Online Streaming Feature Selection

Xindong Wu [1, 2] XWU@CS.UVM.EDU
Kui Yu [1] YKUI713@GMAIL.COM
Hao Wang [1] JSJXWANGH@HFUT.EDU.CN
[1] Department of Computer Science, Hefei University of Technology, 193 Tunxi Road, Hefei, Anhui 230009, China
[2] Department of Computer Science, University of Vermont, 351 Votey Hall, Burlington, VT 05405, USA
Wei Ding [3] DING@CS.UMB.EDU
[3] Department of Computer Science, University of Massachusetts Boston, 100 Morrissey Blvd., Boston, MA 02125, USA

Abstract

We study an interesting and challenging problem, online streaming feature selection, in which the size of the feature set is unknown, and not all features are available for learning while leaving the number of observations constant. In this problem, the candidate features arrive one at a time, and the learner's task is to select a "best so far" set of features from streaming features. Standard feature selection methods cannot perform well in this scenario. Thus, we present a novel framework based on feature relevance. Under this framework, a promising alternative method, Online Streaming Feature Selection (OSFS), is presented to online select strongly relevant and non-redundant features. In addition to OSFS, a faster Fast-OSFS algorithm is proposed to further improve the selection efficiency. Experimental results show that our algorithms achieve more compactness and better accuracy than existing streaming feature selection algorithms on various datasets.

1. Introduction

Feature selection for predictive modeling has received considerable attention during the last three decades both in statistics and in machine learning. A great variety of feature selection algorithms have been developed and proven to be effective in improving predictive accuracy for classification (Kohavi & John 1997; Guyon & Elisseeffm 2003; Loscalzo et al 2009; Aliferis et al 2010). Standard feature selection methods assume that all candidate features are available and presented to a learner before feature selection takes place.

Appearing in *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel, 2010.

In this paper, another interesting scenario is taken into account where the candidate feature set size is unknown, or even infinite instead of all candidate features being known in advance. In this problem, the candidate features are generated dynamically and arrive one at a time while the number of observations is left constant. This scenario is called streaming feature selection which has practical use in many settings. For example, texture-based image segmentation assigns a label to each pixel in a training image according to its texture type, and an image might easily contain tens of thousands of labeled pixels, hence the computational cost is expensive in generating those features. It is not practical to wait until all features have been generated before learning begins, thus it could be far more preferable to generate candidate features one at a time (Perkins & Theiler 2003). Therefore, streaming feature selection seeks to select a minimal yet good set of features from the features generated so far.

Although many standard feature selection algorithms are effective in selecting a subset of predictive features for various classification problems, they are not necessarily reliable to deal with steaming features. In this paper, we present a novel framework for selection of features from a feature stream. Our work is inspired by feature relevance and feature redundancy. The unique contributions that distinguish our work from existing approaches are threefold: (1) our work goes a step further on feature relevance and explicitly expresses feature redundancy between a feature and a target class; (2) a novel framework based on feature relevance is proposed to manage streaming feature selection; and (3) two new online streaming feature selection algorithms are designed with comparative studies.

2. Related Work

For many years, feature selection, as an effective means to deal with large dimensionality, has been generally viewed as a problem of searching for an optimal subset of features. In principle, feature selection methods can be

broadly classified into three categories: filter, wrapper, and embedded methods. The filter method is independent of any classifiers, and applies evaluation measures such as distance, information, dependency and consistency to select features and then later build a classifier using the selected features (Dash & Liu 2003). The wrapper method performs heuristic search in the space of all possible feature subsets, using a classifier of choice to assess each subset (Kohavi & John 1997). Meanwhile, the embedded method attempts to simultaneously maximize classification performance and minimize the number of features used (Tibshirani 1996).

All work discussed above assumes that all candidate features are available from the beginning and pays little attention to candidate feature sets of unknown, or even infinite size, that is, the problem of streaming feature selection.

Two major lines of recent research efforts have tried to address this problem. Perkins and Theiler (2003) proposed a grafting algorithm based on a stagewise gradient descent approach for streaming feature selection. However, grafting requires all candidate features in advance to determine the value of the tuning parameter λ using cross-validation before learning. Thus, grafting is a quasi-streaming feature selection method. Zhou et al. (2005; 2006) presented two algorithms based on streamwise regression, Information-investing and Alpha-investing for streaming feature selection. Since Information-investing gave extremely similar results as Alpha-investing and Alpha-investing was emphasized in their work, we adopt Alpha-investing in this paper. Alpha-investing uses a p-value to determine whether a new feature should be added to the learning model or not, and a linear regression is used to evaluate the modified model. This method needs some prior knowledge about the structure of the feature space to heuristically control the choice of candidate feature selection. However, Alpha-investing might not provide good performance on the original streaming features. In the real world, it is difficult to obtain sufficient prior information about the structure of the candidate features with a feature stream. Thus, more efforts are needed in order to manage real-world feature streams without any transformations of the original features in advance.

