GEOMETRIC APPROACH FOR GAUSSIAN-KERNEL BOLSTERED ERROR ESTIMATION FOR LINEAR CLASSIFICATION IN COMPUTATIONAL BIOLOGY

Arsen Arakelyan, Lilit Nerisyan, Aram Gevorgyan, Anna Boyajyan

***Abstract**: Classification and feature selection techniques are among the most commonly used mathematical approaches for analysis and interpretation of biological data. One of the important characteristics of any classifier is its classification error, which is important to take into consideration for accurate data analysis. The most popular error estimation techniques (resubstitution, bootstrapping, cross-validation) strikingly vary in performance. It is well known that more accurate classifiers such as bootstrapping, cross-validation are very slow, while heavily biased resubstitution is very fast. Recently, a new bolstered error estimation technique has been proposed that optimally combines speed and accuracy. It uses a Monte-Carlo sampling based algorithm for classification for the general case, but for the case of linear classification, an analytical solution may be applied. In this paper we introduce geometric approach for bolstered error estimation and compare its performance with other error estimation algorithms. The results obtained show that geometric bolstered error estimation algorithms are very fast error estimation techniques characterized by accuracy comparable with LOO and having lower variance. These algorithms are useful for analyzing extremely large numbers of features and may find their applications in wide fields of - omics data analysis.*

***Keywords**: Biology and genetics, Classifier design and evaluation, Machine learning.*

***ACM Classification Keywords**: A.0 General Literature - Conference proceedings, I.5.2 – Classifier design and evaluation, Feature evaluation and selection. J.3 - Biology and genetics*

Introduction

Classification techniques have found their wide application in various fields of biomedical research [Sayes, 2007]. The classification problem may be stated as follows: given a set of objects belonging to two or more classes and described by a set of features, the aim is to design a classifier that will correctly predict class memberships of new objects. Support vector machines (SVMs) are among the most popular classifiers widely used for classification and feature extraction in computational biology research. SVMs are used for prediction of protein secondary structure [Nguyen, 2011], analysis of protein and DNA sequences [Choi, 2011; Lee, 2011], protein classification [Cai, 2003], prediction of protein-protein interactions [Cui, 2012], identification of transcription binding sites [Holloway, 2007], and analysis of gene expression data [Golub, 1999; Maulik, 2013].

Occasionally, a classifier may fail to correctly assign the membership of new objects resulting in classification error. Classification true error is the error rate of the classifier if it was tested on the true distribution of cases [Nolan, 1997]. Since the true distribution is generally unknown, there is a need to come up with a proper estimation of the classification true error. Considering its importance both for assessment of the classifier itself and for accurate interpretation of classification results, several algorithms have been developed for classification true error estimation.

The best algorithm for approximation of the true error is considered to be the hold-out estimation [Nolan, 1997], where the dataset is divided into independent training and test sets. The purpose of the training set is to design the classifier, while the test set is used for assessing the classification error. This method, however, requires large

datasets, which are not always available, especially in the field of genomics. In high-throughput gene expression analysis, researchers often deal with very small sample sizes, making the application of the above mentioned strategy basically impossible [Allison, 2006]. In such cases, training set based error estimation approaches, such as resubstitution (Resub), leave-one-out cross-validation (LOO) and bootstrapping methods (BST), are commonly used. These techniques are shown to be strikingly different in terms of speed and accuracy [Dougherty, 2010].

Resub uses the whole training data to estimate the error of a classifier, and is considered to be the fastest among available algorithms [Devroye, 1996]. However, it has been shown that it is heavily biased, especially in the case of small sample sized settings [Devroye, 1996].

LOO error estimation is a case of cross-validation algorithms when a single observation from the original sample is used as validation data, and the remaining observations – as training data. This is repeated until each observation in the sample is used as validation data. LOO error estimation is shown to be nearly unbiased, but to have large variance. Moreover, the speed of LOO slows with the increase of sample size [Lachenbrucha, 1968].

BST error estimation is based on generation of a test set from the training set using sampling with replacement technique [Efron, 1983]. For correct error estimation by bootstrapping, it is suggested to use 100-200 bootstrap samplings [Efron, 1983]. Bootstrapping error estimation is usually pessimistically biased, but has lower variance compared to LOO. In addition, bootstrapping is very slow compared to Resub and LOO.

