Online Feature Selection and Its Applications

Jialei Wang, Peilin Zhao, Steven C.H. Hoi, and Rong Jin

Abstract—Feature selection is an important technique for data mining. Despite its importance, most studies of feature selection are restricted to batch learning. Unlike traditional batch learning methods, online learning represents a promising family of efficient and scalable machine learning algorithms for large-scale applications. Most existing studies of online learning require accessing all the attributes/features of training instances. Such a classical setting is not always appropriate for real-world applications when data instances are of high dimensionality or it is expensive to acquire the full set of attributes/features. To address this limitation, we investigate the problem of **Online Feature Selection** (OFS) in which an online learner is only allowed to maintain a classifier involved only a small and fixed number of features. The key challenge of Online Feature Selection is how to make accurate prediction for an instance using a small number of active features. This is in contrast to the classical setup of online learning where all the features can be used for prediction. We attempt to tackle this challenge by studying sparsity regularization and truncation techniques. Specifically, this article addresses two different tasks of online feature selection: (1) *learning with full input* where an learner is allowed to access all the features to decide the subset of active features, and (2) *learning with partial input* where only a limited number of features is allowed to be accessed for each instance by the learner. We present novel algorithms to solve each of the two problems and give their performance analysis. We evaluate the performance of the proposed algorithms for online feature selection on several public datasets, and demonstrate their applications to real-world problems including image classification in computer vision and microarray gene expression analysis in bioinformatics. The encouraging results of our experiments validate the efficacy and efficiency of the proposed techniques.

Keywords—Feature Selection; Online Learning; Large-scale Data Mining; Classification;

1 Introduction

Feature selection is an important topic in data mining and machine learning, and has been extensively studied for many years in literature [10], [19], [28], [37], [44]. For classification, the objective of feature selection is to select a subset of relevant features for building effective prediction models. By removing irrelevant and redundant features, feature selection can improve the performance of prediction models by alleviating the effect of the curse of dimensionality, enhancing the generalization performance, speeding up the learning process, and improving the model interpretability. Feature selection has found applications in many domains, especially for the problems involved high dimensional data.

Despite being studied extensively, most existing studies of feature selection are restricted to *batch* learning, which assumes the feature selection task is conducted in an off-line/batch learning fashion and all the features of training instances are given a priori. Such assumptions may not always hold for real-world applications in which training examples arrive in a sequential manner or it is expensive to collect the full information of training data. For example, in an online spam email detection system, training data usually arrive sequentially, making it difficult to deploy a regular batch feature selection technique in a timely, efficient, and scalable manner. Another example is feature selection in bioinformatics, where acquiring the entire set of features/attributes for every training instance is expensive due to the high cost in conducting wet lab experiments.

Unlike the existing feature selection studies, we study the problem of *Online Feature Selection* (OFS), aiming to resolve the feature selection problem in an online fashion by effectively exploring online learning techniques. Specifically, the goal of online feature selection is to develop online classifiers that involve only a small and fixed number of features for classification. Online feature selection is particularly important and necessary when a real-world application has to deal with sequential training data of high dimensionality, such as online spam classification tasks, where traditional batch feature selection approaches can not be applied directly.

In this paper, we address two different types of online feature selection tasks: (i) OFS by learning with full inputs, and (ii) OFS by learning with partial inputs. For the first task, we assume that the learner can access all the features of training instances, and our goal is to efficiently identify a fixed number of relevant features for accurate prediction. In the second task, we consider a more challenging scenario where the learner is allowed to access a fixed small number of features for each training instance to identify the subset of relevant features. To make this problem attractable, we allow the learner to decide which subset of features to acquire for each training instance.

The major contributions of this paper include: (i) we

- *Steven CH Hoi is the corresponding author. J. Wang, P. Zhao and S.C.H. Hoi are with the School of Computer Engineering, Nanyang Technological University, Singapore 639798.*
  E-mail: {jl.wang, zhao0106, chhoi}@ntu.edu.sg
- *R. Jin is with Department of Computer Science and Engineering Michigan State University East Lansing, MI, 48824.*
  E-mail: rongjin@cse.msu.edu

propose novel algorithms to solve both of the above OFS tasks; (ii) we analyze their theoretical properties of the proposed algorithms; (iii) we validate their empirical performance by conducting an extensive set of experiments; (iv) finally, we apply our technique to solve real-world problems in text classification, computer vision and bioinformatics. We note that a short version of this work had been appeared in the KDD-2012 Workshop on the Big Data Mining topic (BigMine-2012) [23]. The rest of this paper is organized as follows. Section 2 reviews related work. Section 3 presents the problem and the proposed algorithms as well as their theoretical analysis. Section 4 discusses our empirical studies and Section 5 concludes this work.

2 Related Work

Our work is closely related to the studies of online learning and feature selection in literature. Below we review important related works in both areas.

One classical online learning method is the well-known Perceptron algorithm [35], [16]. Recently, a large number of online learning algorithms have been proposed [22], [6], [49], [21], [39], in which many of them follow the criterion of maximum margin principle [17], [24], [6], [49]. For example, the Passive-Aggressive algorithm [6] proposes to update a classifier when the incoming training example is either misclassified or fall into the range of classification margin. The PA algorithm is limited in that it only exploits the first order information during the updating. This limitation has been addressed by the recently proposed confidence weighted online learning algorithms that exploit the second order information [14], [7], [8]. Despite the extensive investigation, most studies of online learning requires the access to all the features of training instances. In contrast, we consider an online learning problem where the learner is only allowed to access a small and fixed number of features, a significantly more challenging problem than the conventional setup of online learning.

Feature Selection (FS) has been studied extensively in the literatures of data mining and machine learning [10], [19]. The existing FS algorithms generally can be grouped into three categories: supervised, unsupervised, and semi-supervised FS. *Supervised* FS selects features according to labeled training data. Based on different selection criterions and methodologies, the existing supervised FS methods can be further divided into three groups: *Filter methods*, *Wrapper methods*, and *Embedded methods* approaches. Filter methods [46], [9], [1] choose important features by measuring the correlation between individual features and output class labels, without involving any learning algorithm; wrapper methods [25] rely on a predetermined learning algorithm to decide a subset of important features. Although wrapper methods generally tend to outperform filter methods, they are usually more computationally expensive than the filter methods. Embedded methods [2], [5], [42], [54] aim to integrate the feature selection process into the model training process. They are usually faster than the wrapper methods and able to provide suitable feature subset for the learning algorithm. When there is no label information available, *unsupervised* feature selection attempts to select the important features which preserve the original data similarity or manifold structures. Some representative works include Laplacian Score [20], Spectral Feature Selection [53], and the recently proposed $\ell_{2,1}$-Norm Regularized Discriminative Feature Selection [45]. Feature selection has found many applications [19], including bioinformatics, text analysis and image annotation [29]. Finally, recent years also witness some semi-supervised feature selection methods that exploit both labeled and unlabeled data information [52], [34], [43]. Our OFS technique generally belongs to supervised FS.

We note that it is important to distinguish Online Feature Selection addressed in this work from the previous studies of online streaming feature selection in [33], [18], [41]. In those works, features are assumed to arrive one at a time while all the training instances are assumed to be available before the learning process starts, and their goal is to select a subset of features and train an appropriate model at each time step given the features observed so far. This differs significantly from our online learning setting where training instances arrive sequentially, a more natural scenario in real-world applications.

Our work is closely related to sparse online learning [15], [27], whose goal is to learn a sparse linear classifier from a sequence of high-dimensional training examples. Our work however differs from these studies in that we are motivated to explicitly address the feature selection issue and thus impose a hard constraint on the number of non-zero elements in classifier $\mathbf{w}$, while most of the previous studies of sparse online learning do not aim to explicitly address feature selection, and usually enforce only soft constraints on the sparsity of the classifier. Despite the difference between two kinds of problems and methodologies, we will show empirically in our experiments that our proposed Online Feature Selection algorithm performs better than the cutting-edge sparse online learning algorithms for online classification tasks when the same sparsity level is enforced for the two algorithms.

Finally, we would like to distinguish our work from budget online learning [3], [11], [31], [48], [51] which aims to learn a kernel-based classifier with a bounded number of support vectors. A common strategy behind many budget online learning algorithms is to remove the "oldest" support vector when the maximum number of support vectors is reached, which however is not applicable to online feature selection. Our work is different from some existing online learning work for online dimension reduction, such as the online PCA algorithm [40]. Unlike online feature selection that is a supervised learning, online dimensionality reduction is completely unsupervised and requires the access to the full features.

3 Online Feature Selection

3.1 Problem Setting

In this paper, we consider the problem of online feature selection for binary classification. Let $\{(\mathbf{x}_t, y_t)|\ t = 1, \ldots, T\}$ be a sequence of input patterns received over the trials, where each $\mathbf{x}_t \in \mathbb{R}^d$ is a vector of d dimension and $y_t \in \{-1, +1\}$. In our study, we assume that d is a large number and for computational efficiency we need to select a relatively small number of features for linear classification. More specifically, in each trial t, the learner presents a classifier $\mathbf{w}_t \in \mathbb{R}^d$ that will be used to classify instance $\mathbf{x}_t$ by a linear function $\text{sgn}(\mathbf{w}_t^\top \mathbf{x}_t)$. Instead of using all the features for classification, we require the classifier $\mathbf{w}_t$ to have at most B non-zero elements, i.e.,

$$\|\mathbf{w}_t\|_0 \leq B$$

where $B > 0$ is a predefined constant, and consequently at most B features of $\mathbf{x}_t$ will be used for classification. We refer to this problem as Online Feature Selection (OFS). Our goal is to design an effective strategy for OFS that can make a small number of mistakes. Throughout the paper, we assume $\|\mathbf{x}_t\|_2 \leq 1, t = 1, \ldots, T$.

