

Feature Selection for Microarray Data by AUC Analysis

Juana Canul-Reich*, Lawrence O. Hall*, Dmitry Goldgof* and Steven A. Eschrich†

*Department of Computer Science and Engineering
University of South Florida, Tampa, Florida USA 33620

Email: jcanulre, hall, goldgof@cse.usf.edu

†Department of Biomedical Informatics

H. Lee Moffitt Cancer Center & Research Institute, Tampa, Florida USA 33612

Email: steven.eschrich@moffitt.org

Abstract—Microarray datasets are often limited to a small number of samples with a large number of gene expressions. Therefore, dimensionality reduction through a feature/gene selection process is highly important for classification purposes. In this paper, the feature perturbation method we previously introduced is applied for gene selection on microarray data. A publically available colon cancer dataset is used in our experiments. In comparison to SVM-RFE, our method is better for a number of features between 10 and 80, however for fewer features SVM-RFE results in higher accuracy. An analysis of the area under the curve of the feature perturbation method for the top 50 and 25 features is performed, aiming to determine the proper amount of noise to be applied. We show that a good set of small features/genes can be found using the feature perturbation method.

Index Terms—Microarray data, classification, support vector machines, feature selection.

I. INTRODUCTION

Microarray datasets, characterized by a limited number of samples due to the cost of acquisition and usually a larger number of gene expressions, have highlighted the importance of the dimensionality reduction task. Too many gene expressions (or features), in a dataset lead to poor classification performance. Therefore, feature selection processes are vital to successfully achieve high classification performance on microarray datasets. Several methods for feature selection have been proposed since one of the benefits is the possibility of the reduction of overfitting. For feature selection there exist two highly effective approaches, filters and wrappers. Recent studies have shown that wrapper methods, which involve the performance of the underlying learning algorithm in the process of feature selection, usually result in a selection of genes such that the classification accuracy is noticeably higher [2][3]. In [4], recursive feature elimination for support vector machines, SVM-RFE for short, was introduced. It is a backward selection approach that selects genes according to their influence (weight) on a support vector machine.

We introduced the feature perturbation method in [1], which is more general than SVM-RFE in the sense that it could have any learning algorithm as the base classifier. We use the colon cancer dataset for our experiments in this paper and show results of the performance of our method compared to that of SVM-RFE. We found that the feature perturbation method outperforms SVM-RFE for a large range of selected features,

however for a small number of features SVM-RFE has higher average accuracy. We considered this behavior is likely due to the amount of noise injected for a small set of presumably good features. To determine the proper amount of noise to be applied, we introduced an analysis of the area under the curve (AUC) for the top 50 features using diverse noise levels. Finally we show that with the proper amount of noise injected into a small set of features, the feature perturbation method ends up generating accurate classifiers.

II. METHOD

A. Feature Perturbation

A feature perturbation method was introduced in [1]. It is a wrapper approach that uses backward selection process for feature elimination. The algorithm starts with the entire set of features in the dataset, and the size of the feature set decreases by removing the least important features at every iteration. The hypothesis of the feature perturbation method is: the least important features will have little effect on classification performance when perturbed by noise, therefore these are nonrelevant features. Correspondingly, most important features will highly affect classification performance when perturbed by noise. Nonrelevant features are removed so that only relevant features remain. Fig. 1 shows a flowchart of the feature perturbation method. The initialization process consists of training set X , which is the original dataset with all features (genes) S as the surviving features. The method iterates in three stages. In the first stage, a classification model is created based on the current training set with all existing surviving features S . Any classifier could be used to create the classification model. In the second stage, called feature ranking, the ranking criteria is computed for each of the surviving features in S . The ranking for a feature is determined by the change in accuracy observed on the training samples before and after adding noise. The third stage, called feature elimination, consists of removing a feature set R from S ; R contains the features with the lowest ranks. For computational purposes, it will be more efficient to remove several features at a time rather than only one. These three stages will be repeated until the training set is left with no features. A ranked feature list will result from the iterative stages. Then, for performance evaluation purposes, a

