Joint Semi-Supervised Feature Selection and Classification through Bayesian Approach

Bingbing Jiang,[1] Xingyu Wu,[1] Kui Yu,[2] Huanhuan Chen[1]*
[1]School of Computer Science and Technology, University of Science and Technology of China, Hefei, China.
[2]School of Computer and Information, Hefei University of Technology, Hefei, China.
{jiangbb, xingyuwu}@mail.ustc.edu.cn, yukui@hfut.edu.cn, hchen@ustc.edu.cn.

Abstract

With the increasing data dimensionality, feature selection has become a fundamental task to deal with high-dimensional data. Semi-supervised feature selection focuses on the problem of how to learn a relevant feature subset in the case of abundant unlabeled data with few labeled data. In recent years, many semi-supervised feature selection algorithms have been proposed. However, these algorithms are implemented by separating the processes of feature selection and classifier training, such that they cannot simultaneously select features and learn a classifier with the selected features. Moreover, they ignore the difference of reliability inside unlabeled samples and directly use them in the training stage, which might cause performance degradation. In this paper, we propose a joint semi-supervised feature selection and classification algorithm (JSFS) which adopts a Bayesian approach to automatically select the relevant features and simultaneously learn a classifier. Instead of using all unlabeled samples indiscriminately, JSFS associates each unlabeled sample with a self-adjusting weight to distinguish the difference between them, which can effectively eliminate the irrelevant unlabeled samples via introducing a left-truncated Gaussian prior. Experiments on various datasets demonstrate the effectiveness and superiority of JSFS.

Introduction

In many real-world applications, such as scene classification, text categorization, gene expression data analysis, the data dimensionality becomes larger and larger. Directly using these high-dimensional data might lead to lower time efficiency and performance deterioration due to the existence of noisy or irrelevant features. As a dimensionality reduction technique, feature selection tries to find a lower-dimensional representation of data via removing irrelevant features without altering the original feature space (Chandrashekar and Sahin 2014). Due to its effective representation and better interpretability for data, various feature selection algorithms have been proposed (He, Cai, and Niyogi 2006; Zhao and Liu 2007; Nie et al. 2010; Jiang et al. 2016).

According to the availability of class label, the existing feature selection algorithms can be divided into three different groups: unsupervised, supervised and semi-supervised (Sheikhpour et al. 2017). Without the guidance of class label information, unsupervised algorithms cannot select the features that effectively distinguish different classes. On the other hand, supervised algorithms can select discriminative features with sufficient labeled data. However, the labeled data is usually scarce since it is expensive and time-consuming to collect them. Therefore, it is of great practical significance to design the feature selection algorithm that can exploit both labeled and unlabeled data simultaneously. Inspired by the semi-supervised learning (Chapelle, Schölkopf, and Zien 2006; Sakai et al. 2017; Chen, Jiang, and Yao 2018), semi-supervised feature selection has been proposed, which can select features with the help of the label information of labeled data and the structure information of unlabeled data.

To learn the relevant features in the case of abundant unlabeled data with few labeled data, many state-of-the-art semi-supervised feature selection algorithms have been proposed. Most semi-supervised feature selection algorithms are filter-based, which are implemented by ranking the features based on their capability of maintaining the specific data structure or some information criterion. For example, Zhao *et al* proposed a locality sensitive discriminant algorithm (Zhao, Lu, and He 2008), which selects the features that can maximize the margin between different classes and simultaneously preserve the similarity between data. Based on the trace ratio criterion in supervised learning (Nie et al. 2008), Liu *et al* designed a semi-supervised feature selection algorithm that can avoid selecting the features with very small variance (Liu et al. 2013). However, the filter-based semi-supervised feature selection algorithms ignore the interaction between the selection of feature and the learning tasks. Therefore, it is difficult for them to select the features that are particularly effective for a given learning algorithm (Xu et al. 2010).

More recently, the sparsity constraint combining with the manifold regularization framework (Belkin, Niyogi, and Sindhwani 2006) has been applied to design the embedded algorithms. The embedded semi-supervised feature selection algorithms formulate an objective function to exploit the discriminative information and the local geometric structure of training data to learn a feature projection matrix. For example, Ma *et al.* incorporated l_{21}-norm constraint into manifold regularization and proposed a structural feature selec-

*Corresponding author: Huanhuan Chen.

tion algorithm. Luo *et al.* used l_{2q}-norm ($0 < q \leq 1$) constraint to replace the l_{21}-norm and designed an insensitive sparse regression algorithm (Luo et al. 2018). Unfortunately, this algorithm has at least four free parameters that need to be tuned, which makes it difficult to be practical. Generally, although the embedded algorithms integrate the feature selection process to the training, they are not able to adaptively select features and simultaneously learn a classifier with the selected features. Concretely, they need to predetermine the number of selected features and then select features according to the feature projection matrix. To validate the effectiveness of selected features, an additional classifier (usually SVM or KNN) is required to train a classifier by using the selected features. Sometimes, they could still perform well by using a classifier with tuned parameters even though including some irreverent features. Therefore, it is hard to judge whether their promising performance is attributed to the robust classifier or the selected features.

Moreover, the filter-based or embedded semi-supervised feature selection algorithms mentioned above indiscriminately and directly use the labeled and unlabeled samples to design the evaluation criterion or objective functions for selecting features. In fact, the labeled samples are usually more reliable while there might inevitably exist noisy unlabeled samples since abundant unlabeled samples could be collected from different resources. Therefore, using all unlabeled samples indiscriminately could have an adverse impact on the feature selection and even lead to performance degradation. Although the semi-supervised feature selection has been used to some applications, it fails to take the robustness to the noisy unlabeled samples into consideration.

To alleviate this issue, Chang *et al.* proposed a convex semi-supervised feature selection algorithm (Chang et al. 2014), which preassigns different scores for labeled and unlabeled samples. However, this algorithm lacks the ability to capture the local geometric structure of unlabeled data. In fact, the local geometric structure refers to the local neighborhood relationship, which is very effective for feature selection especially when very few labeled samples with the high-dimension feature are provided (Liu et al. 2014). Moreover, it determines the scores of labeled and unlabeled samples empirically, making it hard to be useful. Thus, it is of vital importance for semi-supervised feature selection to make full use of unlabeled samples and simultaneously take an effective strategy to distinguish their difference of reliability.

Motivated by the aforementioned issues, we propose a novel semi-supervised feature selection algorithm, called joint semi-supervised feature selection and classification through Bayesian approach (JSFS). The main contributions of this paper are summarized as follows:

- Combining with a Bayesian approach, JSFS is able to adaptively identify the relevant features and simultaneously learn a classifier with the selected features. Thus, JSFS does not require the number of selected features to be predetermined, nor an additional classifier to be adopted for training, which are two main limitations of the existing semi-supervised feature selection algorithms.
- To make better use of prior knowledge from labeled and unlabeled samples, we define a prior on feature weight to generate adaptive sparsity in the feature space and then propose an alternative way to optimize the feature weight.
- Instead of using all unlabeled samples indiscriminately and directly, we associate each unlabeled sample with a self-adjusting weight first and then introduce a left-truncated Gaussian prior on it. By this prior, JSFS can eliminate the irrelevant unlabeled samples, improving the robustness against noise.
- JSFS is compared with the state-of-the-art algorithms. The experimental results on different real-world datasets validate the effectiveness and superiority of JSFS.

