

# Semi-supervised Feature Selection via Spectral Analysis

Zheng Zhao \*

Huan Liu \*

## Abstract

Feature selection is an important task in effective data mining. A new challenge to feature selection is the so-called “small labeled-sample problem” in which labeled data is small and unlabeled data is large. The paucity of labeled instances provides insufficient information about the structure of the target concept, and can cause supervised feature selection algorithms to fail. Unsupervised feature selection algorithms can work without labeled data. However, these algorithms ignore label information, which may lead to downgraded performance. In this work, we propose to use both (small) labeled and (large) unlabeled data in feature selection, which is a topic has not yet been addressed in feature selection research. We present a semi-supervised feature selection algorithm based on spectral analysis. The algorithm exploits both labeled and unlabeled data through a regularization framework, which provides an effective way to address the “small labeled-sample” problem. Experimental results demonstrated the efficacy of our approach and confirmed that using labeled and unlabeled data together does help feature selection with small labeled samples.

**Keyword:** Feature Selection, Semi-supervised Learning, Machine Learning, Spectral Analysis

## 1 Introduction

The high dimensionality of data poses a challenge to learning tasks. In the presence of many irrelevant features, learning algorithms tend to overfitting [14]. Various studies show that features can be removed without performance deterioration [8]. Feature selection is one effective means to identify relevant features for dimension reduction [14, 22]. The training data used in feature selection can be either labeled or unlabeled, corresponding to supervised and unsupervised feature selection. In supervised feature selection [8, 14], feature relevance can be evaluated by their correlation with the class label. And in unsupervised feature selection [11, 12], without

label information, feature relevance can be evaluated by their capability of keeping certain properties of the data, such as the variance or the separability. Data are abundant and continue to accumulate in an unprecedented rate, but labeled data are costly to obtain. It is common to have a data set with huge dimensionality but small labeled-sample size. The data sets of this kind present a serious challenge, the so-called “small labeled-sample problem” [1], to supervised feature selection, that is, when the labeled sample size is too small to carry sufficient information about the target concept, supervised feature selection algorithms fail with either unintentionally removing many relevant features or selecting irrelevant features, which seems to be significant only on the small labeled data. Unsupervised feature selection algorithms can be an alternative in this case, as they are able to use the large amount of unlabeled data. However, as these algorithms ignore label information, important hints from labeled data are left out and this will generally downgrades the performance of unsupervised feature selection algorithms.

Under the assumption that labeled and unlabeled data are sampled from the same population generated by target concept, using both labeled and unlabeled data is expected to better estimate feature relevance. The task of learning from mixed labeled and unlabeled data is of semi-supervised learning [6]. In this paper, we present a *semi-supervised feature selection* algorithm based on the spectral graph theory [7]. The algorithm ranks features through a regularization framework, in which a feature’s relevance is evaluated by its fitness with both labeled and unlabeled data. The remainder of the paper is organized as follows. We first define notations, provide definitions and background knowledge that are used in the paper. In Section 3, we propose the algorithm, *sSelect*, and investigate its properties. In Section 4, we present an empirical study in comparison with representative algorithms and conduct a sensitivity study to evaluate various components of the algorithm. Experimental results demonstrates the efficacy of our approach and shows using both labeled and unlabeled data does help feature selection with small labeled samples. We review related work in Section 5 and conclude in Section 6 with a discussion of the proposed algorithm and some future work.

---

\*Technical Report, TR-06-022, Computer Science and Engineering (CSE) Department, Arizona State University (ASU), Tempe, AZ, 85281. {zheng.zhao, huan.liu}@asu.edu

## 2 Notations and Definitions

In semi-supervised learning, a data set of  $n$  data points  $X = (\mathbf{x}_i)_{i \in [n]}$  consists of two subsets depending on the label availability:  $X_L = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_l)$  for which labels  $Y_L = (y_1, y_2, \dots, y_l)$  are provided, and  $X_U = (\mathbf{x}_{l+1}, \mathbf{x}_{l+2}, \dots, \mathbf{x}_{l+u})$  whose labels are not given. Here data point  $\mathbf{x}_i$  is a vector with  $m$  dimensions (features), and label  $y_i$  is an integer from  $\{+1, -1\}^1$ , and  $l + u = n$  ( $n$  is the total number of instances). When  $l = 0$ , data  $X$  is for unsupervised learning; when  $u = 0$ ,  $X$  is for supervised learning. Let  $F_1, F_2, \dots, F_m$  denote the  $m$  features of  $X$  and  $\mathbf{f}_1, \mathbf{f}_2, \dots, \mathbf{f}_m$  be the corresponding feature vectors that record the feature value on each instance. The above is depicted in Figure 1.

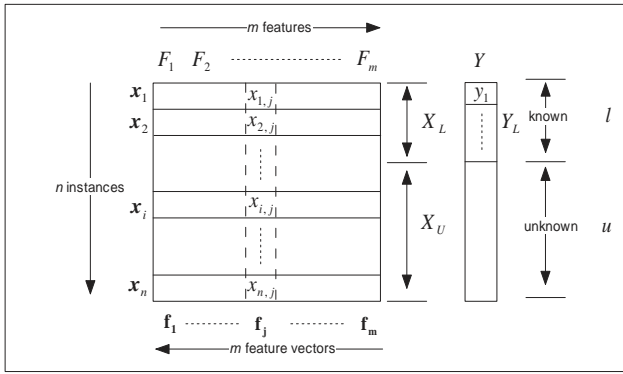


Figure 1: The data used in semi-supervised learning

We give the definition of semi-supervised feature selection as:

**DEFINITION 1. (SEMI-SUPERVISED FEATURE SELECTION)** Given data  $X_L$  and  $X_U \subseteq R^m$ , semi-supervised feature selection is to use both  $X_L$  and  $X_U$  to identify the set of most relevant features  $\{F_{j_1}, F_{j_2}, \dots, F_{j_k}\}$  of the target concept, where  $k \leq m$  and  $j_r \in \{1, 2, \dots, m\}$  for  $r \in \{1, 2, \dots, k\}$ .

In this work, we employ the spectral graph theory [7] to semi-supervised feature selection. In the following, we provide some definitions and basic concepts from the spectral graph theory used in the paper. Given a data set  $X$ , let  $G(V, E)$  be the undirected graph constructed from  $X$ , with  $V$  is its node set and  $E$  is its edge set. The  $i$ -th node  $v_i$  of  $G$  corresponds to  $\mathbf{x}_i \in X$  and there is an edge between each nodes pair  $(v_i, v_j)$ , whose weight  $w_{ij} = w(v_i, v_j)$  is determined by  $\psi(\mathbf{x}_i, \mathbf{x}_j)$ , where  $\psi(\cdot)$  is a similarity func-

<sup>1</sup>This is corresponding to data with binary classes, which is the case we will study in this paper. If the data has multiple classes, we have  $y_i \in \{1, 2, \dots, c\}$ , where  $c$  is the number of classes.

tion defined as:  $\psi(\cdot) : R^n \times R^n \rightarrow R^+$ . A popular similarity function is the *RBF* kernel function:  $\psi(\mathbf{x}_1, \mathbf{x}_2) = \exp(-\|\mathbf{x}_1 - \mathbf{x}_2\|^2 / (2\delta^2))$ . The volume of a node set  $S \subseteq V$  is defined as  $\text{vol}S = \sum_{v_i \in S, v_j \in V} \sum_{(v_i, v_j) \in E} w_{ij}$ . Let  $(S, S^c)$  be a partition of  $V$ , the cut induced by  $(S, S^c)$  is defined as  $\text{cut}(S, S^c) = \sum_{v_i \in S, v_j \in S^c} w_{ij}$ . Instead of connecting each nodes pair with an edge,  $v_i$  and  $v_j$  are connected, if and only if  $v_i$  or  $v_j$  is one of the  $k$  nearest neighbors of the other. This will form a  $k$ -neighborhood graph,  $G$ . In the paper use normal typefaces, such as  $\gamma, \mu$  and  $\lambda$ , to denote scalar variables and use bold typefaces, such as  $\mathbf{x}, \mathbf{g}$  and  $\mathbf{e}$ , to denote vector variables. We use  $I$  to denote the identity matrix,  $\mathbf{e}$  to denote the column vector with all its elements to be 1,  $\mathbf{e} = \{1, 1, \dots, 1\}^T$  and  $\langle \mathbf{x} \cdot \mathbf{y} \rangle$  to denote the inner product of two vectors  $\mathbf{x}$  and  $\mathbf{y}$ ,  $\langle \mathbf{x} \cdot \mathbf{y} \rangle = \mathbf{x}^T \cdot \mathbf{y}$ . Below we give the definitions of adjacency matrix, degree matrix and Laplacian matrix, which are frequently used in spectral graph theory.

