative subgraphs*.

DEFINITION 3. DISCRIMINATIVE SUBGRAPHS: *Given a GS-network $\mathbb{N} = (V_N, E_N, L_i, S_i)$, let $f(G, L)$ be a structure-sensitive prediction function that predicts the global state of a network. If $\mathbb{C} = \{N_i = (V_N, E_N, L_i, S_i) | N_i \in \mathbb{N}, f(G, L_i) = S_i\}$ is the set of correctly predicted networks, then the discriminative potential of subgraph $G \subseteq \mathbb{N}$ is:*

$$\phi(G) = \frac{|\mathbb{C}|}{|\mathbb{N}|} \quad (1)$$

G is discriminative if $\phi(G) \geq \theta$ for a user-provided threshold θ.

Due to our assumption of binary valued global states, $f(G, L)$ is essentially a classification model. For continuous valued global states, $f(G, L)$ would be a regression function. We elaborate on how to learn the prediction function $f(G, L)$ in Secs. 3 and 4.

While one could mine all discriminative subgraphs in the network for a given threshold θ, such an answer set is likely to be informationally sparse. More specifically, given a subgraph G that is discriminative, all of G's supergraphs are discriminative as well. This result follows from the fact that any prediction function $f(G, L)$ learned from $G = (V_G, E_G)$ can be learned from a supergraph $G' = (V_{G'}, E_{G'}) \supseteq G$ as well since the feature set $V_{G'} \supseteq V_G$ contains all the information embedded in G. Therefore , to mitigate this potential issue of information sparsity, our goal is to extract the set of *minimally discriminative* subgraphs.

DEFINITION 4. MINIMALLY DISCRIMINATIVE SUBGRAPHS: *A subgraph G is minimally discriminative if $\phi(G) \geq \theta$ and the set $\{G' | G' \subseteq G, \phi(G') \geq \phi(G)\} = \emptyset$.*

As can be seen, minimally discriminative subgraphs correspond to the smallest possible subnetworks within a GS-network that are influential enough to determine its global state. Consequently, mining minimally discriminative subgraphs allows us to maximize the information density in the answer set and avoid overfitting.

3. NETWORK-CONSTRAINED DECISION TREES

Sec. 2 formalizes the discriminative potential of any graph G. However, we still need to learn a structure-sensitive prediction function $f(G, L)$ so that $\phi(G)$ can be quantified. From Defn. 3, $\phi(G)$ is directly proportional to the probability $P(f(G, L_i) = S_i)$ for any network $N_i \in \mathbb{N}$. Without the constraints of the structure, the problem is essentially that of learning a classification/regression model on the GS-network $\mathbb{N}$ using only nodes in G as features. However, as already discussed in Sec. 1, such an approach lacks semantic meaning and the level of abstraction required to gain meaningful insights from the mined network features.

To concretize the importance of structure in our problem further, consider the hypothesized GS-network in Fig.1 and the local events where protein p_1 over-expresses (i.e., samples where p_1 has node label 1). From the network structure, it is evident that an abnormality in p_1 has a direct impact only on p_2. As a result, out of the five human samples where $p_1 = 1$, p_2 behaves abnormally on four of them. Now, through p_2, the abnormality in p_1 has a cascading effect on the expression levels of p_3, p_4 and p_5. A deeper analysis

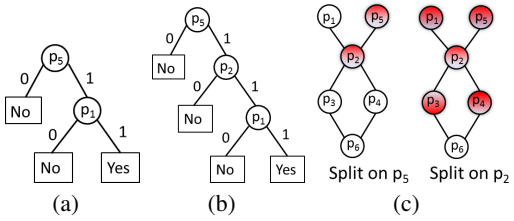


Figure 2: The optimal traditional decision tree (a) and the optimal network-constrained decision tree (b) for the GS-network shown in Fig.1. (c) Demonstrates how infection spreads in the network. The infected nodes are highlighted in red.

of the example reveals that whenever both p_1 and p_5 behave abnormally, the corresponding human samples are susceptible to cancer. Clearly, p_1 and p_5 are statistically the most informative nodes and this fact is reflected in Fig.2(a) which shows the optimal decision tree (DT) for Fig.1. Notice that the learned model is completely oblivious to the fact that the process evolved from p_1 to p_5 through p_2. Even though p_2 is not statistically informative, structurally, it is the "bridge" between p_1 and p_5, and thus, plays a key role in determining the global network state. From a biological viewpoint, if p_2 can somehow be shielded from abnormal behaviors in p_1, then the risk of the disease is greatly diminished. Clearly, the importance of p_2 must be recognized and the failure of traditional local learners in capturing this key structural aspect highlights their limitations in mining structured data such as graphs. To capture the importance of network structure within the framework of a traditional classifier, we introduce the concept of *network-constrained decision trees (NCDT)*.

Similar to the goals of a DT, an NCDT also learns the optimal boolean function that best predicts the global states using the local node labels. However, an NCDT also models the evolution of a process through the network and imposes additional constraints on nodes that can be used to split the training dataset. For the first split, an NCDT is free to choose any node. After the first split node n_1 is selected, an NCDT considers n_1 as "infected". Furthermore, the infection spreads from n_1 to all of n_1's neighbors, and an NCDT can select only one of the infected nodes to decide the next split. Based on this constraint, once the second split node n_2 is selected, n_2's neighbors in turn get affected and this process repeats recursively, like in a DT, till leaf nodes are reached. The additional constraint of splitting only through infected nodes ensures that "structural bridges" are captured and we do not overfit the learned models. Note that the proposed NCDT can easily be employed for learning a regression function as well by incorporating the same strategies used for learning regression DTs. Formally, an NCDT is defined as the following:

DEFINITION 5. NETWORK-CONSTRAINED DECISION TREES: *A decision tree is also an NCDT if all nodes in any path starting from the root form a connected component in the GS-network. Consequently, the nodes in the NCDT are guaranteed to form a connected component as well.*

EXAMPLE 2. *Fig.2(b) shows the optimal NCDT for the GS-network in Fig.1. Fig.2(c) demonstrates how the "infection" spreads in the network. In the optimal NCDT, p_5 is selected as the first split node. As a result, p_5 and p_2 get infected. Among the infected nodes, p_2 is selected for the next split, which results in the infection spreading to p_1, p_3 and p_4. p_1 is selected for the third split to pro-*

duce the optimal NCDT. On the other hand, the DT in 2(a) is not an NCDT since p_5 and p_1 do not form a connected component.