Therefore, our work takes a paradigm shift from the above research efforts and proposes a novel framework which is clearly different from previous work on streaming feature selection. Under this framework, a novel Online Streaming Feature Selection algorithm (OSFS) is presented in this paper. In addition to OSFS, a faster Fast-OSFS algorithm is proposed to further improve the selection efficiency.

3. A Framework for Streaming Feature Selection

In this section, we first review notions of feature relevance. Then we redefine feature redundancy and propose a novel framework for streaming feature selection based on feature relevance.

3.1 Notations and Definitions

Koller et al. proposed a classification of input features X with respect to their relevance to a target T in terms of conditional independence (Koller & Sahami 1996; Kohavi & John 1997). They classified features into three disjoint categories, namely, strongly relevant, weakly relevant and irrelevant features. In the following definitions, let V be a full set of features, X_i denote the ith input feature, and $X_{\setminus i}$ represent all input features excluding X_i.

Definition 1 (Conditional independence) In a feature set V, two features X and Y are conditionally independent given the set of features Z, if and only if

$P(X|Y,Z)=P(X|Z)$, denoted as $Ind(X,Y|Z)$.

Accordingly, for notational convenience we denote conditional dependence as $Dep(X,Y|Z)$.

Definition 2 (Strong relevance) A feature X_i is strongly relevant to a target T if

$$P(T \mid X_{\setminus i}) \neq P(T \mid X_{\setminus i}, X_i)$$

Definition 3 (Weak relevance) A feature X_i is weakly relevant to a target T if X_i is not strongly relevant and

$$\exists S \subseteq X_{\setminus i} : P(T \mid S) \neq P(T \mid S, X_i)$$

Definition 4 (Irrelevance) A feature X_i is irrelevant to a target T if it is neither strongly nor weakly relevant, that is, if

$$\forall S \subseteq X_{\setminus i} : P(T \mid S) = P(T \mid S, X_i)$$

Yu et al. further studied the feature relevance and pointed out that the weakly relevant feature set could be classified into redundant features and non-redundant features. They gave a definition of feature redundancy based on a Markov blanket criterion (Yu & Liu 2004; Tuv et al 2009).

Definition 5 (Markov blanket) Given a feature X_i, assuming $M_i \in V$, $X_i \notin M_i$, M_i is said to be a Markov blanket for X_i, if and only if

$$P(V - M_i - \{X_i\}, T \mid X_i, M_i) = P(V - M_i - \{X_i\}, T \mid M_i)$$

Definition 6 (Redundant feature-1) Let V be the current set of features. A feature is redundant and hence should be removed from V, if and only if it is weakly relevant and has a Markov blanket M_i within V.

According to Definitions 1 and 5, Definition 6 can be rewritten using conditional independence and we attain Definition 7 which expresses the relation of redundancy between a target feature T and a feature X more explicitly.

Definition 7 (Redundant feature-2) Given a candidate Markov blanket of a target feature T, denoted as CMB(T), and a feature X∈CMB(T), X is said to be redundant to T, if and only if

$$\exists S \subseteq CMB(T) : P(T \mid X, S) = P(T \mid S) \qquad (1)$$

Proof: => According to Definition 5 and the term X∈ CMB(T), X is relevant to T. Because X is redundant to T, there must exist a subset S within CMB(T) which subsumes all of the information that X has about T. That is to say, X and T are conditionally independent given the subset S. Then we get formula (1).

<= According to formula (1) and the term X∈CMB(T), the subset S carries all information that X has about T. Thus, X is redundant to T.□

3.2 A Framework for Streaming Feature Selection

Based on the definitions above, an entire feature set is divided into four basic disjoint parts: (1) irrelevant features, (2) redundant features (part of weakly relevant features), (3) weakly relevant but non-redundant features, and (4) strongly relevant features. An optimal subset contains non-redundant and strongly relevant features.

Since the global information of all candidate features is unknown and the features are generated continuously, it is difficult to find all strongly relevant and non-redundant features from streaming features. Our task is to design an efficient and effective way to find an optimal subset from streaming features. Searching for an optimal subset based on the definitions of feature relevance and redundancy is combinatorial in nature. Our novel framework is designed as follows based on online feature relevance and redundancy analysis.

1. **Initialization**
   Best candidate feature set BCF={}, the target feature T
2. **Online relevance analysis**
   (1) Generate a new feature X
   (2) Determine whether X is irrelevant to T or not.
      a. If X is irrelevant to T, then disregarded;
      b. Otherwise, X is added to BCF
3. **Online Redundancy analysis**
   Online identify redundant features from the current subset BCF and remove them by Definition 7.
4. Alternate steps 2 and 3 until the stopping criteria are satisfied.
5. Output the subset BCF.