**DEFINITION 2. (ADJACENCY MATRIX  $W$ )** Let  $G$  be the graph construct from  $X$ , the adjacency matrix of  $G$  is defined as:

$$(2.1) \quad W_{ij} = \begin{cases} w_{ij} & \text{if } (v_i, v_j) \in E \\ 0 & \text{otherwise} \end{cases}$$

**DEFINITION 3. (DEGREE MATRIX  $D$ )** Let  $\mathbf{d}$  denote the vector:  $\mathbf{d} = \{d_1, d_2, \dots, d_n\}$ , where  $d_i = \sum_{k=1}^n w_{ik}$ , the degree matrix  $D$  of the graph  $G$  is defined by:  $D_{ij} = d_i$  if  $i = j$ , and 0 otherwise.

According to the definition, more data points close to  $\mathbf{x}_i$ , means a larger  $d_i$ . Therefore  $d_i$  can be interpreted as an estimation of the density around the node  $v_i$  in graph  $G$ , which is corresponding to  $\mathbf{x}_i$ .

**DEFINITION 4. (LAPLACIAN MATRIX  $L$ )** Given the adjacency matrix  $W$  and the degree matrix  $D$  of  $G$ , the Laplacian Matrix of graph  $G$  is defined as:

$$(2.2) \quad L = D - W$$

The degree matrix  $D$  and the Laplacian matrix  $L$  satisfy the following properties [7]:

**THEOREM 2.1.** Given  $W, L$  and  $D$  of  $G$ , we have:

1. Let  $\mathbf{e} = \{1, 1, \dots, 1\}^T$ ,  $L * \mathbf{e} = 0$ .
2.  $L$  is symmetric and semi-positive definite.
3.  $\forall \mathbf{x} \in R^n$ ,  $\mathbf{x}^T L \mathbf{x} = \sum_{\{v_i, v_j\} \in E} w_{ij} (x_i - x_j)^2$
4.  $D \cdot \mathbf{e} = \mathbf{d}$  and  $\mathbf{e}^T \cdot D \cdot \mathbf{e} = \text{vol}V$ .

Applying the spectral graph theory to unsupervised learning results in spectral clustering algorithms [24, 10], which have been proved to be effective in many applications [3]. Spectral clustering algorithms, such as ratio cut [5] and normalized cut [27], transform the original clustering problem to the cut problems on graph models. And the (local) optimal cluster indicator can be reconstructed from the eigenvectors of the corresponding matrix defined in the cut problem [9]. Instead of reconstructing the cluster indicators from eigenvectors, we show a way to construct them from feature vectors. By doing so, the fitness of cluster indicators can be evaluated by both labeled and unlabeled data, paving the way to evaluate feature relevance using both labeled and unlabeled data.

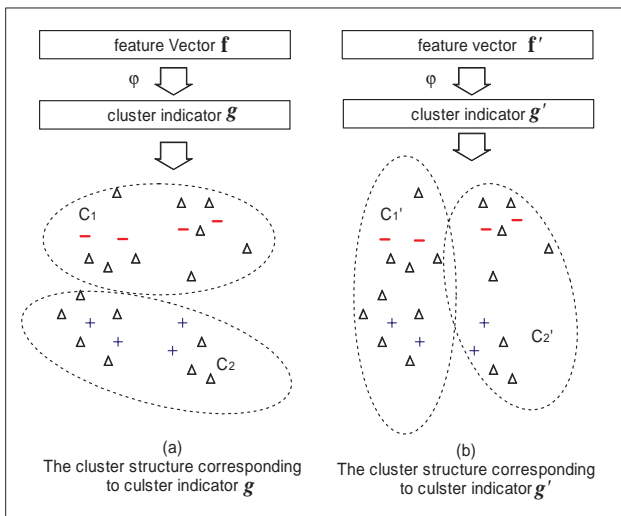


Figure 2: Basic idea for comparing the fitness of cluster indicators according to both labeled and unlabeled data for semi-supervised feature selection. “-” corresponds to negative class, “+” to positive class, and “ $\Delta$ ” to unlabeled instances.

### 3 Semi-supervised Feature Selection

Supervised and unsupervised feature selection methods require to measure feature relevance, but in different ways. Therefore the key for designing an effective semi-supervised feature selection algorithm is to develop a framework, under which the relevance of a feature can be evaluated by both labeled and unlabeled data in a natural way. The clustering assumption is a base assumption for most semi-supervised learning algorithms. It assumes that “if points are in the same cluster, they are likely to be of the same class” [6]. In this spirit, we propose a semi-supervised feature selection algorithm, *sSelect*. The basic idea is illustrated in Figure 2. We

first transform a feature vector  $\mathbf{f}_i$  into a cluster indicator, so each element  $f_{ij}$ , ( $j = 1, 2, \dots, n$ ) of  $\mathbf{f}_i$  indicates the affiliation of the corresponding instance  $\mathbf{x}_j$ . The fitness of the cluster indicator can be evaluated by two factors: (1) separability - whether the cluster structures formed are well separable; and (2) consistency - whether the cluster structures formed is consistent with the given label information. The ideal case is all labeled data in each cluster coming from the same class.

Suppose we have two feature vectors  $\mathbf{f}$  and  $\mathbf{f}'$ , and the corresponding cluster indicators are  $\mathbf{g}$  and  $\mathbf{g}'$  (we will elaborate how they are formed in the next section). The cluster structures formed by  $\mathbf{g}$  and  $\mathbf{g}'$  are shown in Figure 2. Comparing with the cluster structures formed by  $\mathbf{g}'$ , those formed by  $\mathbf{g}$  are preferred. From the unlabeled data point of view, both cluster indicators form clearly separable cluster structures. However, when the label information is considered, the cluster structure formed by  $\mathbf{g}$  turn out to be more consistent, because all labeled data in a cluster are of the same class. Under the clustering assumption,  $\mathbf{g}$  fits the data better than  $\mathbf{g}'$ , suggesting the feature corresponding to the feature vector  $\mathbf{f}$  is more relevant with target concept than the feature corresponding to feature vector  $\mathbf{f}'$ . In the next, we show how to construct cluster indicators from feature vectors semi-supervised feature selection.