Certainly, DT is not the only classifier that can be adapted to learn a structure-sensitive prediction function. We choose NCDTs since it forms a natural and intuitive extension to DT. Additionally, as shown later in Sec. 5, the linear construction cost of NCDTs make it highly efficient when compared to other state-of-the-art classification techniques such as Support Vector Machines.

3.1 Computational challenges

With the formalization of NCDTs, we now have a mechanism to quantify the discriminative potential of any graph. In this section, we analyze the computational challenges faced while mining minimally discriminative subgraphs.

CLAIM 1. *Computing the optimal NCDT is NP-hard.*

PROOF: Learning the optimal DT is known to be NP-complete [9]. Given any dataset $\mathbb{D} = \{d_1, \cdots, d_n\}$ where $d_i = (x_1, \cdots, x_k, y_i)$ with y_i being the class label, the decision tree problem is to determine whether there exists a decision tree of size (i.e., number of nodes in the tree) less than s that classifies each d_i correctly. Given an arbitrary instance of the problem, we construct a clique with all features (or nodes) $1, \cdots, k$ connected to each other. It is easy to see that a DT of size less than s exists if and only if an NCDT of size less than s exists. In other words, learning an NCDT on a clique is equivalent to learning a DT. □

NP-hardness of computing the optimal NCDT is not the only computational challenge. To mine minimally discriminative subgraphs, we need to first enumerate all possible subgraphs of the GS-network, and then compute their discriminative potentials. Unfortunately, the number of subgraphs in a network grows exponentially with its size and as a result, enumerating all possible subgraphs is not feasible. Consequently, the proposed problem presents us with a unique challenge: *how can we mine minimally discriminative subgraphs even without enumerating the entire search space?*

4. MINING DISCRIMINATIVE SUBGRAPHS

Sec. 3.1 outlines the two computational challenges in mining discriminative subgraphs from large GS-networks. In this section, we address these two challenges. First, we devise a strategy to compute NCDTs *greedily*. Next, to combat the exponential search space, we impart an ordering on the candidate subgraphs in the form of an *edit map*, and then perform *Metropolis-Hastings* [1] sampling on the map to compute an accurate approximation.

4.1 Greedy computation of NCDT

As in greedy learning of traditional DTs, the first node to split the training dataset is selected greedily by choosing the one with the highest statistical importance. The statistical importance can be quantified using any of the existing attribute value tests such as *information gain* or *gini index*. For our implementation, we use information gain which is defined as the following

$$IG(\mathbb{N}, u) = E(\mathbb{N}) - \sum_{l \in L^*(u)} \frac{|\mathbb{N}_l|}{|\mathbb{N}|} E(\mathbb{N}_l) \quad (2)$$

where $\mathbb{N}$ is a GS-network, $L^*(u)$ is the set of all possible labels for node u, $\mathbb{N}_l = \{N = (V_N, E_N, L, S) | N \in \mathbb{N}, L(u) = l\}$ is the set of networks where node u has label l and $E(\cdot)$ is the entropy of a set. The first split divides $\mathbb{N}$ into $|L^*(u)|$ subsets. Next, the set of infected nodes is computed, and each of the subsets is split recursively by choosing the infected node with the highest information gain for that subset. As in a DT, this process completes when leaf nodes are reached where either the global states of all snapshots belong to a single class or no feature exists to split further.

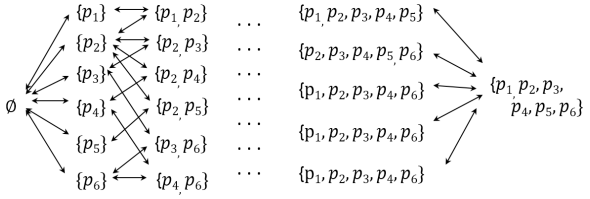


Figure 3: The top three levels and the bottom two levels of the edit map of the GS-network in Fig.1. We just show the set of nodes in each vertex of the edit map since the edges can be concluded from the definition of an induced subgraph.

4.2 Searching greedily in the subgraph space

A greedy learning of the NCDT tackles the NP-hardness challenge outlined earlier. Now, we focus on the second challenge of exploring the exponential search space, which cannot be computed or stored due to its sheer size. Our current capabilities only allow us to compute the discriminative potential of a given subgraph and evaluate local modifications to further improve its discriminative power. Thus, our only hope to reach the globally optimal solution is through locally optimal choices. Towards that goal, a greedy approach could be adopted. First, the node with the highest information gain can be identified as the seed, and an NCDT can be constructed greedily around that seed node. The corresponding subgraph would therefore be the nodes spanning the NCDT. If the subgraph is discriminative, then it is added to the candidate set, otherwise discarded. To continue populating the candidate subgraph set, the process is restarted from the seed node with the second highest information gain. Once all nodes have been explored, the answer set can be computed by identifying only the minimally discriminative subgraphs from the candidate set.

While a greedy strategy is computationally efficient, the discriminative potential of the subgraph is highly restricted by the choice of the initial seed node. If the seed lies in a neighborhood where the rest of the nodes provide low information gain, then the resultant subgraph will be non-discriminative as well. More importantly, a greedy algorithm is constrained to continue expanding the NCDT in the low informative region even after realizing that the initial seed is an informative outlier. What is therefore critical to the success of any local optimization based approach is being sensitive to back-tracking and negating any of the wrong choices already made. MINDS builds upon this intuition by converting the subgraph search space into an *Edit Map*, and then performing MH sampling on the map to mine minimally discriminative subgraphs.