Figure 1 **A framework for streaming feature selection**

The framework described in Figure 1 is composed of two steps: first, online relevance analysis determines a new feature with respect to its relevance to the target T and removes irrelevant ones; and second, online redundancy analysis eliminates redundant features from the features selected so far. The two steps are alternated till some stopping criteria are satisfied.

4. Algorithms and Their Analysis

Under the above framework for stream feature selection, two novel algorithms are presented, OSFS and Fast-OSFS, and an algorithm analysis is given in this section.

4.1 An Online Streaming Feature Selection Algorithm

The pseudo-code of our online streaming feature selection (OSFS for short) algorithm is shown in Figure 2.

The OSFS algorithm

Input: class label C, feature stream X
Output: BCF (the best candidate features so far)

```
1.  BCF={}
2.  i=1
3.  repeat
4.  /*online relevance analysis*/
5.     added=0
6.  /*generating new features*/
7.     Xi←get_next_feature()
8.  if Dep(Xi, C | φ)
9.  /*add relevant feature Xi to BCF*/
10.       BCF = BCF ∪ Xi
11.       added=1
12.    endif
13. /*online redundancy analysis*/
14.    if (added)
15.    for each feature Y ∈ BCF
16.    if ∃S ⊆ BCF \ Y, s.t. Ind(Y, C | S)
17. /*remove redundant feature */
18.    BCF = BCF-Y
19.    endif
20. endfor
21. endif
22. i=i+1
23. until the stopping criteria satisfied
24. output BCF
```

Figure 2 **The OSFS algorithm**

OSFS finds an optimal subset using a two-phase scheme: online relevance analysis (steps 4 - 12) and online redundancy analysis (steps 13 - 21). In the relevance analysis phase, OSFS discovers strongly and weakly relevant features and adds them into BCF. When a new feature arrives, OSFS assesses whether it is irrelevant to the class label C; if so, it is discarded, otherwise it is added to BCF.

If a new feature enters BCF, the redundancy analysis phase is performed. In this phase, OSFS dynamically eliminates redundant features in the subset of the features selected so far. If there exists a subset within BCF to make Y and C conditionally independent, Y is removed from BCF. OSFS alternates the two phases till some stopping criteria are satisfied.

4.2 An Analysis of the OSFS Algorithm

With the OSFS algorithm above, under the assumption that all statistical independence tests are reliable, let us have an analysis about its performance in theory.

Firstly, we can analyze its performance on a small data set with hundreds of thousands of features.

When the data set is so small in size so as to make most of the conditional independence tests unreliable, OSFS might fail. There are two lines in the pseudo-code for conditional independence tests. One is at line 8, and the other is at line 16. OSFS doesn't fail at line 8, since the

conditioning set is an empty set. But OSFS might fail at line 16, when the conditioning set S exponentially grows. However, in the relevance analysis phase, only strongly and weakly relevant features are admitted into BCF. In the redundancy analysis phase OSFS dynamically evaluates each feature within BCF and removes redundant features from BCF when a feature enters into BCF. With many irrelevant and redundant features, BCF keeps as minimal as possible so that conditioning on all subsets of BCF is feasible. Thus, OSFS can deal with a dataset with a small sample-to-variable ratio.

Secondly, we can analyze whether or not the OSFS algorithm can discover all strongly relevant and some non-redundant features.

According to Definition 4, if a feature X is irrelevant to C, X must be discarded in the relevance analysis phase. Thus, from Definitions 2 and 3, all strongly and some weakly relevant features will enter BCF at line 10. According to Definition 7, if X is a strongly relevant feature, there doesn't exist a subset S within BCF to satisfy the term Ind(X,C|S). X cannot be removed in any phase. Thus, OSFS can find all strongly relevant features.

For a redundant feature, one situation is that the size of streaming feature set is unknown, but finite. According to Definition 3, some weakly relevant features, including redundant features and non-redundant features, will enter BCF in the relevance analysis phase. Therefore, OSFS needs to remove redundant features from those weakly relevant features. In the redundancy analysis phase, based on Definition 7, OSFS searches a subset S for each feature within BCF to make it redundant to C. For example, if the term Dep(X,C|S) is satisfied at line 8 where S is an empty set, X will be added into BCF as a relevant feature at line 10. Now assuming that X is redundant to C, as the time goes on, the subset S within BCF must be found in the redundancy analysis phase, and satisfy Ind(X,C|S) according to Definition 7. Then X is removed from BCF at line 18. The other situation is that if the size of the streaming feature set is infinite, OSFS could fail to remove X at line 13 at time t. Because a feature is generated randomly, OSFS doesn't know when S can be found within BCF. Thus, we don't know when OSFS can remove X at line 18. But in theory, since OSFS can find all strongly relevant features, if X is a redundant feature within BCF, there must exist an S to satisfy Ind(X,C|S). Therefore, OSFS can discover all strongly relevant features and some non-redundant features.