The proposed algorithm

In this section, we will propose the joint semi-supervised feature selection and classification algorithm in detail.

Notation and Model Specification

In semi-supervised learning, training data $\boldsymbol{X}$ consists of a labeled dataset with l samples, $\boldsymbol{X}_l = \{\boldsymbol{x}_i\}_{i=1}^{l}$ associated with class labels $\boldsymbol{y} = [y_1, \ldots, y_l]^T$, and an unlabeled dataset with u samples, $\boldsymbol{X}_u = \{\boldsymbol{x}_i\}_{i=l+1}^{n}$, where $\boldsymbol{x}_i \in \mathbb{R}^d$ and $n = l + u$ is the total number of training samples. This paper studies the binary semi-supervised feature selection problem, in which the class label $y_i \in \{0, 1\}$. Given the training data, the main aim is to learn a prediction model based on the selected features, which can make an accurate prediction for out-of-sample data.

The prediction model adopted by previous feature selection algorithms is a linear combination of all features, treating the labeled and unlabeled samples equally in the training stage. This model is improper due to ignoring the quality difference between labeled and unlabeled samples and the difference between unlabeled samples themselves. To overcome this limitation, we propose a separable linear model from training data as:

$$f(\boldsymbol{X}, \boldsymbol{w}, \boldsymbol{\Lambda}) = \boldsymbol{\Lambda}\boldsymbol{X}\boldsymbol{w}, \tag{1}$$

where $\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n]^T \in \mathbb{R}^{n \times d}$ denotes training samples, $\boldsymbol{w} = [w_1, \cdots, w_d]^T$ is the d-dimensional weight vector for the feature, and $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ denotes an $n \times n$ non-negative diagonal matrix, which consists of the weights for training samples. In Eq. (1), each sample $\boldsymbol{x}_i$ is associated with a weight λ_i ($\lambda_i \geq 0$), and the training samples are re-weighted via $\boldsymbol{\Lambda}\boldsymbol{X}$ before combining with the features weight vector $\boldsymbol{w}$. The sample weights evaluate the importance of samples, and their values reflect the difference between training samples. In our algorithm, we care more about the difference between unlabeled samples not that between labeled samples since labeled samples are scarce with more reliable quality while unlabeled samples are abundant with greatly varied properties. Thus, we assign the same weight for labeled samples, e.g., $\lambda_i = 1$ ($i = 1, \ldots, l$). Different from the predetermined scores for the unlabeled samples in (Chang et al. 2014), the weights of the unlabeled sample λ_j ($j = l + 1, \ldots, n$) are adjustable parameters whose values will be adaptively tuned in the training stage.

Priors over Weights of Features and Samples

In semi-supervised classification, the discriminant information included in labeled samples is very limited. To exploit abundant unlabeled samples, previous semi-supervised algorithms define an additional regularizer to capture the local geometric structure underlying the unlabeled samples. From a Bayesian perspective, the regularizer is corresponding to the prior knowledge (Bishop 2006; Jiang et al. 2017). To make better use of few labeled and abundant unlabeled samples, a sparseness-promoting prior over above feature weight vector $\boldsymbol{w}$ is designed as:

$$p(\boldsymbol{w}|\boldsymbol{\alpha}) = (2\pi)^{-\frac{d}{2}}|\boldsymbol{A}+\boldsymbol{B}|^{\frac{1}{2}}\exp\big\{-\frac{1}{2}\boldsymbol{w}^T(\boldsymbol{A}+\boldsymbol{B})\boldsymbol{w}\big\}\,, \quad (2)$$

where $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_d]$ is a d-dimensional hyper-parameter vector, and $\boldsymbol{B} = \gamma \boldsymbol{X}^T \boldsymbol{L} \boldsymbol{X}$, in which γ ($\gamma \geq 0$) is a predetermined parameter that controls the use of local geometric structure of training samples, and $\boldsymbol{L}$ is the graph Laplacian that characterizes the intrinsic local structure. By combining matrix $\boldsymbol{B}$ with feature weight vector $\boldsymbol{w}$, we can integrate the local geometric structure into the prior and conduct semi-supervised feature selection within a Bayesian framework. Note that when the local structure is not obvious, the γ will have a very small value, making matrix $\boldsymbol{B}$ possibly singular. At this point, the prior in Eq. (2) is close to a zero-mean Gaussian prior with the inverse covariance matrix $\boldsymbol{A} = \mathrm{diag}(\alpha_1, \ldots, \alpha_d)$. Therefore, introducing a non-negative diagonal matrix $\boldsymbol{A}$ that acts as a sparse regularizer, could stabilize the prior especially when $\boldsymbol{B}$ may be singular.

To capture the local neighborhood relationship of training samples, the graph Laplacian matrix $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{S}$ is built. $\boldsymbol{S}$ is an affinity matrix whose element S_{ij} reflects the similarity between samples $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, and $\boldsymbol{D}$ denotes a diagonal matrix with the diagonal element $D_{ii} = \sum_j S_{ij}$. To efficiently compute the graph Laplacian, we use a k-nearest neighbor graph to construct the affinity matrix $\boldsymbol{S}$ as follows:

$$S_{ij} = \begin{cases} \eta & \text{if } \boldsymbol{x}_i \text{ and } \boldsymbol{x}_j \text{ have the same label,} \\ 1 & \text{if } \boldsymbol{x}_i \text{ or } \boldsymbol{x}_j \text{ is unlabeled} \\ & \text{but } \boldsymbol{x}_i \in kNN(\boldsymbol{x}_j) \text{ or } \boldsymbol{x}_j \in kNN(\boldsymbol{x}_i), \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

where $kNN(\boldsymbol{x}_j)$ denotes the k nearest neighbors set of sample $\boldsymbol{x}_j$, and η is a constant which is set as 10 in this paper. The affinity matrix $\boldsymbol{S}$ measures the neighborhood relationship of training data while considering the difference between the samples with the same label and the unlabeled samples that are sufficiently close to each other.