Recently, Braga-Neto and Dougherty [Braga-Neto, 2004] have proposed another error estimation technique called "bolstered error estimation" (BOL). The principal idea of this approach is the following. The available data points are used as a training set for the design of the classifier. Next, a bolstered distribution of these points is generated based on class-dependent variances. In simple cases, bolstering is performed by constructing p-dimensional spheres around the points (spherical bolstering), where p is the number of features (Figure 1).

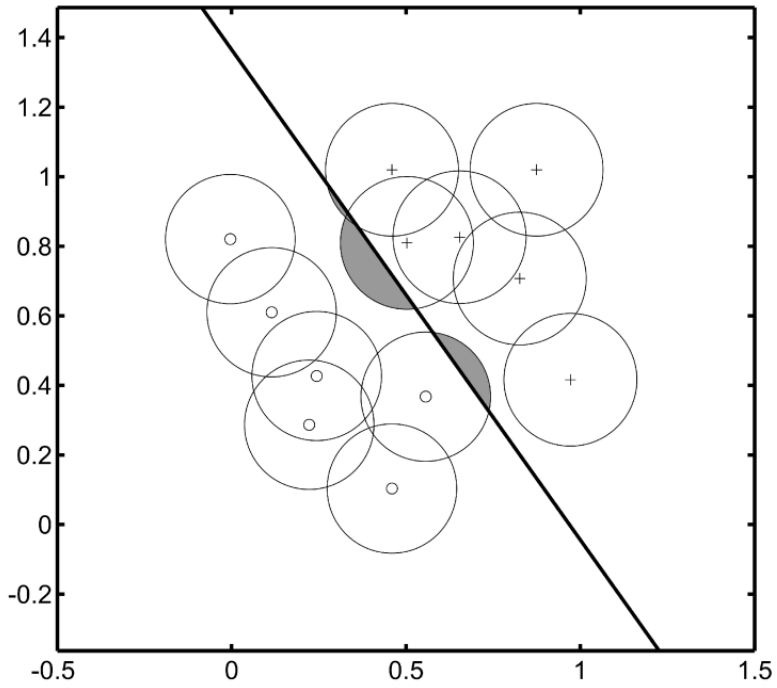

Figure 1. Graphical representation of bolstered error estimation. Shaded areas of the circles represent error contributions of the points. The average of all the error contributions is the bolstered error estimate

The test set is generated by taking new points from these spheres. The use of bolstered space increases the accuracy of error estimation. According to the authors [Braga-Neto, 2004], bolstered error estimation combines high computational speed of resubstitution and the accuracy of LOO algorithms. In addition, they have proposed a semi-bolstering technique, when bolstering is applied only on correctly classified points. In the original paper [Braga-Neto, 2004], bolstering error estimation is calculated using Monte-Carlo integration (mBOL), however, in the case of linear classification and spherical bolstering, it is possible to find analytical solutions for error estimation that may be more accurate and less computationally intensive.

In this paper we introduce an analytical approach for bolstered error estimation for linear classification with SVMs based on computational geometry approaches.

Methods

Geometric bolstered error estimation algorithm (gBOL)

The algorithm is designed for linear classification and spherical bolstering. It proceeds by firstly training the linear classifier (e.g. SVM) based on dataset points. Next, spherical bolsters are generated around the points, with sphere radiuses being equal to the variance in each class, and sphere dimensions representing the number of features (p). If the signed distance of a sphere center from the hyperplane is less than the sphere radius, the hyperplane may cut the sphere resulting in generation of a spherical cap. The ratio of the cap volume to the whole volume of the sphere is the error contribution of the given point. In the most extreme cases the whole sphere may appear in the opposite decision region, resulting in ratio equal to 1 (100% misclassification). Finally, classification error is estimated by computing the ratio of the volume of spherical caps or spheres appearing in the opposite decision region to the overall volume of the spheres (Figure 2).