**3.1 Clustering Indicator Construction** The normalized min-cut clustering algorithm was first proposed by Shi and Malik in [27], and has been shown to be superior to other cluster algorithms, such as ratio cut [17]. Our method resorts to transforming feature vectors to the cluster indicators of normalized min-cut. Given a graph  $G = (V, E)$  constructed from data  $X$ , the normalized min-cut clustering algorithm finds a cut  $(S, S^c)$  for  $G$ , that minimizes the cost function:

$$(3.3) \quad Ncut(S, S^c) = \frac{cut(S, S^c)}{\text{vol}S} + \frac{cut(S^c, S)}{\text{vol}S^c}$$

**3.1.1 Cluster indicator space** Let  $\mathbf{g} = \{g_1, g_2, \dots, g_n\}$  be the clustering indicator and  $\gamma = \text{vol}S/\text{vol}V$ , the minimization of (3.3) can be rewritten as [17]:

$$(3.4) \quad \min \frac{\sum_{(v_i, v_j) \in E} (g_i - g_j)^2 \times w_{ij}}{2 \sum_{v_i \in V} g_i^2 \times d_i} = \frac{\mathbf{g}^T L \mathbf{g}}{\mathbf{g}^T D \mathbf{g}}$$

$$s.t. \quad g_i \in \{2(1 - \gamma), -2\gamma\} \quad \text{and} \quad \langle \mathbf{g}, \mathbf{d} \rangle = 0$$

The combinatorial optimization problem specified in (3.4) is intractable. In [27] the problem is relaxed to allow  $g_i$ ,  $i \in \{1, \dots, n\}$  to have any real value, instead of only one of the two discrete values,  $2(1 -$

$\gamma$ ) and  $-2\gamma$ . The relaxed problem can be solved efficiently by calculating the harmonic eigenfunction of the normalized Laplacian matrix  $\mathcal{L} = D^{-1/2}LD^{-1/2}$  [7]. It can be proved that the harmonic eigenfunction of  $\mathcal{L}$  is orthogonal to  $\mathbf{d}$  [27]. Since elements in  $\mathbf{d}$  estimate the density around the nodes in  $G$ , the orthogonality between the cluster indicator and  $\mathbf{d}$  implies that the data density of clusters should be balance. For the relaxed problem, we give the definition for the cluster indicator space of normalized min-cut.

DEFINITION 5. (CLUSTER INDICATOR SPACE) Given a graph  $G$ , the cluster indicator space  $\mathcal{S}$  of normalized min-cut clustering on  $G$  is defined as:

$$(3.5) \quad \mathcal{S} = \{\mathbf{g} \mid \mathbf{g} \in R^n, \langle \mathbf{g} \cdot \mathbf{d} \rangle = 0\}$$

A vector is a member of the cluster indicator space  $\mathcal{S}$ , if and only if it is orthogonal to  $\mathbf{d}$ .

**3.1.2 Transformation for features vectors** Given a cluster indicator, the fitness of the indicator can be evaluated by both labeled and unlabeled data. If a feature vector  $\mathbf{f}$  is orthogonal to  $\mathbf{d}$ , it is a cluster indicator and its fitness can be evaluated by using the way we mentioned above. However, not every feature vector of  $X$  is naturally orthogonal to  $\mathbf{d}$ . Therefore, we introduce an  $F$ - $C$  transformation  $\varphi$ , which transforms an  $n$  dimensional feature vector  $\mathbf{f} \in R^n$  to a vector in cluster space  $\mathcal{S}$ .

DEFINITION 6. ( $F$ - $C$  TRANSFORMATION) Let  $\mathbf{f} \in R^n$  and  $\mathbf{e} = \{1, \dots, 1\}^T$ , the  $F$ - $C$  transformation  $\varphi$  is defined as:

$$(3.6) \quad \varphi(\mathbf{f}) = \mathbf{f} - \frac{\sum_i f_i d_i}{\text{volV}} \cdot \mathbf{e};$$

The  $F$ - $C$  transformation  $\varphi$  defines a linear transformation and has the following properties: first, it transforms  $\forall \mathbf{f} \in R^n$  into a vector in space  $\mathcal{S}$ ; second, working in  $R^n$  via  $\varphi$ , we can achieve the same optimal cut value for Equation (3.4), as the one we achieved in  $\mathcal{S}$ ; and third, among all linear transformations in the form of  $\ell(\mathbf{f}) = \mathbf{f} + \mu \cdot \mathbf{e}$ , where  $\mu \in R$ , the cluster indicator generated from  $\varphi$  upper bounds the value of Equation (3.4), and provides a reliable estimation of the fitness of  $\mathbf{f}$  with data  $X$ . We prove the first two properties through Theorem 3.1, and show the third one through Theorem 3.2.

THEOREM 3.1.  $\varphi$  satisfies following properties:

$$i. \forall \mathbf{f} \in R^n, \langle \varphi(\mathbf{f}) \cdot \mathbf{d} \rangle = 0$$

$$ii. \forall \mathbf{f}_1, \mathbf{f}_2 \in R^n, \mathbf{f}_1^T \cdot L \cdot \mathbf{f}_2 = (\varphi(\mathbf{f}_1))^T \cdot L \cdot \varphi(\mathbf{f}_2)$$

$$iii. \forall \mathbf{g} \in \mathcal{S}, \varphi(\mathbf{g}) = \mathbf{g}$$

$$iv. Ncut^*(\varphi(R^n)) = Ncut^*(\mathcal{S})$$

Here,  $Ncut^*$  denotes the optimal cut value.

PROOF:

$$\langle \varphi(\mathbf{f}) \cdot \mathbf{d} \rangle = \left( \mathbf{f} - \frac{\sum_i f_i \times d_i}{\text{volV}} \cdot \mathbf{e} \right)^T \cdot \mathbf{d}$$

$$\begin{aligned} 1. \quad &= \mathbf{f}^T \cdot \mathbf{d} - \frac{\mathbf{f}^T \cdot \mathbf{d}}{\text{volV}} \cdot \mathbf{e}^T \cdot \mathbf{d} = \mathbf{f}^T \cdot \mathbf{d} - \left( \frac{\mathbf{f}^T \cdot \mathbf{d}}{\mathbf{e}^T \cdot \mathbf{d}} \right) \cdot \mathbf{e}^T \cdot \mathbf{d} \\ &= \mathbf{f}^T \cdot \mathbf{d} - \mathbf{f}^T \cdot \mathbf{d} = 0 \end{aligned}$$

2. Follows from the definition of  $F$ - $C$  transformation and property 3 of Theorem 2.1.

$$\begin{aligned} 3. \quad \varphi(\mathbf{g}) &= \mathbf{g} - \frac{\sum_i g_i \times d_i}{\text{volV}} \cdot \mathbf{e} = \mathbf{g} - \frac{\langle \mathbf{g} \cdot \mathbf{d} \rangle}{\text{volG}} \\ \therefore \langle \mathbf{g} \cdot \mathbf{d} \rangle &= 0 \quad \therefore \varphi(\mathbf{g}) = \mathbf{g} \end{aligned}$$

4. According to property (i), we have  $\varphi(R^n) \subseteq \mathcal{S}$ , which means  $Ncut^*(\varphi(R^n)) \geq Ncut^*(\mathcal{S})$ . By property (iii), we also have  $\varphi(R^n) \supseteq \mathcal{S}$ , which means  $Ncut^*(\varphi(R^n)) \leq Ncut^*(\mathcal{S})$ . Therefore the equality holds.  $\square$

THEOREM 3.2. Among all linear transformations in the form:  $\ell(\mathbf{f}) = \mathbf{f} + \mu \cdot \mathbf{e}$ , where  $\mu \in R$  and  $\mathbf{e} = (1, \dots, 1)^T$ ,  $Ncut(\varphi(\mathbf{f}))$  upper bounds the value of Equation (3.4).

