Semi-Supervised Feature Selection for Graph Classification

Xiangnan Kong
Department of Computer Science
University of Illinois at Chicago
851 S. Morgan Street
Chicago, IL 60607-0753
xkong4@uic.edu

Philip S. Yu
Department of Computer Science
University of Illinois at Chicago
851 S. Morgan Street
Chicago, IL 60607-0753
psyu@cs.uic.edu

ABSTRACT

The problem of graph classification has attracted great interest in the last decade. Current research on graph classification assumes the existence of large amounts of labeled training graphs. However, in many applications, the labels of graph data are very expensive or difficult to obtain, while there are often copious amounts of unlabeled graph data available. In this paper, we study the problem of semi-supervised feature selection for graph classification and propose a novel solution, called gSSC, to efficiently search for optimal subgraph features with labeled and unlabeled graphs. Different from existing feature selection methods in vector spaces which assume the feature set is given, we perform semi-supervised feature selection for graph data in a progressive way together with the subgraph feature mining process. We derive a feature evaluation criterion, named gSemi, to estimate the usefulness of subgraph features based upon both labeled and unlabeled graphs. Then we propose a branch-and-bound algorithm to efficiently search for optimal subgraph features by judiciously pruning the subgraph search space. Empirical studies on several real-world tasks demonstrate that our semi-supervised feature selection approach can effectively boost graph classification performances with semi-supervised feature selection and is very efficient by pruning the subgraph search space using both labeled and unlabeled graphs.

Categories and Subject Descriptors

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

General Terms

Algorithm, Performance, Experimentation

KDD'10, July 25–28, 2010, Washington, DC, USA.

Keywords

Semi-Supervised Learning, Feature Selection, Graph Classification, Data Mining

1. INTRODUCTION

Graphs are ubiquitous and have become increasingly important in modeling diverse kinds of objects. In many real-world applications, instances are not represented as feature vectors, but as graphs with complex structures, *e.g.*, chemical compounds, program flows and XML web documents. One central issue in graph mining research is graph classification, which has a wide variety of real world applications, *e.g.* drug activity predictions, toxicology tests and kinase inhibitions. A major difficulty in graph classification lies in the complex structure of graphs and lack of vector representations. Selecting a proper set of features for graph data is an essential and important procedure for graph classification.

The general problem of feature selection is well studied in the literature. Semi-supervised feature selection problem for graph data, however, has not been studied in this context so far. Conventional feature selection approaches on graph data assume, explicitly or implicitly, that there exists a large amount of labeled training data. However, in many real world applications, the labels of graph data are very expensive or difficult to obtain. Creating a large training dataset can be too expensive, time-consuming or even infeasible. For example, in molecular medicine, it requires time, efforts and excessive resources to test drugs' anti-cancer efficacies by pre-clinical studies and clinical trials, while there are often copious amounts of unlabeled drugs or molecules available from various sources.

Thus it is much desired that the large amounts of unlabeled graphs can be effectively utilized to select better features for graphs, and improve the graph classification performances. For example, in Figure 1, we show a dataset with two labeled graphs and four unlabeled graphs. Based only on the two labeled graphs, subgraph feature "a-b" and "a-c" are both discriminative features. Clearly, when we consider the distribution of the four unlabeled graphs, "a-b" is more likely to be useful than "a-c". This is because the unlabeled graphs are not separable based on the subgraph feature "a-c".

Despite its value and significance, the semi-supervised feature selection for graph classification is a much more challenging task due to the specific characteristics of the task. The reasons are listed as follows.

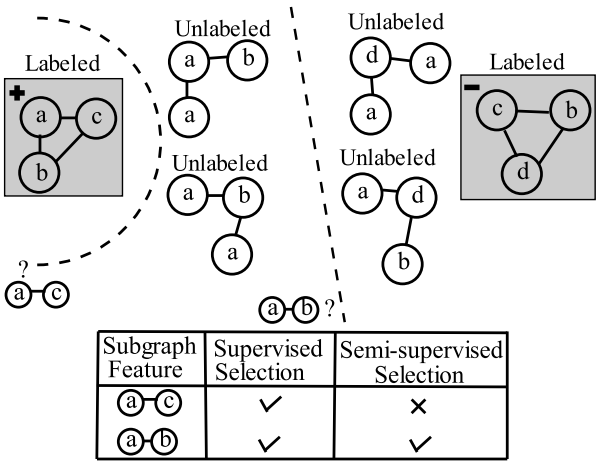


Figure 1: An example of semi-supervised feature selection on graph data. The subgraph feature "a-b" is more useful than "a-c" based on both labeled and unlabeled graphs.

1. *Lack of labels.* Conventional feature selection in graph classification approaches focuses on supervised settings [7, 14, 13]. The mining strategy of discriminative subgraph patterns strictly follows the assumption that there exists a large amount of labeled graphs. However, many real-world graph classifications usually suffer from a lack of training graphs. It is usually laborious, or even infeasible to create a large training set of graph instances.

2. *Lack of features.* Another fundamental problem in semi-supervised feature selection on graph data lies in the complex structures and lack of feature representations of graphs. Conventional feature selection approaches in vector spaces, which assume a candidate feature set is available, cannot be directly applied to graph data, because it is usually infeasible to generate all the subgraph features of a graph dataset before feature selection. The number of subgraphs is usually too large to be fully generated, since it grows exponentially with the graph size. Furthermore checking subgraph isomorphism is NP-complete.

   In order to efficiently find discriminative subgraph features, conventional supervised subgraph feature mining approaches rely on the label information from a large training set to prune the subgraph search space and select useful features [14]. However, when the number of labeled graphs is not large enough, the usefulness of the mined subgraph features can be weak, and the pruning of the subgraph mining process can be ineffective.

Figure 2 illustrates the feature selection process in conventional graph classification approaches. Obviously, when there is only a small number of labeled graphs available, supervised approaches cannot work well due to two reasons: (1) During the subgraph features mining procedure, supervised feature selection approaches for graph classification need to employ evaluation criteria to select discriminative subgraph features based on labeled graphs. However, when the labeled graphs are too few, the usefulness of the selected subgraph features can be weak, and thus detriment to the classification performances. (2) During the subgraph feature mining procedure, most supervised graph classification approaches require a branch-and-bound search to avoid exhaustive enumeration of all subgraphs in a dataset. However, when there are not enough labeled graphs, the pruning ability of the upper-bound based on labeled graphs can be poor, thus making it infeasible to find discriminative subgraph features within a reasonable amount of time.