The pseudocode for gsBOL calculation:

Let N data points $\{x_i, y_i\}$ for $i = 1...N, x \in R^p$ and classes $y = \{-1, 1\}$ be the training set.
Set error $err = 0$;
Train the linear classifier;
For each class in y
  Calculate the radius σ of the spherical bolstering kernel;
  Calculate the volume of the sphere V with radius σ;
    For each point belonging to the class in y
      Calculate the signed distance d_i from the point x_i to the separating hyperplane;
      Calculate the volume of the spherical cap C_i in opposite decision region using σ and d_i;
      Calculate the bolstered error $errp_i$ for the point x_i as C_i / V;
      Accumulate $errp_i$ in $err = err + errp_i$;
    End
End
The total bolstered error is calculated as err/N

Figure 2. The pseudocode for geometric bolstered error estimation

Geometric semi-bolstered error estimation (gsBOL)

The semi-bolstered error estimation algorithm is based on the gBOL algorithm. The difference is that bolstering kernel is calculated only for correctly classified objects [Braga-Neto, 2004], while misclassified point are assigned 100% error.

Formulas used in calculations

Linear support vector machines

For N data points $\{x_i, y_i\}$ for $i = 1...N, x \in R^p$ and classes $y = \{-1, 1\}$, the linear support vector solution should satisfy the following conditions [Vapnik, 1995]:

$$\begin{cases} W(x_i) = w^T x_i + b \geq +1, \text{if } y_i = +1 \\ W(x_i) = w^T x_i + b \leq -1, \text{if } y_i = -1 \end{cases} \quad (1)$$

where $w^T x_i + b = 0$ is the separation hyperplane. By minimizing $w^T w$, the margin between both classes is maximized. The signed distance d_i from a data point x_i to the separating hyperplane equals $d_i = W(x_i)\|w\|$.

Bolstering kernel radius

Bolstering kernel radiuses were calculated according to the approach given in [Braga-Neto, 2004]. The Gaussian *p*-variate (*p* is the number of features) zero-mean bolstering kernel generates spheres centered at the data points. The radius of the sphere, which is equal to the standard deviation σ for a given class, can be easily calculated from Euclidean pairwise distances between data points in the given class multiplied by a correction factor. The correction factor is the inverse of χ^2 cumulative distribution at point 0.5 with degrees of freedom equal to *p* [Braga-Neto, 2004].

Cap volume computation and bolstered error estimation

If the distance from a given data point to the separation hyperplane is less than σ, then the hyperplane will divide the bolstering sphere into caps, one of which will be located in the opposite decision region. The volume of this spherical cap actually represents the error contribution of a given data point.

Spherical cap volume can be computed using formulas introduced by Li [Li, 2011] as follows:

$$\begin{aligned} V_n^{cap}(r) &= \int_{\theta-\varphi}^{\theta-0} V_{n-1}(r\sin\theta) d\cos\theta = \\ &= 1/2\, \mathrm{V}_n(r) I_{\sin^2\phi}\left(\frac{n+1}{2}, \frac{1}{2}\right) \end{aligned} \quad (2)$$

where n is the sphere dimension, r is the radius of the sphere, $I_{\sin^2\phi}\left(\frac{n+1}{2}, \frac{1}{2}\right)$ is the incomplete beta function and $0 \leq \phi \leq \pi/2$ is the colatitude angle.

Algorithms and scripts for Monte-Carlo sampling (mBOL), LOO and BST error estimation

The algorithms and MATLAB codes for BST, LOO, and mBOL error estimation were obtained from previously published sources [Efron, 1994; Hastie, 2009] and [Okun, 2011].

The scripts for gBOL and gsBOL as well as the general framework for comparison of abovementioned estimation techniques were written in MATLAB. For SVM linear classification, MATLAB's built-in svmtrain function was used.

The Matlab source code for geometric bolstering is located at

http://www.mathworks.com/matlabcentral/fileexchange/40118/.

All the used scripts are available at http://www.molbiol.sci.am/big/jbc/Arakelyan_JBC_src.zip.

Datasets

For evaluation of different error estimation techniques we have used two datasets from previously published results on microarray-based classification of non–small cell lung cancer [Showe, 2009] and prediction of survival in sub-optimally debulked patients with ovarian cancer [Bonome, 2008]. These datasets are available in Gene Expression Omnibus (http://www.ncbi.nlm.nih.gov/geo/) under identifiers GSE13255 and GSE26712, respectively.