4.3 Edit map

The edit map (*EM*) of a GS-network represents all possible *edits* that can be performed on any subgraph $G \subseteq \mathbb{N}$ in the form of an edge-weighted *partial-order* graph.

DEFINITION 6. EDIT MAP: *The edit map of a GS-network $\mathbb{N} = (V_N, E_N, L_i, S_i)$ is a directed edge-weighted graph $M = (V_M, E_M)$, where $V_M = \{G | G \subseteq \mathbb{N}\}$, $E_M = \{(G = (V, E), G' = (V', E')) \mid either\ G' \supseteq G,\ V' = V \cup \{u\}, u \notin V,\ u \in V_N\ or\ G' \subseteq G,\ V' = V \backslash \{u\}, u \in V\}$, and $f_M : E_M \rightarrow \mathbb{R}$ is a function that assigns a weight to each edge in E_M.*

As can be seen, the EM structures the search space into an edge-weighted graph where each vertex corresponds to a distinct subgraph $G \subseteq \mathbb{N}$. Besides, G is connected to all of its subgraphs with one less node, denoted as $G \rightarrow u$, and supergraphs with one additional node, denoted as $G \leftarrow u$. Each edge in the EM, incident on

some vertex G, $G \subseteq \mathbb{N}$, corresponds to an *edit* which either inserts or deletes a node from G. By performing a series of edits, G can be transformed to any subgraph $G' \subseteq \mathbb{N}$. The edge weights quantify the impact of the edits on the discriminative potential. We elaborate on how to compute these edge weights in Sec. 4.5. Hereon, we use the term node to denote an entity in the GS-network, and vertex to denote a candidate subgraph in the EM. Fig.3 shows the EM of the GS-network in Fig.1. The leftmost vertex in the EM represents the *null graph*, and the rightmost vertex represents the entire network structure. The EM is always connected.

As noted earlier, the size of the EM is exponential with respect to the GS-network size and thus cannot be computed or stored in its entirety. However, given any subgraph, we can make local edits to enhance our chances of finding the minimally discriminative subgraphs. We formalize this idea by initiating a Metropolis-Hastings sampling on the EM to guide us towards discriminative subgraphs.

4.4 Metropolis-Hastings sampling

The Metropolis-Hastings (MH) algorithm is a Monte Carlo Markov Chain sampling algorithm whose goal is to sample from a *target distribution* τ. Given a state space $\Omega = \{s_1, \cdots, s_n\}$, let $v_i \geq 0$ be the value of item $s_i \in \Omega$. Our goal is to draw state s_i from τ, where

$$\tau_i = \frac{v_i}{C} \tag{3}$$

$C = \sum_{i=1}^{n} v_i$ is a normalizing constant. For large n, C is difficult to compute and thus, computing τ directly is not feasible. MH allows us to simulate τ by converting the state space into an n-state Markov chain with an arbitrary transition matrix Q. Let the current state, X_t, at time step t be i. The MH algorithm performs the following three steps to determine X_{t+1}:

- Draw a random state j with probability Q_{ij}
- Compute the acceptance probability α_{ij}, where

$$\alpha_{ij} = min\left\{1, \frac{\tau_j Q_{ji}}{\tau_i Q_{ij}}\right\} = min\left\{1, \frac{v_j Q_{ji}}{v_i Q_{ij}}\right\} \tag{4}$$

- $$X_{t+1} = \begin{cases} j, & \text{with probability } \alpha_{ij} \\ i & \text{with probability } 1 - \alpha_{ij} \end{cases} \tag{5}$$

In this formulation, the transition matrix Q is called the *proposal distribution matrix* and α_{ij} is termed as the *acceptance probability*. The optimal proposal distribution is the one that best approximates the target distribution, and the acceptance probability should model how precise the approximation is. The proposal distribution and the acceptance probability can be combined to define the following *one-step transition matrix* T:

$$T_{ij} = \begin{cases} Q_{ij}\alpha_{ij} & \text{if } i \neq j \\ 1 - \sum_{k \neq i} Q_{ik}\alpha_{ik} & \text{if } i = j \end{cases} \tag{6}$$

The Markov chain with transition matrix T is *reversible* and *ergodic*. Additionally, the stationary distribution π of the Markov chain converges to the target distribution τ.

4.5 MH sampling on the edit map

Sec. 4.4 describes how the MH algorithm can be used to sample from a target distribution. In this section, we utilize the MH algorithm to sample discriminative subgraphs from the exponential subgraph search space. In our problem, each subgraph (or vertex) in the EM is a state. The target is to approximate the answer set by sampling only a small subset of highly discriminative subgraphs from the entire search space. Clearly, the quality of the sampled set is critical to the accuracy of our approximation. In MH algorithm, the quality of the stationary distribution depends on two key parameters: the proposal distribution and the acceptance probability. We thus focus on defining these parameters to best approximate the answer set of minimally discriminative subgraphs.

The proposal distribution matrix Q is a function of the edge weights in the EM. The edge weights reflect the quality of the edits on a given subgraph G. An edit is "good" if the newly constructed subgraph increases our chances of finding a minimally discriminative subgraph. While an increase in the discriminative potential can be observed only when nodes are added, Q should allow node deletions so that the sampler does not converge to local optimums. Furthermore, since our goal is to mine minimally discriminative subgraphs and maximize information density, deletions should be preferred over "bad" additions that do not increase the discriminative potential. Thus, to summarize, given a subgraph G, we group all possible edits on G into three classes:

1. **Good addition:** $\phi(G)$ increases due to addition of a node.
2. **Bad addition:** $\phi(G)$ does not increase.
3. **Deletion:** Delete nodes to avoid converging to local optimums.

We next formalize these intuitions.