In this paper, we introduce a novel framework to the above problems by mining subgraph features using both labeled and unlabeled graphs. Our framework is illustrated in Figure 3. Different from existing supervised feature selection methods for graph classification, our approach, called gSSC, can utilize both labeled and unlabeled graphs to find optimal subgraph features for graph classification. We first derive a feature evaluation criterion, named gSemi, based upon a given graph dataset with both labeled and unlabeled graphs. Then we propose a branch-and-bound algorithm to efficiently search for optimal subgraph features by deriving an upper-bound of gSemi and pruning the subgraph search space using labeled and unlabeled graphs. In order to evaluate our model, we perform comprehensive experiments on real-world graph classification tasks. The experiments demonstrate that the proposed semi-supervised feature selection method for graph classification outperforms supervised approaches and is very efficient by pruning the subgraph search space using both labeled and unlabeled graphs.

The rest of the paper is organized as follows. We start by a brief review on related works of graph feature selection and semi-supervised feature selection in Section 2. We then introduce the preliminary concepts, give the problem analysis and present the gSemi criterion in Section 3. In Section 4, we derive an upper-bound of gSemi and propose the gSSC method. Then Section 5 reports the experiment results on real-world graph classification tasks. In Section 6, we conclude the paper.

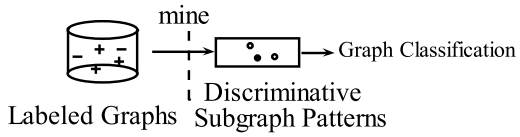


Figure 2: Supervised Feature Selection Process for Graph Classification

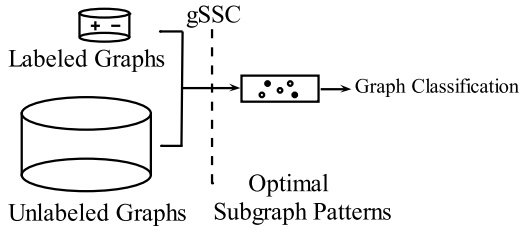


Figure 3: gSSC Semi-Supervised Feature Selection Process for Graph Classification

2. RELATED WORK

To the best of our knowledge, this paper is the first work on semi-supervised feature selection problem for graph classification. Some research works have been done in related areas.

Extracting subgraph features from graph data have been investigated by many researchers. The goal of such approaches is to extract informative subgraph features from a set of graphs. Typically some filtering criteria are used. Upon whether considering the label information, there are two types of approaches: unsupervised and supervised. A typical evaluation criterion is frequency, which aims at collecting frequently appearing subgraph features. Most of the frequent subgraph feature extraction approaches are unsupervised. For example, Yan and Han develop a depth-first search algorithm: gSpan [15]. This algorithm builds a lexicographic order among graphs, and maps each graph to an unique minimum DFS code as its canonical label. Based on this lexicographic order, gSpan adopts the depth-first search strategy to mine frequent connected subgraphs efficiently. Many other approaches for frequent subgraph feature extraction have also been developed, *e.g.* AGM [5], FSG [8], MoFa [2], FFSM [4], and Gaston [10]. Moreover, supervised subgraph feature extraction problem has also been studied in literature, such as LEAP [14] and CORK [13], which look for discriminative subgraph patterns for graph classifications.

Dimensionality reduction and feature selection in vector spaces have also been studied. Several recent works use pairwise constraints as weak supervision for dimensionality reduction, *i.e. must-link* constraints [1] (pairs of instances with the same class) and *cannot-link* constraints [12] (pairs of intstances with different classes). Feature selection methods in vector spaces using both labeled and unlabeled instances have also been proposed [16, 11], which select useful features within a pre-defined feature set. These methods assume that a set of candidate features is given before the feature selection. However, conventional semi-supervised feature selection approaches cannot be directly applied to graph data, because it is usually infeasible to generate all the subgraph features of a graph dataset before feature selection. The number of subgraphs is usually too large to be fully generated, since it grows exponentially with the graph size. Instead, our proposed semi-supervised feature selection for graph data works in a progressive way: the semi-supervised feature selection is integrated to the subgraph feature generation, which can skip most of the bad subgraph features without even generating them.

3. PROBLEM FORMULATION

In this section, we formulate the semi-supervised feature selection problem for graph classification based on subgraph features.

3.1 Semi-Supervised Feature Selection

Before presenting the semi-supervised feature selection model for graph classification, we first introduce the notations that will be used throughout this paper. Let $\mathcal{D} = \{G_1, \cdots, G_n\}$ denote the entire graph dataset, which consists of n graph objects, represented as *connected graphs*. The data set includes both labeled and unlabeled graphs. We assume that the first l graphs within $\mathcal{D}$ are labeled by $\{y_1, \cdots, y_l\}$, where $y_i \in \{-1, +1\}$ denotes the binary class label assigned to G_i. For convenience, we also denote the labeled graph dataset by $\mathcal{D}_l = \{G_1, \cdots, G_l\}$, and the unlabeled graph dataset as $\mathcal{D}_u = \{G_{l+1}, \cdots, G_n\}$, $\mathcal{D} = \mathcal{D}_l \cup \mathcal{D}_u$.

Definition 1 (Connected Graph). *A graph is represented as $G = (\mathcal{V}, E, \mathcal{L})$, where $\mathcal{V}$ is a set of vertices $\mathcal{V} = \{v_1, \cdots, v_{n_v}\}$, $E \subseteq \mathcal{V} \times \mathcal{V}$ is a set of edges, $\mathcal{L}$ is the set of symbols for the vertices and the edges. A connected graph is a graph such that there is a path between any pair of vertices.*

Definition 2 (Subgraph). *Let $G' = (\mathcal{V}', E', \mathcal{L}')$ and $G = (\mathcal{V}, E, \mathcal{L})$ be connected graphs. G' is a subgraph of $G (G' \subseteq G)$ iff: (1) $\mathcal{V}' \subseteq \mathcal{V}$, (2) $E' \subseteq E$, (3) $\mathcal{L}' \subseteq \mathcal{L}$. If G' is a subgraph of G, then G is a supergraph of G'.*

In this paper, we adopt the idea of subgraph-based graph classification approaches, which assume that each graph object G_i is represented as a feature vector $\boldsymbol{x}_i = [x_i^1, \cdots, x_i^m]^\top$ corresponding to a set of subgraph patterns $\{g_1, \cdots, g_m\}$. Denote x_i^k as the binary feature associated with the subgraph pattern g_k. $x_i^k = 1$ iff g_k is a subgraph of G_i ($g_k \subseteq G_i$), otherwise $x_i^k = 0$.