PROOF: According to property 3 of Theorem 2.1, to prove  $\mu^*$  is the optimal value is equivalent to prove  $\mu = \mu^*$  is the value that minimizes:

$$(3.7) \quad (\mathbf{f} + \mu \cdot \mathbf{e})^T \cdot D \cdot (\mathbf{f} + \mu \cdot \mathbf{e})$$

we can solve the  $\mu^*$  directly by setting:

$$\frac{\partial (\mathbf{f} + \mu \cdot \mathbf{e})^T \cdot D \cdot (\mathbf{f} + \mu \cdot \mathbf{e})}{\partial \mu} = 0$$

Solving this equation, we obtain  $\mu = -\frac{\mathbf{f}^T \cdot \mathbf{d}}{\text{volV}}$ . The positive second derivative assures the minimum. Note that  $\mathbf{e}^T \cdot \mathbf{d} = \text{volV}$ , and  $d_i$  estimates the density around the  $i$ th instance, therefore  $\mu$  can be interpreted as the density weighted mean of feature vector  $\mathbf{f}$ .  $\square$

**3.2 Algorithm *sSelect*** The  $F$ - $C$  transformation  $\varphi$  transforms a feature vector  $\mathbf{f}$  into a cluster indicator  $\mathbf{g}$ , which forms a basis for us to evaluate the feature on both labeled and unlabeled data. Given a cluster indicator  $\mathbf{g}$ , labeled data  $X_L$  and unlabeled data  $X_U$ , the fitness should be evaluated by: (1) whether the clusters formed by the indicator are well separable (renders a small cut value), and (2) whether it is consistent with the label information as shown in Figure 2. In this spirit we design a regularization framework, which enables us to evaluate the fitness of the cluster indicator using both labeled and unlabeled data. Let  $\mathbf{g}$  be the cluster indicator generated from a feature vector  $\mathbf{f}$  and  $\hat{\mathbf{g}} = \text{sign}(\mathbf{g})$ ,<sup>2</sup> the regularization framework is defined as:

$$(3.8) \quad \lambda \frac{\sum_{v_i \sim v_j} (g_i - g_j)^2 \times w_{ij}}{2 \sum_{v_i \in V} g_i^2 \times d_i} + (1 - \lambda)(1 - \text{NMI}(\hat{\mathbf{g}}, \mathbf{y}))$$

In Equation (3.8),  $\text{NMI}(\hat{\mathbf{g}}, \mathbf{y})$  is the normalized mutual information between  $\hat{\mathbf{g}}$  and  $\mathbf{y}$ . The first term of Equation (3.8) calculates the cut value of using  $\mathbf{g}$  as the cluster indicator for data  $X$ . The second term estimates the corresponding classification loss of  $\hat{\mathbf{g}}$  according to the labeled data. In this framework, the evaluation with either labeled or unlabeled data is based on the cluster indicator  $\mathbf{g}$ , which serves as a common base and makes the integration of the two terms of Equation (3.8) reasonable.

The normalized mutual information used in Equation (3.8) measures the consistency between the discretized cluster indicator and the label data, irrespective to how the cluster indicator is mapped to classes (i.e.  $(1, -1) \rightarrow (+, -)$  or  $(1, -1) \rightarrow (-, +)$ ). Here we give the definition of normalized mutual information. Suppose  $\hat{y}_i \in \hat{C} = (\hat{C}_1, \dots, \hat{C}_c)$  and  $y_i \in C = (C_1, \dots, C_c)$ . Let  $p(\hat{C}_i)$  and  $p(C_j)$  denote the probability that an instance randomly selected from  $X$  belongs cluster  $\hat{C}_i$  and class  $C_j$ , respectively. Let  $p(\hat{y}_i, y_j)$  denote the joint probability, and  $H(\hat{\mathbf{y}})$ ,  $H(\mathbf{y})$  denote the entropy of  $\hat{\mathbf{y}}$  and  $\mathbf{y}$  respectively,  $\text{NMI}$  is defined as:

$$(3.9) \quad \text{NMI}(\hat{\mathbf{y}}, \mathbf{y}) = \frac{\sum_{\hat{C}_i, C_j} p(\hat{C}_i, C_j) \cdot \log \frac{p(\hat{C}_i, C_j)}{p(\hat{C}_i) \cdot p(C_j)}}{\max(H(\hat{\mathbf{y}}), H(\mathbf{y}))}$$

Given the framework specified in Equation (3.8), we propose a semi-supervised feature **Selection** algorithm, *sSelect*, below:

<sup>2</sup>For the 2nd term (labeled part) in Equation (3.8), we need discretized cluster indicator. To transform a continuous cluster indicator to a discretized one, we use 0 as the cut point and binarize the continuous cluster indicator, which is one option suggested in [27]. More sophisticated methods can be found in [9, 24]

**Algorithm 1:** The spectral graph based semi-supervised feature selection algorithm (*sSelect*)

**Input:**  $X, Y_L, \lambda, k$   
**Output:**  $SF_{sSelect}$ , the ranked feature list

- 1 construct  $k$ -neighborhood graph  $G$  from  $X$ ;
- 2 build  $W, \mathbf{d}$  and  $L$  from  $G$ ;
- 3 **for** each feature vector  $\mathbf{f}_i$  **do**
- 4     construct  $\mathbf{g}_i$  from  $\mathbf{f}_i$  using  $\varphi$ ;
- 5     calculate  $s_i$ , the score of  $F_i$  using Eq. (3.8);
- 6 **end**
- 7  $SF_{sSelect} \leftarrow$  ranking  $F_i$  in descending order;
- 8 **return**  $SF_{sSelect}$ ;

Algorithm 1 has three parts. (1) In line 1-2, graph matrices are built using the training data. (2) In line 3-6, features are transformed and evaluated based on the graph. (3) In line 7-8, features are ranked in descending order in terms of relevance (the smaller the  $s_i$ , the more relevance the feature), so that the most relevant ones are ranked as top features in the returned feature list. Features selection can be done based on the returned feature list according the desired number of features. The time complexity of the proposed algorithm, *sSelect*, can be obtained as follow. First, we need  $O(mn^2)$  operations to build  $W, \mathbf{d}$  and  $L$ . Next, we need  $O(n^2)$  operations to calculate  $s_i$  for each feature: transforming  $\mathbf{f}_i$  to  $\mathbf{g}_i$  requires  $O(n)$  operations; calculating the cut value needs  $O(n^2)$  operations; and using the confusion table to calculate NMI takes  $O(c^2n)$  operations (for binary class data  $c^2 = 4$ ). Therefore, we need  $O(mn^2)$  operations to calculate scores for  $m$  features. Last, we need  $O(m \log m)$  operations to rank the features. Hence, the overall time complexity of *sSelect* is  $O(mn^2)$ . In the next, we evaluate the performance of *sSelect*.

## 4 Empirical Study

We now empirically evaluate the performance of *sSelect*. We compare the proposed algorithm with two representative feature selection algorithms: Laplacian Score [16] is a recent spectral graph-based unsupervised feature selection algorithm and Fisher Score [4] is a popular supervised feature selection algorithm which is employed in [16] for comparison. We implement *sSelect* algorithm in the Matlab environment. All experiments were conducted on a PENTIUM IV 2.4G PC with 1.5GB RAM. In the experiment, Euclidian distance and the RBF kernel function are used for building a neighborhood graph with the neighborhood size of 10. We will also discuss how to choose the regularization parameter  $\lambda$  in this section.

Table 1: Summary of the three data sets

Data Set	PCMAC	HOCKEYBASE	MACBASE
instances	1943(982:961)	1993(994:999)	1955(961:994)
features	8298	8298	8298

**4.1 Data sets** We test the three feature selection algorithms on three real data sets generated from the 20-new-group data. The three data sets are: (1) PC *vs.* MAC (PCMAC), (2) BASEBALL *vs.* HOCKEY (HOCKEYBASE) and (3) MAC *vs.* BASEBALL (MACBASE). The four topics addressed in the three data sets are also used in [30, 32] and are widely used for performance evaluation for learning algorithms. The three data sets are generated from the version 20-news-18828. The articles in four topics were processed by the TMG software package [29] with the following options: (1) passing all words through the Porter stemmer before counting them; (2) tossing out any token which is on the stoplist; and (3) ignoring words that occur in 4 or fewer documents. No further preprocessing was done. We summarize the three data sets in Table 1. The numbers in the brackets of the second row show the sizes of the instances from each class. As the preprocessing for the articles from all four topics are done together, the three data sets share the same dimensionality. In the experiment, the  $\lambda$  value is set to 0.1. More discussion for the regularization parameter,  $\lambda$ , will be given in Section 4.4.1.