3.2 OFS: Learning with Full Input

In this task, we assume the learner is provided with full inputs of every training instance (i.e. $\mathbf{x}_1, \ldots, \mathbf{x}_T$). To motivate our algorithm, we first present a simple but noneffective algorithm that simply truncates the features with small weights. The failure of this simple algorithm motivates us to develop effective algorithms for OFS.

3.2.1 A Simple Truncation Approach

A straightforward approach to online feature selection is to modify the Perceptron algorithm by applying truncation. Specifically, In the t-th trial, when being asked to make prediction, we will truncate the classifier $\mathbf{w}_t$ by setting everything but the B largest (absolute value) elements in $\mathbf{w}_t$ to be zero. This truncated classifier, denoted by $\mathbf{w}_t^B$, is then used to classify the received instance $\mathbf{x}_t$. Similar to the Perceptron algorithm, when the instance is misclassified, we will update the classifier by adding the vector $y_t\mathbf{x}_t$ where $(\mathbf{x}_t, y_t)$ is the misclassified training example. Algorithm 1 shows the steps of this approach.

Unfortunately, this simple approach does not work: it cannot guarantee a small number of mistakes. To see this, consider the case where the input pattern $\mathbf{x}$ can only take two possible patterns, either $\mathbf{x}_a$ or $\mathbf{x}_b$. For $\mathbf{x}_a$, we set its first B elements to be 1 and the remaining elements to be 0. For $\mathbf{x}_b$, we set its first B elements to be 0 and the remaining elements to be 1. An instance $\mathbf{x}$ is assigned to the positive class (i.e., $y = +1$) when it is $\mathbf{x}_a$, and assigned to the negative class (i.e., $y = -1$) when it is $\mathbf{x}_b$. Let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{2T}, y_{2T})$ be a sequence of $2T$ examples, with $\mathbf{x}_{2k+1} = \mathbf{x}_a, y_{2k+1} = 1, k = 0, \ldots, T-1$ and $\mathbf{x}_{2k} = \mathbf{x}_b, y_{2k} = -1, k = 1, \ldots, T$. It is clear that Algorithm 1 will always make a mistake while a simple classifier that uses only two attributes (i.e., the first feature and the $(B+1)$-th feature) will make almost no mistakes.

Algorithm 1 Modified Perceptron by Truncation for OFS

Input
- B: the number of selected features

Initialization
- $\mathbf{w}_1 = 0$

for $t = 1, 2, \ldots, T$ **do**
  Receive $\mathbf{x}_t$
  Make prediction $\text{sgn}(\mathbf{x}_t^\top \mathbf{w}_t)$
  Receive y_t
  if $y_t \mathbf{x}_t^\top \mathbf{w}_t \leq 0$ **then**
    $\widehat{\mathbf{w}}_{t+1} = \mathbf{w}_t + y_t \mathbf{x}_t$
    $\mathbf{w}_{t+1} = Truncate(\widehat{\mathbf{w}}_{t+1}, B)$
  else
    $\mathbf{w}_{t+1} = \mathbf{w}_t$
  end if
end for

Algorithm 2 $\mathbf{w} = Truncate(\widehat{\mathbf{w}}, B)$

if $\|\widehat{\mathbf{w}}\|_0 > B$ **then**
  $\mathbf{w} = \widehat{\mathbf{w}}^B$ where $\widehat{\mathbf{w}}^B$ is $\widehat{\mathbf{w}}$ with everything but the B largest elements set to zero.
else
  $\mathbf{w} = \widehat{\mathbf{w}}$
end if

3.2.2 A Sparse Projection Approach

One reason for the failure of Algorithm 1 is that although it selects the B largest elements for prediction, it does not guarantee that the numerical values for the unselected attributes are sufficiently small, which could potentially lead to many classification mistakes. We can avoid this problem by exploring the sparsity property of $L1$ norm, given in the following proposition from [13].

Proposition 1: For $q > 1$ and $\mathbf{x} \in \mathbb{R}^d$, we have

$$\|\mathbf{x} - \mathbf{x}^m\|_q \leq \xi_q \|\mathbf{x}\|_1 (m+1)^{1/q-1}, m = 1, \ldots, d$$

where ξ_q is a constant depending only on q and $\mathbf{x}^m$ stands for the vector $\mathbf{x}$ with everything but the m largest elements set to 0.

Proposition 1 indicates that when a vector $\mathbf{x}$ lives in a $L1$ ball, most of its numerical values are concentrated in its largest elements, and therefore removing the smallest elements will result in a small change to the original vector measured by the L_q norm. Thus, we will restrict the classifier to be restricted to a $L1$ ball, i.e.,

$$\Delta_R = \{\mathbf{w} \in \mathbb{R}^d : \|\mathbf{w}\|_1 \leq R\} \tag{1}$$

Based on this idea, we present a new approach for Online Feature Selection (OFS), as shown in Algorithm 3. The online learner maintains a linear classifier $\mathbf{w}_t$ that has at most B non-zero elements. When a training instance $(\mathbf{x}_t, y_t)$ is misclassified, the classifier is first updated by online gradient descent and then projected to a $L2$ ball to ensure that the norm of the classifier is bounded. If the resulting classifier $\widehat{\mathbf{w}}_{t+1}$ has more than

B non-zero elements, we will simply keep the B elements in $\widehat{\mathbf{w}}_{t+1}$ with the largest absolute weights. Finally, Theorem 1 gives the mistake bound of Algorithm 3.

Algorithm 3 OFS via Sparse Projection. (**OFS**)

Input
- λ: regularization parameter
- η: step size
- B: the number of selected features

Initialization
- $\mathbf{w}_1 = 0$

for $t = 1, 2, \ldots, T$ **do**
  Receive $\mathbf{x}_t$
  Make prediction $\text{sgn}(\mathbf{w}_t^\top \mathbf{x}_t)$
  Receive y_t
  if $y_t \mathbf{w}_t^\top \mathbf{x}_t \leq 1$ **then**
    $\widetilde{\mathbf{w}}_{t+1} = (1 - \lambda\eta)\mathbf{w}_t + \eta y_t \mathbf{x}_t$
    $\widehat{\mathbf{w}}_{t+1} = \min\{1, \frac{\frac{1}{\sqrt{\lambda}}}{\|\widetilde{\mathbf{w}}_{t+1}\|_2}\}\widetilde{\mathbf{w}}_{t+1}$
    $\mathbf{w}_{t+1} = Truncate(\widehat{\mathbf{w}}_{t+1}, B)$
  else
    $\mathbf{w}_{t+1} = (1 - \lambda\eta)\mathbf{w}_t$
  end if
end for

Theorem 1: Let $\ell(z)$ be a convex loss function decreasing in z, with $|\ell'(1)| \geq G$ and $\ell(0) = 1$. After running Algorithm 3 over a sequence of training examples $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$ with $\|\mathbf{x}_t\|_2 \leq 1$ and $\mathbf{x}_t \in \mathbb{R}^d, t \in [T]$, we have the following bound for the number of mistakes M made by Algorithm 3

$$M \leq \frac{1}{\Omega}\left\{\min_{\mathbf{w}\in\Delta_{\sqrt{d}/\lambda}} \|\mathbf{w}\|_2^2 + 2\sum_{t=1}^{T}\ell(y_t\mathbf{w}^\top\mathbf{x}_t)\right\}$$

$$\Omega = 2\eta - \eta^2 G^2 - \frac{4\xi_2 d}{\sqrt{B+1}\lambda} - \frac{\xi_2^2 d}{\lambda(B+1)}$$

The detailed proof can be found in our online supplemental file http://OFS.stevenhoi.org/OFS_proofs.pdf.

3.3 OFS: Learning with Partial Inputs

In the above discussion, although the classifier $\mathbf{w}$ only consists of B non-zero elements, it requires the full knowledge of the instances, namely, every attribute in $\mathbf{x}_t$ has to be measured and computed. We can further constrain the problem of online feature selection by requiring no more than B attributes of $\mathbf{x}_t$ when soliciting input patterns. We note that this may be important for a number of applications when the attributes of objects are expensive to acquire [36], [4]. Evidently, we can not just acquire the B attributes that have non-zero values in the classifier $\mathbf{w}_t$. This is because in this way, the classifier will never be able to change the subset of attributes with non-zero elements, and it is easy to generate a sequence of training examples that lead to a poor classification performance for this approach.

To address this challenge, we propose an ε-greedy online feature selection approach with partial input information by employing a classical technique for making tradeoff between exploration and exploitation [30]. In this approach, we will spend ε of trials for exploration by randomly choosing B attributes from all d attributes, and the remaining $1-\varepsilon$ trials on exploitation by choosing the B attributes for which classifier $\mathbf{w}_t$ has non-zero values. Algorithm 4 shows the detailed steps of the proposed **OFS**$_P$ algorithm. Finally, Theorem 2 gives the mistake bound of Algorithm 4.