First, we focus on quantifying the edge-weights corresponding to additions. As can be seen, the impact of a node addition on the discriminative potential can be computed only after the NCDT is constructed on the new subgraph following the edit. Consequently, if G has m supergraph neighbors, we need to build m NCDTs. On dense networks, m can be significantly large. Furthermore, this operation needs to be repeated for each graph that we sample in the EM. As a result, an accurate computation of the edge weights is computationally expensive. To reduce this computational burden, we compute an approximation of the actual edge weight based on information gain. Assuming the current state $X_t = G = (V_G, E_G)$, first, the NCDT on G is built. Next, we construct the set $\mathbb{M} = \{N_i = (V_N, E_N, L_i, S_i) | N_i \in \mathbb{N}, f(G, L_i) \neq S_i\}$ of misclassified networks, where $f(G, L)$ is the prediction function. For each node $u \in V_N$ that can be added to G to construct a supergraph $G' = G \leftarrow u \in G_{sup}$, we group them into two sets based on their information gain:

$$A_- = \{u | IG(\mathbb{M}, u) \leq 0 \mid G' = G \leftarrow u \in G_{sup}\}$$
$$A_+ = \{u | IG(\mathbb{M}, u) > 0 \mid G' = G \leftarrow u \in G_{sup}\}$$

A_- and A_+ represent the "bad" and "good" additions respectively, and G_{sup} represents the set of all possible supergraphs of G. The quality of performing an addition is now quantified as follows:

$$A(u) = \begin{cases} \frac{\Delta}{|A_-|} & \text{if, } u \in A_- \\ (1 - \Delta)\frac{IG(\mathbb{M}, u)}{\sum_{v \in A_+} IG(\mathbb{M}, v)} & \text{if, } u \in A_+ \end{cases} \tag{7}$$

where Δ is a small probability distributed evenly among the "bad" additions. As can be seen, the sampler is most likely to select one of the "good" additions based on its information gain. However, since information gain is only an approximation of the actual increase in discriminative potential, with a small probability Δ, the sampler would explore "bad" additions as well. Furthermore, as in the case of deletions, being open to "bad" additions avoids convergence to local optimums. We discuss how to select Δ in Sec. 4.5.1.

Next, we focus on quantifying the utility of deletions. As discussed earlier, deletions are necessary to maximize the information density in the sampled subgraphs, and ensure that the sampler does not explore non-minimally discriminative subgraphs. To achieve this property, the need for deletions on a subgraph G should be dependent on $\phi(G)$. If $\phi(G)$ is low, additions are preferred so that the sampler adds more information and moves to discriminative subgraphs. On the other hand, if $\phi(G)$ is high, it is preferable to delete nodes and explore other regions of the subgraph search space. We model these requirements using the following proposal

Algorithm 1 MINDS($\mathbb{N}, \theta$)

1: $\mathbb{A} := \emptyset$
2: $t := 0$
3: $X_t :=$ A randomly selected subgraph $G = (V_{X_t}, E_{X_t}) \subseteq \mathbb{N}$
4: Build NCDT on X_t
5: **while** $t < maxiter$ **do**
6: **if** X_t is minimally discriminative **then**
7: $\mathbb{A} := \mathbb{A} \cup \{X_t\}$
8: $\mathbb{A} := \mathbb{A} \backslash \{G' \in \mathbb{A} | G' \supseteq X_t,\ \phi(X_t) = \phi(G')\}$
9: $X_{t_{sub}} := \{X_t \rightarrow u | u \in V_{X_t}, u$ is not a cut-vertex$\}$
10: $X_{t_{sup}} := \{X_t \leftarrow u | u \notin V_{X_t}, \exists v \in V_{X_t}, (u, v) \in E_N\}$
11: Compute $Q_{X_t G'}, \forall G' \in X_{t_{sub}} \cup X_{t_{sup}}$
12: Choose neighbor G' from proposal distribution $Q_{X_t G'}$
13: Update NCDT for G'
14: $\alpha := \frac{v_{G'} Q_{G' X_t}}{v_{X_t} Q_{X_t G'}}$
15: **if** $uniform(0, 1) \leq \alpha$ **then**
16: $t := t + 1$
17: $X_t := G'$
18: **return** $\mathbb{A}$

distribution:

$$Q_{GG'} = \begin{cases} \frac{\beta}{|G_{sub}|} & \text{if } G' = G \rightarrow u \in G_{sub} \\ (1 - \beta) A(u) & \text{if } G' = G \leftarrow u \in G_{sup} \end{cases} \quad (8)$$

where β models the need for deletions based on $\phi(G)$ and is quantified as the following:

$$\beta = \frac{e^{K\phi(G)}}{e^K} \quad (9)$$

and K is some large constant. As $\phi(G)$ increases, most of the probability is distributed among deletes, whereas at a low $\phi(G)$, additions are preferred.

The definition of β completes the formalization of the proposal distribution matrix Q. We next focus on defining the acceptance probability $\alpha_{GG'}$. Since our goal is to sample discriminative subgraphs, v_G in Eq. 4 can be set to $\phi(G)$. However, with such a score assignment, any supergraphs of $G' \supseteq G$ where $\phi(G') = \phi(G) \geq \theta$ will be considered as a "good" state even though from Definition 4, G' will never be part of the answer set. Therefore, to model this property, we compute v_G as follows:

$$v_G = \begin{cases} \epsilon \approx 0 & \text{if } \exists G' = G \rightarrow u \in G_{sub}, \phi(G') = \phi(G) \\ \epsilon \approx 0 & \text{if } \exists G' = G \leftarrow u \in G_{sup}, \phi(G') > \phi(G) \\ \phi(G) & \text{otherwise} \end{cases} \quad (10)$$

Eq. 10 ensures that transitions from non-minimally discriminative states are always accepted (cases 1 and 2). If no such conclusion can be drawn from the current state and its neighbors, then transitions are accepted based on their discriminative potentials.

4.5.1 Parameters

Although the proposed model contains two parameters, Δ and K, none of them have a profound impact on the results as long as the parameters are set within an appropriate range. Δ is a small probability that allows exploration of locally "bad" node additions in hope of a globally optimal solution. The results are consistent for any values in the range [0.001,0.005]. In our experiments, we set $\Delta = \frac{|A_-|}{|V_N|}$. For K in Eq. 9, any large value in the range [100, 300] would produce consistent results.