Finally, the complexity of OSFS depends on the number of independent tests. At time t, assuming V features are arriving, then the worst-case complexity is $O(|V||BCF|k^{|BCF|})$ where k is the maximum allowable size that a conditioning set may grow. Assuming $SF \subseteq V, |SF| << |V|$ where SF contains all of strongly relevant features, then the average time complexity is $O(|SF||BCF|k^{|BCF|})$ at time t.

4.3 The Fast-OSFS Algorithm

According to the above analysis, the most time-consuming part of OSFS is the redundancy analysis phase. When a new feature enters BCF, the redundancy analysis phase will re-examine each feature of BCF with respect to its relevance to C. Therefore, in order to further improve the selection efficiency, Fast-OSFS is designed in Figure 3.

The Fast-OSFS algorithm

Input: class labels C, feature stream X Output: BCF (the candidate best features)

1. BCF ={}
2. i=1
3. repeat
4. /* online relevance analysis*/
5. added=0
6. /*generating new features*/
7. X_i←get_next_feature()
8. if $Dep(X_i, C \mid \phi)$
9. /*add relevant feature X_i to BCF*/
10. BCF = BCF ∪ X_i
11. added=1
12. endif
13. /*the inner-online redundancy analysis*/
14. if (added)
15. if $\exists S \subseteq BCF \setminus X_i, s.t.\ Ind(X_i, C \mid S)$
16. /*remove redundant feature X_i */
17. BCF= BCF-X_i
18. endif
19. endif
20. i=i+1
21. until the stopping criteria satisfied
22. /*the outer-online redundancy analysis*/
23. for each feature $Y \in$ BCF
24. if $\exists S \subseteq BCF \setminus Y, s.t.\ Ind(Y, C \mid S)$
25. BCF = BCF-Y
26. endif
27. endfor
28. output BCF

Figure 3 The Fast-OSFS algorithm

The key difference between Fast-OSFS and OSFS is that Fast-OSFS divides the redundancy analysis phase into two phases, inner-redundancy analysis and outer-redundancy analysis. Fast-OSFS only alternates the relevance analysis and the inner-redundancy analysis phase. In the inner-redundancy analysis phase Fast-OSFS only re-examines the feature just added into BCF, whereas the outer-redundancy analysis phase re-examines each feature of BCF only when the process of generating a feature is stopped. The worst-case complexity is $O(|V|k^{|BCF|}+|BCF|k^{|BCF|})$ and the average is $O(|SF|k^{|BCF|}+|BCF|k^{|BCF|})$ at time t. Thus, Fast-OSFS is more efficient.

5. Experimental Results

In order to have a comprehensive comparison of existing streaming feature selection methods on various data sets with our algorithms, we apply these algorithms in traditional feature selection settings, that is, those of fixed features, but the features arrive one at a time in a random order to simulate the situation of streaming features.

Our data sets include 8 UCI benchmark databases and 10 challenge databases. We used three classifiers, *k*-nn, J48 and Randomforest (Spider 2010), and selected the best accuracy as the result. The experiments were conducted on a computer with Windows XP, 2.6GHz CPU and 2GB memory. Grafting and Alpha-investing were performed using their original implementations. The tuning parameter λ for Grafting was selected using cross-validation and the parameters of Alpha-investing used default settings, W_0=0.5 and α_Δ=0.5. The conditional

independence tests in our implementation are G^2 tests and the parameter alpha is the statistical significance level.

5.1 Results on UCI Benchmark Data Sets

8 data sets in the traditional form are selected from the UCI Machine Learning Repository. These data sets, including their problem type and numbers of features and instances, are shown in Table 1. Either the original test data or 10-fold cross validation is used in these datasets.

Table 1. Summary of UCI benchmark data sets

DATASET	DOMAIN	FEATURES	INSTANCES
SPECT	MEDICINE	22	267
SPECTF	MEDICINE	44	267
WDBC	MEDICINE	30	569
IONOSPHERE	RADAR DATA	34	351
SPAMBASE	SPAM E-MAIL	57	4601
INFANT MORTALITY	MEDICINE	86	5337
BANKRUPTY	FINANCIAL	147	7063
SYLVA	ECOLOGY	216	14374

Two measurements for solving the feature selection problem are compactness (the proportion of selected features) and predictive accuracy (%). A maximally compact method which cannot achieve a good predictive accuracy doesn't solve our feature selection problem. Therefore, Figure 4 reports the compactness and predictive accuracy by 4 algorithms where the value of alpha is up to 0.01. The best possible mark for each graph is at the upper left corner, which selects the fewest features with the best accuracy. According to Figure 4, we analyze the experimental results as follows.