Sample weights reflect the importance of unlabeled samples. For important unlabeled samples, the corresponding weights should have relatively large values and vice versa. To avoid the indiscriminative use of unlabeled samples and simultaneously ensure $\lambda_j \geq 0$, a left truncated Gaussian prior is introduced over unlabeled sample weights

$$p(\boldsymbol{\lambda}|\boldsymbol{c}) = \prod_{j=l+1}^{n} p(\lambda_j \mid c_j) = \prod_{j=l+1}^{n} \mathcal{N}_t(\lambda_j|0, c_j^{-1}), \quad (4)$$

where $\boldsymbol{\lambda} = [\lambda_{l+1}, \ldots, \lambda_n]^T$ is the weight vector of unlabeled samples, c_j is the inverse of variance which is also referred as a hyper-parameter, $\boldsymbol{c}$ is the corresponding hyper-parameter vector and $\mathcal{N}_t(\lambda_j|0, c_j^{-1})$ denotes the left truncated Gaussian distribution. For each unlabeled sample weight λ_j ($j = l+1, \ldots, n$), its prior is formalized as follows

$$\begin{aligned} p(\lambda_j \mid c_j) &= \begin{cases} 2\mathcal{N}(\lambda_j \mid 0, c_j^{-1}) & \text{if } \lambda_j \geq 0 \\ 0 & \text{otherwise} \end{cases} \\ &= 2\mathcal{N}(\lambda_j \mid 0, c_j^{-1}) \cdot \delta(\lambda_j), \end{aligned} \quad (5)$$

where $\delta(\lambda_j)$ is an indicator function that returns 1 for each $\lambda_j \geq 0$ and 0 otherwise. According to (Chen, Tiňo, and Yao 2009; 2014), this truncated Gaussian prior can generate adaptive sparseness in the estimation of $\boldsymbol{\lambda}$.

In Bayesian research (Tipping 2001; Li and de Rijke 2018), the non-informative Gamma distribution is often introduced as hyperpriors for hyper-parameters α_i and c_j, which can promote the sparsity of feature and unlabeled sample space. Specifically, the posterior estimate of the parameter (w_i or λ_j) will be very close to zero for a large value of hyper-parameter, thus the corresponding feature or unlabeled sample will be removed from the current model due to less contribution.

Optimization

Based on the prediction model in Eq. (1), the posterior probability for sample $\boldsymbol{x}_i$ belonging to positive class can be calculated by applying a logistic sigmoid function $\sigma(x) = 1/(1+e^{-x})$ to its prediction, i.e., $p(y_i = 1|\boldsymbol{x}_i, \boldsymbol{w}, \lambda) = \sigma(\lambda_i \boldsymbol{w}^T \boldsymbol{x}_i)$. Also, the posterior probability for negative class, $p(y_i = 0|\boldsymbol{x}_i, \boldsymbol{w}, \lambda) = 1 - \sigma(\lambda_i \boldsymbol{w}^T \boldsymbol{x}_i)$. By providing pseudo labels for unlabeled samples, the likelihood function $p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}, \boldsymbol{\lambda})$ can be decomposed as:

$$p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}, \boldsymbol{\lambda}) = p(\boldsymbol{y}_l|\boldsymbol{X}, \boldsymbol{w}, \boldsymbol{\lambda}) \cdot r\, p(\widetilde{\boldsymbol{y}}_u|\boldsymbol{X}, \boldsymbol{w}, \boldsymbol{\lambda})\,, \quad (6)$$

where $p(\boldsymbol{y}_l|\boldsymbol{X}, \boldsymbol{w}, \boldsymbol{\lambda})$ denotes the likelihood function on labeled samples with true label vector $\boldsymbol{y_l} = [y_1, \ldots, y_l]^T$, $p(\widetilde{\boldsymbol{y}}_u|\boldsymbol{X}, \boldsymbol{w}, \boldsymbol{\lambda})$ denotes the likelihood function on unlabeled samples with pseudo label vector $\widetilde{\boldsymbol{y}}_u = [\widetilde{y}_{l+1}, \ldots, \widetilde{y}_n]^T$, and $\boldsymbol{y} = [\boldsymbol{y}_l, \widetilde{\boldsymbol{y}}_u]^T$. To enlarge the label information, we introduce pseudo labels $\widetilde{y}_j$ for unlabeled samples, which can be obtained by the label propagation algorithm (Zhou et al. 2004). The pseudo labels extend the definition of likelihood, but also contain some false labels for unlabeled samples, which affects the reliability of likelihood. Therefore, a trade-off parameter r is introduced to balance $p(\boldsymbol{y}_l|\boldsymbol{X}, \boldsymbol{w}, \boldsymbol{\lambda})$ and $p(\widetilde{\boldsymbol{y}}_u|\boldsymbol{X}, \boldsymbol{w}, \boldsymbol{\lambda})$.

The MAP Estimates of Parameters $\boldsymbol{w}$ and $\boldsymbol{\lambda}$: Having the priors and likelihood, the posterior distribution over $\boldsymbol{w}$ and $\boldsymbol{\lambda}$ can be computed by using the Bayesian rule:

$$p(\boldsymbol{w}, \boldsymbol{\lambda}|\boldsymbol{y}, \boldsymbol{\alpha}, \boldsymbol{c}) = \frac{p(\boldsymbol{w}|\boldsymbol{\alpha})p(\boldsymbol{\lambda}|\boldsymbol{c})p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}, \boldsymbol{\lambda})}{p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\alpha}, \boldsymbol{c})}\,. \quad (7)$$

Due to the non-Gaussian distribution of the likelihood, $p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\alpha}, \boldsymbol{c}) = \int\int p(\boldsymbol{w}|\boldsymbol{\alpha})\, p(\boldsymbol{\lambda}|\boldsymbol{c})\, p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}, \boldsymbol{\lambda}) d\boldsymbol{w}\, d\boldsymbol{\lambda}$ in Eq. (7) is intractable, and thus we cannot derive an analytical solution for $p(\boldsymbol{w}, \boldsymbol{\lambda}|\boldsymbol{y}, \boldsymbol{\alpha}, \boldsymbol{c})$. Adopting the Bernoulli

distribution for likelihood and putting the priors in Eqs. (2) and (4) into the log of posterior $p(\boldsymbol{w}, \boldsymbol{\lambda}|\boldsymbol{y}, \boldsymbol{\alpha}, \boldsymbol{c})$, we have

$$\begin{aligned} Q(\boldsymbol{w}, \boldsymbol{\lambda}) &= \log p(\boldsymbol{w}|\boldsymbol{\alpha}) + \log p(\boldsymbol{\lambda}|\boldsymbol{c}) + \log p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}, \boldsymbol{\lambda}) \\ &= \sum_{i=1}^{l} [y_i \log \sigma_i + (1 - y_i) \log(1 - \sigma_i)] \\ &+ \mu \sum_{j=l+1}^{n} [\widetilde{y}_j \log \sigma_j + (1 - \widetilde{y}_j) \log(1 - \sigma_j)] \\ &- \frac{1}{2} \boldsymbol{w}^T (\boldsymbol{A} + \boldsymbol{B}) \boldsymbol{w} - \frac{1}{2} \boldsymbol{\lambda}^T \boldsymbol{C} \boldsymbol{\lambda} + \sum_{j=l+1}^{n} \log \delta(\lambda_j) \,, \end{aligned} \tag{8}$$

where $\sigma_i = \sigma(\lambda_i \boldsymbol{w}^T \boldsymbol{x}_i)$, $\mu = \ln r$ $(0 \leq \mu \leq 1)$ is a parameter that controls the importance of the likelihood function defined on unlabeled samples, and $\boldsymbol{C}$=diag$(c_{l+1}, \ldots, c_n)$. As the indicator function $\delta(\lambda_j)$ is not differentiable, we can approximate it by a sigmoid function $\sigma(\beta\lambda_j)$ with $\beta = 5$. The maximum a posterior (MAP) estimates of $\boldsymbol{w}$ and $\boldsymbol{\lambda}$ can be boiled down to how to maximize $Q(\boldsymbol{w}, \boldsymbol{\lambda})$. Due to the non-convexity of the likelihood, it is difficult to maximize $Q(\boldsymbol{w}, \boldsymbol{\lambda})$ with respect to parameters $\boldsymbol{w}$ and $\boldsymbol{\lambda}$ directly and simultaneously. To solve this maximization problem, we propose an alternative way to optimize $Q(\boldsymbol{w}, \boldsymbol{\lambda})$ with respect to one parameter assuming the other to be fixed.