4.6 Implementation details

Alg. 1 presents the pseudocode of MINDS. MINDS starts exploring the search space from a random subgraph $X_t \subseteq \mathbb{N}$ and the NCDT on X_t is built (lines 2-4). By leveraging the memoryless property of the MH sampling algorithm, MINDS constructs only the local neighborhood of X_t in the EM (lines 9-10). To further reduce computation costs, only those entries of Q that involve graph X_t are computed (line 11). Furthermore, to optimize storage costs in the exponential search space, at any time step, only two copies of NCDTs are maintained in memory: one for the current state and the other for the proposed state. As a result, MINDS achieves both of the desired goals: accurate simulation of the target distribution through MH sampling, and computational efficiency through memoryless property of Markov chains.

Table 1: Summary of the GS-networks used. The 'Event' column denotes the event being modeled.

Dataset	#Nodes	#Edges	#Events	Event
$D_1 1001[6]$	11203	57235	371	Breast Cancer
$D_2 1001[5]$	9673	39240	183	Liver Metastasis
$D_3 1001[6]$	1321	5227	35	Embryonic Origin

5. EXPERIMENTS

The objectives of our evaluation procedure are the following:
• Evaluate the sampling quality and scalability of MINDS.
• Investigate the importance of network structure.
• Study the impact of noise in GS-networks on the mined patterns.
Furthermore, based on the observed results, quantify the statistical significance of the results obtained in the cleaned datasets.
• Analyze the power of minimally discriminative subgraphs on predicting network states.

5.1 Datasets

To benchmark MINDS on real GS-networks, we use three different PPIs. Each of the PPIs represents the human protein interaction network. Although all three networks are drawn from the same species, they are curated by three different agencies and differ in the various cellular processes being modeled. Consequently, no mapping exists between nodes across networks. Table 1 summarizes the GS-networks. Fig.4(a) shows the degree distribution of each of these networks. As expected, they display a scale-free behavior. The "#Events" column denotes the number of network events/snapshots observed. Each event is associated with local node labels and a global state. The local node labels represent the protein expression levels and the global state indicates the clinical outcome of the event being modeled. To discretize the protein expression levels, we follow the standard procedure from system biology [5]. First, the expression levels of each protein are standard normalized so that the mean and the standard deviation is 0 and 1 respectively. Next, the expression levels of all proteins across all events are sorted and the values in the top 25% are set to 1. The remaining values are set to 0. A node label 1 therefore indicates the corresponding protein to over-express and 0 indicates normal behavior.

5.2 Experimental setup

For experiments evaluating quality of MINDS, we select the maximum possible subset of network events from each dataset such that the distribution of the global states is balanced. A balanced set ensures that a majority-class classifier can only achieve an accuracy of 0.5. Otherwise, we use the entire datasets. Unless specifically mentioned, we iterate the sampler for $100,000$ time steps. We set the default threshold for discriminative potential to 0.8. The value of constant K in Eq. 9 is set to 200. Typically, K has minimal impact on the results as long as $K > 100$.

5.2.1 MH with SVM

To highlight the importance of capturing the underlying network structure, we replace NCDT with SVM as the learning methodology in the MH sampling step. More specifically, at any subgraph

$G \subseteq \mathbb{N}$, $\phi(G)$ is computed based on SVM with linear kernel. The SVM is not constrained by network connectivity as long as all features (nodes) are part of G. In MH sampling with SVM, only the proposal distribution matrix is altered based on the feature ranking mechanism outlined in [4]. Instead of information gain, the importance of a node u is quantified based on its absolute weight value $w(u)$ in the learned SVM model. Thus, at each state with graph G, two SVM models are learned: SVM model M_G on G, and SVM model M_{Gsup} that uses all nodes in G in addition to the nodes that can be added to G on the EM. $w(u) \in M_G$ quantifies the importance of deleting node u, and $w(u) \in M_{Gsup}$ quantifies the importance of adding node u to G. Thus,

$$Q_{GG'} = \begin{cases} (\beta)\frac{\frac{1}{|w(u)|}}{\sum_{G \to v \in G_{sub}} \frac{1}{|w(v)|}} & \text{if } G' = G \to u \in G_{sub} \\ (1-\beta)\frac{|w(u)|}{\sum_{G \leftarrow v \in G_{sup}} |w(v)|} & \text{if } G' = G \leftarrow u \in G_{sup} \end{cases}$$

The formulation of β (9) and α (10) remains the same.

5.3 Performance analysis of sampling

First, we evaluate the quality of the subgraphs sampled from the EM. The quality of the sampling procedure is the single most important aspect that affects the accuracy of the computed answer set. To obtain an accurate approximation, the sampler should often visit graphs that have high discriminative potential. Therefore, to analyze the desired correlation between visit count and the discriminative potential of a subgraph, we plot the likelihood of a subgraph being visited given its discriminative potential. Figs.4(b)-4(c) demonstrate the results on two of the largest datasets D_1 and D_2. To set the baseline, we perform random sampling of subgraphs. As can be seen, majority of the subgraphs visited by MINDS have $0.9 \leq \phi(G) \leq 1$. On the other hand, if we select subgraphs randomly from the network, the visit count is uniformly distributed across all values of discriminative potential. During random selection, we ensure that the subgraph sizes are drawn from the same distribution visited by MINDS. For SVM-guided MH sampling, a trend similar to MINDS is also observed. However, the sampler spends more time in the range $0.7 \leq \phi(G) \leq 0.9$. This result shows that the proposed formulations of the proposal distribution matrix and the acceptance probability, for both SVM and NCDT, are effective in separating out the discriminative subgraphs from those that are non-discriminative.