**4.2 Evaluation framework** A common hypothesis used for evaluating the quality of a feature subset is: if a feature subset is more relevant with the target concept than others, a classifier learning with the feature subset should achieve better accuracy. In the normal evaluation framework, feature selection is carried out on the training data, and a classifier is trained and evaluated on the training and testing data, respectively, using selected features. To simulate the small labeled sample context, we set  $l$ , number of labeled data, to be 2, 6 and 10 respectively. So few labeled instances, however, induce insufficiency for sensibly obtaining a classifier, whose estimated accuracy is used to evaluate the quality of a feature subset. Hence, we use 5-fold cross validation (CV) on the whole data  $X$  (recall that all instances in  $X$  have class labels). to estimate the accuracy for evaluating the quality of a feature subset. The details of the evaluation framework is shown in Algorithm 2. We define a projection operator  $\Pi_{SF}(X)$  which retains the selected features in  $SF$  and removes unselected features. The process specified in Algorithm 2 is repeated for 20 times. The obtained accuracy is averaged and used for evaluating the quality of the feature subset selected ac-

ording to each algorithm. In the framework, line 2-3, construct  $X_L$  and  $X_U$ , the labeled and unlabeled data, from the whole data  $X$ . Line 5-9, use  $X_L$  for Fisher Score,  $X_U$  for Laplacian Score and  $X_L + X_U$  for  $sSelect$  to form ranked feature lists  $SF_{sSelect}$ ,  $SF_{LP}$  and  $SF_F$ , respectively<sup>3</sup>. Line 12-15, use  $X$ , the whole data, as the evaluating data and filter  $X$  by the top features from each ranked feature list and generate data sets corresponding to each feature selection algorithm. Line 16, classification algorithm is applied on each obtained data set with cross validation. Accuracy on each data set is reported as an evaluation of the quality of the feature subset selected by each feature selection algorithm.

**Algorithm 2:** Feature Evaluation Framework

```

1 for each data set do
2   Generate labeled data  $X_L$  by randomly sampling
    $\frac{1}{2} \cdot l$  instances from each class;
3    $X_U = X - X_L$ ;
4   /*using each algorithm to rank features*/;
5   begin
6      $SF_{sSelect} \leftarrow sSelect$  with  $X_L + X_U$ ;
7      $SF_{LP} \leftarrow Laplacian\ Score$  with  $X_U$ ;
8      $SF_F \leftarrow Fisher\ Score$  with  $X_L$ ;
9   end
10  /*evaluating the quality of feature sets*/;
11  for  $i = 5$  to 50 step 5 do
12    Select top  $i$  features from  $SF_{sSelect}$ ,  $SF_{LP}$ ,
     $SF_F$  and  $SF_{IG}$ ;
13     $X_{sSelect} \leftarrow \Pi_{SF_{sSelect}}(X)$ ;
14     $X_{LP} \leftarrow \Pi_{SF_{LP}}(X)$ ;
15     $X_F \leftarrow \Pi_{SF_F}(X)$ ;
16    Run 5-fold CV on  $X_{sSelect}$ ,  $X_{LP}$  and  $X_F$ 
    using 1NN and record accuracy;
17  end
18 end

```

**4.3 Comparison of feature quality** Using the framework defined in Algorithm 2, we test the three algorithms on the three benchmark data sets. Figure 3 shows the plots for accuracy vs. different numbers of selected features and different numbers of labeled data. As shown in the figure,  $sSelect$  works consistently better than the other two feature selection algorithms. Generally,  $sSelect$  works best and is followed by Fisher Score and Laplacian Score. From the figure, we can see, generally, the more features we select, the better accuracy we can achieve. A closer study reveals, generally, the accuracy of  $sSelect$  increases fast in the beginning (the

<sup>3</sup>Features are ranked in descending order in terms of relevance according to each feature selection algorithm, so that the most relevant features are the top ones in feature lists.

number of selected feature is small) and slows down at the end (the number of selected feature is already large). This suggests that *sSelect* ranks features properly as important features are selected first. From the figure, we can also observe that Laplacian Score does not perform well, as it does not utilize the label information provided with the training data.

$L$	Fisher Score	Laplacian Score
PCMAC		
2	+0.0610	+0.1254
6	+0.0431	+0.1447
10	+0.0873	+0.1753
HOCKEYBASE		
2	+0.0560	+0.1389
6	+0.0836	+0.1775
10	+0.1150	+0.1997
MACBASE		
2	+0.0550	+0.1875
6	+0.0859	+0.2338
10	+0.1377	+0.2682
AVERAGE		
	+0.0805	+0.1834

Table 2: A comparison of the average accuracy for the three feature selection algorithms with *sSelect*.  $L$  is for number of labeled data.

For each data set and different numbers of labeled data, we average the accuracy for different number of selected features. The differences of the averaged accuracy between *sSelect* and the other two feature selection algorithms on the benchmark data sets are list in Table 2. We can see that in terms of average accuracy gains, *sSelect* is 0.0805 better than Fisher Score and 0.1834 better than Laplacian Score. One trend can be clearly observed is that comparing with Laplacian Score, the accuracy differences become bigger when more labeled data is provided for training *sSelect*. This observation suggests that the label information is important for feature selection. This is also consistent with our understanding for the role of the label information in semi-supervised learning. The experiment results on the benchmark data sets confirm that using both labeled and unlabeled data does help feature selection. In the next, we will study the effects of the regularization parameter  $\lambda$ , number of labeled data, number of selected features and their interactions.

**4.4 Sensitivity study** The three factors, regularization parameter, number of labeled data and number of selected features, can have influence on the performance of *sSelect*. We study their effect and interactions in this section. We will also provide a way for choosing proper  $\lambda$  for *sSelect*.

#### Algorithm 3: Evaluating $\lambda$

**Input:**  $X, Y_L, \lambda_{list}, k, l$   
**Output:**  $\mathbf{c}$ , the cut value achieved

- 1 Construct  $k$ -neighborhood graph  $G$  from  $X$ ;
- 2 Build  $W, \mathbf{d}$  and  $L$  from  $G$ ;
- 3 **for** each feature  $F_r$  **do**
- 4 | Construct  $\mathbf{g}_r$  from  $F_r$  using  $\varphi$ ;
- 5 **end**
- 6 **for** each  $\lambda_i$  in  $\lambda_{list}$  **do**
- 7 | Calculate  $\mathbf{s}$ , the score vector for features using Equation (3.8) with  $\lambda_i$ ;
- 8 | Select  $\mathbf{F}_l$ , the top  $l$  features according to  $\mathbf{s}$ ;
- 9 |  $D = \Pi_{\mathbf{F}_l}(X)$ ;
- 10 | Calculate  $\mathbf{c}(i)$ , the cut value achieved on  $D$ ;
- 11 **end**
- 12 **return**  $\mathbf{c}$ ;

**4.4.1 Effect of regularization parameter** Figure 4 (a-2, b-2, c-2) shows the effect of  $\lambda$  on feature quality. The regularization parameter has a significant influence on the performance of *sSelect*. With  $\lambda$  set to 0.9, there is a big performance decrease comparing with  $\lambda = 0.1$ . We study how to select a proper  $\lambda$ . In *sSelect*, we select features that are able to provide good separability to both labeled data and unlabeled data. However, this does not assure that combining them will find a good cluster indicator providing well separable cluster structures. This motivate us to use the separability of the reduced data generated from a feature subset in evaluating the quality of the feature subset. Base on this idea, we propose Algorithm 3 for evaluating the properness of the regularization parameter  $\lambda$ .

In Algorithm 3,  $\Pi_{\mathbf{F}}$  is the projection operator. Given a data  $D$ , the cut value (line 10, Algorithm 3) can be calculated by: (1) constructing the adjacent matrix  $W$  from  $D$ , (2) forming the normalized Laplacian matrix  $\mathcal{L}$  using  $W$ , and (3) obtaining the cut value by calculating the second smallest eigenvalue of  $\mathcal{L}$ . Figure 4 shows the effect of  $\lambda$  on both cut value and the accuracy with the top 10 features are selected. From the figure we can see the cut value and the accuracy coincide very well with different  $\lambda$  value. By observing the cut values, we infer that a good  $\lambda$  value is in the region of  $[0.1, 0.3]$ . This conjecture is verified by the results shown in the second row of Figure 4 (the lower the cut value, the higher the accuracy). It also indicates that the evaluating approach proposed in Algorithm 3 is a reasonable one.