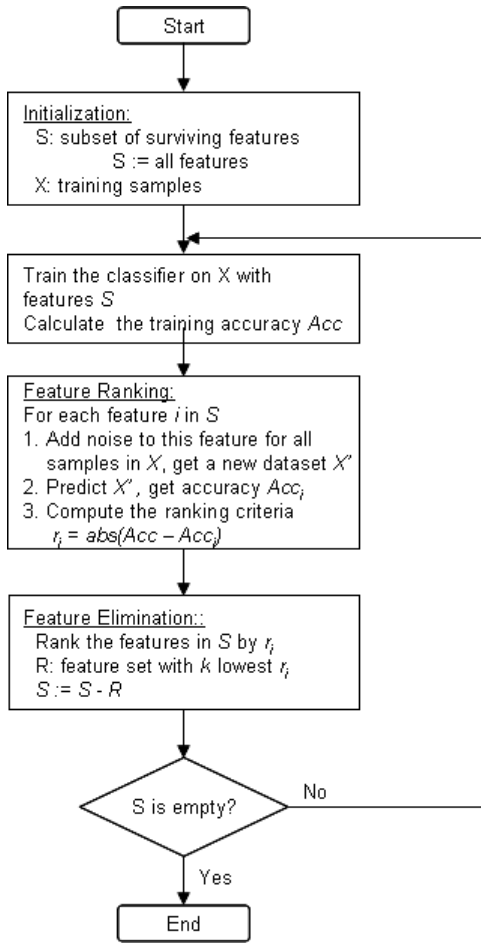


Fig. 1. Feature Perturbation Method

classification model can be created based on a selected feature set from the ranked list, and its accuracy calculated on the test set. Feature ranking is the most important stage of the feature perturbation method. Individually, each feature i in S has randomly generated noise added to it for each of the samples in X , resulting in a new dataset X' . The trained model whose training accuracy is denoted as Acc is used to do classification on the new dataset, which was perturbed on feature i , to get a prediction accuracy denoted as Acc_i . The ranking criterion for feature i is defined by the absolute value of difference between Acc and Acc_i . In the feature ranking stage, for noise generation, we assume the features are independent of each other. A uniform distribution is used to randomly sample noise with parameters related to the feature being perturbed. Noise is generated following (1):

$$Noise_i \approx Uniform(-c * sd_i, c * sd_i) \quad (1)$$

where sd_i is the standard deviation of the feature being perturbed across all samples in training set, and c is a constant factor indicating the noise level being injected in the perturbation process. For random number generation the function `rand48` was used.

In the feature elimination stage, the feature set R to be

removed from S contains the features with the k lowest r_i , however, in feature set S there may be features that have exactly the same r_i value as that of “the last” k in set R . So, in order to avoid keeping features with this r_i value in the feature set S , k is decreased such that no features in S contain the same r_i value as that of k in R . Given that microarray datasets usually have few samples and a large number of gene expressions, the feature elimination stage will be computationally time consuming if features are removed one at a time. Moreover, only a few features are expected to be relevant for classification. Based on this, an adaptive feature elimination strategy is applied so that the number of features being removed is determined by the current number of surviving features. The adaptive feature elimination strategy consists of removing half the features at a time rather than just one when there are likely to be a large number of nonrelevant features; when the number of surviving features reaches a threshold, one-at-a-time feature removal is initiated.

B. Feature Perturbation vs. SVM-RFE Method

SVM-RFE is an alternative method for feature selection [4]. It follows a backward removal approach as does the feature perturbation method. The main difference between SVM-RFE and the feature perturbation method is that SVM-RFE ranks features according to the weight of each feature as calculated from the support vectors. The feature perturbation method could be applied using any learning algorithm as the base classifier, in contrast to SVM-RFE that only works with support vector machines.

III. EXPERIMENTAL RESULTS

A. Data and Parameters

Experiments were performed on the colon cancer dataset, a well-known microarray benchmark [5] downloadable from [6]. The colon cancer dataset is made up of 62 microarrays from tissue samples including 22 normal and 40 colon cancer tissues. There are 2000 gene expression values for each sample. Ten fold cross validation accuracy was used for performance comparison between the feature perturbation method and SVM-RFE. Even though the feature perturbation method could be applied to any classifier, to be able to compare against SVM-RFE we used support vector machines (SVM) as the base classifier. The SVM we used is a modified version of libSVM [7]. We used a linear kernel with parameter $C=1$, which is the default value, for our experiments to reduce training time as well as the probability of overfitting. The sequential minimal optimization (SMO) algorithm was the optimization algorithm used. Coding and implementation differences lead to differences in results from those presented in [1].