In the first study authors have identified a 29-gene signature that separates these two patient classes with 86% accuracy [Showe, 2009]. The classification was performed using an SVM with recursive feature elimination with 10-fold cross-validation. In the second study, expression profiles of 57 genes were used for prediction of survival in sub-optimally debulked patients. Classification was performed with hierarchical clustering and 10-fold cross-validation [Bonome, 2008].

Additionally, we have generated synthetic data (200 samples per class, 5 dimensions) drawn from Gaussian class-conditional distribution with means equal to -0.5 and 0.5 and spherical variance equal to 2, with Bayes error equal to 0.29. This very high error rate was chosen to evaluate the performance of error estimators in very extrim settings.

Experimental setup

We have assessed the performance of gBOL, gsBOL, as well as Resub, LOO and BST error estimation algorithms according to the following setup:

1. For all datasets we have performed classification error estimation using p = 1, 2, 3, and 5 top ranked genes.
2. All calculations were performed for samples of size n = 5, 10, 20 and 50 per class. For the GSE26712 dataset, n = 50 was not used because the maximum sample size among the classes was 45.
3. The true error for each experiment was calculated using hold-out estimation: In each experiment a sample of size n was independently drawn from the pool. This sample was used as a training set, while the remainder was used as a test set.
4. Each experiment was performed 1000 times.
5. The true error deviation distribution parameters (mean, SD and root mean square (RMS)) were calculated for each experiment.

Results

We have performed an evaluation of geometrical bolstered and semi-bolstered error estimation using synthetic and experimental data, and compared the results with other commonly used error estimation algorithms, as well as the original Monte-Carlo based bolstering algorithm.

Simulation data

The accuracy of error estimation techniques for the synthetic data is shown in Table 1. Because the true error of the synthetic data was known we also evaluated the performance of hold-out estimator, which is thought to be the most accurate true error estimator. However, we found that hold-out estimator was not the best in our experiments. Simple average rank calculation of true error differences showed that mBOL, gBOL and BST are top tree error estimators.