Algorithm 4 Learning with Partial Inputs. (**OFS**$_P$)

Input
- R: maximum $L2$ norm
- B: the number of selected features
- ϵ: the exploration-exploitation tradeoff
- η: step size

Initialization
- $\mathbf{w}_1 = 0$

for $t = 1, 2, \ldots, T$ **do**
  Sample Z_t from a Bernoulli distribution with probability ϵ.
  if $Z_t = 1$ **then**
    Randomly choose B attributes $\mathcal{C}_t$ from $[d]$
  else
    Choose the attributes that have non-zero values in $\mathbf{w}_t$, i.e., $\mathcal{C}_t = \{i : [\mathbf{w}_t]_i \neq 0\}$
  end if
  Receive $\widetilde{\mathbf{x}}_t$ by only requiring the attributes in $\mathcal{C}_t$
  Make prediction $\text{sgn}(\mathbf{w}_t^\top \widetilde{\mathbf{x}}_t)$
  Receive y_t
  if $y_t \mathbf{w}_t^\top \widetilde{\mathbf{x}}_t \leq 1$ **then**
    Compute $\widehat{\mathbf{x}}_t$ as

$$[\widehat{\mathbf{x}}_t]_i = \frac{[\widetilde{\mathbf{x}}_t]_i}{\frac{B}{d}\epsilon + I([\mathbf{w}_t]_i \neq 0)(1-\epsilon)}, i = 1, \ldots, d$$

    $\widetilde{\mathbf{w}}_{t+1} = \mathbf{w}_t + y_t \eta \widehat{\mathbf{x}}_t$
    $\widehat{\mathbf{w}}_{t+1} = \min\{1, \frac{R}{\|\widetilde{\mathbf{w}}_{t+1}\|_2}\}\widetilde{\mathbf{w}}_{t+1}$
    $\mathbf{w}_{t+1} = Truncate(\widehat{\mathbf{w}}_{t+1}, B)$
  else
    $\mathbf{w}_{t+1} = \mathbf{w}_t$
  end if
end for

Theorem 2: After running Algorithm 4 over a sequence of training examples $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_T, y_T)$ with $\|\mathbf{x}_t\|_2 \leq 1, t \in [T]$, we have the following bound for the number of mistakes M made by Algorithm 4

$$M \leq \frac{\{\frac{1}{2\eta}\|\mathbf{w}\|_2^2 + \sum_{t=1}^{T}\ell(y_t\mathbf{w}^\top\mathbf{x}_t)\}}{[\frac{\epsilon}{C_d^B} + (1-\epsilon)] - \frac{\eta d}{2B\epsilon} - \frac{2\xi_2\sqrt{d}R^2}{\eta\sqrt{B+1}} - \frac{\xi_2^2\sqrt{d}R^2}{2\eta(B+1)}}$$

where C_d^B is the number of B-combinations from a given set of d elements. The detailed proof can be found in our online supplemental file http://OFS.stevenhoi.org/OFS_proofs.pdf.

4 Experimental Results

In this section, we conduct an extensive set of experiments to evaluate the performance of the proposed online feature selection algorithms. We will first evaluate the online predictive performance of the two OFS tasks on several benchmark datasets from UCI machine learning repository. We will then demonstrate the applications of the proposed online feature selection technique for two real-world applications by comparing the proposed OFS techniques with state-of-the-art batch feature selection techniques in literature [32]. We will also compare the proposed technique with regular the existing online learning technique [15]. Finally, all the source code and data sets used in this paper can be downloaded from our project website http://OFS.stevenhoi.org/.

4.1 Experiment I: OFS with Full Input Information

In this subsection, we will introduce the empirical results of the proposed Online Feature Selection algorithms in full information setting.

4.1.1 Experimental Testbed on UCI and Text Classification Datasets

We test the proposed algorithms on a number of publicly available benchmarking datasets. All of the datasets can be downloaded either from LIBSVM website [1] or UCI machine learning repository [2]. Besides the UCI data sets, we also adopt two high-dimensional real text classification datasets based on the bag-of-words representation: (i) the Reuters Corpus Volume 1 (RCV1) [3]; (ii) 20 Newsgroups datasets [4], we extract the "comp" versus "sci" and "rec" versus "sci" to form two binary classification tasks. Table 1 shows the statistics of the datasets used in our following experiments.

TABLE 1
List of UCI and Text Classification datasets in our experiments.

Dataset	# Samples	# Dimensions
magic04	19020	10
svmguide3	1243	21
german	1000	24
splice	3175	60
spambase	4601	57
a8a	32561	123
RCV1	4086	29992
20Newsgroup("rec"vs"sci")	8928	26214
20Newsgroup("comp"vs"sci")	9840	26214

1. http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/
2. http://www.ics.uci.edu/~mlearn/MLRepository.html
3. http://datahub.io/dataset/rcv1
4. http://qwone.com/~jason/20Newsgroups/

4.1.2 Experimental Setup and Baseline Algorithms

We compare the proposed OFS algorithm against the following two baselines:

- the modified perceptron by the simple truncation step shown in Algorithm 1, denoted as "PE_{trun}" for short;
- a randomized feature selection algorithm, which randomly selects a fixed number of active features in an online learning task, denoted as "RAND" for short.

To make a fair comparison, all algorithms adopt the same experimental settings. We set the number of selected features as $round(0.1 * dimensionality)$ for every dataset, the regularization parameter λ to 0.01, and the learning rate η to 0.2. The same parameters are used by all the baseline algorithms. After that, all the experiments were conducted over 20 times, each with a random permutation of a dataset. All the experimental results were reported by averaging over these 20 runs.

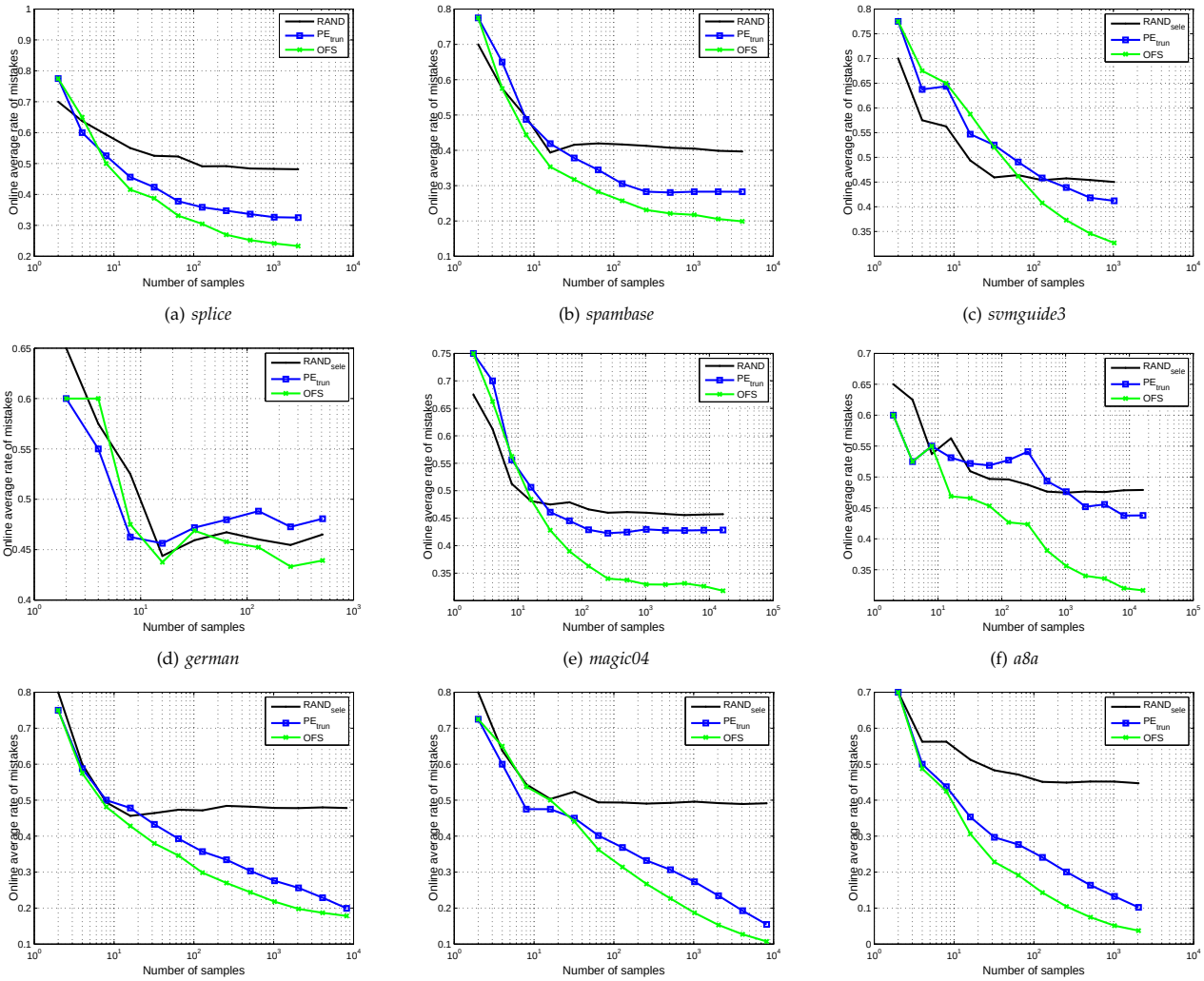
4.1.3 Evaluation of Online Predictive Performance

Table 2 summarizes the online predictive performance of the compared algorithms with a fixed fraction of selected features (10% of all dimensions) on the datasets.

TABLE 2
Evaluation of the average number of mistakes by three algorithms on the six datasets.