Due to the unequal distribution of the two classes in the dataset, the weighted accuracy was used as the classifier performance measure instead of total accuracy. Weighted accuracy gives a better estimate of the classifier performance than total accuracy [8]. Weighted accuracy is defined in (2),

$$Weighted Accuracy = \left(\frac{tp}{tp + fn} + \frac{tn}{fp + tn} \right) / 2 \quad (2)$$

TABLE I
CONFUSION MATRIX

	Predicted Cancer	Predicter Normal Tissue
Actual Cancer	True Positive (TP)	False Negative (FN)
Actual Normal tissue	False Positive (FP)	True Negative (TN)

TABLE II
THREE BEST NOISE LEVEL PERFORMANCES

Noise level	Number of features	Weighted accuracy
2000	34	84.45%
4000	41	85.77%
6000	49	85.98%

where tp , fp , tn , and fn respectively are the number of true positives, false positives, true negatives and false negatives in a confusion matrix, as shown in Table I.

All experiments in this paper are reported as weighted accuracy. Feature perturbation code was written in the C language on a Linux-based system.

B. Data Preprocessing

A data preprocessing phase is usually performed prior to the application of any learning algorithm [9]. A \log_2 -transformation was applied to the colon cancer data on each of the gene expression values, in order to normalize the data. In addition, data were scaled to a range from 0 through 1 for use in the SVM.

C. Experiments and Results

According to the Student's t-test, we found that only 478 out of 2000 features in the colon dataset have p-values < 0.05. This suggests that the majority of the features are not informative for classification purposes. Therefore we chose the adaptive feature elimination strategy described earlier in section II-A in experiments for both the feature perturbation and the SVM-RFE methods. Thus, starting with 2000 features, 50% of them were removed at every iteration until 10% (200) of total initial features remain. Then features were removed one at a time.

For all experiments, 10-fold cross validation accuracy is reported. Each of the cross-validation experiments was performed five times with different random seeds and the average of the accuracies are reported. The feature perturbation method was done with diverse noise levels. The parameter c in (1), was set to 2000, 3000, 4000, 5000, 6000, 8000, and 10000. In terms of the highest accuracy reached as well as the number of features that it was reached for, we selected 2000, 4000, 6000 as the three best noise level performances, as shown in Table II.

Fig. 2 shows the feature perturbation method's performance for the three best noise levels compared to that of SVM-RFE. It was observed that our method outperforms SVM-RFE for many sets of features, however its accuracy drops off early in contrast to SVM-RFE whose performance was good even for few features. We believe that the 2000, 4000, or 6000 noise levels represented too much noise for a small number of good features, therefore a smaller amount of noise was needed. We decided to reduce the amount of noise when perturbing a few

“good” features. The noise level of 6000 was chosen for further experiments since the highest accuracy was reached using this approach.

Two issues had to be solved to proceed with the experiments:

- 1) the point at which noise reduction was needed
- 2) how much to reduce the noise.

Issue 1) was tackled by observing that at the 6000 noise level the highest accuracy occurred for 49 features. Therefore we chose 50 features to start noise reduction and observe how favourable/unfavourable its impact would be. Hence, p was set to 50 as the switching point.

Issue 2) required the determination of an appropriate noise level to be applied to improve the accuracy for a smaller number of “good” features. To this end, the parameter c of (1) is calculated as follows:

$$c = \alpha * p \quad (3)$$

where, α is in the range [0.1, 2.5]. For simplicity, once (3) was applied, we looked at the resultant accuracies for just one out of ten folds, however, an average of accuracies over five runs was used. The curve formed by the average weighted accuracies of the top 50 features was used to calculate the AUC. The appropriate noise level to be applied maximizes the area under this curve for each α value. The AUC was calculated with the trapezoidal method using the student version of the DADiSP software [10].

For comparison purposes, the AUC was also calculated for SVM-RFE in the same way. Fig. 3 shows a tie for $\alpha = 0.4$, $\alpha = 1.5$, and SVM-RFE in terms of the AUC values for the top 50 features. We also looked at the AUC for the top 25 features. Fig. 4 shows these resultant AUC values with $\alpha = 0.5$ being best. Based on the AUC values obtained, we performed experiments on the colon dataset, starting with the 6000 noise level and at the point where 50 features were left, the following noise settings were applied:

- 1) $\alpha = 1.5$ constant until no features left
- 2) $\alpha = 0.5$ constant until no features left
- 3) $\alpha = 1.5$ reducing it again from 25 features to be $\alpha = 0.5$

Fig. 5 shows the performance compared to that of SVM-RFE as well as to the feature perturbation method performance for the case where a simple 6000 noise level was applied and kept constant for all 2000 features. Clearly, any experiment where (3) was applied to the data outperformed the case where it was not applied. The latter now had the worst performance in this scenario.