Table 1. Performance of the error estimation algorithms on the synthetic data

		Class 1: Mean = -0.5, variance: 2; Class 2: Mean = 0.5, variance: 2 True Error = 0.29						
		Hold out	Resub	LOO	BST	mBOL	gBOL	gsBOL
		M_{diff} ± SD (RMS)						
p = 1	n = 5	-0.16 ± 0.07 (0.17)	-0.07 ± 0.12 (0.14)	-0.17 ± 0.20 (0.26)	-0.03 ± 0.10 (0.10)	-0.08 ± 0.09 (0.12)	-0.07 ± 0.10 (0.12)	-0.12 ± 0.12 (0.17)
p = 1	n = 10	-0.15 ± 0.06 (0.16)	-0.09 ± 0.09 (0.13)	-0.14 ± 0.14 (0.20)	-0.08 ± 0.07 (0.11)	-0.10 ± 0.08 (0.13)	-0.10 ± 0.08 (0.13)	-0.13 ± 0.09 (0.16)
p = 1	n = 20	-0.13 ± 0.04 (0.14)	-0.11 ± 0.07 (0.13)	-0.13 ± 0.10 (0.17)	-0.10 ± 0.06 (0.12)	-0.11 ± 0.06 (0.13)	-0.11 ± 0.07 (0.13)	-0.13 ± 0.07 (0.15)
p = 1	n = 50	-0.12 ± 0.02 (0.12)	-0.12 ± 0.04 (0.13)	-0.12 ± 0.05 (0.13)	-0.12 ± 0.04 (0.12)	-0.12 ± 0.04 (0.13)	-0.12 ± 0.04 (0.13)	-0.13 ± 0.04 (0.13)
p = 2	n = 5	-0.13 ± 0.07 (0.15)	0.02 ± 0.13 (0.13)	-0.15 ± 0.20 (0.25)	0.07 ± 0.09 (0.11)	-0.03 ± 0.09 (0.09)	0.01 ± 0.10 (0.10)	-0.04 ± 0.12 (0.13)
p = 2	n = 10	-0.11 ± 0.06 (0.13)	-0.03 ± 0.10 (0.10)	-0.13 ± 0.15 (0.20)	0.00 ± 0.07 (0.07)	-0.06 ± 0.07 (0.09)	-0.04 ± 0.08 (0.09)	-0.07 ± 0.09 (0.12)
p = 2	n = 20	-0.09 ± 0.03 (0.10)	-0.05 ± 0.07 (0.08)	-0.09 ± 0.09 (0.13)	-0.03 ± 0.06 (0.07)	-0.06 ± 0.06 (0.08)	-0.05 ± 0.06 (0.08)	-0.08 ± 0.07 (0.10)
p = 2	n = 50	-0.07 ± 0.02 (0.07)	-0.05 ± 0.04 (0.07)	-0.07 ± 0.05 (0.08)	-0.05 ± 0.04 (0.06)	-0.06 ± 0.04 (0.07)	-0.06 ± 0.04 (0.07)	-0.07 ± 0.04 (0.09)
p = 3	n = 5	-0.11 ± 0.07 (0.13)	0.09 ± 0.12 (0.15)	-0.12 ± 0.20 (0.23)	0.14 ± 0.08 (0.16)	0.00 ± 0.08 (0.08)	0.07 ± 0.10 (0.12)	0.03 ± 0.12 (0.12)
p = 3	n = 10	-0.08 ± 0.04 (0.10)	0.03 ± 0.09 (0.10)	-0.09 ± 0.14 (0.16)	0.07 ± 0.07 (0.10)	-0.01 ± 0.07 (0.07)	0.03 ± 0.08 (0.08)	-0.01 ± 0.09 (0.09)
p = 3	n = 20	-0.07 ± 0.03 (0.07)	0.00 ± 0.07 (0.07)	-0.07 ± 0.09 (0.11)	0.02 ± 0.06 (0.06)	-0.03 ± 0.05 (0.06)	-0.01 ± 0.06 (0.06)	-0.04 ± 0.07 (0.08)
p = 3	n = 50	-0.05 ± 0.02 (0.06)	-0.03 ± 0.04 (0.05)	-0.06 ± 0.04 (0.07)	-0.02 ± 0.03 (0.04)	-0.04 ± 0.03 (0.05)	-0.03 ± 0.03 (0.05)	-0.06 ± 0.04 (0.07)
p = 5	n = 5	-0.08 ± 0.07 (0.11)	0.19 ± 0.09 (0.21)	-0.10 ± 0.20 (0.23)	0.23 ± 0.05 (0.24)	0.03 ± 0.07 (0.08)	0.16 ± 0.08 (0.18)	0.14 ± 0.10 (0.17)
p = 5	n = 10	-0.04 ± 0.05 (0.07)	0.12 ± 0.09 (0.15)	-0.05 ± 0.14 (0.15)	0.16 ± 0.06 (0.17)	0.02 ± 0.06 (0.06)	0.10 ± 0.07 (0.13)	0.08 ± 0.08 (0.11)
p = 5	n = 20	-0.01 ± 0.03 (0.03)	0.07 ± 0.06 (0.10)	-0.02 ± 0.09 (0.09)	0.10 ± 0.05 (0.11)	0.01 ± 0.04 (0.04)	0.06 ± 0.06 (0.08)	0.04 ± 0.06 (0.07)
p = 5	n = 50	0.02 ± 0.02 (0.03)	0.06 ± 0.04 (0.07)	0.02 ± 0.05 (0.05)	0.06 ± 0.03 (0.07)	0.01 ± 0.03 (0.03)	0.05 ± 0.03 (0.06)	0.02 ± 0.04 (0.04)

Experimental data

For experimental validation, we have used GSE13255 and GSE26712 datasets (see 2.5).

The accuracy of error estimation techniques in all 16 experiments for the GSE13255 dataset (experiment 1) showed that the distributions of the deviation of the error estimates from the true error obtained by gBOL and gsBOL were comparable with the results of LOO, BST and mBOL (Table 2) and outperformed Resub. Meanwhile, gsBOL appeared to be more accurate than gBOL. Moreover, the data obtained showed that gBOL and gsBOL demonstrate much lower variance compared to LOO.