To find the MAP estimate $\widehat{\boldsymbol{w}}$, we first optimize $Q(\boldsymbol{w}, \boldsymbol{\lambda})$ with respect to $\boldsymbol{w}$ when $\boldsymbol{\lambda}$ is fixed. We adopt the iteratively re-weighted least squares method to update $\boldsymbol{w}$ as follows:

$$\widehat{\boldsymbol{w}}^{\text{new}} = \widehat{\boldsymbol{w}} - \boldsymbol{H}_w^{-1} \boldsymbol{g}_w, \tag{9}$$

where $\boldsymbol{g}_w$ denotes the gradient vector of $Q(\boldsymbol{w}, \boldsymbol{\lambda})$ with respect to $\boldsymbol{w}$, and $\boldsymbol{H}_w$ is the Hessian matrix. When $\boldsymbol{\lambda}$ is fixed, $Q(\boldsymbol{w}, \boldsymbol{\lambda})$ can be written as $Q(\boldsymbol{w}|\boldsymbol{\lambda})$. Therefore, the gradient $\boldsymbol{g}_w$ is given by:

$$\boldsymbol{g}_w = \frac{\partial Q(\boldsymbol{w}|\boldsymbol{\lambda})}{\partial \boldsymbol{w}} = \boldsymbol{X}^T \widetilde{\boldsymbol{\Lambda}} (\boldsymbol{y} - \boldsymbol{\sigma}) - (\boldsymbol{A} + \boldsymbol{B}) \boldsymbol{w}, \tag{10}$$

where $\widetilde{\boldsymbol{\Lambda}}$ is a diagonal matrix with $\widetilde{\Lambda}_{ii} = 1$ if $\boldsymbol{x}_i$ is labeled and $\widetilde{\Lambda}_{ii} = \mu\lambda_i$ otherwise, and $\boldsymbol{\sigma} = [\sigma_1, \ldots, \sigma_n]^T$. The Hessian matrix of $\boldsymbol{w}$ is given by:

$$\boldsymbol{H}_w = \frac{\partial^2 Q(\boldsymbol{w}|\boldsymbol{\lambda})}{\partial \boldsymbol{w}^2} = -(\boldsymbol{X}^T \boldsymbol{E} \boldsymbol{X} + \boldsymbol{A} + \boldsymbol{B}), \tag{11}$$

where $\boldsymbol{E}$ is a $d \times d$ diagonal matrix with $E_{ii} = \sigma_i(1 - \sigma_i)$ if $\boldsymbol{x}_i$ is labeled and $E_{ii} = \mu\lambda_i^2 \sigma_i (1 - \sigma_i)$ otherwise.

Likewise, we update $\boldsymbol{\lambda}$ with fixed $\boldsymbol{w}$ as follows:

$$\widehat{\boldsymbol{\lambda}}^{\text{new}} = \widehat{\boldsymbol{\lambda}} - \boldsymbol{H}_\lambda^{-1} \boldsymbol{g}_\lambda, \tag{12}$$

where $\boldsymbol{g}_\lambda$ denotes the gradient vector of $\boldsymbol{\lambda}$, and $\boldsymbol{H}_\lambda$ is the corresponding Hessian matrix. The gradient $\boldsymbol{g}_\lambda$ is given by:

$$\boldsymbol{g}_\lambda = \frac{\partial Q(\boldsymbol{\lambda}|\boldsymbol{w})}{\partial \boldsymbol{\lambda}} = \mu \boldsymbol{P} (\widetilde{\boldsymbol{y}}_u - \boldsymbol{\sigma}_u) - \boldsymbol{C} \boldsymbol{\lambda} + \boldsymbol{k}_\lambda, \tag{13}$$

where $\mathbf{k}_\lambda = [\beta(1 - \sigma(\beta\lambda_{l+1})), \ldots, \beta(1 - \sigma(\beta\lambda_n))]^T$ and $\boldsymbol{\sigma}_u = [\sigma_{l+1}, \ldots, \sigma_n]^T$ are u-dimensional vectors, $\boldsymbol{P}$ is an $u \times u$ diagonal matrix with the i-th diagonal elements $P_{ii} = \boldsymbol{w}^T \boldsymbol{x}_{l+i}$. Then, the Hessian matrix of $\boldsymbol{\lambda}$ is given by:

$$\boldsymbol{H}_\lambda = \frac{\partial^2 Q(\boldsymbol{\lambda}|\boldsymbol{w})}{\partial \boldsymbol{\lambda}^2} = -(\mu \boldsymbol{P}^T \boldsymbol{E}_u \boldsymbol{P} + \boldsymbol{C} + \boldsymbol{O}), \tag{14}$$

where $\boldsymbol{E}_u$ and $\boldsymbol{O}$ are $u \times u$ diagonal matrices with their i-th diagonal elements $E_{u,ii} = \sigma_{l+i}(1 - \sigma_{l+i})$, and $O_{ii} = \beta^2 \sigma(\beta\lambda_{l+i})(1 - \sigma(\beta\lambda_{l+i}))$.

Based on the above alternative update rules, the MAP estimates of feature weights $\boldsymbol{w}$ and unlabeled sample weights $\boldsymbol{\lambda}$ can be obtained. Provided with the MAP estimates of parameters, we can update hyper-parameters $\boldsymbol{\alpha}$ and $\boldsymbol{c}$.