The key issue of semi-supervised feature selection for graph classification is how to find the most informative subgraph patterns from a limited number of labeled graphs and a large number of unlabeled graphs. So, in this paper, the studied research problem can be described as follow: in order to train an effective graph classifier, how to efficiently find a set of optimal subgraph features from both labeled and unlabeled graphs?

Mining the optimal subgraph features from both labeled and unlabeled graphs is a non-trivial task due to the following problems:

(P1) How to properly evaluate the usefulness of a set of subgraph features based upon both labeled and unlabeled graphs?

(P2) How to find the optimal subgraph features within a reasonable amount of time by avoiding the exhaustive enumeration? The subgraph feature space of graph objects is usually too large, because the number of subgraphs grows exponentially with the size of the graphs. It is infeasible to completely enumerate all the subgraph features for a given graph dataset.

In the following sections, we will first introduce the optimization framework for selecting informative subgraph features from labeled and unlabeled graphs. Next we will describe our subgraph mining strategy using the evaluation criteria derived from the optimization solution.

3.2 Optimization Framework

We first address the problem (P1) discussed in Section 3.1 by defining the subgraph feature selection as an optimization problem. Our target is to find an optimal set of subgraph features from both labeled and unlabeled graphs. Formally, let us introduce the following notations:

- $\mathcal{S} = \{g_1, g_2, \cdots, g_m\}$: the given set of all the subgraph features, which are used to predict class membership of graph instances. Usually there is only a subset of the subgraph features $\mathcal{T} \subseteq \mathcal{S}$ relevant to the graph classification task.
- $\mathcal{T}^*$: the optimal set of subgraph features $\mathcal{T}^* \subseteq \mathcal{S}$.

- $J(\mathcal{T})$: an evaluation criterion to estimate the usefulness of subgraph feature subset $\mathcal{T}$.

- X: the matrix consisting binary feature vectors using $\mathcal{S}$ to represent the graph dataset $\{G_1, G_2, \cdots, G_n\}$. $X = [\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n] = [\boldsymbol{f}_1, \boldsymbol{f}_2, \cdots, \boldsymbol{f}_m]^\top \in \{0,1\}^{m \times n}$, where $X = [X_{ij}]^{m \times n}$, $X_{ij} = 1$ iff $g_i \subseteq G_j$. The first l graphs are labeled as $y_1, \cdots, y_l$.

- $\mathcal{C}$ and $\mathcal{M}$: $\mathcal{C} = \{(i,j)|y_i y_j = -1\}$ denotes the *cannot-link* pairwise constraint sets among labeled graphs. $\mathcal{M} = \{(i,j)|y_i y_j = 1\}$ denotes the *must-link* pairwise constraint sets among labeled graphs.

We propose the following general optimization framework to select optimal subgraph feature set:

$$\mathcal{T}^* = \underset{\mathcal{T} \subseteq \mathcal{S}}{\operatorname{argmax}} J(\mathcal{T}) \qquad \text{s.t. } |\mathcal{T}| \leq t, \tag{1}$$

where $|\cdot|$ denotes the size of the feature set and t is the maximum number of feature selected. The objective function in Eq. 1 has two components: the evaluation criterion $J(\mathcal{T})$ and the subgraph features of graphs $\mathcal{S}$.

We assume that the optimal subgraph features set should have the following properties: (a) *cannot-link*: labeled graphs in different classes should be far away from each other; (b) *must-link*: labeled graphs in the same class should be close to each other; (c) *separability*: unlabeled graphs should be able to be separated from each other. Intuitively, (a) and (b) only consider the constraints from labeled graphs, and tend to select the most discriminative subgraph features based on the graph labels. They are similar to the LDA [9] criterion. Note (c) incorporates the distribution of unlabeled graphs, and tends to select the subgraph features that can separate graphs far from each other. It is similar to the PCA's assumption, which is expressed as the average squared distance between unlabeled samples. An opposite example for property (c) is: The subgraph features that are too rare or too frequent in the dataset are not useful at all, because unlabeled graphs cannot be separated from each other using these subgraph features. Similar assumptions have also been used by previous works on dimensionality reduction in vector spaces [16].

Based upon the above properties, we derive an evaluation criterion $J(\mathcal{T})$ as follow:

$$\begin{aligned} J(\mathcal{T}) = \; & \frac{\alpha}{2|\mathcal{C}|} \sum_{y_i y_j = -1} (D_\mathcal{T} \boldsymbol{x}_i - D_\mathcal{T} \boldsymbol{x}_j)^2 \\ & - \frac{\beta}{2|\mathcal{M}|} \sum_{y_i y_j = 1} (D_\mathcal{T} \boldsymbol{x}_i - D_\mathcal{T} \boldsymbol{x}_j)^2 \\ & + \frac{1}{2|\mathcal{D}_u|^2} \sum_{G_i, G_j \in \mathcal{D}_u} (D_\mathcal{T} \boldsymbol{x}_i - D_\mathcal{T} \boldsymbol{x}_j)^2 \end{aligned} \tag{2}$$

where $D_\mathcal{T} = \operatorname{diag}(\boldsymbol{d}(\mathcal{T}))$ is a diagonal matrix indicating which features are selected into feature set $\mathcal{T}$ from $\mathcal{S}$, $d(\mathcal{T})_i = I(g_i \in \mathcal{T})$. α, β are two parameters, which control the weights of the three types of constraints. Different settings of α and β can refer to different scenarios, and reflect different beliefs we have for the problem. A discussion on the parameter setting will be presented analytically in Section 4.4 and empirically in Section 5.4.