The feature perturbation method outperformed SVM-RFE for certain feature sets, as was shown in Fig. 2. The introduction of (3) to the data made the accuracies of our method no longer drop off as early for small number of features. Fig. 6 shows a view of the top 25 feature comparison of our method for the case where $\alpha = 1.5$, $\alpha = 0.5$ and SVM-RFE. At 25 features, feature perturbation has higher accuracy, with fewer features accuracy crossover occurs between features 15 and 14. SVM-RFE is superior for features 14, 13, and 12. A tie

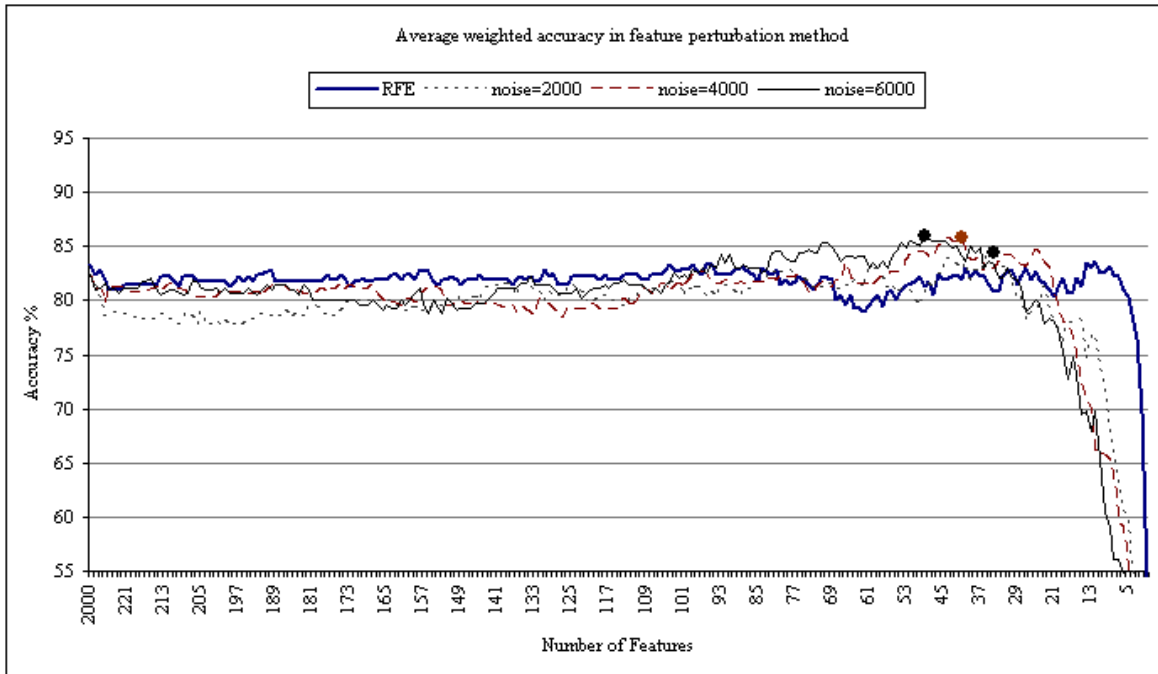


Fig. 2. Feature perturbation method performance for three best noise levels vs. SVM-RFE.