Algorithm 1 The proposed JSFS algorithm

Input: Training data $\boldsymbol{X} \in \mathbb{R}^{n \times d}$, parameters γ and μ.
Output: The selected feature indexes and their corresponding weight vector $\boldsymbol{w}$ for the linear classifier.
Initialize w_i, λ_j, α_i, and c_j for $i = 1, \ldots, d$ and $j = l + 1, \ldots, n$.
Construct the affinity matrix $\boldsymbol{S}$ and graph Laplacian $\boldsymbol{L}$.
Obtain the pseudo label vector $\widetilde{\boldsymbol{y}}_u$ via label propagation.
While $\max_i |w_i^{\text{new}} - w_i^{\text{old}}| > 10^{-3}$ **do**
  If $\|\boldsymbol{g}_w\|/d < 10^{-3}$ **then**
    Fix $\boldsymbol{\lambda}$, compute $\boldsymbol{g}_w$ and $\boldsymbol{H}_w$ by Eqs . (10) and (11), and update $\boldsymbol{w} \leftarrow \boldsymbol{w} - \boldsymbol{H}_w^{-1} \boldsymbol{g}_w$;
  end if
  Remove the i-th feature if $|w_i| < 10^{-3}$;
  If $\|\boldsymbol{g}_\lambda\|/u < 10^{-3}$ **then**
    Fix $\boldsymbol{w}$, compute $\boldsymbol{g}_\lambda$ and $\boldsymbol{H}_\lambda$ by Eqs . (13) and (14), and update $\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} - \boldsymbol{H}_\lambda^{-1} \boldsymbol{g}_\lambda$;
  end if
  Remove the j-th unlabeled sample if $|\lambda_j| < 10^{-3}$;
  Update $\boldsymbol{\alpha}$ and $\boldsymbol{c}$ using Eqs. (18) and (20);
end while

Optimizing the Hyper-parameters $\boldsymbol{\alpha}$ and $\boldsymbol{c}$: Given the MAP estimate of parameters $\boldsymbol{w}$ and $\boldsymbol{\lambda}$, the learning objective becomes the optimization of hyper-parameters $\boldsymbol{\alpha}$ and $\boldsymbol{c}$, which boils down to maximize the hyper-parameter posterior, i.e., $p(\boldsymbol{\alpha}, \boldsymbol{c}|\boldsymbol{X}, \boldsymbol{y}) \propto p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\alpha}, \boldsymbol{c}) p(\boldsymbol{\alpha}) p(\boldsymbol{c})$. Due to the noninformative hyperpriors over $\boldsymbol{\alpha}$ and $\boldsymbol{c}$, we need only maximize $p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\alpha}, \boldsymbol{c})$, which is known as the marginal likelihood. Therefore, we can update hyper-parameters by

$$\begin{aligned} (\widehat{\boldsymbol{\alpha}}, \widehat{\boldsymbol{c}}) &= \arg\max_{(\boldsymbol{\alpha}, \boldsymbol{c})} \; p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\alpha}, \boldsymbol{c}) \\ &= \arg\max_{(\boldsymbol{\alpha}, \boldsymbol{c})} \int \int p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}, \boldsymbol{\lambda}) \, p(\boldsymbol{w}|\boldsymbol{\alpha}) \, p(\boldsymbol{\lambda}|\boldsymbol{c}) d\boldsymbol{w} \, d\boldsymbol{\lambda}. \end{aligned} \tag{15}$$

Since the integral in Eq. (15) is intractable, the maximization with respect to $\boldsymbol{\alpha}$ and $\boldsymbol{c}$ cannot be simultaneously derived. To update the hyper-parameters, we use an iterative re-estimation method to alternately optimize the marginal likelihood between $\boldsymbol{\alpha}$ and $\boldsymbol{c}$. First, we attempt to maximize the marginal likelihood with respect to $\boldsymbol{\alpha}$ assuming $\boldsymbol{c}$ to be fixed. Then we have $\widehat{\boldsymbol{\alpha}} = \arg\max_{\boldsymbol{\alpha}} \mathcal{L}(\boldsymbol{\alpha})$, where

$$\begin{aligned} \mathcal{L}(\boldsymbol{\alpha}) &= \int p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}, \boldsymbol{\lambda}) \, p(\boldsymbol{w}|\boldsymbol{\alpha}) d\boldsymbol{w} \\ &\approx p(\boldsymbol{y}|\boldsymbol{X}, \widehat{\boldsymbol{w}}, \boldsymbol{\lambda}) p(\widehat{\boldsymbol{w}}|\boldsymbol{\alpha}) (2\pi)^{\frac{d}{2}} |-\boldsymbol{H}_{\widehat{w}}|^{-\frac{1}{2}} \,, \end{aligned} \tag{16}$$

where $\widehat{\boldsymbol{w}}$ is the MAP estimate of $\boldsymbol{w}$, and $\boldsymbol{H}_{\widehat{w}}$ is the Hessian matrix computed at $\widehat{\boldsymbol{w}}$. Substituting $\boldsymbol{w}$ in the prior and

likelihood with $\widehat{\boldsymbol{w}}$, we have the log marginal likelihood

$$\log \mathcal{L}(\boldsymbol{\alpha}) \approx \log p(\boldsymbol{y}|\boldsymbol{X}, \widehat{\boldsymbol{w}}, \boldsymbol{\lambda}) - \frac{1}{2}\widehat{\boldsymbol{w}}^T \boldsymbol{A}\widehat{\boldsymbol{w}} + \frac{1}{2}\log|\boldsymbol{A}+\boldsymbol{B}| - \frac{1}{2}\log|-\boldsymbol{H}_{\widehat{w}}|\,. \tag{17}$$

To find the $\boldsymbol{\alpha}$ that can maximize this approximation of log marginal likelihood, we set the first order derivative of $\log \mathcal{L}(\boldsymbol{\alpha})$ to zero, and get the update rule

$$\alpha_i^{\text{new}} = \frac{1}{\widehat{w}_i^2 + G_{ii} + \Sigma_{w,ii}}\,, \tag{18}$$

where $\widehat{w}_i$ is the i-th element of $\widehat{\boldsymbol{w}}$, G_{ii} is the i-th diagonal element of the matrix $\boldsymbol{G} = \boldsymbol{A}^{-1}\boldsymbol{B}(\boldsymbol{I}+\boldsymbol{A}^{-1}\boldsymbol{B})^{-1}\boldsymbol{A}^{-1}$, and $\Sigma_{w,ii}$ denotes the i-th diagonal element of $-\boldsymbol{H}_{\widehat{w}}^{-1}$.

Likewise, we maximize the marginal likelihood with respect to $\boldsymbol{c}$ when $\boldsymbol{\alpha}$ is fixed, i.e., $\widehat{\boldsymbol{c}} = \arg\max_{\boldsymbol{c}} \mathcal{L}(\boldsymbol{c})$, where

$$\mathcal{L}(\boldsymbol{c}) \approx p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}, \widehat{\boldsymbol{\lambda}}) p(\widehat{\boldsymbol{\lambda}}|\boldsymbol{c}) (2\pi)^{\frac{u}{2}} |-\boldsymbol{H}_{\widehat{\lambda}}|^{-\frac{1}{2}}, \tag{19}$$

where $\widehat{\boldsymbol{\lambda}}$ is the MAP estimate of $\boldsymbol{\lambda}$, and $\boldsymbol{H}_{\widehat{\lambda}}$ is the Hessian matrix computed at $\widehat{\boldsymbol{\lambda}}$. Setting the first order derivative of $\log \mathcal{L}(\boldsymbol{c})$ to zero, we have

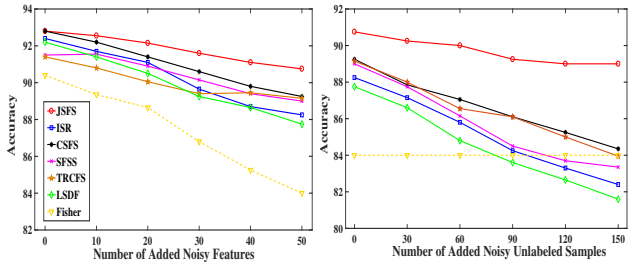
$$c_j^{\text{new}} = \frac{1}{\widehat{\lambda}_j^2 + \Sigma_{\lambda,jj}}\,, \tag{20}$$

where $\widehat{\lambda}_j$ is the j-th element of $\widehat{\boldsymbol{\lambda}}$, and $\Sigma_{\lambda,jj}$ denotes the j-th diagonal element of $-\boldsymbol{H}_{\widehat{\lambda}}^{-1}$.