(1) Our algorithms vs the Alpha-investing algorithm. On 7 out of the 8 data sets, our algorithms achieve more compact and higher accurate results than Alpha-investing where Alpha-investing selects almost 80 percent of the features on the spambase data and all features on the wdbc data. On the bankrupty dataset, although a little lower in accuracy than Alpha-investing, our algorithms achieve more compact results.

(2) Our algorithms vs the Grafting algorithm. Our algorithms outperform Grafting on 6 out of the 8 data sets on both compactness and accuracy. On the bankrupty and ionosphere data sets our algorithms are competitive with Grafting.

(3) Grafting vs Alpha-investing. Grafting is more compact than Alpha-investing on all data sets, and its accuracy is higher on 50% of the datasets.

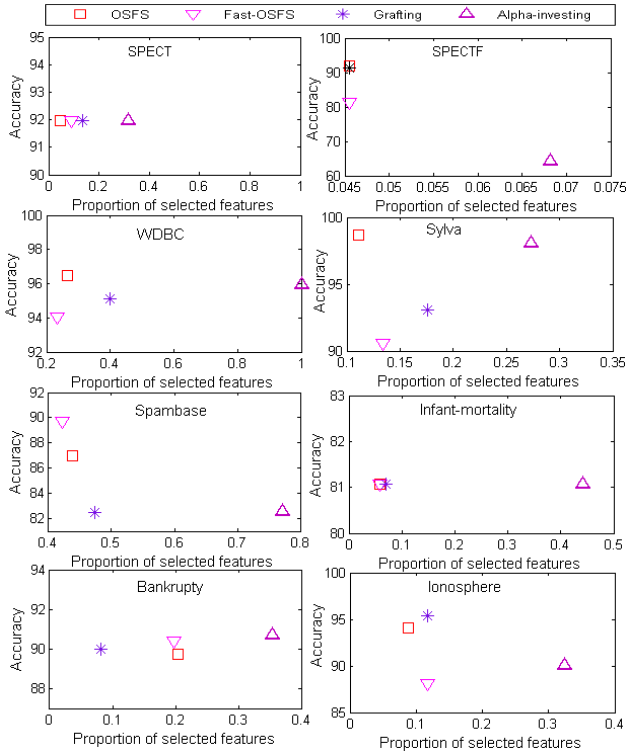

Figure 4 The compactness and predictive accuracy of 4 algorithms (alpha=0.01)

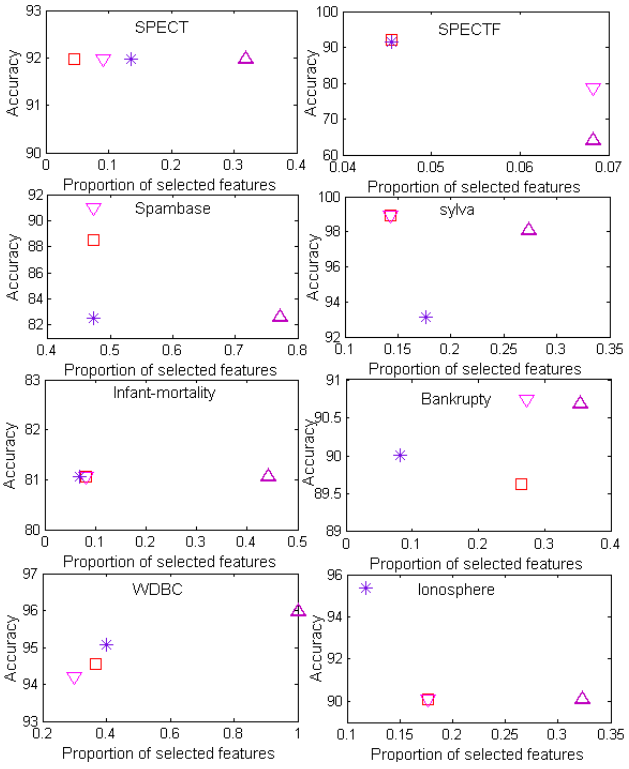

Figure 5 The compactness and predictive accuracy of 4 algorithms (alpha=0.05)

We can conclude that our algorithms have achieved more compactness and higher accuracy than the two state-of-the-art algorithms.

From Figure 5, with alpha up to 0.05, our algorithms also outperform Alpha-investing on 7 out of the 8 datasets on both compactness and accuracy. On the wdbc data, Alpha-investing has the best accuracy than the other algorithms, but it selected all features.

Compared with Grafting, our algorithms outperform on 4 out of the 8 datasets on both compactness and accuracy. On the Ionosphere dataset, Grafting achieves more compactness and higher accuracy than the others. On the remaining three datasets, our algorithms are competitive with Grafting.

Finally, we give an analysis of the performance of our two algorithms with different values of alpha, as shown in Figures 6 and 7. When alpha is up to 0.05, our two algorithms tend to select more features, but the performance of the two algorithms is different. OSFS degrades a little while Fast-OSFS improves a little. In summary, when alpha is equal to 0.01 and when alpha is up to 0.05, two algorithms have similar performance in our experiments.