In terms of computational speed, gBOL and gsBOL clearly "beat" mBOL, LOO and BST, being very similar to hold-out and Resub error estimations. While gBOL and gsBOL demonstrated almost no variability in computational speed depending on the sample size. The speed of mBOL, LOO, and, especially, BST dramatically slowed down when the sample size became more than 10 (Figure. 3).

Table 2. Performance of the error estimation algorithms in experiment 1 (GSE13255)

		True errror	Resub	LOO	BST	mBOL	gBOL	gsBOL
Fea-tures	Sample size	M ± SD	M_{diff}± SD (RMS)					
p = 1	n = 5	0.47±0.05	0.12±0.12 (0.17)	0.04±0.18 (0.18)	0.17±0.09 (0.19)	0.12±0.09 (0.15)	0.12±0.10 (0.16)	0.09±0.11 (0.14)
p = 1	n = 10	0.46±0.03	0.10±0.09 (0.13)	0.05±0.13 (0.14)	0.12±0.07 (0.14)	0.09±0.08 (0.12)	0.09±0.08 (0.12)	0.08±0.09 (0.12)
p = 1	n = 20	0.46±0.02	0.08±0.07 (0.11)	0.06±0.08 (0.10)	0.09±0.06 (0.11)	0.08±0.06 (0.10)	0.08±0.06 (0.10)	0.07±0.07 (0.10)
p = 1	n = 50	0.49±0.03	0.11±0.05 (0.12)	0.10±0.05 (0.11)	0.11±0.05 (0.12)	0.11±0.05 (0.12)	0.11±0.05 (0.12)	0.10±0.05 (0.11)
p = 2	n = 5	0.35±0.08	0.13±0.13 (0.18)	-0.02±0.19 (0.19)	0.17±0.10 (0.20)	0.06±0.11 (0.13)	0.11±0.11 (0.16)	0.07±0.13 (0.15)
p = 2	n = 10	0.33±0.06	0.07±0.10 (0.12)	-0.00±0.13 (0.13)	0.10±0.09 (0.13)	0.03±0.09 (0.09)	0.06±0.09 (0.11)	0.02±0.10 (0.10)
p = 2	n = 20	0.31±0.05	0.04±0.08 (0.09)	0.01±0.09 (0.09)	0.05±0.07 (0.09)	0.02±0.07 (0.08)	0.04±0.08 (0.08)	0.01±0.08 (0.08)
p = 2	n = 50	0.30±0.04	0.02±0.06 (0.07)	0.01±0.07 (0.07)	0.03±0.06 (0.07)	0.02±0.06 (0.06)	0.02±0.06 (0.07)	-0.00±0.06 (0.06)
p = 3	n = 5	0.35±0.08	0.16±0.13 (0.21)	-0.03±0.19 (0.19)	0.22±0.10 (0.24)	0.07±0.10 (0.12)	0.14±0.11 (0.18)	0.11±0.13 (0.16)
p = 3	n = 10	0.32±0.05	0.10±0.10 (0.14)	-0.00±0.14 (0.14)	0.14±0.08 (0.16)	0.04±0.08 (0.09)	0.09±0.09 (0.13)	0.05±0.10 (0.11)
p = 3	n = 20	0.3±0.04	0.06±0.08 (0.10)	0.01±0.09 (0.09)	0.08±0.07 (0.11)	0.02±0.07 (0.07)	0.05±0.07 (0.09)	0.02±0.08 (0.08)
p = 3	n = 50	0.31±0.03	0.06±0.06 (0.08)	0.03±0.06 (0.07)	0.07±0.06 (0.09)	0.03±0.05 (0.06)	0.05±0.06 (0.08)	0.02±0.06 (0.06)
p = 5	n = 5	0.34±0.07	0.24±0.11 (0.27)	-0.04±0.20 (0.20)	0.29±0.08 (0.30)	0.08±0.10 (0.13)	0.21±0.10 (0.23)	0.19±0.11 (0.22)
p = 5	n = 10	0.30±0.05	0.15±0.09 (0.18)	-0.00±0.13 (0.13)	0.19±0.07 (0.21)	0.04±0.07 (0.08)	0.13±0.08 (0.15)	0.10±0.09 (0.14)
p = 5	n = 20	0.28±0.04	0.09±0.08 (0.12)	0.01±0.09 (0.10)	0.09±0.07 (0.12)	0.02±0.06 (0.06)	0.08±0.07 (0.11)	0.05±0.08 (0.09)
p = 5	n = 50	0.28±0.04	0.06±0.06 (0.09)	0.03±0.07 (0.07)	0.06±0.06 (0.08)	0.02±0.05 (0.06)	0.06±0.06 (0.08)	0.03±0.06 (0.07)