By defining a matrix $W = [W_{ij}]^{n \times n}$ as

$$W_{ij} = \begin{cases} \frac{\alpha}{|\mathcal{C}|} & \text{if } y_i y_j = -1 \\ -\frac{\beta}{|\mathcal{M}|} & \text{if } y_i y_j = 1 \\ \frac{1}{|\mathcal{D}_u|^2} & \text{if } G_i, G_j \in \mathcal{D}_u \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

we can rewrite the $J(\mathcal{T})$ in Eq. 2 as follow:

$$\begin{aligned} J(\mathcal{T}) = \; & \frac{1}{2} \sum_{i,j} (D_\mathcal{T} \boldsymbol{x}_i - D_\mathcal{T} \boldsymbol{x}_j)^2 W_{ij} \\ = \; & \operatorname{tr}(D_\mathcal{T}{}^\top X (D - W) X^\top D_\mathcal{T}) \\ = \; & \operatorname{tr}(D_\mathcal{T}{}^\top X L X^\top D_\mathcal{T}) \\ = \; & \sum_{g_k \in \mathcal{T}} (\boldsymbol{f}_k{}^\top L \boldsymbol{f}_k) \end{aligned} \tag{4}$$

where $\operatorname{tr}(\cdot)$ is the trace of a matrix, D is a diagonal matrix whose entries are column sums of W, *i.e.* $D_{ii} = \sum_j W_{ij}$. $L = D - W$ is a Laplacian matrix.

By denoting function $h(g_k, L) = \boldsymbol{f}_k{}^\top L \boldsymbol{f}_k$, the optimization in Eq. 1 can be written as

$$\begin{aligned} \max_{\mathcal{T}} \quad & \sum_{g_k \in \mathcal{T}} h(g_k, L) \\ \text{s.t.} \quad & \mathcal{T} \subseteq \mathcal{S}, |\mathcal{T}| \leq t \end{aligned} \tag{5}$$

Definition 3 (gSemi). *Let $\mathcal{D} = \{G_1, \cdots, G_n\}$ denote a graph dataset, with first l graphs labeled as $y_1, \cdots, y_l$. Suppose W is a matrix defined as Eq. 3. L is a Laplacian matrix defined as $L = D - W$, where D is a diagonal matrix, $D_{ii} = \sum_j W_{ij}$. We define a quality criterion q called gSemi, for a subgraph feature g as*

$$q(g) = h(g, L) = \boldsymbol{f}_g{}^\top L \boldsymbol{f}_g \tag{6}$$

where $\boldsymbol{f}_g = [f_g^{(1)}, \cdots, f_g^{(n)}]^\top \in \{0,1\}^n$ is the indicator vector for subgraph feature g, $f_g^{(i)} = 1$ iff $g \subseteq G_i$ $(i = 1, 2, \cdots, n)$. Since the Laplacian matrix L is positive semi-definite, for any subgraph pattern g, $q(g) \geq 0$.

The optimal solution to the problem in Eq. 5 can be found by using gSemi to make feature selection on a set of subgraphs $\mathcal{S}$. Suppose the gSemi values for all subgraphs are denoted as $q(g_1) \geq q(g_2) \geq \cdots \geq q(g_m)$ in sorted order. Then the optimal solution to the optimization problem in Eq. 5 is:

$$\mathcal{T}^* = \{g_i | i \leq t\}. \tag{7}$$

4. gSSC

In this section, we address the problem (P2) discussed in Section 3.1 by proposing an efficient method to find the optimal set of subgraphs features from a dataset with both labeled and unlabeled graphs.

The straightforward method is the exhaustive enumeration: We first enumerate all subgraph patterns in the graph dataset, and then calculate the gSemi values for all subgraph patterns. This method is usually impractical, because the number of subgraphs grows exponentially with the size of the graphs. Inspired by recent graph classification approaches, *e.g.* [14], which put their evaluation criteria into the subgraph pattern mining process and develop constraints to

prune search spaces, we take a similar approach by deriving a different constraint from both labeled and unlabeled graphs. In order to avoid the exhaustive search, we proposed a branch-and-bound algorithm, named gSSC, which is summarized as follow: a) Adopt a canonical search space where all the subgraph patterns can be enumerated. b) Search through the space, and find the optimal subgraph features by gSemi. c) Propose an upper bound of gSemi and prune the search space. Details with these three steps will be described in the next subsections.

4.1 Subgraph Mining

In this paper, we adopted a depth first search algorithm, gSpan proposed by Yan et al[15], to enumerate all subgraphs from a graph dataset. The key idea of gSpan[15] is that, instead of enumerating subgraphs and testing for isomorphism, they first build a lexicographic order of all the edges of a graph, and then map each graph to an unique minimum DFS code as its canonical label. The minimum DFS codes of two graphs are equivalent iff they are isomorphic. Details can be found in [15]. Based on this lexicographic order, a depth-first search (DFS) strategy is used to efficiently search through all the subgraphs in a DFS code tree. By a depth-first search through the DFS code tree's nodes, we can enumerate all the subgraphs of a graph in their DFS codes' order. And the nodes with non-minimum DFS codes can be directly pruned in the tree, which saves us from performing an explicit isomorphic test among the subgraphs.

4.2 Upper Bound of gSemi

By adopting gSpan's DFS Code Tree, we can efficiently enumerate all the subgraph patterns of a graph dataset in a canonical search space. We now derive an upper bound for the gSemi value which can be used to prune the subgraph search space. A convenient method to compute a upper-bound on gSemi value is given as follow:

THEOREM 1 (UPPER BOUND OF GSEMI). *Given any two subgraphs $g, g' \in \mathcal{S}$, g' is a supergraph of g ($g' \supseteq g$). The gSemi value of g' ($q(g')$) is bounded by $\hat{q}(g)$ (i.e., $q(g') \leq \hat{q}(g)$). $\hat{q}(g)$ is defined as follow:*

$$\hat{q}(g) \triangleq \boldsymbol{f}_g{}^\top \hat{L} \boldsymbol{f}_g \tag{8}$$

where the matrix $\hat{L}$ is defined as $\hat{L}_{ij} \triangleq \max(0, L_{ij})$. $\boldsymbol{f}_g = \{I(g \subseteq G_i)\}_{i=1}^n \in \{0,1\}^n$ is a vector indicating which graphs in a graph dataset $\{G_1, \cdots, G_n\}$ contain the subgraph g, $I(\cdot)$ is the indicator function. Suppose the gSemi value of g is $q(g) = \boldsymbol{f}_g{}^\top L \boldsymbol{f}_g$.