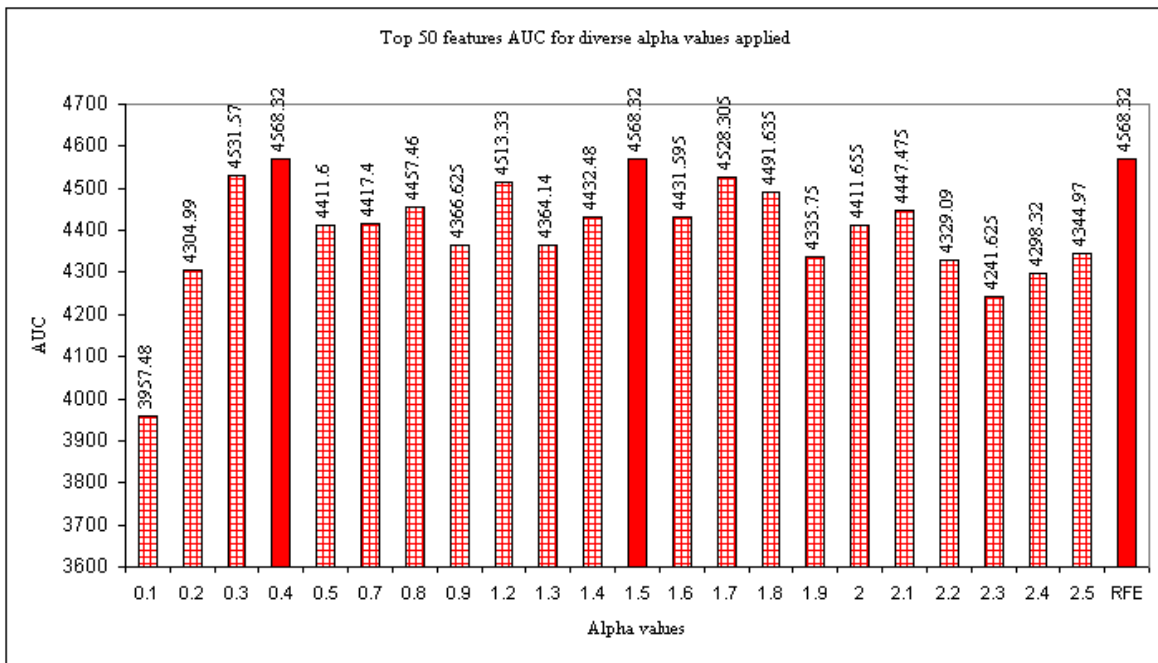


Fig. 3. AUC for the top 50 features for diverse alpha values and for SVM-RFE.

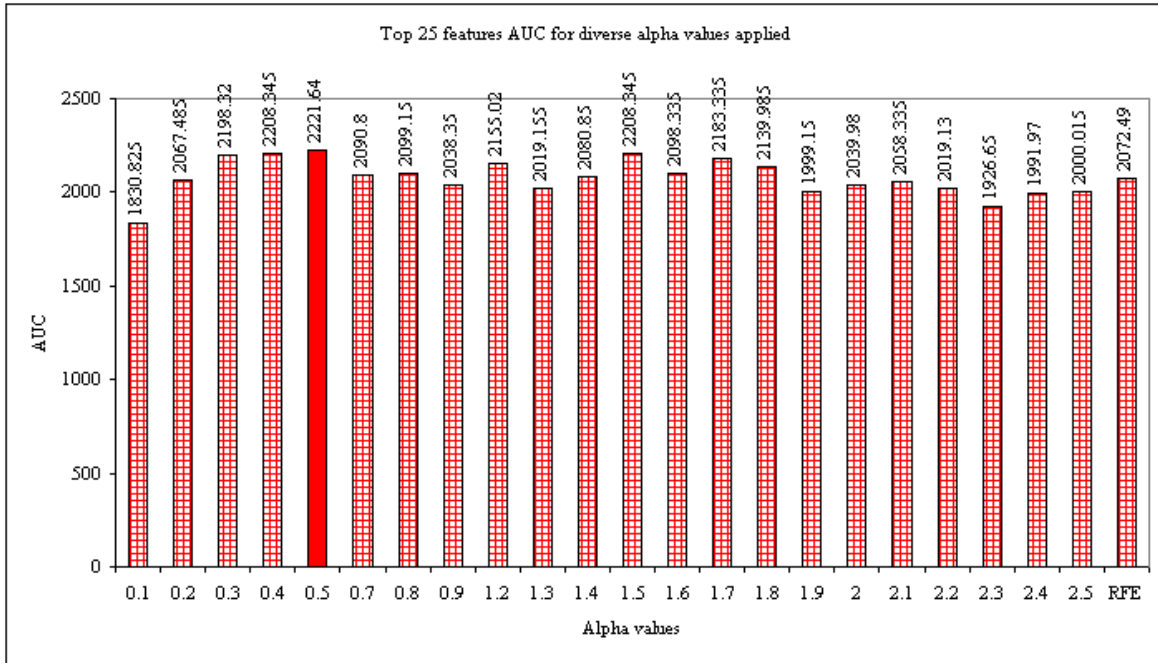


Fig. 4. AUC for the top 25 features for diverse alpha values and for SVM-RFE.