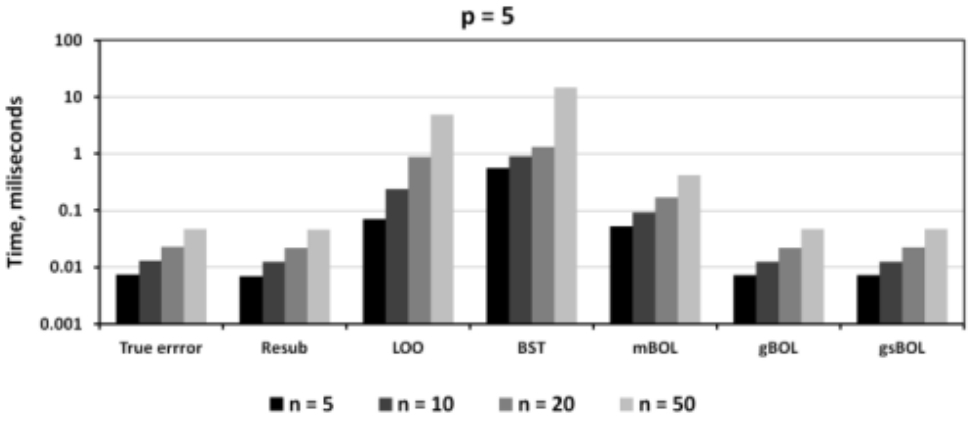

Figure 3. Representative timings in milliseconds for error estimators' performance depending on sample size for p = 5. Dataset – GSE13255

The calculation results also showed that the speed of the computation did not substantially depend on dimensionality (Figure 4).

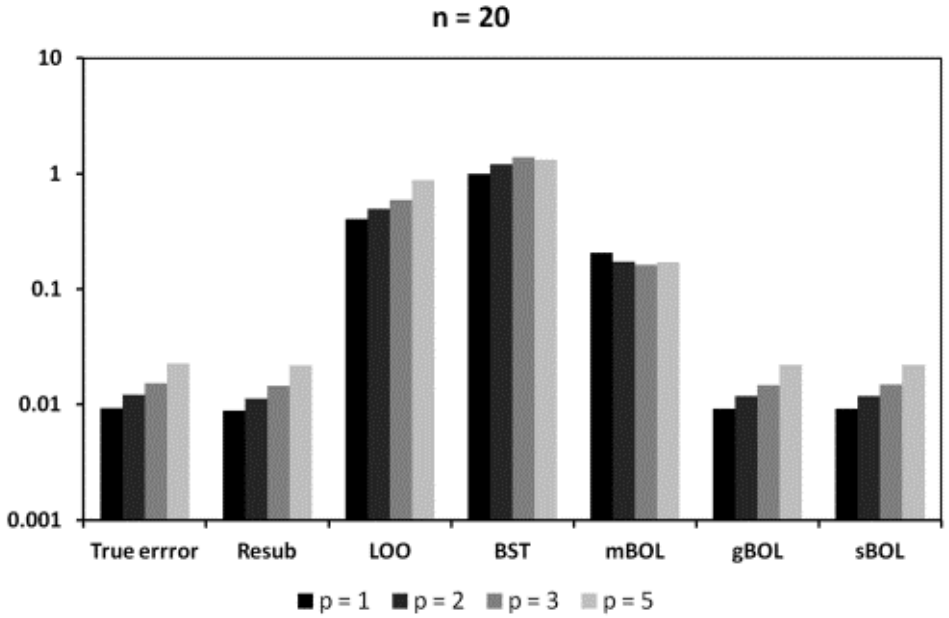

Figure 4. Representative timings in milliseconds for error estimators' performance depending on feature dimensionality for n = 20. Dataset – GSE13255

Similar results were obtained for the second dataset (GSE26712). The overall performance of the error estimators for this dataset is presented in Table 3 and Figure 5 and Figure 6.