The second important aspect of the sampling procedure that affects the quality of the answer set is the size of the visited subgraphs. Recall that for subgraph G to be minimally discriminative, none of G's subgraphs can have a higher discriminative potential than G. Clearly, that reduces to sampling small subgraphs but with high discriminative potentials. Thus, to analyze how well the proposed technique conforms to this desired sampling property, we analyze the distribution of the subgraph sizes that are sampled. First, we plot the distribution of the subgraph sizes against the discriminative potential. As can be seen in Figs. 4(d)-4(e), the information density in subgraphs sampled by MINDS is significant higher than in SVM for subgraph sizes above 5. While the discriminative potential of subgraphs sampled by MINDS saturates at sizes around 20, to achieve the same potential, SVM requires significantly larger subgraphs. This result highlights the importance of capturing the network structure through NCDTs. Since SVM is oblivious to network connectivity, it only utilizes the information encoded in the network nodes. On the other hand, the structural constraint in NCDT captures the network through which the process being modeled evolves, and consequently, utilizes the information encoded in both the nodes as well as edges. The importance of capturing the structure is further established in Fig.4(f). Fig. 4(f) analyzes the sampled subgraph sizes and plots their distribution against visit count. Since the information densities in SVM sampled subgraphs are significantly lower than MINDS, much of the SVM sampling is restricted on large subgraphs to achieve a high discriminative potential. Consequently, most of the subgraphs sampled by SVM are not minimally discriminative. We further analyze the importance of structure in Sec. 5.4.

Next, we focus on analyzing the scalability of MINDS. First, we evaluate the growth rate of the running time against network size in Fig. 4(g). To construct GS-networks of varying sizes, we select subgraphs from D_1. To set the baseline, we first attempt an exhaustive subgraph exploration on a network containing only 60 nodes. However, due to the exponential subgraphs search space, the exhaustive search failed to complete even after 12 hours, during which it analyzed more than 100×10^6 subgraphs. The projected time based on the number of subgraphs that were left unprocessed was 200 hours. Given this context, even on a network containing $10,000$ nodes, MINDS is more than three orders of magnitude faster than an exhaustive exploration on a network of size 60. Compared to SVM, MINDS is more than a magnitude faster than SVM. Fig. 4(h) analyzes scalability against the number of events in the network. As can be seen, the running time of MINDS grows linearly and is more than 10 times faster than SVM. Finally, in Fig. 4(i), we evaluate the growth rate of the running time against the discriminative potential threshold θ. As expected from the formulation in Sec. 4.5, the running time is constant since other than the randomness in the sampling procedure, θ does not change the number of computations performed.

Fig. 4(j) analyzes the quality of the answer set with the number of iterations. To quantify quality, we verify whether the answer set captures the entire spectrum of the minimally discriminative subgraphs. For that purpose, we use the metric of *information density span*. First, we define *information density* of a graph $G = (V_G, E_G)$ as $\frac{\phi(G)}{|V_G|}$. The information density span of the answer set is the difference between the highest and the lowest information densities among all graphs in the answer set. The highest and the lowest information densities define the boundaries of the answer set, and we consider the answer set to converge once the density span stops expanding. As can be seen in Fig.4(j), after $100,000$ iterations, the increase in the span is minimal. We use information density instead of discriminative potential, since graphs in the answer set should be both discriminative and compact.

Although the above experiments indicate an excellent performance, an important question remains to be answered: *How accurate is our approximation?* To answer this question, we select a sub-network of D_1 containing 60 nodes and try computing the ground truth answer set. Unfortunately, due to the huge computational cost mentioned above, computing the ground truth even on miniature networks is not feasible. Thus, we use an alternative strategy for constructing the ground truth. We synthetically implant NCDTs on the network structure of D_1 and generate a balanced set of network events ensuring that the implanted NCDTs have an accuracy of 1. While generating the network events and the accompanying node labels, we set the node labels according to the functions encoded by the implanted NCDTs. For nodes that are not used by the NCDTs, we set the labels arbitrarily. Due to this controlled construction, subgraphs spanning the implanted NCDTs have discriminative potentials of 1.0. Now, to evaluate the accuracy, we execute MINDS on the constructed GS-network and verify whether the discriminative subgraphs are extracted.

Table 2 presents the results averaged over 800 runs as the sizes of the implanted subgraphs are varied (we explain the results for graph $N = (V_N, E_N)$ in Sec. 5.4). In all of our experiments, MINDS

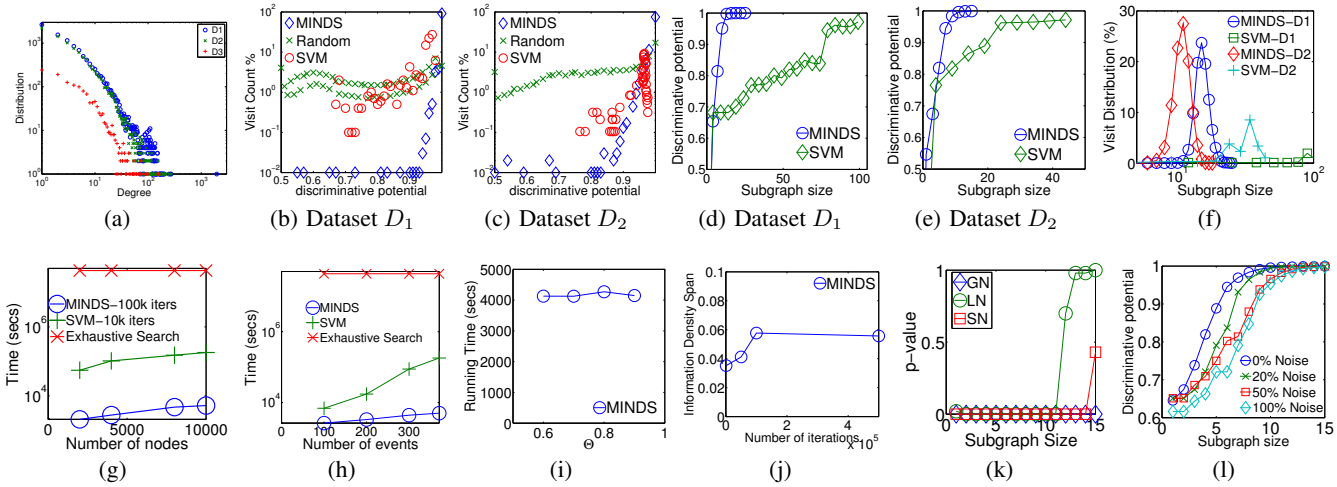


Figure 4: (a) Degree distribution in datasets D_1, D_2 and D_3. (b-c) Growth rate of visit count with discriminative potential. (d-e) Growth rate of discriminative potential with subgraph size. (f) Distribution of sampled subgraph sizes. Growth rate of the running time with (g) network size, (h) number of events, and (i) θ. (j) Quality of the answer set against number of iterations. (k) Statistical significance of the patterns mined by MINDS. (l) Impact of structural noise on discriminative subgraphs.