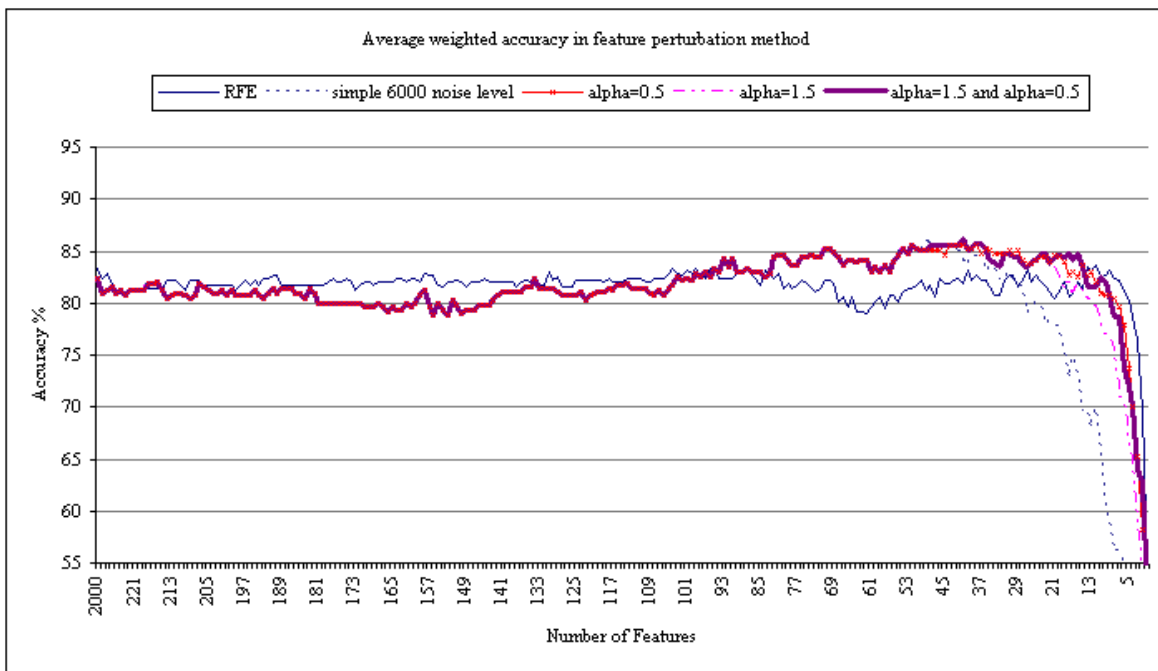


Fig. 5. Comparison of average weighted accuracy of the feature perturbation method for diverse alpha values with SVM-RFE