Algorithm	svmguide3	german	magic04
RAND	567.6 ± 17.3	472.4 ± 11.1	8689.8 ±58.9
PE_{trun}	512.2 ± 32.6	489.6 ± 29.8	8153.1 ±79.3
OFS	**400.9 ± 66.8**	**432.8 ± 13.6**	**6023.4 ±1342.3**
Algorithm	splice	spambase	a8a
RAND	1517.0 ±25.7	1827.7 ±45.2	15610.7 ±78.8
PE_{trun}	1039.9 ±35.0	1294.8± 66.3	14086.8 ±300.4
OFS	**735.4 ±68.3**	**913.1 ± 157.8**	**9424.4 ±2545.8**
Algorithm	RCV1	"rec" vs "sci"	"comp" vs "sci"
RAND	1818.9 ±41.8	4379.6 ±44.0	4697.2 ±44.8
PE_{trun}	314.3 ±19.8	1343.1± 43.8	1886.7 ±60.1
OFS	**117.2 ±13.8**	**943.81 ± 59.5**	**1725.5 ±60.8**

Several observations can be drawn from the results. First of all, we found that among all the compared algorithms, the RAND algorithm has the highest mistake rate for all the cases. This shows that it is important to learn the active features in an OFS task. Second, we found that the simple "PE_{trun}" algorithm can outperform the RAND algorithm considerably, which further indicates the the importance of selecting informative features for online learning tasks. Finally, among the three algorithms, we found that the OFS algorithm achieved the smallest mistake rate, which is significantly smaller than the two algorithms. This shows that the proposed algorithm is able to considerably boost the performance of the simple "PE_{trun}" approach.



(a) *splice* (b) *spambase* (c) *svmguide3*

(d) *german* (e) *magic04* (f) *a8a*

(g) *20Newsgroup "comp" vs "sci"* (h) *RCV1* (i) *20Newsgroup "rec" vs "sci"*

Fig. 1. Performance evaluation of online feature selection in the online learning process.

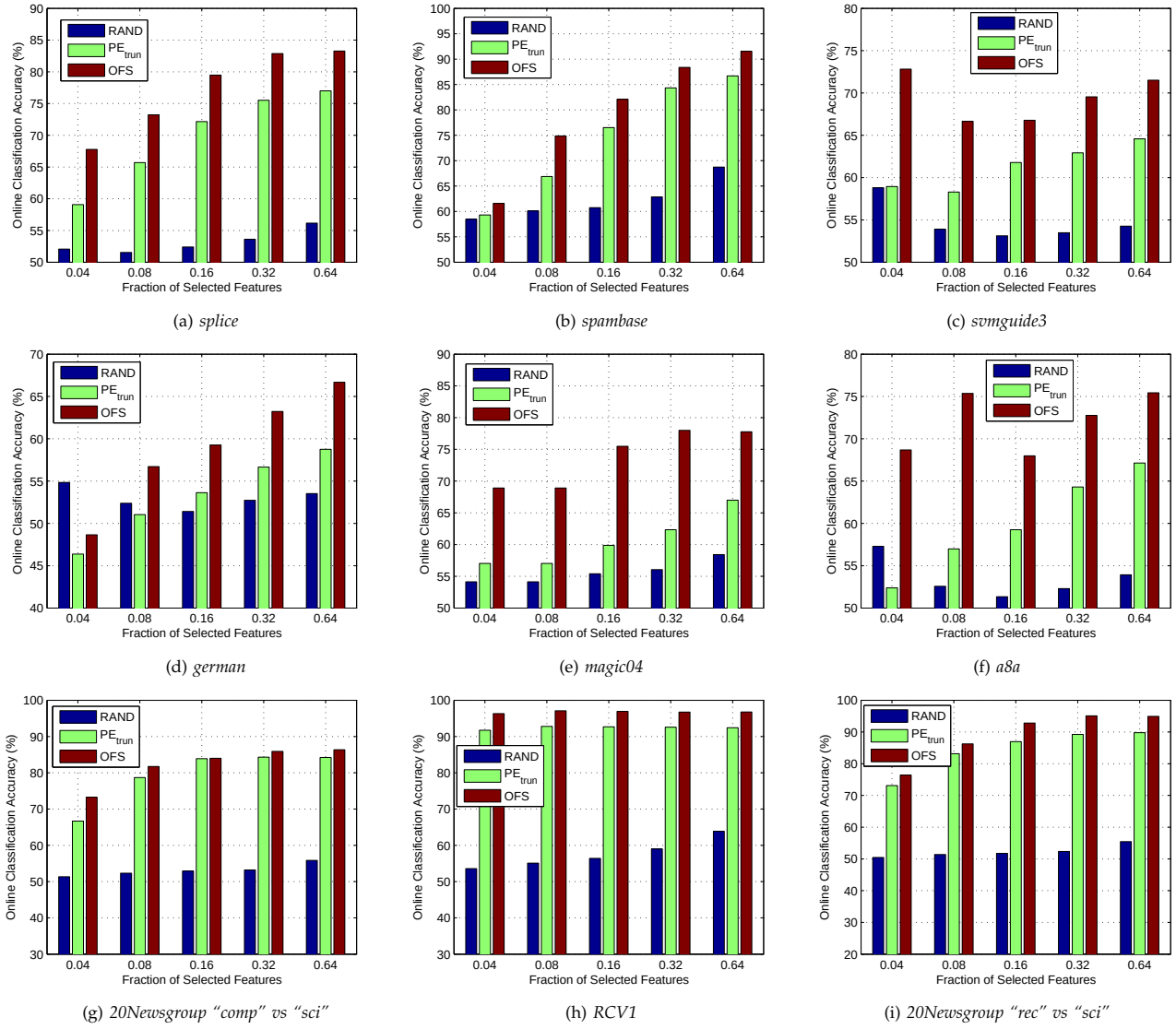
To further examine the online predictive performance, Figure 1 shows how the mistake rates is varied over iterations accord the entire OFS process on the three randomly chosen datasets (similar observations can be found on the other three datasets, we simply omit them due to space limitation). Similar to the previous observations, we can see that the proposed OFS algorithm consistently surpassed the other two algorithms for all the situations. Besides, we also found that the more the training instances received, the more significant the gain achieved by the proposed OFS algorithm over the other baselines. This again verifies the efficacy of the proposed OFS algorithm and its promising potential for large-scale data mining tasks.

Finally, Figure 2 further shows the details of the online performance of the compared online feature selection algorithms with varied fractions of selected features. The proposed OFS algorithm outperform the other two baselines for most cases. This encouraging result further verifies the efficacy of the proposed technique.

4.2 Experiment II: Comparison with Sparse Online Learning

We also compare OFS with the sparse online learning method, i.e., the Forward Backward Splitting (FOBOS) algorithm [15]. Although we mentioned that there is a distinct difference between these two family of algorithms in the related work section, it is interesting and useful to compare them directly in online learning settings. To make a fair comparison, we set the learning rate η to 0.2 for both algorithms, and vary the regularization parameter in FOBOS in order to obtain different levels of sparsity; we then apply OFS to select the exact number of features as FOBOS does, and compare the online classification performances of the two algorithms under the different levels of sparsity.

The experimental results are summarized in Figure 3. From the results, it is clear to see that when the sparsity level is 0 (all the features are selected, for some text datasets, which adopt the bag-of-words representation, the features are already somewhat sparse), FOBOS and



(a) *splice* (b) *spambase* (c) *svmguide3*

(d) *german* (e) *magic04* (f) *a8a*

(g) *20Newsgroup "comp" vs "sci"* (h) *RCV1* (i) *20Newsgroup "rec" vs "sci"*

Fig. 2. Online Classification Accuracy with various fractions of selected features.

OFS perform almost identically, which indicates the two methods have very similar predictive performance for online learning(On newsgroup datasets OFS performs even worse than FOBOS when use all features, but when we select only a small faction of the features, OFS performs much better). When the sparsity level increases, we observe that the proposed OFS algorithm significantly outperforms FOBOS. The FOBOS algorithm adopts the ℓ_1 norm regularization based approach, in which the optimization task of FOBOS leads to the soft-thresholding operations in order to achieve the sparse solutions. In contrast, OFS have two important advantages: (i) OFS can select the exact number of features specified by users, while FOBOS has to carefully tune the regularization parameter in order to achieve the desired sparsity level; (ii) The soft-thresholding operations may achieve different sparsity levels at different iterations during the online learning process, while OFS is able to guarantee the sparsity level of the learner keeps unchanged during the entire online learning process. This promising observation shows that the proposed OFS algorithm is able to identify and exploit more effective features for online learning tasks.