PROOF.

$$q(g') = \boldsymbol{f}_{g'}{}^\top L \boldsymbol{f}_{g'} = \sum_{i,j:G_i,G_j \in \mathcal{G}(g')} L_{ij}$$

where $\mathcal{G}(g') \triangleq \{G_i | g' \subseteq G_i, 1 \leq i \leq n\}$. Since g' is the supergraph of g ($g' \supseteq g$), according to anti-monotonic property, we have $\mathcal{G}(g') \subseteq \mathcal{G}(g)$. Also $\hat{L}_{ij} \triangleq \max(0, L_{ij})$, we have $\hat{L}_{ij} \geq L_{ij}$ and $\hat{L}_{ij} \geq 0$. So,

$$\begin{aligned} q(g') &= \sum_{i,j:G_i,G_j \in \mathcal{G}(g')} L_{ij} \leq \sum_{i,j:G_i,G_j \in \mathcal{G}(g')} \hat{L}_{ij} \\ &\leq \sum_{i,j:G_i,G_j \in \mathcal{G}(g)} \hat{L}_{ij} = \hat{q}(g) \end{aligned}$$

Thus, for any $g' \supseteq g$, $q(g') \leq \hat{q}(g)$. □

$\mathcal{T}$ = gSSC($\mathcal{D}$, $\boldsymbol{y}_l$, *min_sup*, t)

Input:

$\mathcal{D}$:	Graph data set $\{G_1, \cdots, G_n\}$
$\boldsymbol{y}_l$:	The first l graphs' labels, where $\boldsymbol{y}_l = [y_1, \cdots, y_l]^\top$
min_sup :	Minimum support threshold
t :	number of subgraph feature selected

Process:

1 $\mathcal{T} = \emptyset$, $\theta = 0$;
2 Recursively visit the DFS Code Tree in gSpan:
3 g = currently visited subgraph in DFS Code Tree
4 if $|\mathcal{T}| < t$, then
5 $\mathcal{T} = \mathcal{T} \cup \{g\}$;
6 else if $q(g) > \min_{g' \in \mathcal{T}} q(g')$, then
7 $g_{min} = \text{argmin}_{g' \in \mathcal{T}} q(g')$ and $\mathcal{T} = \mathcal{T} / g_{min}$;
8 $\mathcal{T} = \mathcal{T} \cup \{g\}$ and $\theta = q(g_{min})$;
9 if $\hat{q}(g) \geq \theta$ and $freq(g) \geq min\_sup$, then
10 Depth-first search the subtree rooted from node g;
11 return $\mathcal{T}$;

Output:

$\mathcal{T}$: Set of optimal subgraph features

Figure 4: The gSSC algorithm

4.3 Pruning Search Space

We can now utilize the upper bound to efficiently prune the DFS Code Tree with a branch-and-bound method. During the depth-first search through the DFS Code Tree, we always maintain the temporally suboptimal gSemi value (denoted by θ) among all the gSemi values calculated before. If $\hat{q}(g) < \theta$, the gSemi value of any supergraph g' of g ($g' \supseteq g$) is no greater than θ. Thus, we can safely prune the subtree from g in the search space. If $\hat{q}(g) \geq \theta$, we cannot prune this space since there might exist a supergraph $g' \supseteq g$ that $q(g') \geq \theta$.

The algorithm gSSC is summarized in Figure 4. We initialize a set of selected subgraphs $\mathcal{T}$ as an empty set. In order to speed up the mining process, we can prune the search space from gSpan by always maintaining the currently top-t best subgraphs according to q. During the course of mining, whenever we reach a subgraph g with $\hat{q}(g) \leq \min_{g_i \in \mathcal{T}} q(g_i)$, we can prune the branches originating from g. This is because for any supergraph $g' \supseteq g$ we have $q(g') \leq \hat{q}(g)$, according to the bound defined in Eq. 8. As long as the resulting subgraph g can improve the gSemi value of any subgraphs $g_i \in \mathcal{T}$, it is accepted into $\mathcal{T}$ and the least best subgraph is dropped off from $\mathcal{T}$. And then we start searching for the next subgraph in the DFS Code Tree.

We further note that in our experiments among almost all datasets gSemi provides such a bound that we can even omit the support threshold *min_sup* and still find a set of optimal subgraphs within a reasonable time cost.

4.4 Discussion

In this section we show the connection between our framework and various application scenarios of graph classification.

Parameter Setting: There are two parameters in the objective function: α and β, which represent the weights of different constraints based on both labeled and unlabeled

graphs. Different settings of these parameters fit the optimization to different scenarios of graph classification:

- $\alpha \neq 0$, $\beta = 0$. In this case, we only consider the *cannot-link* constraints and unlabeled graph's *separability* in subgraph feature selection. No *must-link* constraint is considered, *i.e.* labeled graphs within the same classes are not necessarily close together. α controls how much we assume labeled graphs within different classes should be far from each other. This setting of parameters is useful when there is a large diversity within graphs from the same class. For example, drug molecules that have the same toxicology activities on one animal can have very different structures. Furthermore, if $\alpha = +\infty$, we only trust the cannot-link constraints. This reduce the problem into a supervised feature selection task.

- $\alpha = 0$, $\beta \neq 0$. In this setting of parameter, we only consider the *must-link* constraints and unlabeled graph's *separability* in subgraph feature selection. The larger β is, the more we trust the must-link constraints in feature selection. No *cannot-link* constraint is considered, *i.e.* labeled graphs in different classes are not necessarily far from each other.

- $\alpha = 0$, $\beta = 0$. In this case, we don't trust label constraints. Only unlabeled graph's *separability* is considered in subgraph feature selection. This reduce the problem into an unsupervised feature selection task for the unlabeled graph data.

- $\alpha \neq 0$, $\beta \neq 0$. In this case, we consider all constraints (must-link, cannot-link, unlabeled separability) with different weights. This setting is a typical setting for semi-supervised feature selection, where we need to consider both labeled and unlabeled graphs. The smaller the values of α and β, the more we trust the separability constraints from unlabeled graphs.

5. EXPERIMENTS

In this section, we conduct extensive experiments to examine the effectiveness and efficiency of gSSC in semi-supervised feature selection for graph classification.

5.1 Experimental Setup

Data Collections: In order to evaluate the performances of our semi-supervised feature selection approach for graph classification, we tested our algorithm on five real-world graph classification datasets including the following tasks: (Summarized in Table 1)

Table 1: Summary of experimental datasets. "Pos%" denotes the average percentage of postive graphs in each dataset.