With the above update formulas, the pseudo code of the proposed JSFS algorithm is provided as Algorithm 1. Specifically, JCFS first finds the MAP estimator $\widehat{\boldsymbol{w}}$ and $\widehat{\boldsymbol{\lambda}}$ given the hyper-parameters α_i and c_j by iteratively using Eqs. (9) and (12), and then updates the hyper-parameters by Eqs. (18) and (20). During updating, we generally find that most of w_i and λ_j tend to zero, thus the corresponding irrelevant feature and unlabeled sample will be automatically removed from the current model through Bayesian automatic relevance determination (Bishop 2006), which avoids predetermining the number of selected features and simultaneously accelerates the training speed. Except for measuring the importance of features, the feature weights $\boldsymbol{w}$ can be used to learn a direct classifier for the samples represented by the selected features. For any unseen sample $\hat{\boldsymbol{x}}$, its label $\hat{y} = 1$ whenever $f(\hat{\boldsymbol{x}}, \boldsymbol{w}) = \boldsymbol{w}^T\hat{\boldsymbol{x}} \geq 0$, and $\hat{y} = 0$ whenever $f(\hat{\boldsymbol{x}}, \boldsymbol{w}) = \boldsymbol{w}^T\hat{\boldsymbol{x}} < 0$. Thus, JSFS does not require an additional classifier to be adopted for training as well.

Experiments

In this section, we conduct a series of experiments to evaluate the effectiveness of JSFS. The first experiment aims to validate the robustness of JSFS against noisy features and unlabeled samples. Then, eight high-dimensional datasets from various domains are used to verify the performance of classification and feature selection of JSFS.



(a) Accuracies versus added noisy features (b) Accuracies versus added unlabeled noise samples

Figure 1: Accuracies (in %) of different algorithms with the increase of noisy features and unlabeled samples in data.

Experimental Setup

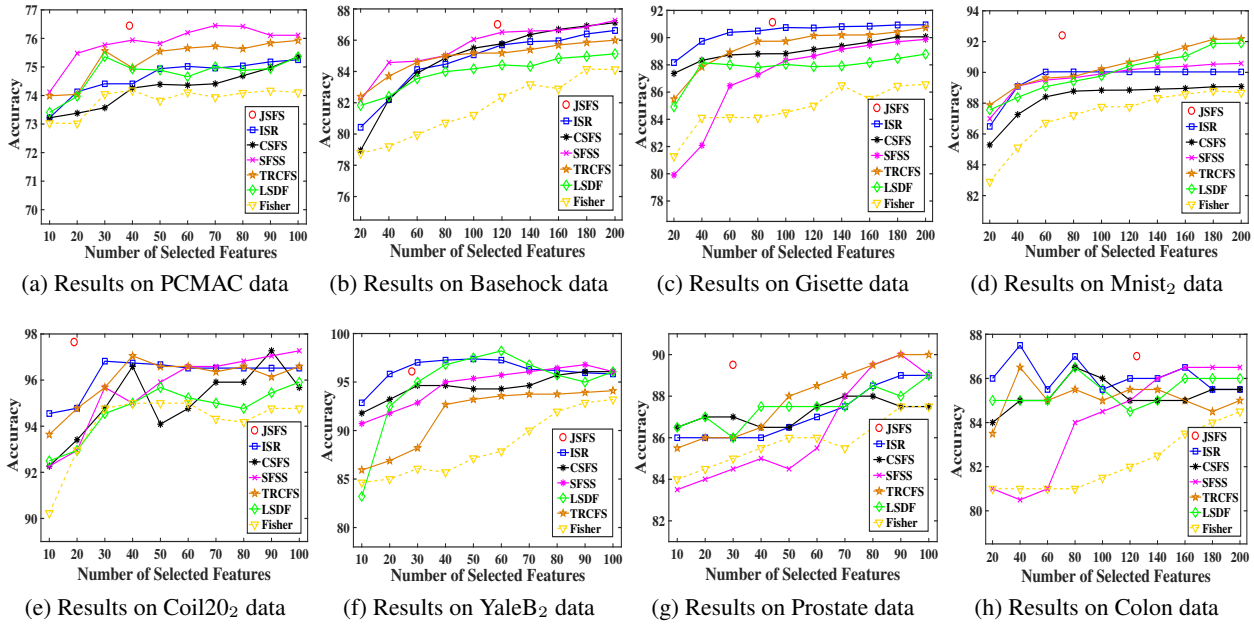
To illustrate the effectiveness and superiority of JSFS, it is compared with some state-of-the-art feature selection algorithms, including three embedded semi-supervised feature selection algorithms: semi-supervised feature selection via insensitive sparse regression (ISR) (Luo et al. 2018), structural feature selection with sparsity (SFSS) (Ma et al. 2012) and convex semi-supervised feature selection (CSFS) (Chang et al. 2014), two filter-based semi-supervised feature selection algorithm: locality sensitive discriminant feature (LSDF) (Zhao, Lu, and He 2008) and semi-supervised feature selection with trace ratio criterion (TRCFS) (Liu et al. 2013), and one supervised feature selection algorithm: fisher score (Fisher) (Bishop and others 1995).

For a fair comparison, the regularization or trade-off parameters of all comparing algorithms are tuned from $\{10^{-2}, 10^{-1}, \cdots, 10^{2}\}$ by grid search, the number of nearest neighbor k is set as five for all algorithms, and parameters μ and $\gamma \in [0, 1]$ for JSFS. Since the comparing algorithms are not able to select features and simultaneously learn a classifier with the selected features. To evaluate the quality of selected features, the linear Support Vector Machine (SVM) (Chang and Lin 2011) is adopted to compute their classification accuracies. Specifically, we first apply these feature selection algorithms to select features and then use SVM to train a classifier with the selected features. For each comparing algorithm, we report their best classification accuracies on the test samples represented by the selected features, employing the trained classifier.

Robustness Against Noisy Features and Samples

The first experiment is to verify the robustness of JSFS against the increasing scales of noisy features and unlabeled samples. We use G50C dataset (Chapelle and Zien 2005), containing 550 samples with two classes, in which each sample has 50 features generated by a 50-dimensional multivariate Gaussian distribution. G50C is randomly partitioned by 350 samples for training and 200 for testing, in which the training set includes 20 labeled samples for each class.