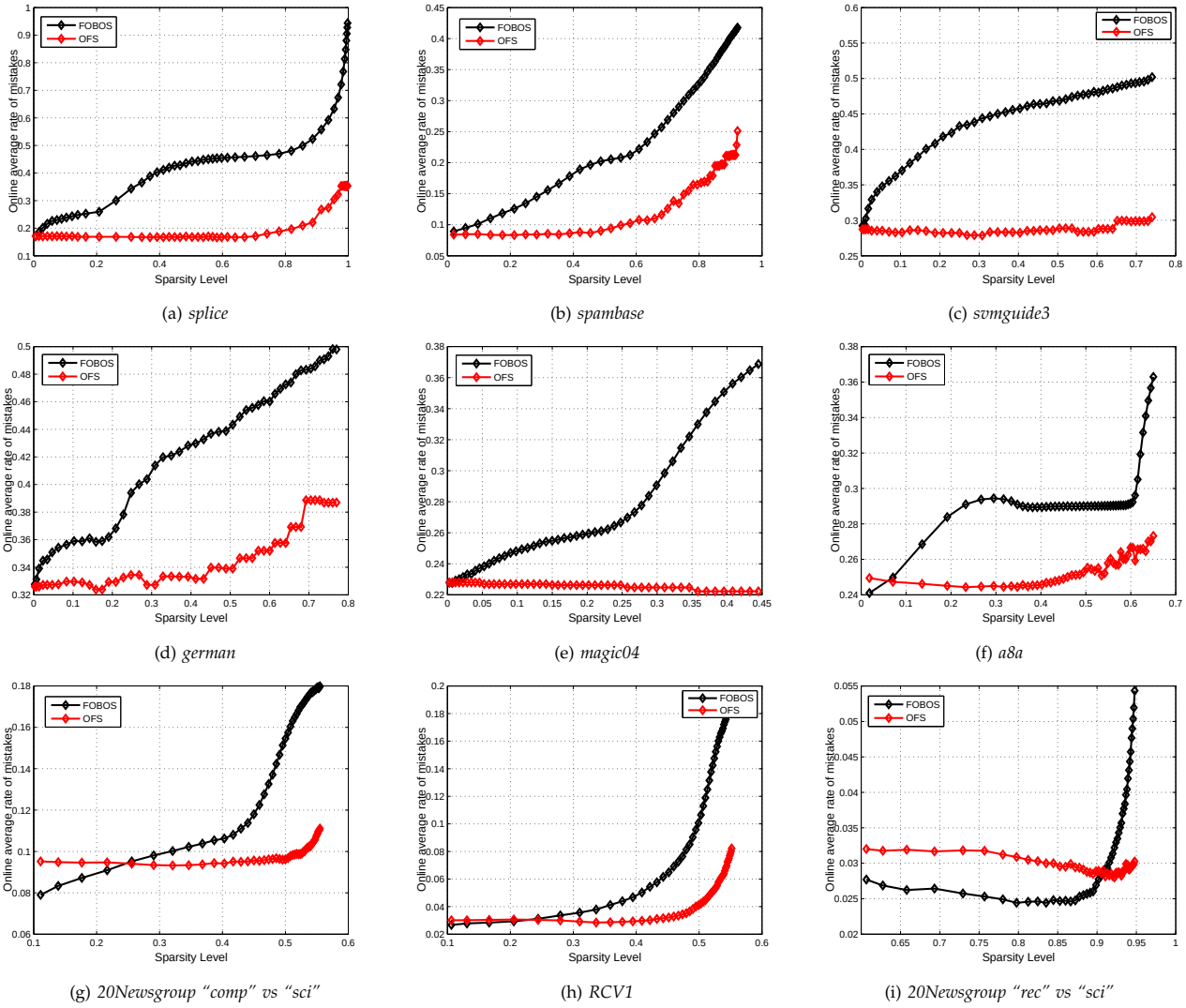
4.3 Experiment III: OFS with Partial Input

We now evaluate the empirical performance of the proposed Online Feature Selection (OFS) algorithm by learning with partial input.

4.3.1 Experimental Setup and Compared Algorithms

We compare the OFS$_P$ algorithm with three other baseline algorithms of learning with partial input:

- modified perceptron by using the truncation step, referred to as "RAND", which randomly selects a fixed number of active features for the input and for learning the weight vector;



(a) *splice* (b) *spambase* (c) *svmguide3*

(d) *german* (e) *magic04* (f) *a8a*

(g) *20Newsgroup "comp" vs "sci"* (h) *RCV1* (i) *20Newsgroup "rec" vs "sci"*

Fig. 3. Comparison of Sparse Online Learning and Online Feature Selection under Varied Levels of Sparsity.

- another modified perceptron, referred to as "PE$_{rand}$", which randomly selects a fixed number of active features for the inputs but treats the top largest elements in the weight vector as active dimensions; and
- a modified OFS algorithm, referred to as OFS$_{rand}$, which randomly selects the active features as the inputs for the OFS algorithm.

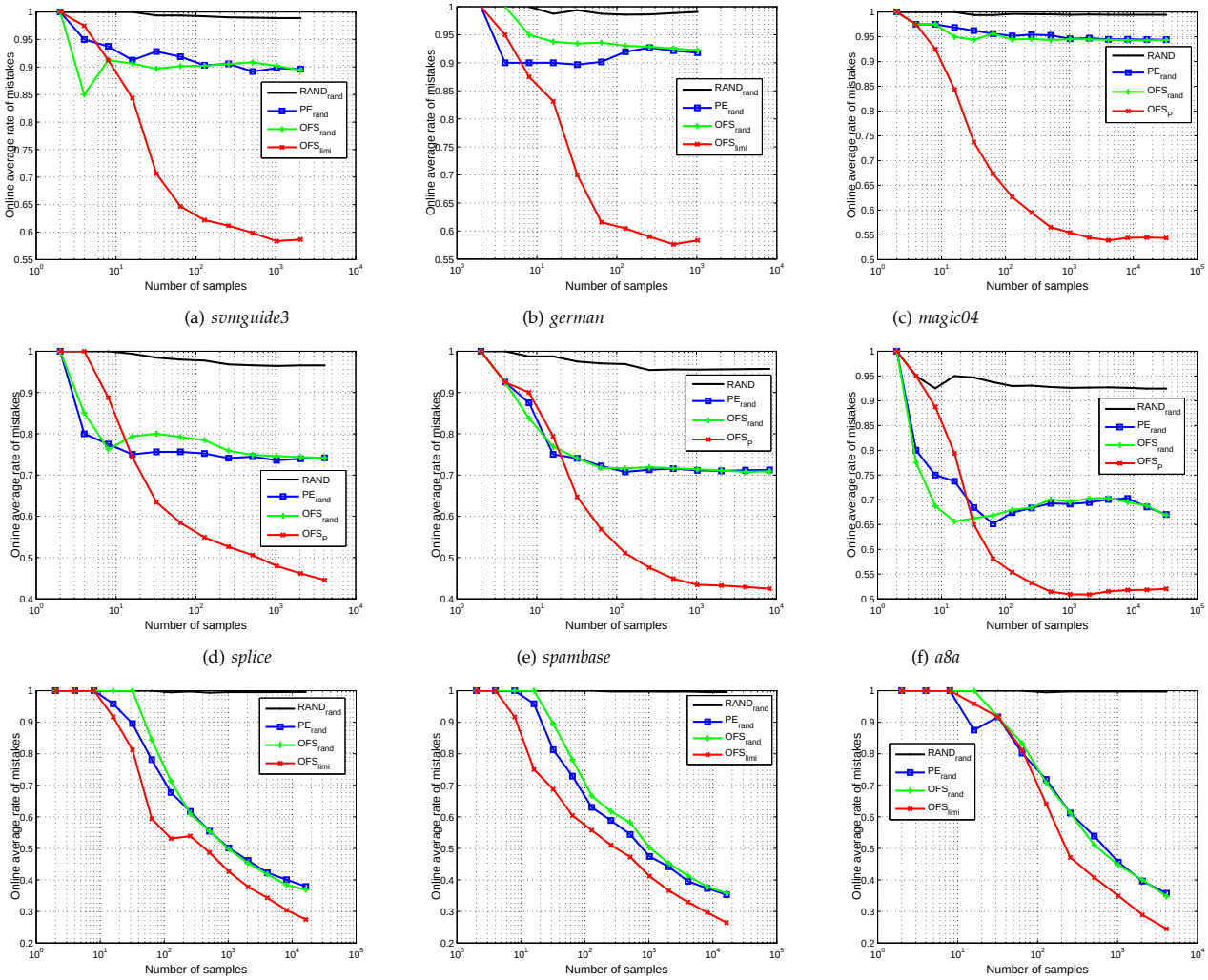
We test all the algorithms on all of the datasets listed in the Table 1. To make a fair comparison, all algorithms adopt the same experimental setup on all the datasets. We set the number of selected features as $round(0.1 * Dimensionality)$ for every dataset. R is set to 10 for OFS$_P$ and OFS$_{rand}$ algorithms. Furthermore, we set $\epsilon = 0.2$ and $\eta = 0.2$ for OFS$_P$. All the experiments were conducted over 20 random permutations for each dataset. All the results were reported by averaging over these 20 runs.

4.3.2 Performance Evaluation

Table 3 summarizes the online prediction performance of the compared algorithms on the six datasets.

Several observations can be drawn from the results. First, we found that the RAND algorithm suffered the highest mistake rate for all cases. This again shows that it is important to learn the active features for the inputs and the weight vector. Second, OFS$_{rand}$ made significantly more mistakes than OFS$_P$ for all the datasets, which validates the importance and efficacy of exploring the knowledge of the active features. Finally, we found that the proposed OFS$_P$ algorithms achieved the smallest mistake rates. This shows that the proposed OFS$_P$ technique is effective for learning the most informative features under the partial input situation.

To further examine the online predictive performance, Figure 4 shows the details of online average mistake rates for the entire OFS process on the six datasets, respectively. Similar to the previous observations, we can see that the proposed OFS$_P$ algorithm consistently



(a) *svmguide3* (b) *german* (c) *magic04*

(d) *splice* (e) *spambase* (f) *a8a*

(g) *20Newsgroup "comp" vs "sci"* (h) *RCV1* (i) *20Newsgroup "rec" vs "sci"*

Fig. 4. Performance evaluation of online feature selection with partial input.

outperformed the other algorithms for all the situations. This again validates the effectiveness of the proposed technique.

4.4 Experiment IV: Applications to Image Classification and Bioinformatics Tasks

In this section, we apply the proposed OFS technique to tackle feature selection tasks of real-world applications in computer vision and Bioinformatics.

4.4.1 Experimental Datasets and Setup

The first application is to solve feature selection problem for image classification. We adopt the CIFAR-10 image dataset [26] [5] in our experiment. It consists of 10 classes of images, which as a subset of the well-known 80-million images. In this experiment, we randomly choose two classes "airplane" and "bird" to form a binary classification task. In our dataset, the CIFAR-10 dataset consists of 3,992 images, where each image is represented by a 3073-dimensional feature vector.

The second application is to solve feature selection task for microarray gene expression data in bioinformatics. We adopt the Colon dataset, which is a microarray gene expression data of tumor and normal colon tissues [38] [6]. This dataset consists of 62 samples and each sample contains 2000 gene features.

The parameter settings are the same as the previous section. All the experiments are conducted over 20 random permutations. All the results are reported by averaging over these 20 runs.