Name	#Graph	Pos%	Details
MCF-7	27784	8.19	Breast Cancer
NCI-H23	40460	5.06	Lung Cancer
OVCAR-8	40626	5.08	Ovarian Cancer
PTC-MM	336	41.0	Male Mice Toxicology
PTC-FM	349	38.4	Female Mice Toxicology

1) Anti-cancer activity prediction: The first three benchmark datasets are collect from PubChem Website[1]. The task is to classify chemical compounds' anti-cancer activities on three types of cancers, *i.e.* breast, lung and ovarian. The datasets consist information on the biological activities of small molecules, containing anti-cancer activity records of more then 10,000 chemical compounds against the three types of cancers. Each chemical compound is represented as a graph. We collected 3 graph datasets with *active* and *inactive* labels from PubChem Website. The original datasets are unbalanced, where the active class is around 5%. We randomly sample 500 inactive compounds and 500 active compounds from each dataset for performance evaluation.

2) Toxicology prediction (PTC): The last two benchmark datasets are collected from PTC datasets[2] [3]. The task is to classify chemical compounds' carcinogenicity on two animal models, *i.e.* MM (Male Mouse) and FM (Female Mouse). The datasets consist carcinogenicity records of more than 300 chemical compounds. Each chemical compound is assigned with carcinogenicity labels for these animal models. On each animal model the carcinogenicity label is one of {CE, SE, P, E, EE, IS, NE, N}. We assume {CE, SE, P} as 'positive' labels, and {NE, N} as 'negative', which is the same setting as [6, 7]. Each chemical compound is represented as a graph with an average of 25.7 vertices.

Comparing Methods: In order to demonstrate the effectiveness of our semi-supervised features selection approach for graph classification, we compare our methods with two baseline methods, including a supervised feature selection approach and an unsupervised approach.

The compared methods are summarized as follows:

- Semi-Supervised **(gSSC)**: The proposed semi-supervised feature selection method for graph classification. We first use gSSC to find a set of subgraph features. The parameters in gSSC are set to $\alpha = \beta = 1$ unless otherwise specified.

- Supervised **(IG)**: We compare with a supervised feature selection method for graph classification. In this approach, a set of frequent subgraphs within labeled graphs are first mined. Then a supervised feature selection based upon Information Gain (IG), an entropy based measure, is used to select a subset of discriminative features from frequent subgraphs.

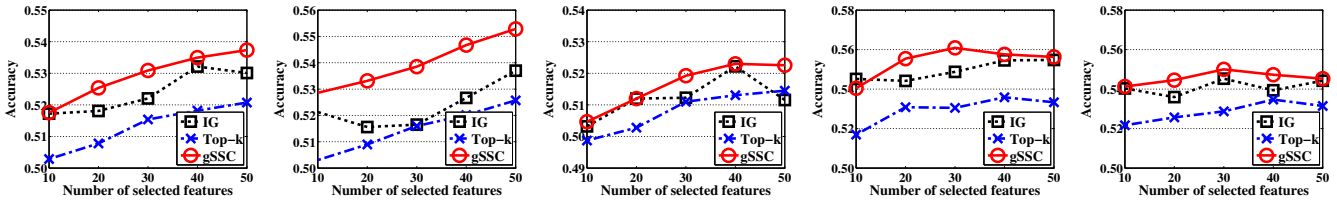
- Unsupervised **(Top-k)**: We also compare with an unsupervised feature selection method. In this approach, the evaluation criterion for subgraph feature selection is based upon frequency. The top-k frequent subgraph features in labeled graphs are selected.

All experiments are conducted on machines with 4 GB RAM and Intel Xeon^{TM}Quad-Core CPUs of 2.40 GHz.

5.2 Performances on Graph Classification

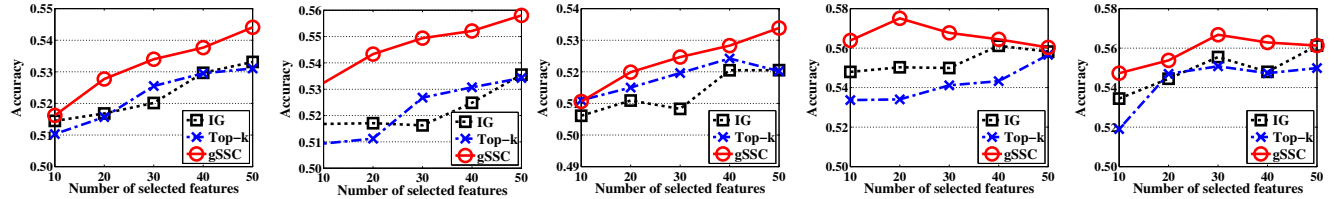
In our experiments, the labeled training graphs are randomly sampled from each datasets. All the remaining graphs

[1]http://pubchem.ncbi.nlm.nih.gov
[2]http://www.predictive-toxicology.org/ptc/



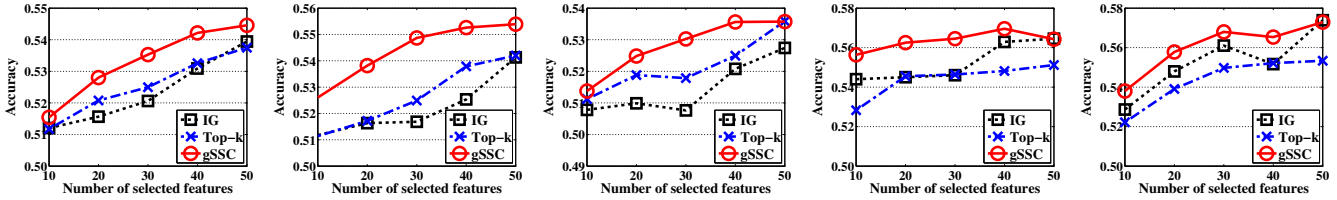
a) MCF-3 b)NCI-H23 c) OVCAR-8 d) PTC-MM e)PTC-FM

Figure 5: Classification accuracy with different number of features. (#label=30)



a) MCF-3 b)NCI-H23 c) OVCAR-8 d) PTC-MM e)PTC-FM

Figure 6: Classification accuracy with different number of features. (#label=50)



a) MCF-3 b)NCI-H23 c) OVCAR-8 d) PTC-MM e)PTC-FM

Figure 7: Classification accuracy with different number of features. (#label=70)

are used as unlabeled testing graphs. The results are average of over 30 runs of randomly sampled graph dataset. After the subgraph feature sets are selected by each method, the nearest neighbor (1-NN) classifier is used for classification.

The result of the feature selection methods with different number of labeled training graphs are displayed in Figure 5 (# labeled graphs =30), Figure 6 (# labeled graphs =50) and Figure 7 (# labeled graphs =70). We show the number of selected subgraphs t among frequent subgraphs (min_sup = 10%), together with classification accuracy as the evaluation metric.