is able to identify a subgraph with a discriminative potential of 1. Interestingly, for $|V_I| \geq 5$, MINDS is able to identify a smaller subgraph G and still achieve an accuracy of 1. Due to the human-mediated construction of the implanted NCDTs, the trees are not always optimal. MINDS is able to identify that non-optimality and construct a more complex and compact NCDT while retaining the same accuracy. This result establishes that MINDS is efficient in both accurately approximating the answer set and regularizing the network.

5.4 Impact of noise

What happens when the GS-network is noisy? How much of the signal is lost due to inaccuracies in the network structure? In this section, we answer these questions. There are three sources of noise: the network structure (SN), the local expression levels at nodes (LN), and the global network state (GN). To understand the impact of noise, we perform permutation tests [7]. More specifically, first, we create a null hypotheses by introducing noise in the PPI and run MINDS on the noisy network. Due to the addition of noise, the discriminative signal in the original network is lost. This process of introducing noise is repeated one million times to compute the distribution of discriminative potentials for a subgraph of a given size. Based on the null hypothesis, we compute the p-value for the distribution observed in Fig.4(e). For example, in Fig.4(e), a subgraph of size 6 has an average discriminative potential of 0.94 and our goal is to compute the statistical significance of this event. The higher the statistical significance, the more discriminative is the information that is lost due to the addition of noise.

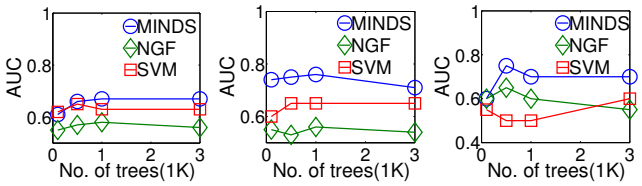
To introduce structural noise (SN), we randomly construct edges between nodes while keeping the total number of edges in the original network, and the local and global labels intact. For LN and GN, we adopt similar strategies by permuting the local node labels and global snapshot labels respectively. As in SN, we ensure that the distributions of the noisy local and global states are the same as in the original datasets. Fig.4(k) shows the results for all subgraph sizes sampled by MINDS in dataset D_2. As can be seen, regardless of the noise introduction policy, the p-values for the majority of the subgraph sizes are 0. The significance of the results decreases for sizes above 12 in the SN and LN methods due to diminishing return of marginal gains. More specifically, in the original dataset, the discriminative potential saturates for subgraphs above a size of 8. In the noisy dataset, even after perturbing the local states, MINDS is able to identify discriminative subgraphs when their sizes are above 12. In other words, due to the permutation, the saturation happens from size 12 onwards instead of 8.

To further understand the impact of noise, we analyze the information density of the discriminative subgraphs as the amount of SN is varied. A noise level of 20% indicates that 20% of the edges are randomly constructed; the remaining edges are the same as in the original network. Fig. 4(l) demonstrates the results. As expected, with increase in noise level the structural signal is lost, and consequently, the average discriminative potential for a given subgraph size decreases. Similar results are observed for LN and GN as well.

Finally, we investigate the impact of SN on discovery of the ground-truth answer set. As in the verification procedure earlier, synthetic NCDTs are implanted on the GS-network. Next, SN is introduced and we compare the discriminative subgraphs $N = (V_N, E_N)$ identified in the noisy network with the implanted ones. As can be seen in Table 2, although discriminative subgraphs are still identified, they are significantly larger in size. This is in sharp contrast to the results in original network where MINDS is able to identify subgraphs that are actually smaller than the implanted

Table 2: Accuracy of MINDS against ground-truth answer set and the impact of noise on network structure. $I = (V_I, E_I)$ denotes the implanted discriminative subgraph, $G = (V_G, E_G)$ and $N = (V_N, E_N)$ denote the best subgraph discovered in the original and noisy GS-networks respectively.

$\|V_I\|$	$\|V_G\|$	$\|V_N\|$	$\|V_G \cap V_I\|$	$\|V_N \cap V_I\|$	$\frac{\|V_G \cap V_I\|}{\|V_G \cup V_I\|}$	$\frac{\|V_N \cap V_I\|}{\|V_N \cup V_I\|}$	$\phi(G)$	$\phi(N)$
3	4.93	10.02	2.59	2.31	0.64	0.28	1	1
5	5.08	8.53	3.42	3.41	0.57	0.38	1	1
8	6.52	8.02	4.61	4.61	0.48	0.42	1	1
10	8.02	8.47	5.38	5.26	0.43	0.4	1	1



(a) Breast Cancer (b) Liver Metastasis (c) Embryonic Origin

Figure 5: (a-c) Growth rate of the AUC with number of trees for the events.

ones. This is a direct consequence of the structural signal getting lost due to shuffling of edges.

Overall, the analysis reveals the following:

1. The network structure contains discriminative information that should be captured in the classifier for optimum performance.

2. If the network is noisy, the sizes of the discriminative subgraphs grow to compensate for the missing information.

5.5 Analysis of prediction quality

In this section, we verify the predictive power of the mined subgraphs for network state classification. To evaluate prediction performance, we perform 5-fold cross validation. First, we compute the answer set on the training dataset. Then, from each of the minimally discriminative subgraphs, we extract the learned NCDT. The state predicted on the testing set by the majority of the NCDTs is considered as the collective predicted state. The accuracy is quantified by the area under the ROC curve (AUC).