4.4.2 Evaluation of Online Prediction Performance

The experimental results are shown in Table 4 and Table 5. Note that the average mistake rates of online gradient descent with full input dimension on CIFAR-10 and Colon datasets are 0.226 ± 0.003 and 0.354 ± 0.054, respectively.

5. http://www.cs.toronto.edu/~kriz/cifar.html

6. http://genomics-pubs.princeton.edu/oncology

TABLE 6
Evaluation of the classification error rates with two different classifiers (OGD vs. KNN) on "Colon".

OGD	0.005	0.01	0.02	0.04	0.08	0.16	0.32
mRMR	0.464 ± 0.146	**0.400 ± 0.055**	0.361 ± 0.067	0.358 ± 0.075	0.387 ± 0.076	0.393 ± 0.083	0.380 ± 0.058
OFS	**0.441 ± 0.088**	0.409 ± 0.127	**0.348 ± 0.062**	**0.306 ± 0.065**	**0.345 ± 0.057**	**0.341 ± 0.066**	**0.332 ± 0.050**
KNN	0.005	0.01	0.02	0.04	0.08	0.16	0.32
mRMR	0.377 ± 0.066	0.341 ± 0.094	0.251 ± 0.042	0.280 ± 0.088	0.271 ± 0.055	0.319 ± 0.068	0.325 ± 0.110
OFS	**0.345 ± 0.097**	**0.335 ± 0.107**	**0.212 ± 0.097**	**0.180 ± 0.120**	**0.222 ± 0.068**	**0.290 ± 0.068**	**0.287 ± 0.105**

TABLE 7
Evaluation of the classification error rates with two different classifiers (OGD vs. KNN) on "CIFAR-10".

OGD	0.005	0.01	0.02	0.04	0.08	0.16	0.32
mRMR	0.437 ± 0.018	0.428 ± 0.009	0.427 ± 0.012	0.424 ± 0.012	0.364 ± 0.012	0.359 ± 0.017	0.342 ± 0.013
OFS	**0.341 ± 0.028**	**0.318 ± 0.021**	**0.309 ± 0.018**	**0.295 ± 0.015**	**0.270 ± 0.012**	**0.254 ± 0.010**	**0.241 ± 0.008**
KNN	0.005	0.01	0.02	0.04	0.08	0.16	0.32
mRMR	0.325 ± 0.009	0.327 ± 0.014	0.327 ± 0.015	0.323 ± 0.011	**0.238 ± 0.006**	0.264 ± 0.013	0.256 ± 0.010
OFS	**0.293 ± 0.013**	**0.273 ± 0.012**	**0.269 ± 0.012**	**0.259 ± 0.014**	0.257 ± 0.010	**0.253 ± 0.017**	**0.255 ± 0.007**

TABLE 3
Evaluation of the average number of mistakes by learning with partial input on the datasets.

Algorithm	svmguide3	german	magic04
RAND	1229.7± 5.2	991.0± 3.4	18908.3± 11.8
PE_{trun}	1112.3 ± 11.4	918.3 ± 7.8	17939.6± 45.2
OFS_{rand}	1112.3 ± 16.9	924.0±8.4	17948.7± 37.3
OFS_P	**701.5± 42.5**	**589.3 ±33.9**	**10274.2±172.1**
Algorithm	splice	spambase	a8a
RAND	3066.9 ± 8.1	4408.8± 12.8	30089.1± 43.0
PE_{trun}	2361.7± 29.1	3275.4± 40.4	21431.8± 653.6
OFS_{rand}	2361.3 ± 26.9	3278.6± 31.2	21575.3± 728.3
OFS_P	**1418.1± 70.5**	**1954.2± 78.7**	**16931.0 ± 164.6**
Algorithm	RCV1	"rec" vs "sci"	"comp" vs "sci"
RAND	4074.6 ±4.0	8889.3 ±8.5	9795.6 ±2.5
PE_{trun}	1293.0 ±3.0	3134.0± 85.5	3694.0 ±67.8
OFS_{rand}	1280.6 ± 8.5	3176.3± 46.3	3586.6± 26.8
OFS_P	**853.0± 21.1**	**2325.3± 62.0**	**2639.3 ± 45.5**

TABLE 4
Evaluation of online mistake rates on CIFAR-10. ρ is the fraction of selected features.

ρ	RAND	PE_{trun}	OFS
0.01	0.471 ±0.006	0.367 ± 0.009	**0.323 ± 0.020**
0.02	0.466 ±0.006	0.340 ±0.006	**0.309 ±0.017**
0.04	0.466 ±0.006	0.321 ±0.006	**0.288 ±0.012**
0.08	0.464 ±0.008	0.306 ±0.003	**0.267 ±0.008**
0.16	0.461 ±0.006	0.295 ±0.004	**0.252 ±0.004**
0.32	0.455 ±0.009	0.291 ±0.005	**0.238 ±0.003**
0.64	0.436 ±0.008	0.288 ±0.005	**0.226 ± 0.001**

Several observations could be drawn. First, our OFS algorithm performs significantly better than the RAND and the PE_{trun} approaches on both datasets, which demonstrates the effectiveness of our algorithms; Second, on CIFAR-10, We observed a significant improvement in the performance of the proposed OFS algorithm, as the fraction of selected features is increased. It achieves the same performance as OGD when the fraction reaches 64%; on Colon, as the fraction of features increases, we first observed an improvement in the performance, and a decline afterwards. The best performance is achieved when 2% of the features are selected. We attribute the decline in the prediction to the potentially large number of noisy features, which is also acknowledged by the other studies [32].

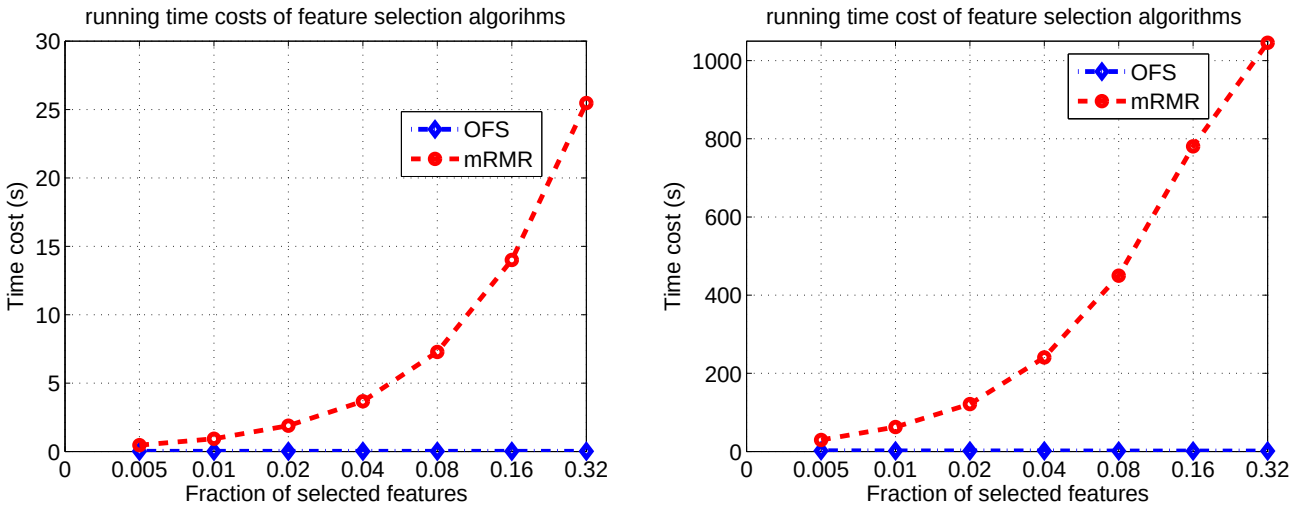
TABLE 5
Evaluation of online mistake rates on "Colon". ρ is the fraction of selected features.

ρ	RAND	PE_{trun}	OFS
0.01	0.496 ±0.061	0.406 ±0.060	**0.362 ± 0.068**
0.02	0.485 ±0.075	0.391 ±0.048	**0.325 ±0.053**
0.04	0.512 ±0.050	0.366 ±0.045	**0.337 ±0.066**
0.08	0.496 ±0.056	0.361 ±0.046	**0.334 ±0.053**
0.16	0.506 ±0.075	0.375 ±0.052	**0.341 ±0.055**
0.32	0.508 ±0.060	0.372 ±0.035	**0.347 ±0.055**
0.64	0.457 ±0.046	0.391 ±0.040	**0.350 ±0.059**

4.4.3 Online vs. Batch Feature Selection Methods

All the above experiments are conducted in an online learning setting. In this experiment, we compare our OFS algorithm with a state-of-the-art batch feature selection method: minimum Redundancy Maximum Relevance Feature Selection (mRMR) [32], [12].

We divide the datasets into two equal size: the first part is used to select features by running FS algorithms (OFS and mRMR), and the second part is used to test the performance of selected features. To examine the efficacy of the selected features invariant to different classifiers, we adopt two types of widely used classifiers: (i) Online gradient descent (OGD) which is an online learning



(a) Colon (b) CIFAR-10

Fig. 5. Evaluation of time efficiency: online feature selection (OFS) v.s. batch feature selection (mRMR).

classifier, and (ii) K-nearest neighbor classifier (KNN), which is a batch learning classifier. In this experiment, we simply fix $K = 5$ for the parameter K in the KNN classifier. We evaluate the performance in terms of both the classification error rates and the computational time efficiency of the two different feature selection algorithms. Table 6 and Table 7 show the classification errors of both data with respect to the percentage of selected features, and Figure 5 shows the time efficiency evaluation.