5.2 Results on Challenge Data Sets

On UCI data, we used datasets with no more than 300 features to simulate the situation of streaming features. In this section, we further assess our algorithms on 10 public challenge data sets with tens of thousands of features, as shown in Table 2.

Table 2. Summary of challenge datasets

DATASET	DOMAIN	FEATURES	INSTANCES
LYMPHOMA	GENE	7399	227
OVARIAN-CANCER	PROTEOMICS	2190	216
BREAST-CANCER	GENE	17816	286
HIVA	DRUG	1617	4229
NOVA	TEXT	16969	1929
MANELON	SYNTHETIC	500	2000
ARCENE	CLINICAL	10000	100
DEXTER	TEXT	20000	300
DOROHTHEA	DRUG	100000	800
SIDO0	GENOMICS	4932	12768

Ovarian_cancer and Breast-cancer are bio-medical datasets (Conrads et al 2004; Wang et al 2005). Hiva, Nova and Sido0 are from the WCCI 2006 and WCCI 2008 Performance Prediction Challenges, respectively. The other datasets are from the NIPS 2003 feature selection challenge. Ovarian_cancer, Breast-cancer, Lymphoma and Sido0 used 10-fold cross validations; and the NIPS 2003 challenge data sets used their original training and validation sets.

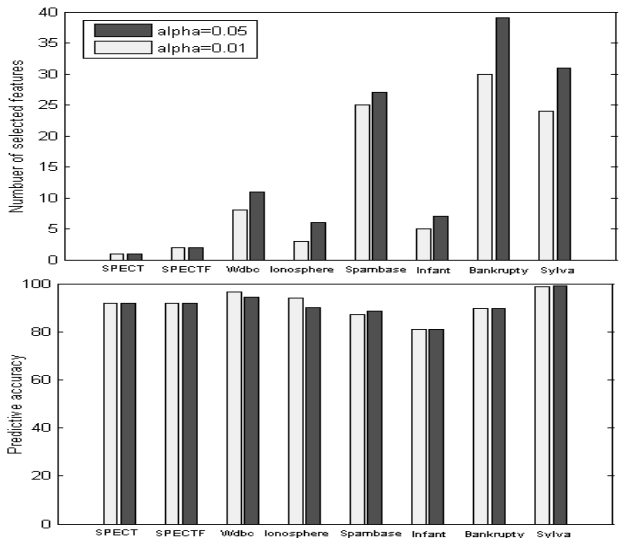

Figure 6 OSFS performance with different alpha values

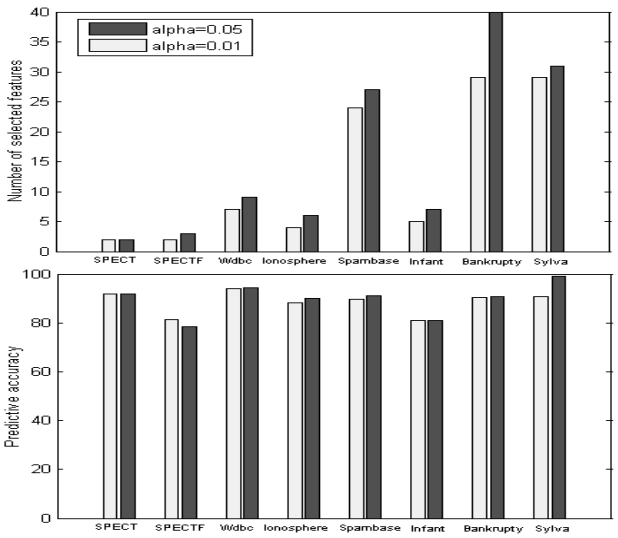

Figure 7 Fast-OSFS performance of with different alpha values

With alpha equal to 0.01, Figure 7 gives the compactness and predictive accuracy (%) of the 4 algorithms on 10 challenge datasets. Our algorithms achieve more compactness and higher accuracy than Alpha-investing on 8 out of the 10 datasets. On the hiva dataset, our algorithms select fewer features and the results are competitive with Alpha-investing. On the ovarian-cancer dataset, Alpha-investing selects more features to achieve

the best accuracy. But on the Dexter dataset, Alpha-investing failed to select any features.