In all these datasets, our semi-supervised feature selection algorithm (gSSC) outperform the supervised approach (IG). gSSC can achieve a good performances with a few labeled training graphs together with a large amount of unlabeled graphs. Although the performance of IG improves with a larger number of features, the IG cannot reach the best performance achievable by gSSC. These results support our first intuition that semi-supervised feature selection methods based on gSemi can boost the performance of graph classification with large amount of unlabeled graphs.

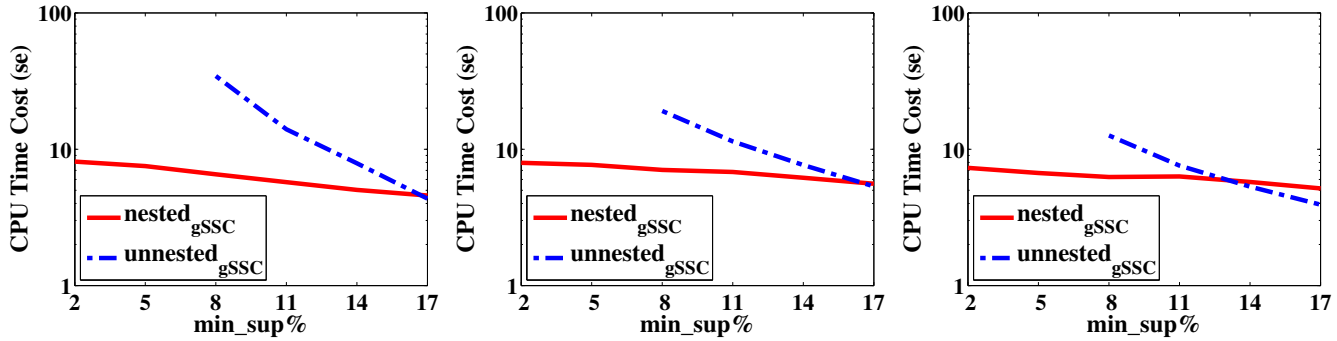
We further observe that gSSC's performances are better than our second baseline Top-k, *i.e.* unsupervised feature selection approaches without label information. These results support our second intuition that the gSemi evaluation criterion in gSSC can find better subgraph patterns for graph classification than unsupervised top-k frequent subgraph selection approaches.

5.3 Pruning Search Space

In our second experiment, we evaluated the effectiveness of the upper-bound for gSemi proposed in Section 4.2. In this section we compare the runtime performance of two versions of implementation for gSSC: 'nested gSSC' versus 'un-nested gSSC'. The 'nested gSSC' denotes the proposed method using the upper-bound proposed in Section 4.2 to prune the search space of subgraph enumerations; the 'un-nested gSSC' denotes the method without the gSemi's upper-bound pruning, which first uses gSpan to find a set of frequent subgraphs, and then selects the optimal set of subgraphs via gSemi. We run both approaches and record the average CPU time used on feature mining and selection. The result is shown in Figure 8.

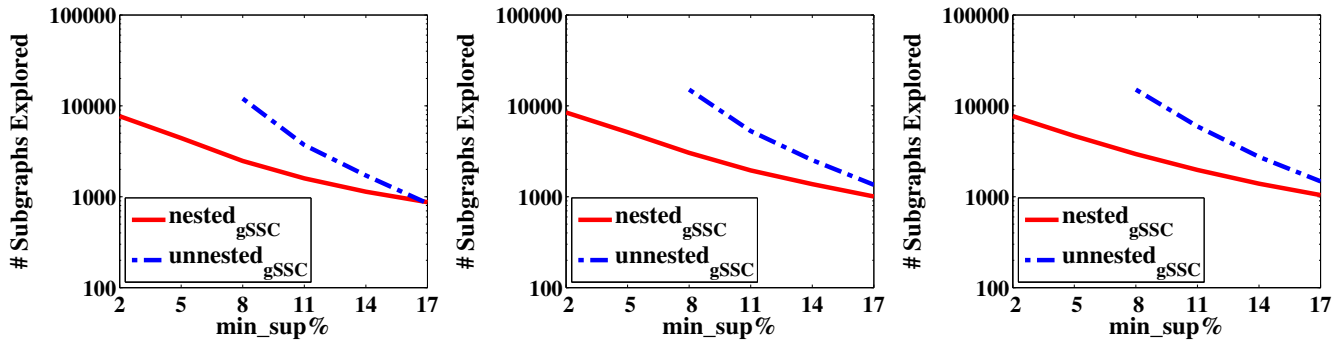
In all these datasets, the un-nested gSSC needs to explore increasingly larger subgraph search spaces as we decrease the min_sup in the frequent subgraph mining. The size increases exponentially when decreasing min_sup. In the MCF-7 dataset, when the min_sup get too low ($min\_sup <$ 8%), the subgraph feature enumeration step in un-nested gSSC can run out of the computer memory. However, the nested gSSC's running time does not increase as much, because the gSemi can help pruning the subgraph search space using both labeled and unlabeled graphs. As we can see, the min_sup can go to very low value in all datasets for the "nested gSSC".

Figure 9 shows the number of subgraph feature explored in the process of subgraph pattern enumeration. In all datasets,



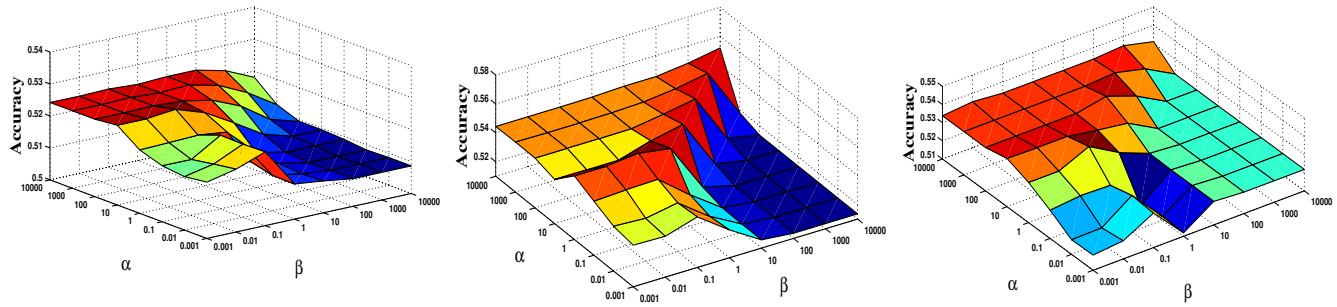
a) MCF-3 b) NCI-H23 c) OVCAR-8

Figure 8: Average CPU time for nested gSSC versus un-nested gSSC with varying min_sup.



a) MCF-3 b) NCI-H23 c) OVCAR-8

Figure 9: Average number subgraph patterns explored during mining for nested gSSC versus un-nested gSSC with varying min_sup.



a) MCF-3 b) NCI-H23 c) OVCAR-8

Figure 10: Classification accuracy of gSSC with different α and β. (#label=50)

we observe that the number of searched subgraph patterns in nested gSSC is much smaller than that of un-nested gSSC. In our experiments, we further noticed that on most datasets, nested gSSC provides such a strong bound that we may even allow nested gSSC to omit the minimum support threshold *min_sup* and still receive an optimal set of subgraph features within a reasonable time.