**4.4.2 Effect of number of labeled data** Figure 5 shows the relationship between number of labeled data

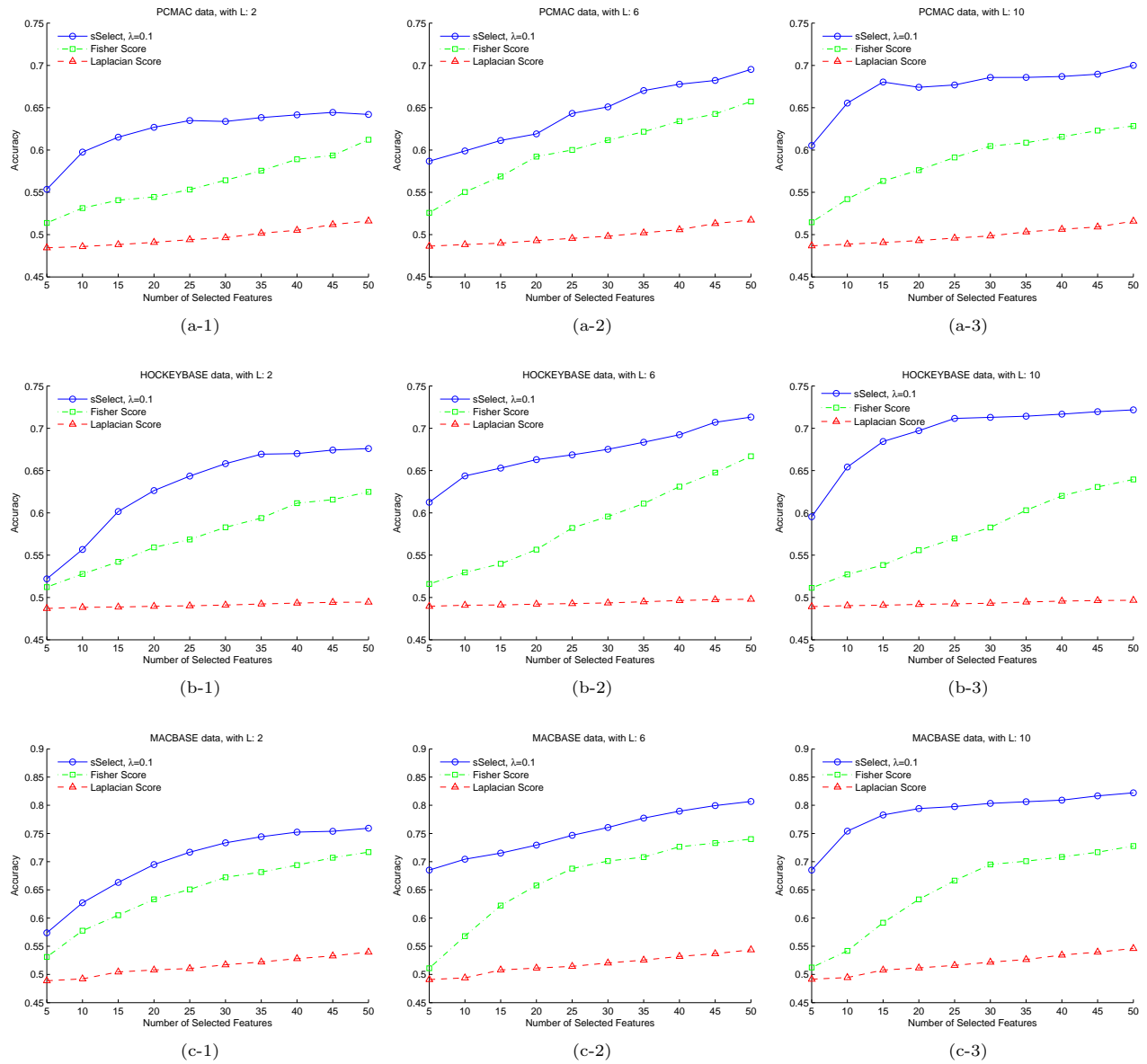


Figure 3: Accuracy vs. different numbers of selected features and different numbers of labeled data ( $l = 2, 6, 10$ ). The first, second and third row are for *PCMAC*, *HOCKEYBASE* and *MACBASE*, respectively. In each figure the x-axis corresponds to different numbers of selected features and the y-axis corresponds to accuracy.

with accuracy. We can observe a general trend: the more labeled data we have, the better accuracy we can achieve, meaning that with more labeled data, we are able to select features with higher quality. Figure 6 gives us a picture of the relationship among number of labeled data, number of selected feature and accuracy.

The general trend shows that the more features and labeled data used, the better accuracy we can achieve. In the next, we study the interaction among the number of labeled data, the number of selected features and the regularization parameter  $\lambda$ .

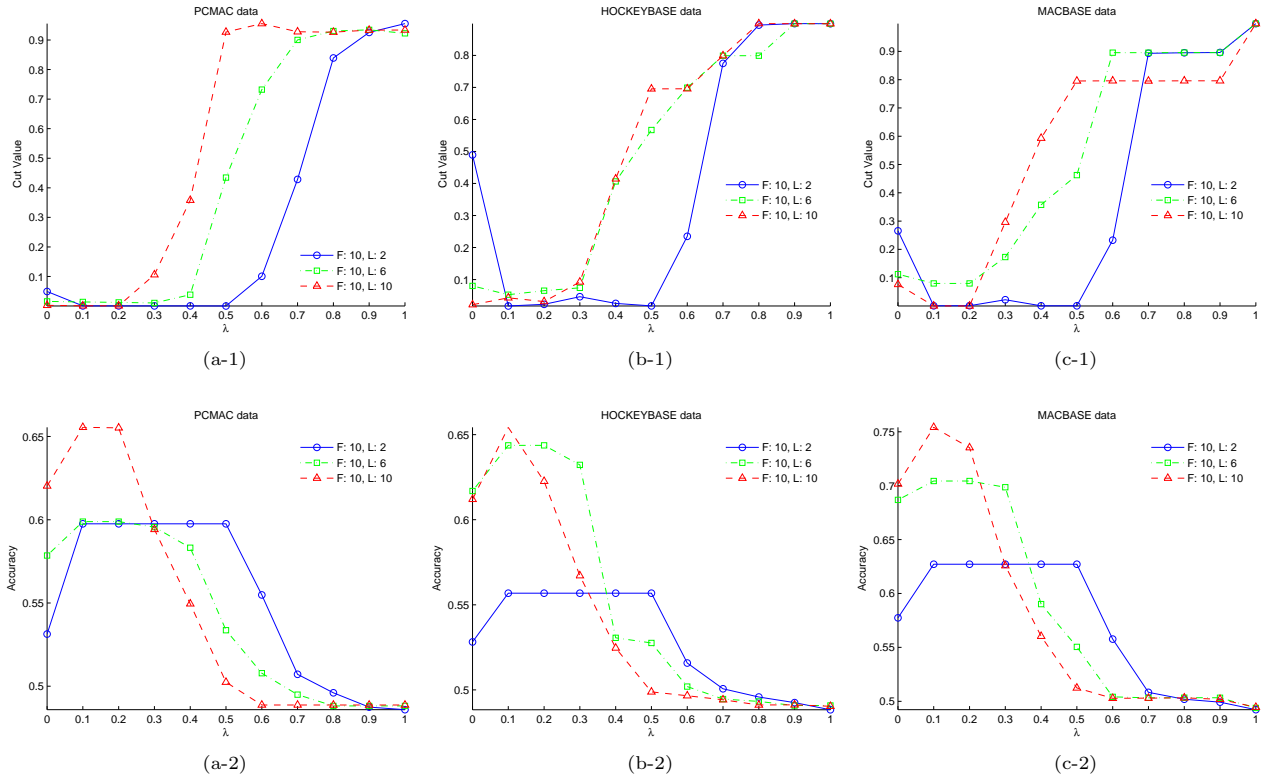


Figure 4: The effect of the regularization parameter  $\lambda$  on feature quality (number of selected features is 10). (a-1), (b-1) and (c-1) are for cut value vs.  $\lambda$ . (a-2), (b-2) and (c-2) are for accuracy vs.  $\lambda$ . The x-axis is for different  $\lambda$  value. The y-axis of (a-1), (b-1) and (c-1) is for cut value (the smaller the better). The y-axis of (a-2), (b-2) and (c-2) is for accuracy (the bigger the better).