In this experiments, we gradually increase the numbers of noisy features and unlabeled samples to the data, in which the noise is generated from independent and identically dis-



(a) Results on PCMAC data (b) Results on Basehock data (c) Results on Gisette data (d) Results on Mnist_2 data

(e) Results on Coil20_2 data (f) Results on YaleB_2 data (g) Results on Prostate data (h) Results on Colon data

Figure 2: Illustration of the accuracy of JSFS and other feature selection algorithms with different scales of selected features.

tributed $\mathcal{N}(0, 1)$. Each algorithm in independently run on 20 disparate data partitions, and the average classification accuracy on the test samples is used to evaluate them. The accuracy variation curves are illustrated in Figure 1, in which Figure 1(a) demonstrates the accuracies of seven feature selection algorithms against gradually increasing noisy features, and Figure 1(b) shows accuracies versus the increasing number of noisy unlabeled samples with 50 noisy features. As shown in Figure 1, all algorithms trend to achieve reduced accuracy with the increase of the noise in data, and JSFS can consistently outperform the state-of-the-art.

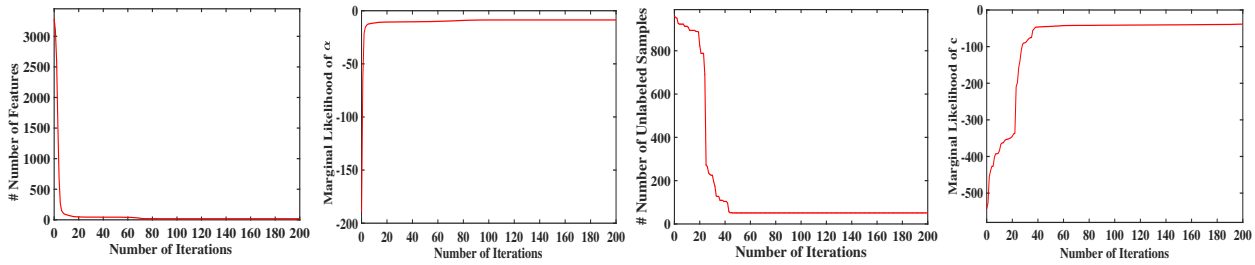
Furthermore, from Figures 1(a) and 1(b) we note that the variation trend of accuracy with added noisy features or unlabeled samples is obviously different. Specifically, the accuracies of comparing algorithms are slightly reduced with increasing noisy features in Figures 1(a). This result is reasonable because they are feature selection algorithms and largely immune to the noisy features. However, their accuracies as depicted in Figure 1(b) are rapidly deteriorated with the increase of unlabeled samples. The main reason is that they directly use all available unlabeled samples and fail to take the reliability of them into full consideration. Fortunately, except for selecting relevant features for classification, our algorithm provides a selective mode to effectively exploit unlabeled samples via the adopted prior in unlabeled sample space, which can automatically eliminate the noisy unlabeled samples. The results in Figure 1 demonstrate the significant superiority of JSFS in terms of robustness against the added noise in data, especially when there exist noisy unlabeled samples.

Performance on High-dimensional Datasets

To evaluate the effectiveness of our proposed joint semi-supervised feature selection and classification learning algorithm, 8 high-dimensional datasets that are collected from different fields are used, including two text datasets: PAMAC and Basehock; four image datasets: Gisette, Mnist_2, Coil20_2 and YaleB_2; and two biological datasets: Prostate and Colon. Mnist_2 is the most challenging binary version of the MNIST dataset, which aims to separate digit 4 from digit 9. Coil20_2 and YaleB_2 denote the binary version of the Coil20 and extended YaleB datasets, respectively. The goal is to discriminate object 1 from object 2. The detailed characteristics of the datasets are summarized in Table 1. Due to different numbers of samples in training set, we randomly sample 5, 5, 5, 5, 10, 20, 20, and 20 labeled samples each class for Coil20_2, YaleB_2, Colon, Prostate, Mnist_2, Gisette, Basehock and PCMAC, and the rest of samples in training set are unlabeled. For these comparing feature selection algorithms, it is still hard to determine the optimal number of features. Thus, we select various numbers of selected features for different datasets since they have different dimensions of features. We perform the experiments 10 times on 10 disparate training and test sets to reduce the statistical variability. The average classification accuracies on different datasets with respect to the number of selected features

Table 1: Characteristics of 8 experimental data sets.

Data	# features	# training	# test
PCMAC	3,289	1,000	943
Basehock	4,862	1,000	993
Gisette	5,000	3,500	3,500
Mnist_2	784	3,782	1,0000
Coil20_2	1,024	100	144
YaleB_2	1,024	100	28
Prostate	5,966	82	20
Colon	2,000	42	20



(a) (b) (c) (d)

Figure 3: Variation curves of the number of features and unlabeled samples selected by our algorithm and the marginal likelihood functions in Eqs. (17) and (19) on PCMAC dataset.

are shown in Figure 2. Different from other feature selection algorithms that require to predetermine the number of selected features, our algorithm can adaptively select relevant features and simultaneously remove out the features with less contribution, automatically determining number of selected features. Therefore, the accuracy of JSFS does not vary with increasing number of selected features in Figure 2. Table 2 presents the average number of features selected by JSFS, which demonstrates the sparsity of JSFS.

Table 2: The average number of features selected by JSFS.

Data	PCMAC	Basehock	Gisette	Mnist_2
SF	39	117	90	72
Data	Coil20_2	YaleB_2	Prostate	Colon
SF	19	28	30	125

* *SF* denotes the average number of selected features.

As shown in Figure 2, the semi-supervised feature selection algorithms achieve higher accuracies than the supervised algorithm (i.e., Fisher) in most of cases, which validates the usefulness of unlabeled samples for performance improvement. Since our algorithm can perform feature selection and simultaneously learn an optimal classifier with the selected features, together with taking the difference of importance between unlabeled samples into consideration. From Figure 2, we observe that our algorithm outperforms other algorithms on 4 out of 8 datasets and also achieves very competitive accuracies on the remaining datasets in comparison with the state-of-the-art algorithms.

Complexity and Convergence Analysis

In the initial training stage, our algorithm contains labeled and all available unlabeled samples. The main computational cost of our algorithm is to update the weights of feature and unlabeled samples in Eqs. (9) and (12), which requires to use the Cholesky decomposition to compute the inverse of their corresponding Hessian matrices in Eqs. (11) and (14). Therefore, the computational complexity of JSFS is $O(d^3 + u^3)$, in which d and u denote the number of features and unlabeled samples, respectively.

Due to the adaptive sparsity in feature and unlabeled sample space, most of w_i and λ_j will be restricted to a small neighborhood around 0 and then we remove their corresponding features and unlabeled samples in future iterations. As iteration goes on, d and u rapidly decrease to relatively small values in the first few iterations, resulting in $O(\bar{u}^3 + \bar{d}^3)$ computational complexity, where $\bar{d} \ll d$ and $\bar{u} \ll u$. In fact, our proposed algorithm is efficient with fast convergence. To illustrate the speed of convergence, Figure 3 provides the variation curves of the number of features and unlabeled samples selected by JSFS and their corresponding marginal likelihoods on PCMAC dataset. From this figure, we can observe that our proposed algorithm can converge stably in the first 40 iterations, which can accelerate the training speed and guarantee the efficiency for practical applications.