5.4 Parameter Settings

In our model we can take different weights on constraints from labeled graphs and unlabeled graphs. If we use different setting for the two parameters α and β, we can take the feature selection with different weights for the three types of constraints: must-link, cannot-link and unlabeled separability. α represents how much we weight the cannot-link constraints, and β denotes how much we weight the must-link constraints. The larger α is, the further away the graphs with different classes are separated from each other. The larger β is, the closer the graphs with the same classes are from each other. We test α and β with values among $\{0.001, 0.01, \cdots 10000\}$ separately. The result in Figure 10 shows that the performance of our model using α with large values and β with small values is often better than other settings. The reason is that in these real-world graph classification tasks, graphs in the same class are not always similar with each other, actually graphs can be very different within a same class.

In Figure 10, we find the best parameter setting for MCF-

3 dataset is $\alpha = 1$, $\beta = 0.1$ (accuracy = 0.526), and with our default parameter setting ($\alpha = \beta = 1$) the accuracy is 0.523. For NCI-H23 dataset, the best parameter setting is $\alpha = 1$, $\beta = 0.1$ (accuracy= 0.556), and the accuracy with default setting is 0.553. For OVCAR-8 dataset, the best parameter setting is $\alpha = 1$, $\beta = 0.1$ (accuracy= 0.539), and the accuracy with default setting is 0.530. Generally, we can see that the performance of gSSC with default setting ($\alpha = \beta = 1$) is pretty good. If we try to optimize the selection of α and β value, the accuracy improvement relative the two base line schemes will be even bigger.

6. CONCLUSION

In this paper, we study the problem of semi-supervised feature selection for graph classification. It is significantly more challenging than the conventional setting of supervised feature selection in graph data because of the lack of labeled training graphs. To address this challenge, we propose a feature evaluation criterion, named gSemi, to evaluate subgraph features with both labeled and unlabeled graphs, and derive an upper-bound for gSemi to prune the subgraph search space. Then we propose a branch-and-bound algorithm to efficiently find a set of optimal subgraph feature which is useful for graph classification. Empirical studies on real-world tasks show that our semi-supervised feature selection approach for graph classification outperforms supervised and unsupervised approaches and is very efficient by pruning the subgraph search space using both labeled and unlabeled graphs.

7. ACKNOWLEDGMENTS

This work is supported in part by NSF through grant IIS-0905215.

8. REFERENCES

[1] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning a mahalanobis metric from equivalence constraints. *Journal of Machine Learning Research*, 6:937–965, 2005.

[2] C. Borgelt and M. Berthold. Mining molecular fragments: Finding relevant substructures of molecules. In *Proceedings of the 2nd International Conference on Data Mining*, pages 211–218, Maebashi City, Japan, 2002.

[3] C. Helma, R. King, S. Kramer, and A. Srinivasan. The predictive toxicology challenge 2000-2001. *Bioinformatics*, 17(1):107–108, 2001.

[4] J. Huan, W. Wang, and J. Prins. Efficient mining of frequent subgraph in the presence of isomorphism. In *Proceedings of the 3rd International Conference on Data Mining*, pages 549–552, Melbourne, FL, 2003.

[5] A. Inokuchi, T. Washio, and H. Motoda. An apriori-based algorithm for mining frequent substructures from graph data. In *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery*, pages 13–23, Lyon, France, 2000.

[6] H. Kashima, K. Tsuda, and A. Inokuchi. Marginalized kernels between labeled graphs. In *Proceedings of the 20th International Conference on Machine Learning*, pages 321–328, Washington, DC, 2003.

[7] T. Kudo, E. Maeda, and Y. Matsumoto. An application of boosting to graph classification. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 729–736. Cambridge, MA: MIT Press, 2005.

[8] M. Kuramochi and G. Karypis. Frequent subgraph discovery. In *Proceedings of the 1st International Conference on Data Mining*, pages 313–320, San Jose, CA, 2001.

[9] K. V. Mardia, J. T. Kent, and J. M. Bibby. *Multivariate Analysis*. Academic Press, San Diego, CA, 1980.

[10] S. Nijssen and J. Kok. A quickstart in frequent structure mining can make a difference. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 647–652, Seattle, WA, 2004.

[11] J. Ren, Z. Qiu, W. Fan, H. Cheng, and P. S. Yu. Forword semi-supervised feature selection. In *Proceedings of the 12th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 970–976, Osaka, Japan, 2008.

[12] W. Tang and S. Zhong. Pairwise constraints-guided dimensionality reduction. In *SIAM International Conference on Data Mining Workshop on Feature Selection for Data Mining*, Bethesda, MD, 2006.

[13] M. Thoma, H. Cheng, A. Gretton, J. Han, H. Kriegel, A. Smola, L. Song, P. Yu, X. Yan, and K. Borgwardt. Near-optimal supervised feature selection among frequent subgraphs. In *Proceedings of the SIAM International Conference on Data Mining*, pages 1075–1086, Sparks, Nevada, 2009.

[14] X. Yan, H. Cheng, J. Han, and P. Yu. Mining significant graph patterns by leap search. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 433–444, Vancouver, BC, 2008.

[15] X. Yan and J. Han. gSpan: Graph-based substructure pattern mining. In *Proceedings of the 2nd International Conference on Data Mining*, pages 721–724, Maebashi City, Japan, 2002.

[16] Z. Zhao and H. Liu. Semi-supervised feature selection via spectral analysis. In *Proceedings of the SIAM International Conference on Data Mining*, pages 641–646, Minneapolis, MN, 2007.