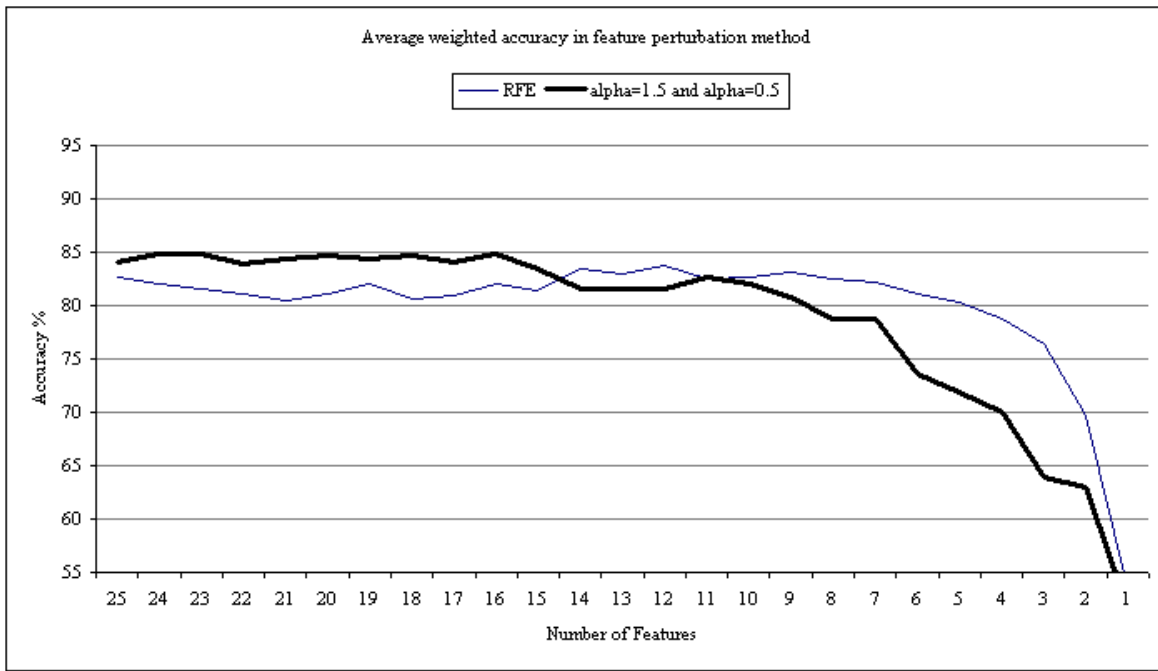


Fig. 6. Comparison of the top 25 average weighted accuracy of the feature perturbation method for $\alpha = 1.5$ and 0.5 with SVM-RFE

is found for 11 features and thereafter SVM-RFE is higher in average accuracy.

IV. CONCLUSIONS

We compared the performance of our method against that of SVM-RFE and found that ours outperformed SVM-RFE most sets of features, however SVM-RFE resulted in better performance for ≤ 14 features. We hypothesized that the same amount of noise applied to 2000 features represented too much noise for a smaller number of features which are presumably better. This paper introduces an analysis based on the area under the curve for the top 50 and 25 features in the process of determining an appropriate amount of noise to be applied for a selected number of features with the ultimate goal of improving performance for the feature perturbation method. According to the AUC analysis, diverse noise levels were applied to the top 50 features and we ended up with an amount of noise which resulted in improved performance for more feature sets. It does not drop off very early as before, however for the top 10, 12, 13, and 14 features SVM-RFE is still better.

Two issues need to be addressed in the feature perturbation method for it to be successfully applied to any data. These are the determination of:

- initial amount of noise to be applied such that good genes are selected for classification purposes
- the switching point, since fewer features require a smaller amount of noise

ACKNOWLEDGMENT

This research was partially supported by the Department of Energy through the ASCI PPPE Data Discovery Program, Contract number: DE-AC04-76DO00789, and by the

Department of Defense, National Functional Genomics Center Project, under award number DAMD 17-02-2-0051. Views and opinions of, endorsements by, the author(s) do not reflect those of the US Army or the Department of Defense.

REFERENCES

- [1] L. Chen, D.O. Goldgof, L.O. Hall, S. A. Eschrich, *Noise-based featured perturbation as a selection method for microarray data*, ISBRA 2007
- [2] I. Inza, P. Larranaga, R. Blanco, and A.J. Cerrolaza, *Filter versus wrapper gene selection approaches in DNA microarray domains*, Artificial Intelligence Medicine, 31 (2004) 91-103.
- [3] H. Xiong, M.N.S. Swamy, and M.O. Ahmad, *Optimizing the kernel in the empirical feature space*, IEEE transactions on Neural Networks, 16 (2001) 460-474.
- [4] I. Guyon, J. Weston, S. Barnihill, and V. Vapnik., *Gene selection for cancer classification using support vector machines*, Machine Learning, 46 (2002) 389-422.
- [5] U. Alon, et al., *Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays*, Proc. Nat. Acad. Sci. USA, 96 (1999), Issue 12, 6745-6750.
- [6] *Data pertaining to the article Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays* <http://microarray.princeton.edu/oncology/affydata/index.html>, 2000
- [7] C.C.. Chang and C. J. Lin, *A library for support vector machines, libsvm*, <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- [8] I. H. Witten and E. Frank, *Data mining: Practical machine learning tools and techniques with Java implementations*, Morgan Kaufman Publishers, 2000.
- [9] D. Pyle, *Data preparation for data mining*, California, USA: Morgan Kaufman Publishers, 1999.
- [10] *DADiSP The ultimate engineering spreadsheet*, <http://www.dadispc.com>, 2008