Conclusion

In this paper, we proposed a novel semi-supervised feature selection algorithm, called JSFS. JSFS adopts a Bayesian approach to solve the problems existing in semi-supervised feature selection, which is able to adaptively select a relevant feature subset and simultaneously learn a classifier with the selected features. To make full use of the unlabeled samples, JSFS takes the local geometry structure underlying labeled and unlabeled samples as an important knowledge and defines a prior on feature weight. Instead of the indiscriminative use of all unlabeled samples, a left truncated Gaussian prior is introduced on the unlabeled sample weight, which can measure the importance of unlabeled sample and eliminate the outliers adaptively. These priors act as regularization terms and jointly encourage the sparsity in the utilization of both features and unlabeled samples. We conduct experiments on different field datasets, and compare JSFS with the state-of-the-art supervised and semi-supervised feature selection algorithms. The results demonstrate the superiority of JSFS over others in terms of the robustness against noise and the effectiveness and efficiency on high-dimensional data. Future work could study how to use the incremental learning or online strategy (Jiang et al. 2017; Mohsenzadeh, Sheikhzadeh, and Nazari 2016; Shivaswamy and Joachims 2015) to further reduce the computational complexity of JSFS. Additionally, it would be an important direction to extend JSFS to solve the multi-class problems for future research.

Acknowledgments

This work was supported in part by the National Key Research and Development Program of China under Grant 2016YFB1000905, and in part by the National Natural Science Foundation of China under Grant 91546116, Grant 91846111, Grant 61876206, and Grant 91746209.

References

Belkin, M.; Niyogi, P.; and Sindhwani, V. 2006. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research* 7:2399–2434.

Bishop, C., et al. 1995. *Neural networks for pattern recognition*. Oxford university press.

Bishop, C. M. 2006. *Pattern Recognition and Machine Learning*. Springer.

Chandrashekar, G., and Sahin, F. 2014. A survey on feature selection methods. *Computers & Electrical Engineering* 40(1):16–28.

Chang, C.-C., and Lin, C.-J. 2011. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* 2(3):27.

Chang, X.; Nie, F.; Yang, Y.; and Huang, H. 2014. A convex formulation for semi-supervised multi-label feature selection. In *AAAI*, 1171–1177.

Chapelle, O., and Zien, A. 2005. Semi-supervised classification by low density separation. In *Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics*, volume 1, 57–64.

Chapelle, O.; Schölkopf, B.; and Zien, A. 2006. *Semi-Supervised Learning*. MIT Press Cambridge.

Chen, H.; Jiang, B.; and Yao, X. 2018. Semisupervised negative correlation learning. *IEEE Transactions on Neural Networks and Learning Systems* 29(11):5366–5379.

Chen, H.; Tiňo, P.; and Yao, X. 2009. Probabilistic classification vector machines. *IEEE Transactions on Neural Networks* 20(6):901–914.

Chen, H.; Tiňo, P.; and Yao, X. 2014. Efficient probabilistic classification vector machine with incremental basis function selection. *IEEE Transactions on Neural Networks and Learning Systems* 25(2):356–369.

He, X.; Cai, D.; and Niyogi, P. 2006. Laplacian score for feature selection. In *Advances in neural information processing systems*, 507–514.

Jiang, B.; Li, C.; Chen, H.; Yao, X.; and de Rijke, M. 2016. Probabilistic feature selection and classification vector machine. *arXiv preprint arXiv:1609.05486*.

Jiang, B.; Chen, H.; Yuan, B.; and Yao, X. 2017. Scalable graph-based semi-supervised learning through sparse bayesian model. *IEEE Transactions on Knowledge and Data Engineering* 29(12):2758–2771.

Li, C., and de Rijke, M. 2018. Incremental sparse bayesian ordinal regression. *Neural Networks* 106:294–302.

Liu, Y.; Nie, F.; Wu, J.; and Chen, L. 2013. Efficient semi-supervised feature selection with noise insensitive trace ratio criterion. *Neurocomputing* 105:12–18.

Liu, X.; Wang, L.; Zhang, J.; Yin, J.; and Liu, H. 2014. Global and local structure preservation for feature selection. *IEEE Transactions on Neural Networks and Learning Systems* 25(6):1083–1095.

Luo, T.; Hou, C.; Nie, F.; Tao, H.; and Yi, D. 2018. Semi-supervised feature selection via insensitive sparse regression with application to video semantic recognition. *IEEE Transactions on Knowledge and Data Engineering*.

Ma, Z.; Nie, F.; Yang, Y.; Uijlings, J. R.; Sebe, N.; and Hauptmann, A. G. 2012. Discriminating joint feature analysis for multimedia data understanding. *IEEE Transactions on Multimedia* 14(6):1662–1672.

Mohsenzadeh, Y.; Sheikhzadeh, H.; and Nazari, S. 2016. Incremental relevance sample-feature machine: A fast marginal likelihood maximization approach for joint feature selection and classification. *Pattern Recognition* 60:835–848.

Nie, F.; Xiang, S.; Jia, Y.; Zhang, C.; and Yan, S. 2008. Trace ratio criterion for feature selection. In *AAAI*, volume 2, 671–676.

Nie, F.; Huang, H.; Cai, X.; and Ding, C. H. 2010. Efficient and robust feature selection via joint l2, 1-norms minimization. In *Advances in neural information processing systems*, 1813–1821.

Sakai, T.; Plessis, M. C.; Niu, G.; and Sugiyama, M. 2017. Semi-supervised classification based on classification from positive and unlabeled data. In *International Conference on Machine Learning*, 2998–3006.

Sheikhpour, R.; Sarram, M. A.; Gharaghani, S.; and Chahooki, M. A. Z. 2017. A survey on semi-supervised feature selection methods. *Pattern Recognition* 64:141–158.

Shivaswamy, P., and Joachims, T. 2015. Coactive learning. *Journal of Artificial Intelligence Research* 53:1–40.

Tipping, M. E. 2001. Sparse bayesian learning and the relevance vector machine. *Journal of Machine Learning Research* 1:211–244.

Xu, Z.; King, I.; Lyu, M. R.-T.; and Jin, R. 2010. Discriminative semi-supervised feature selection via manifold regularization. *IEEE Transactions on Neural networks* 21(7):1033–1047.

Zhao, Z., and Liu, H. 2007. Semi-supervised feature selection via spectral analysis. In *Proceedings of the 2007 SIAM international conference on data mining*, 641–646. SIAM.

Zhao, J.; Lu, K.; and He, X. 2008. Locality sensitive semi-supervised feature selection. *Neurocomputing* 71(10-12):1842–1849.

Zhou, D.; Bousquet, O.; Lal, T. N.; Weston, J.; and Schölkopf, B. 2004. Learning with local and global consistency. *Advances in Neural Information Processing Systems* 16(16):321–328.