From Table 6 and Table 7, we can see that the proposed OFS algorithm outperforms mRMR for most cases in terms of classification accuracy. We also noticed that for both feature selection methods, the classification error decreases at first as the percentage of selected features increases; the classification error however increases as the percentage of selected features goes beyond 4% or 8%. This phenomenon can be explained by the fact that many of the input features are irrelevant or redundant. In terms of computational efficiency as shown in Figure 5, we observe that the OFS algorithm has a significant advantage over the batch feature selection algorithm, especially when the number of features to be selected is large. For example, when choosing 32% of features on the CIFAR-10 dataset, the mRMR algorithm spends about 1045.83 seconds for learning, while the proposed OFS algorithm takes only 1.08 seconds, which is almost 1000 times faster than the mRMR algorithm.

4.5 Experiment V: Evaluation on Large-scale Data

In this section, we evaluate the performance of the proposed OFS algorithms on large data sets that contain at least 100,000 instances. The statistics of these data sets are shown in Table 8. All the experiments were implemented in Matlab and run on a regular PC.

Table 9 shows the experimental results of the average numbers of mistakes achieved by three different algorithms. From the results, it is clear that the proposed OFS algorithm significantly outperforms the other two baselines. Figure 6 shows how online predictive performance of different algorithms varies over the iteration. We can observe that the advantage of the proposed OFS algorithm with more and more training examples received.

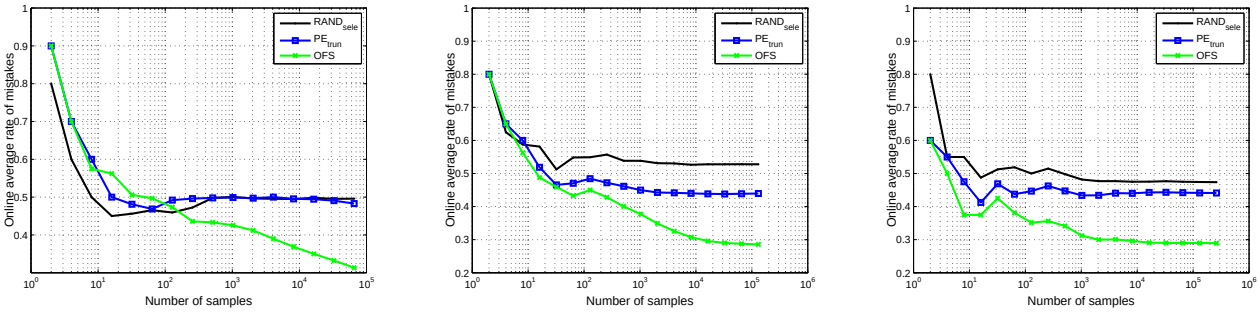
Figure 7 shows the running time of various fractions of selected features of the proposed algorithms. We first observe that all the three online algorithms are very efficient, requiring only a few minutes in solving the online

TABLE 8
The list of large datasets used in our experiments.

Dataset	Number	Dimension
KDDCUP08	102294	117
ijcnn1	141691	22
codrna	271617	8
covtype	581012	54
KDDCUP99	1131571	127

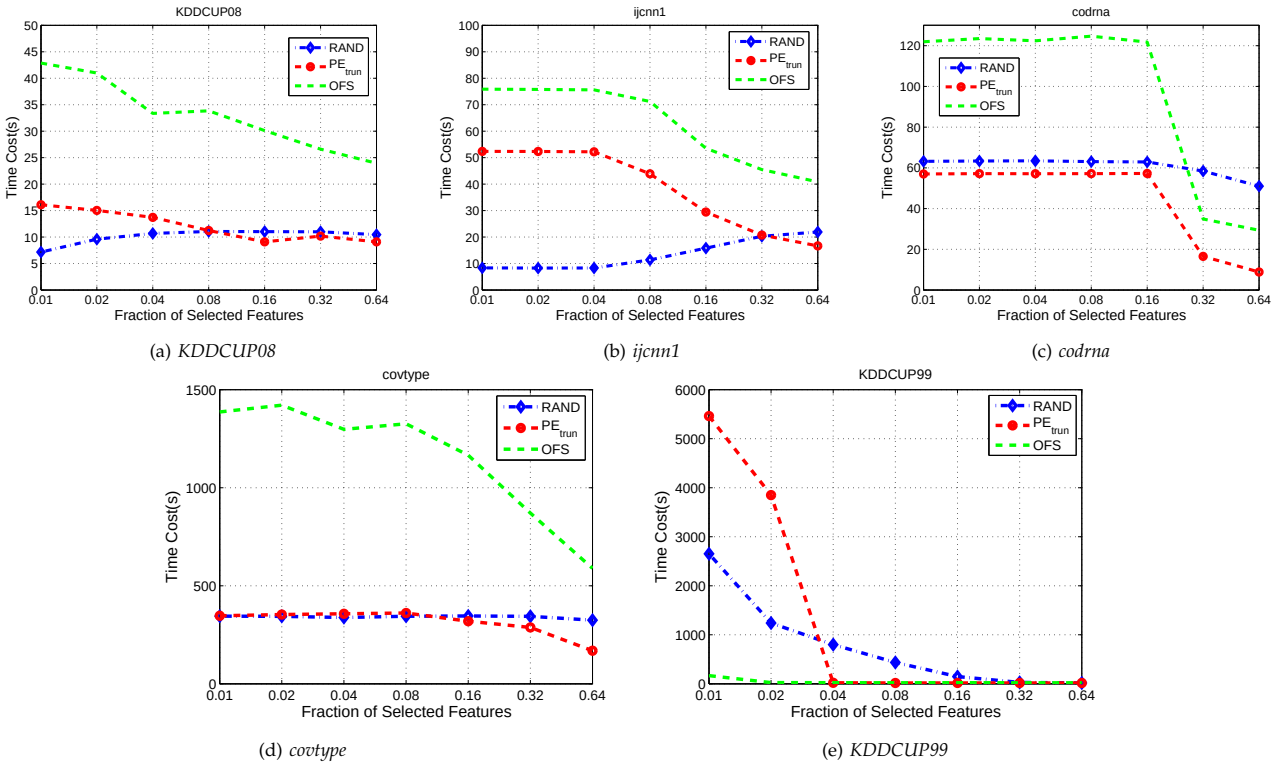
TABLE 9
Evaluation of the average number of mistakes by three algorithms on the big data sets.

Algorithm	KDDCUP08	ijcnn1
RAND	50718.4 ± 210.2	74778.4 ± 159.9
PE_{trun}	48714.4 ± 2654.8	62282.6 ± 188.7
OFS	**31155.6 ± 2733.1**	**40512.8 ± 257.0**
Algorithm	codrna	covtype
RAND	128709.0 ± 284.2	277394.5 ±127.0
PE_{trun}	119838.4 ±165.8	198661.6± 125.6
OFS	**78685.8 ±42.8**	**148223.1 ± 193.0**
Algorithm	KDDCUP99	N.A
RAND	53614.6 ± 3567.9	N.A
PE_{trun}	213.0 ±10.5	N.A
OFS	**98.3 ±4.6**	N.A



(a) *KDDCUP08* (b) *ijcnn1* (c) *codrna*

Fig. 6. Performance evaluation of online feature selection in the entire online learning process on three big datasets.



(a) *KDDCUP08* (b) *ijcnn1* (c) *codrna*

(d) *covtype* (e) *KDDCUP99*

Fig. 7. Time cost under various fractions of selected features of online feature selection in big datasets.

feature selection tasks for all the large-scale data sets. It is interesting to observe that the runtime of the proposed OFS algorithm generally decreases as the number of selected features increases, which seems to be counter-intuitive as we tend to observe a longer running time with increasing number of selected features. (see Figure 5). This is primarily because the main time consuming part of the proposed OFS algorithm lies in the online updating part; when the number of selected features increases, the learner becomes more accurate and thus requires less number of updates. This also explain why the proposed OFS algorithm can even run faster than the other baselines on some data sets. All these encouraging results again validate the efficacy and potential of the proposed OFS method for mining large-scale data sets in the era of big data.

5 Conclusions

This paper investigated a new research problem, **Online Feature Selection** (OFS), which aims to select a small and fixed number of features for binary classification in an online learning fashion. In particular, we addressed two kinds of OFS tasks in two different settings: (i) OFS by learning with full inputs of all the dimensions/attributes, and (ii) OFS by learning with partial inputs of the attributes. We presented a family of novel OFS algorithms to solve each of the OFS tasks, and offered theoretical analysis on the mistake bounds of the proposed OFS algorithms. We extensively examined their empirical performance and applied the proposed techniques to solve two real-world applications: image classification in computer vision and microarray gene expression analysis in bioinformatics. The encouraging

results show the proposed algorithms are fairly effective for feature selection tasks of online applications, and significantly more efficient and scalable than some state-of-the-art batch feature selection technique. Future work could extend our framework to other settings, e.g., online multi-class classification and regression problems, or to help tackle other emerging online learning tasks, such as online transfer learning [47] or online AUC maximization [50].

ACKNOWLEDGEMENTS

This work is in part supported by Singapore MOE Academic tier-1 grant (RG33/11), Singapore NRF under its Interactive & Digital Media (IDM) Program administrated by Media Development Authority (Research Grant No. MDA/IDM/2012/8/8-2VOL01), Microsoft Research grant, and Office of Navy Research (ONR Award N000141210431 and N00014-09-1-0663).

REFERENCES

[1] R. Bekkerman, R. El-Yaniv, N. Tishby, and Y. Winter. Distributional word clusters vs. words for text categorization. *Journal of Machine Learning Research*, 3:1183–1208, 2003.

[2] J. Bi, K. P. Bennett, M. J. Embrechts, C. M. Breneman, and M. Song. Dimensionality reduction via sparse support vector machines. *Journal of Machine Learning Research*, 3:1229–1243, 2003.