**4.4.3 Interaction among factors** One phenomenon observed in experiments is: When the number of selected feature becomes big, e.g., 200, the optimal  $\lambda$  value begins to shift from a small value (such as 0.1) to a larger value. However, the more labeled data we have, the less evident this phenomenon. As shown in Figure 7-(a), when 200 features are selected, the optimal  $\lambda$  values for  $l=2, 6, 10$  are 0.8, 0.4 and 0.2, respectively. A similar trend can be observed in Figure 7-(b). A closer study reveals that when labeled data is little, the feature score generated from the 2nd term (labeled part) of Equation (3.8) tends to be discrete. As shown in Figure 7-(c), when there are only 2 labeled instances, the distribution of feature score calculated from the 2nd term of Equation (3.8) is: 84 features with a score of 0 and 8214 features with a score of  $1^4$ . So if many

<sup>4</sup>When  $X_L$  contains 10 labeled instances, the score distribution is: 4 features with a score of 0.6042, 36 features with 0.7635, 3 features with 0.8755, 427 features with 0.8920, 11 feature with 0.9651, and 7817 features with 1.00000.

features are selected, the score from  $X_L$  (or the 2nd term of Equation (3.8)) becomes less sensible, especially for the features near the end of the top list of selected features. In this case, the 1st term (unlabeled part) of Equation (3.8) should play a more important role. The curve in Figure 7 (d) is smoother than that in Figure 7 (c) as the large amount of  $X_U$  is used.

## 5 Related Work

**5.1 Feature selection** According to the training data  $X$  is whether labeled or not the feature selection algorithms can be either supervised or unsupervised [22, 12]. Supervised feature selection methods [8] can be broadly categorized into *wrapper* models [19, 18] and *filter* models [15, 26]. The wrapper model uses the predictive accuracy of a predetermined learning algorithm to determine the quality of selected features. These methods are prohibitively expensive to run for data with a large number of features. The filter model separates feature selection from classifier learning so

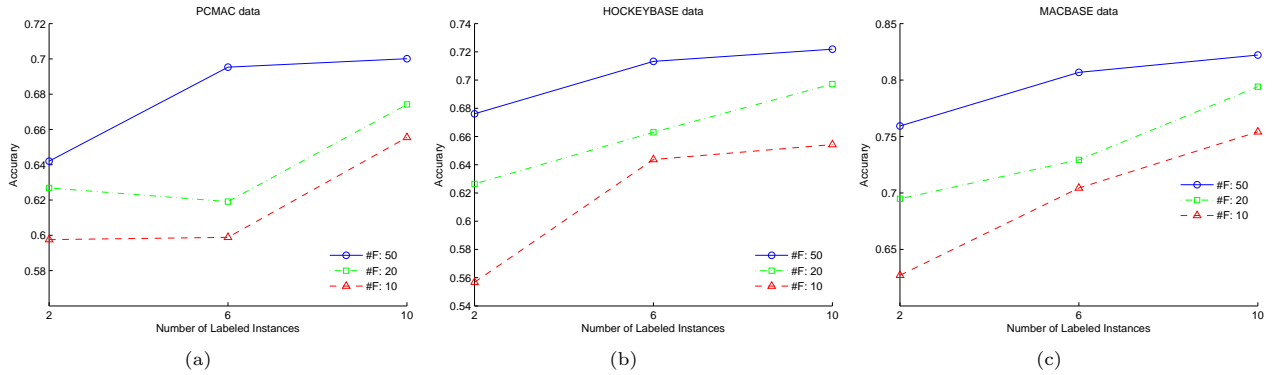


Figure 5: Effect of the number of labeled data on feature quality. The x-axis is for different number of labeled data, the y-axis is for accuracy.

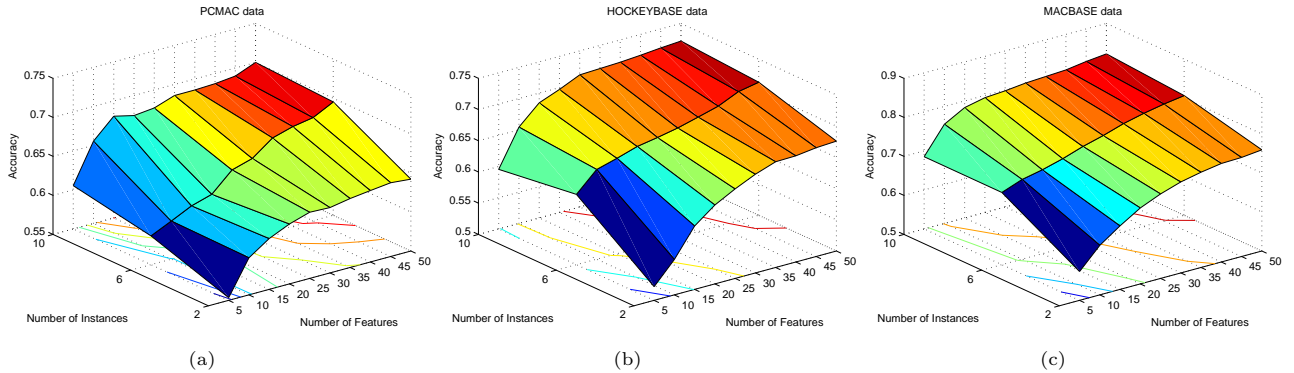


Figure 6: Effect of the number of selected features and the number of instances. The z-axis is for accuracy.

that the bias of a learning algorithm does not interact with the bias of a feature selection algorithm. It relies on measures of the general characteristics of the training data such as distance, consistency, dependency, information, and correlation [22]. Recently, an increasing number of researchers paid attention to developing unsupervised feature selection [16, 20]. It is a more loosely constrained search problem without class labels, depending on clustering quality measures [11], and can eventuate many equally valid feature subsets. The key issue of unsupervised feature selection is how to objectively measure the results of feature selection. A wrapper model involves a clustering algorithm is usually used for evaluating the quality of feature selection.

**5.2 Semi-supervised learning (SSL)** It learns from mixed labeled and unlabeled data [6]. One can interpret SSL as supervised learning with additional information from unlabeled data, or interpret it as unsupervised learning guided by constraints formed from labeled data. Generally the algorithms for semi-supervised learning fall into three categories: Generative Models [25, 2], Low Density Separation [21, 13] and Graph-Based Methods [31, 30]. The paucity of labeled data in semi-supervised learning requires that inductive bias [23] has to be introduced to make the learning possible. Inductive bias is some prior assumption(s). In semi-supervised learning, common assumptions include smoothness (or continuity) assumption, clustering (or low density) assumption, and manifold assumption [6].