To benchmark our technique, we use the state-of-the-art classifier *Network Guided Forests (NGF)* [6] designed specifically for PPIs, and SVM. NGF employs a greedy sampling strategy similar to the algorithm outlined in Sec. 4.2. By sampling multiple times, NGF constructs a random forest. Furthermore, NGF incorporates domain specific information, such as favoring high-degree proteins, and a second round of clustering to identify decision modules to boost the performance. In contrast, MINDS incorporates no domain specific information. Figs. 5(a)-5(c) demonstrate the classification accuracy as the number of trees is varied for MINDS and NGF. As can be seen, MINDS achieves a higher AUC across all network events that is up to 65% higher than NGF and SVM.

6. RELATED WORK

While the idea of a GS-network has not been formalized before, the problem of mining protein modules from PPIs has been studied. Prior work in systems biology has indicated that the network structure is critical towards identifying discriminative protein modules and have focused on network regularization [13, 14] and classification [5, 6]. However, with the exception of [6], existing techniques assume homogeneous activity on entire protein modules and are only capable of identifying simple logic functions such as the sum or the multiplication of the expression levels. On the other hand, [6] employs a greedy strategy by starting from the most informative node in the network and then building a tree within that neighborhood. As illustrated in Sec. 4.2, a greedy strategy is susceptible to converging to a local optima. Furthermore, as opposed to solving the problem only in the context of PPIs, our work proposes a generalized algorithm for GS-networks. The features mined by MINDS can not only be employed for classification, but also for regression, network regularization and real-time monitoring.

As discussed earlier in Sec. 1, dynamic networks [2, 3, 18] and mining discriminative subgraphs from graph databases [8, 10, 15–17, 19] are the two closest lines of work from the computer science community. However, both fail to model the problem being proposed here. Dynamic networks do not contain global states and each snapshot is ordered temporally. Mining discriminative subgraphs from graph databases, on the other hand, assumes a database of multiple graphs and mines subgraphs that are statistically "over-represented" in one of the classes.

7. CONCLUSION

In this paper, we formalized the concept of a global-state network and designed a technique called *MINDS* to learn the network-encoded evolution rules that determine the global network state. MINDS learns local prediction models by constructing network-constrained decision trees on minimally discriminative subgraphs. The mined patterns regularize the network and provide the platform for an array of higher level tasks such as classification, regression and network monitoring. To tackle the exponential subgraph search space, MINDS structures the space in the form of an Edit Map and performs MH sampling on it to mine discriminative subgraphs. Extensive experiments performed on real GS-networks demonstrate MINDS to be efficient in mining patterns that are accurate and statistically significant. MINDS is up to 4 orders of magnitudes faster than baseline techniques.

Acknowledgements: The work was supported by NSF grants IIS-1219254 and IIS-0917149.

8. REFERENCES

[1] B. A. Berg. Introduction to markov chain monte carlo simulations and their statistical analysis. *NATIONAL UNIVERSITY OF SINGAPORE*, 7, 2005.

[2] M. Berlingerio, F. Bonchi, B. Bringmann, and A. Gionis. Mining graph evolution rules. In *ECML PKDD*, pages 115–130, 2009.

[3] P. Bogdanov, M. Mongiovi, and A. K. Singh. Mining heavy subgraphs in time-evolving networks. In *ICDM*, pages 81–90, 2011.

[4] Y.-W. Chang and C.-J. Lin. Feature ranking using linear svm. *Journal of Machine Learning Research*, 3:53–64, 2008.

[5] S. A. Chowdhury, R. K. Nibbe, M. R. Chance, and M. Koyutürk. Subnetwork state functions define dysregulated subnetworks in cancer. *Journal of Computational Biology*, 18(3):263–281, 2011.

[6] J. Dutkowski and T. Ideker. Protein networks as logic functions in development and cancer. *PLoS Comput Biol*, 7, 09 2011.

[7] R. A. Fisher. *The design of experiments / by Sir Ronald A. Fisher*. Oliver Boyd, Edinburgh :, 7th ed. edition, 1960.

[8] M. A. Hasan and M. J. Zaki. Output space sampling for graph patterns. *PVLDB*, 2(1):730–741, 2009.

[9] L. Hyafil and R. L. Rivest. Constructing optimal binary decision trees is np-complete. *Information Processing Letters*, 5(1):15 – 17, 1976.

[10] N. Jin, C. Young, and W. Wang. Gaia: graph classification using evolutionary computation. In *SIGMOD Conference*, pages 879–890, 2010.

[11] K. K. Kumar, B. Rajagopalan, and M. A. Cane. On the weakening relationship between the Indian monsoon and ENSO. *Science*, 284(5423):2156–2159, 1999.

[12] D. Lee, O.-R. Jeong, and S.-g. Lee. Opinion mining of customer feedback data on the web. In *ICUIMC*, pages 230–235, 2008.

[13] C. Li and H. Li. Network-constrained regularization and variable selection for analysis of genomic data. *Bioinformatics*, 24(9):1175–1182, 2008.

[14] J. Noirel, G. Sanguinetti, and P. C. Wright. Identifying differentially expressed subnetworks with mmg. *Bioinformatics*, 24(23):2792–2793, 2008.

[15] S. Ranu, B. T. Calhoun, A. K. Singh, and S. J. Swamidass. Probabilistic substructure mining from small-molecule screens. *Molecular Informatics*, 30:809–815, 2011.

[16] S. Ranu and A. K. Singh. Graphsig: A scalable approach to mining significant subgraphs in large graph databases. In *ICDE*, pages 844–855, 2009.

[17] S. Ranu and A. K. Singh. Mining statistically significant molecular substructures for efficient molecular classification. *J. Chem. Inf. Model.*, 49:2537–2550, 2009.

[18] C. Robardet. Constraint-based pattern mining in dynamic graphs. In *ICDM*, pages 950–955, 2009.

[19] X. Yan, H. Cheng, J. Han, and P. S. Yu. Mining Significant Graph Patterns by Scalable Leap Search. In *SIGMOD*, pages 433–444, 2008.