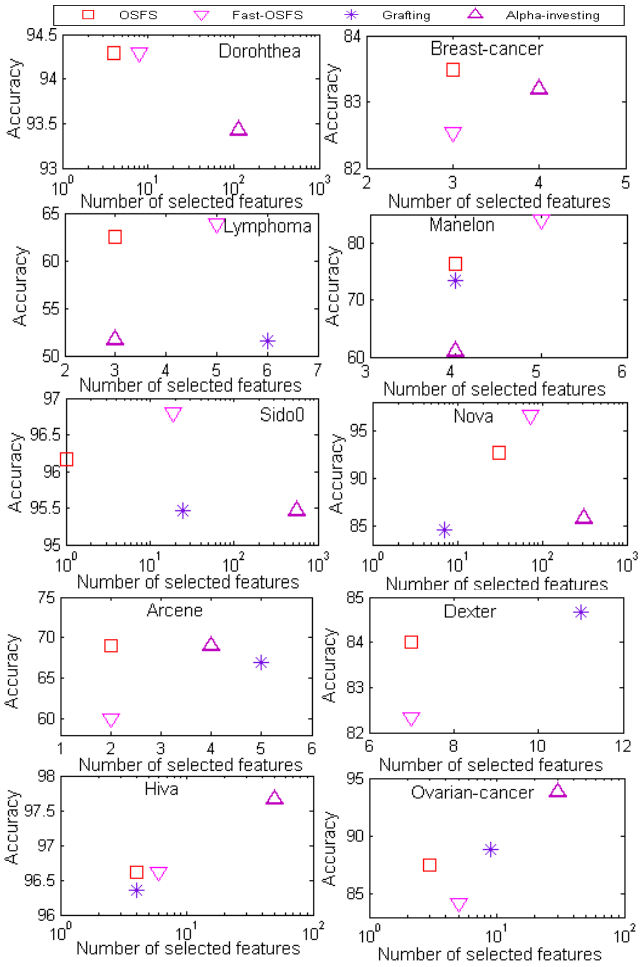

Figure 8 The compactness and prediction accuracy (%) of four algorithms (alpha=0.01)

Compared with Grafting, our algorithms have better compactness and higher accuracy on 7 out of the 10 datasets. On the nova dataset, OSFS and Fast-OSFS achieve much higher accuracy, up to 0.926 and 0.966, respectively, than Grafting with an accuracy up to 0.846, though they selected a few more features than Grafting. On the ovarian-cancer and dexter data, OSFS selects many fewer features than Grafting, but its accuracy is still competitive with Grafting. Moreover, Grafting fails to select any features on the dorohthea and breast-cancer data because of the problem of out of memory.

On most of those challenge data sets, our algorithms have outperformed Grafting and Alpha-investing.

5.3 Running Time Analysis

Since the Grafting and Alpha-investing code used in the experiments was implemented in Matlab and our algorithms were written in the C language, a direct time comparison between them and our algorithms was not performed.

Although we had a theoretical analysis of time complexity for OSFS and Fast-OSFS, a summary of the running time results of the execution of OSFS and Fast-OSFS is also given in Figure 8. The time reported is the normalized time which is the running time of OSFS for a data set divided by the corresponding running time of Fast-OSFS. Thus, a greater normalized running time than one implies that OSFS is slower than Fast-OSFS on the same learning task.

On the UCI data sets, the speed of Fast-OSFS is at least twice faster than that of OSFS. Since the running time of Fast-OSFS and OSFS is less than one second on most of these data sets, we only report the running time longer than ten seconds on five data sets in Figure 8 (left: alpha=0.01; right: alpha=0.05).

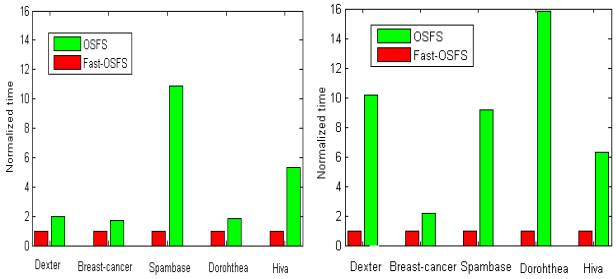

Figure 9 Normalized time results

On the challenge data sets, the selected features of Fast-OSFS are competitive with OSFS, and Fast-OSFS gets a higher accuracy on most of the datasets and is much faster on all datasets.

5.4 Discussion

In this section, we further analyze our algorithms. Firstly, although Grafting is compared with our algorithms, it is a quasi-streaming feature selection algorithm. So we don't further discuss it. As for Alpha-investing, our algorithms outperform Alpha-investing on most of the 18 datasets. Therefore, our framework can manage streaming feature selection better than Alpha-investing on the original streaming features.

Secondly, with prior knowledge of the structure of the candidate features, Alpha-investing can achieve good performance. If we have the prior knowledge, our framework can also deal with the task well. For example, with domain knowledge, we can place potentially more informative features earlier in the streaming features, making it easier for our algorithms to find strongly relevant and useful features. If strongly relevant features can be found earlier, the corresponding redundant features can be earlier eliminated in the online redundancy analysis step of our framework. Fast-OSFS could especially benefit from prior domain knowledge.