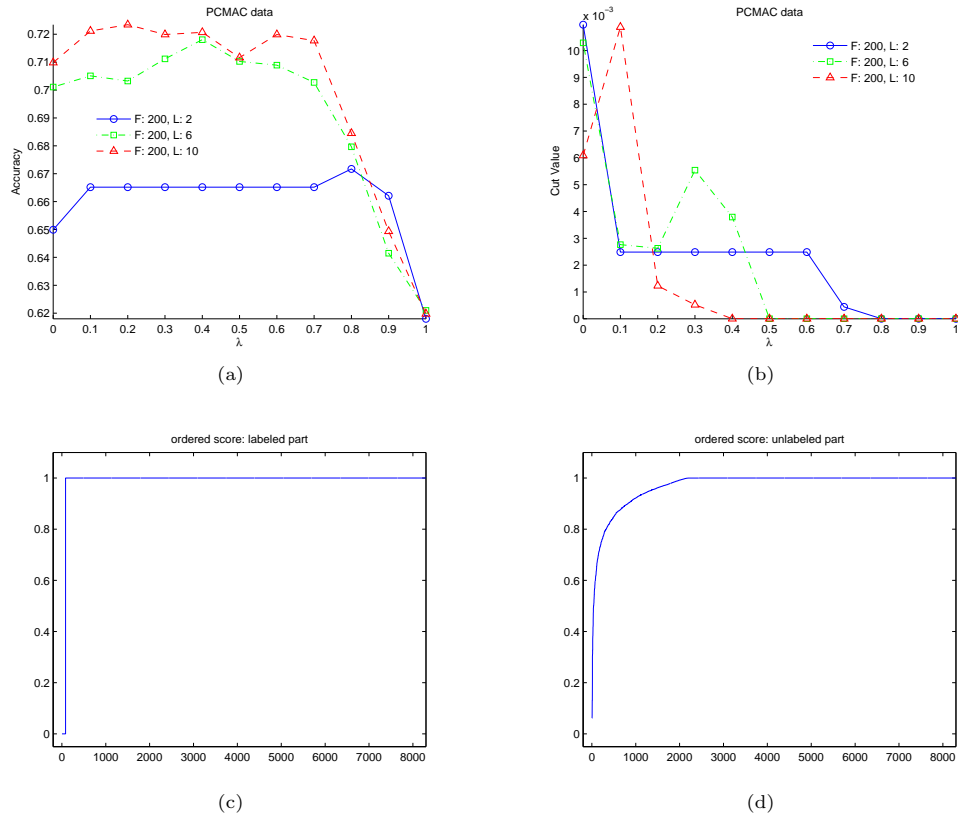


Figure 7: The interaction among factors. Plots (a) and (b) are for  $\lambda$  vs. accuracy, and  $\lambda$  vs. cut value when 200 features are selected. The  $x$ -axes of plot (a) and (b) is for the  $\lambda$  value and  $y$ -axes are for accuracy and cut value respectively. Plot (c) and (d) are for the order score obtained from the labeled part and the unlabeled part of Equation (3.8), when 2 labeled data is provided.  $y$ -axes of plot (c) and (d) is for score value.

## 6 Conclusion

This work presents an extensive initial attempt to semi-supervised feature selection. We propose an algorithm based on the special graph theory. We show that one can construct cluster indicators for normalized min-cut clustering from feature vectors which allows to evaluate fitness on both labeled and unlabeled data in determining feature relevance. Experimental results confirm that using labeled and unlabeled data together does help feature selection. Extending *sSelect* to multi-class data is one line of our future work. With the preliminary success to semi-supervised feature selection, we strengthen our belief that one key issue of semi-supervised feature selection is how to reasonably convert features to cluster indicators of the corresponding clustering algorithm. We continue exploring the possibility of leveraging other clustering mechanisms in this endeavor. Another direc-

tion for semi-supervised feature selection is to iteratively propagate labels from labeled data to unlabeled data while carrying out feature selection. This requires feature selection and label propagation to be considered in an EM framework. Our preliminary experiment (to be reported elsewhere) shows that the performance of this method is unstable and heavily depends on the starting point (initial labeled data). Further work is needed to deepen our understanding and make it a feasible approach.

## References

- [1] A.Jain and D.Zongker. Feature selection: Evaluation, application, and small sample performance. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(2):153–158, 1997.

- [2] S. Basu. *Semi-supervised Clustering: Probabilistic Models, Algorithms and Experiments*. PhD thesis, Department of Computer Sciences, University of Texas at Austin, 2005.
- [3] Yonatan Bilu. *On spectral properties of graphs, and their application to clustering*. PhD thesis, Hebrew University, 2004.
- [4] C.M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.
- [5] P. Chan, D. Schlag, and J. Zien. Spectral k-way ratio cut partitioning and clustering. In *the 30th ACM/IEEE Design Automation Conference*, pages 749–754, 1993.
- [6] O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning*. MIT Press, Cambridge, 2006.
- [7] Fan R. K. Chung. *Spectral graph theory*. AMS, 1997.
- [8] M. Dash and H. Liu. Feature selection for classification. *Intelligent Data Analysis: An International Journal*, 1(3):131–156, 1997.
- [9] I. S. Dhillon, Y. Guan, and B. Kulis. A unified view of kernel k-means, spectral clustering and graph partitioning. Technical report, Department of Computer Sciences, University of Texas at Austin, 2005.
- [10] C. Ding, X. He, and H. D. Simon. On the equivalence of nonnegative matrix factorization and spectral clustering. In *Proc. SIAM Int'l Conf. Data Mining (SDM'05)*, pages 606–610, 2005.
- [11] J. G. Dy, C. E. Brodley, A. C. Kak, L. S. Broderick, and A. M. Aisen. Unsupervised feature selection applied to content-based retrieval of lung images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:373–378, 2003.
- [12] J.G. Dy and C. E. Brodley. Feature selection for unsupervised learning. *J. Mach. Learn. Res.*, 5:845–889, 2004.
- [13] Y. Grandvalet and Y. Bengio. Semi-supervised learning by entropy minimization. In *The 19th Annual Conf. on Neural Information Processing Systems (NIPS)*, 2004.
- [14] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.
- [15] M. A. Hall. Correlation-based feature selection for discrete and numeric class machine learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 359–366, 2000.
- [16] X. He, D. Cai, and P. Niyogi. Laplacian score for feature selection. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*. MIT Press, Cambridge, MA, 2005.
- [17] J. Huang. A combinatorial view of graph laplacians. Technical report, Max Planck Institute for Biological Cybernetics, 2005.
- [18] Y. Kim, W. Street, and F. Menczer. Feature selection for unsupervised learning via evolutionary search. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 365–369, 2000.
- [19] R. Kohavi and G. H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2):273–324, 1997.
- [20] M. H. C. Law, M. A. T. Figueiredo, and A. K. Jain. Simultaneous feature selection and clustering using mixture models. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 26:1154–1166, 2004.
- [21] N. D. Lawrence and M. I. Jordan. Semi-supervised learning via Gaussian processes. In *In 19th Annual Conf. on Neural Information Processing Systems (NIPS)*, 2004.
- [22] H. Liu and L. Yu. Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on Knowledge and Data Engineering*, 17:491–502, 2005.
- [23] T. M. Mitchell. *Machine Learning*. McGraw Hill, 1997.
- [24] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *The 14th Advances in Neural Information Processing Systems (NIPS)*, 2001.
- [25] K. Nigam, A.K. McCallum, S. Thrun, and T.M. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39:103–134, 2000.
- [26] M. Robnik-Sikonja and I. Kononenko. Theoretical and empirical analysis of Relief and ReliefF. *Machine Learning*, 53:23–69, 2003.
- [27] J. B. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.
- [28] J. Ye, Q. Li, H. Xiong, H. Park, R. Janardan, and V. Kumar. Idr/qr: An incremental dimension reduction algorithm via qr decomposition. In *in Proc. of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 364 – 373, 2004.
- [29] D. Zaimpekis and E. Gallopoulos. Tmg: A matlab toolbox for generating term-document matrices from text collections. Technical report, Computer Engineering & Informatics Department, University of Patras, Greece, 2005.
- [30] D. Zhou, O. Bousquet, T. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *NIPS '04: In 19th Annual Conf. on Neural Information Processing Systems*, 2004.
- [31] X. J. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *The Twentieth International Conference on Machine Learning (ICML)*, 2003.
- [32] X. J. Zhu, J. Lafferty, and Z. Ghahramani. Combining active learning and semi-supervised learning using gaussian fields and harmonic functions. In *ICML 2003 workshop on The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining*, 2003.