[3] G. Cavallanti, N. Cesa-Bianchi, and C. Gentile. Tracking the best hyperplane with a simple budget perceptron. *Machine Learning*, 69(2-3):143–167, 2007.

[4] N. Cesa-Bianchi, S. Shalev-Shwartz, and O. Shamir. Efficient learning with partially observed attributes. *Journal of Machine Learning Research*, pages 2857–2878, 2011.

[5] A. B. Chan, N. Vasconcelos, and G. R. G. Lanckriet. Direct convex relaxations of sparse svm. In *ICML*, pages 145–153, 2007.

[6] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. *J. Mach. Learn. Res. (JMLR)*, 7:551–585, 2006.

[7] K. Crammer, M. Dredze, and F. Pereira. Exact convex confidence-weighted learning. In *NIPS*, pages 345–352, 2008.

[8] K. Crammer, A. Kulesza, and M. Dredze. Adaptive regularization of weight vectors. In *NIPS*, pages 414–422, 2009.

[9] M. Dash and V. Gopalkrishnan. Distance based feature selection for clustering microarray data. In *DASFAA*, pages 512–519, 2008.

[10] M. Dash and H. Liu. Feature selection for classification. *Intell. Data Anal.*, 1(1-4):131–156, 1997.

[11] O. Dekel, S. Shalev-Shwartz, and Y. Singer. The forgetron: A kernel-based perceptron on a budget. *SIAM J. Comput.*, 37(5):1342–1372, 2008.

[12] C. H. Q. Ding and H. Peng. Minimum redundancy feature selection from microarray gene expression data. *J. Bioinformatics and Computational Biology*, 3(2):185–206, 2005.

[13] D. Donoho. Compressed sensing. *IEEE Trans. on Information Theory*, 52(4):1289 – 1306, 2006.

[14] M. Dredze, K. Crammer, and F. Pereira. Confidence-weighted linear classification. In *ICML*, pages 264–271, 2008.

[15] J. Duchi and Y. Singer. Efficient online and batch learning using forward backward splitting. *J. Mach. Learn. Res.*, 10:2899–2934, 2009.

[16] Y. Freund and R. E. Schapire. Large margin classification using the perceptron algorithm. *Machine Learning*, 37(3):277–296, 1999.

[17] C. Gentile. A new approximate maximal margin classification algorithm. *J. Mach. Learn. Res. (JMLR)*, 2:213–242, 2001.

[18] K. A. Glocer, D. Eads, and J. Theiler. Online feature selection for pixel classification. In *ICML*, pages 249–256, 2005.

[19] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[20] X. He, D. Cai, and P. Niyogi. Laplacian score for feature selection. In *NIPS*, 2005.

[21] S. C. H. Hoi, R. Jin, P. Zhao, and T. Yang. Online multiple kernel classification. *Machine Learning*, 90(2):289–316, 2013.

[22] S. C. H. Hoi, J. Wang, and P. Zhao. *LIBOL: A Library for Online Learning Algorithms*. Nanyang Technological University, 2012.

[23] S. C. H. Hoi, J. Wang, P. Zhao, and R. Jin. Online feature selection for mining big data. In *Proceedings of the 1st International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications*, BigMine '12, pages 93–100, New York, NY, USA, 2012. ACM.

[24] J. Kivinen, A. J. Smola, and R. C. Williamson. Online learning with kernels. In *Advances in Neural Information Processing Systems*, pages 785–792, 2001.

[25] R. Kohavi and G. H. John. Wrappers for feature subset selection. *Artif. Intell.*, 97(1-2):273–324, 1997.

[26] A. Krizhevsky. Learning multiple layers of features from tiny images. *Technical Report, University of Toronto*, 2009.

[27] J. Langford, L. Li, and T. Zhang. Sparse online learning via truncated gradient. *J. Mach. Learn. Res.*, 10:777–801, 2009.

[28] H. Liu and L. Yu. Toward integrating feature selection algorithms for classification and clustering. *IEEE Trans. Knowl. Data Eng.*, 17(4):491–502, 2005.

[29] Z. Ma, Y. Yang, F. Nie, J. R. R. Uijlings, and N. Sebe. Exploiting the entire feature space with sparsity for automatic image annotation. In *ACM Multimedia*, pages 283–292, 2011.

[30] J. G. March. Exploration and Exploitation in Organizational Learning. *Organization Science*, 2(1):71–87, 1991.

[31] F. Orabona, J. Keshet, and B. Caputo. The projectron: a bounded kernel-based perceptron. In *Proceedings of International Conference on Machine Learning*, pages 720–727, 2008.

[32] H. Peng, F. Long, and C. H. Q. Ding. Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(8):1226–1238, 2005.

[33] S. Perkins and J. Theiler. Online feature selection using grafting. In *ICML*, pages 592–599, 2003.

[34] J. Ren, Z. Qiu, W. Fan, H. Cheng, and P. S. Yu. Forward semi-supervised feature selection. In *PAKDD*, pages 970–976, 2008.

[35] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–407, 1958.

[36] A. Rostamizadeh, A. Agarwal, and P. L. Bartlett. Learning with missing features. In *UAI*, pages 635–642, 2011.

[37] Y. Saeys, I. Inza, and P. Larrañaga. A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19):2507–2517, 2007.

[38] U.Alon, N.Barkai, D.A.Notterman, K.Gish, S.Ybarra, D.Mack, and A.J.Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences*, pages 6745–6750, 1999.

[39] J. Wang, P. Zhao, and S. C. H. Hoi. Cost-sensitive online classification. In *ICDM*, pages 1140–1145, 2012.

[40] M. K. Warmuth and D. Kuzmin. Randomized pca algorithms with regret bounds that are logarithmic in the dimension. In *Advances in Neural Information Processing Systems 19 (NIPS 06)*, 2006.

[41] X. Wu, K. Yu, H. Wang, and W. Ding. Online streaming feature selection. In *ICML*, pages 1159–1166, 2010.

[42] Z. Xu, R. Jin, J. Ye, M. R. Lyu, and I. King. Non-monotonic feature selection. In *ICML*, page 144, 2009.

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

[44] S. Yang, L. Yuan, Y.-C. Lai, X. Shen, P. Wonka, and J. Ye. Feature grouping and selection over an undirected graph. In *KDD*, pages 922–930, 2012.

[45] Y. Yang, H. T. Shen, Z. Ma, Z. Huang, and X. Zhou. $l_{2,1}$-norm regularized discriminative feature selection for unsupervised learning. In *IJCAI*, pages 1589–1594, 2011.

[46] L. Yu and H. Liu. Feature selection for high-dimensional data: A fast correlation-based filter solution. In *ICML*, pages 856–863, 2003.

[47] P. Zhao and S. C. H. Hoi. Otl: A framework of online transfer learning. In *ICML*, pages 1231–1238, 2010.

[48] P. Zhao and S. C. H. Hoi. Bduol: Double updating online learning on a fixed budget. In *ECML/PKDD*, number 1, pages 810–826, 2012.

[49] P. Zhao, S. C. H. Hoi, and R. Jin. Double updating online learning. *Journal of Machine Learning Research*, 12:1587–1615, 2011.
[50] P. Zhao, S. C. H. Hoi, R. Jin, and T. Yang. Online auc maximization. In *ICML*, pages 233–240, 2011.
[51] P. Zhao, J. Wang, P. Wu, R. Jin, and S. C. H. Hoi. Fast bounded online gradient descent algorithms for scalable kernel-based online learning. In *ICML*, 2012.
[52] Z. Zhao and H. Liu. Semi-supervised feature selection via spectral analysis. In *SDM*, 2007.
[53] Z. Zhao and H. Liu. Spectral feature selection for supervised and unsupervised learning. In *ICML*, pages 1151–1157, 2007.
[54] Y. Zhou, R. Jin, and S. C. H. Hoi. Exclusive lasso for multi-task feature selection. *Journal of Machine Learning Research - Proceedings Track*, 9:988–995, 2010.

Jialei Wang is currently a Research Assistant in the School of Computer Engineering at the Nanyang Technological University, Singapore. He received his bachelor degree from University of Science and Technology of China, Hefei, P.R. China, in 2011. His research interests are machine learning, statistics, and optimization with applications to data mining, computer vision and natural language processing.

Peilin Zhao is currently a PhD candidate in the School of Computer Engineering at the Nanyang Technological University, Singapore. He received his bachelor degree from Zhejiang University, Hangzhou, P.R. China, in 2008. His research interests are statistical machine learning, and data mining.

Steven C. H. Hoi is an Assistant Professor of the School of Computer Engineering at Nanyang Technological University, Singapore. He received his Bachelor degree from Tsinghua University, P.R. China, in 2002, and his Ph.D degree in computer science and engineering from The Chinese University of Hong Kong, in 2006. His research interests are machine learning and data mining and their applications to multimedia information retrieval (image and video retrieval), social media and web mining, and computational finance, etc. He has published over 100 referred papers in top conferences and journals in related areas. He has served as general co-chair for ACM SIGMM Workshops on Social Media (WSM'09, WSM'10, WSM'11), program co-chair for the fourth Asian Conference on Machine Learning (ACML'12), book editor for "Social Media Modeling and Computing", guest editor for ACM Transactions on Intelligent Systems and Technology (ACM TIST), technical PC member for many international conferences, and external reviewer for many top journals and worldwide funding agencies, including NSF in US and RGC in Hong Kong. He is a member of IEEE and ACM.

Rong Jin is currently an Associate Professor in the department of Computer Science and Engineering at Michigan State University. He received his Ph.D. degree in Computer Science from Carnegie Mellon University, 2003. His research interests are statistical learning and its application to large-scale information management, including web text retrieval, content-based image retrieval, gene regulatory network reconstruction, neuron data analysis, and visual object recognition.