Thirdly, on all datasets, regarding the compactness, Fast-OSFS is competitive with OSFS. As to the predictive accuracy, from our experimental results, OSFS outperforms Fast-OSFS on the datasets with a very small sample-to-variable ratio while Fast-OSFS is superior to OSFS on the datasets with a large sample. The explanation is that OSFS performs a redundant analysis for all features within BCF so that it keeps BCF as minimal as possible. This is very beneficial to dealing with datasets with a very small sample-to-variable ratio. But with large samples, OSFS increases the total number of tests performed on redundant features or strongly relevant features. This reduces the test of statistical power. As for Fast-OSFS, it only performs a redundant analysis for the feature just added into BCF in the process of feature generation. Thus, with large samples, Fast-OSFS significantly reduces the total number of tests, and has stronger statistical power than OSFS. But this leads to more redundant features into BCF, and so some tests are unreliable in the outer-redundant analysis phase when the sample-to-variable ratio is small.

Finally, to control false positives, our algorithms use two strategies: multiple comparisons and the parameter *k*. The parameter *k* is the maximum allowable size that a conditioning set may grow, and is a key parameter to control false positive features. At each iteration, the selected features are added into the set BCF. In the online redundancy analysis phase, OSFS uses multiple statistical comparisons to filter redundant features. It needs to find all subsets from BCF to perform multiple tests, and the size of the maximum subset is *k*. Under the assumption that all independence tests are reliable, with a right value of *k*, the false positives will be well controlled. Thus，the experimental results show that our algorithms exhibit little sensitivity to false positive features because of these control strategies, even when a fixed significance threshold is used.

6. Conclusions

In this paper, we have proposed a novel framework with two new algorithms to deal with streaming feature selection. Compared with two state-of-the-art algorithms Grafting and Alpha-investing, our algorithms have demonstrated more compactness and better accuracy in supervised learning on databases that contain many irrelevant and redundant features.

In our experiments, we stimulated the feature set with an unknown but finite size. In our future work, we will explore how to dynamically assess the predictive accuracy with an infinite size, when reaching a certain threshold. We will also study the impact of stopping criteria on the OSFS and Fast-OSFS algorithms. Furthermore, we plan to apply online streaming feature selection to real Mars crater data, where craters are represented by thousands of texture-based features that call for efficient feature selection.

Acknowledgements

This work is supported by the 973 Program of China (2009CB326203), the National Science Foundation of China (NSFC 60828005 and 60975034), the U.S. NSF (CCF-0905337), and the U.S. NASA (NNX09AK86G).

References

C. F. Aliferis, A. Statnikov and I. Tsamardinos et al., Local Causal and Markov Blanket Induction for Causal Discovery and Feature Selection for Classification Part I: Algorithms and Empirical Evaluation, *Journal of Machine Learning Research* , 11, 171−234, 2010

T. P. Conrads et al., High-Resolution Serum Proteomic Features for Ovarian Cancer Detection, *Endocrine-Related Cancer*, 11, 163-178, 2004

M. Dash and H. Liu, Consistency-Based Search in Feature Selection, *Artificial Intelligence*, 151(1-2), 155-176, 2003

I. Guyon and A. Elisseeffm, An Introduction to Variable and Feature Selection, *Journal of Machine Learning Research*, 3, 1157-1182, 2003

R. Kohavi and G. H. John, Wrappers for Feature Subset Selection, *Artificial Intelligence*, 97, 273-324, 1997

D. Koller and M. Sahami, Toward Optimal Feature Selection. *ICML*, 284-292, 1996

S. Loscalzo, L. Yu and C. Ding, Consensus Group Based Stable Feature Selection, 567-576, *KDD*, 2009

S. Perkins and J. Theiler, Online Feature Selection Using Grafting, *ICML*, 592-599, 2003

Spider: a Matlab Machine Learning Tool, 2010. http://www.kyb.mpg.de/bs/people/spider/main.html

R. Tibshirani Regression, Shrinkage and Selection via the Lasso, *J. Royal. Statist. Soc. B*. 58, 267–288, 1996

E. Tuv, A. Borisov and G. Runger et al., Feature Selection with Ensembles, Artificial Variables, and Redundancy Elimination, *Journal of Machine Learning Research*, 10, 1341-1366, 2009

Y. Wang et al., Gene-expression Profiles to Predict Distant Metastasis of Lymph-Nodenegative Primary Breast Cancer, *Lancet*, 365, 671-679, 2005

L. Yu and H. Liu, Efficient Feature Selection Via Analysis of Relevance and Redundancy, *Journal of Machine Learning Research*, 5, 1205-1224, 2004

J. Zhou, D. P. Foster and R. Stine et al., Streaming Feature Selection using Alpha-Investing, *KDD*, 384-393, 2005

J. Zhou, D. P. Foster and R.A. Stine et al., Streamwise Feature Selection, *Journal of Machine Learning Research*, 7, 1861-1885, 2006