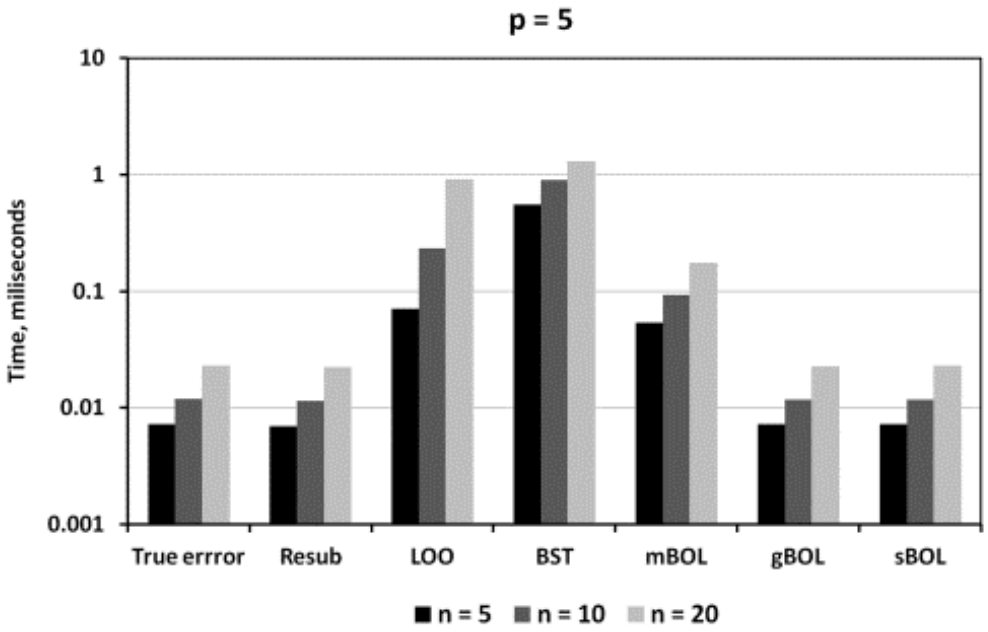

Figure 5. Representative timings in milliseconds for error estimators' performance depending on sample size for p = 5. Dataset – GSE26712

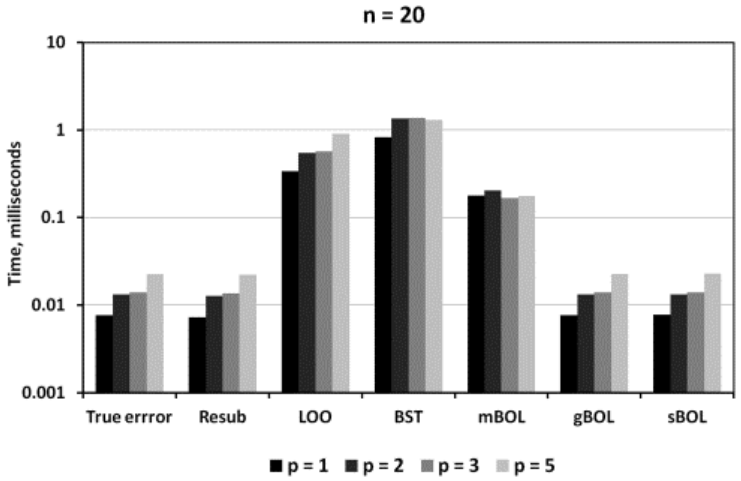

Figure 6. Representative timing in milliseconds for error estimators' performance depending on feature dimensionality for n = 20. Dataset – GSE26712

Table 3. Performance of the error estimation algorithms in experiment 2 (GSE26712)

		True errror	Resub	LOO	BST	mBOL	gBOL	gsBOL
Features	Sample size	M ± SD	M_{diff} ± SD (RMS)					
p = 1	n = 5	0.44±0.09	0.07±0.13 (0.15)	-0.03±0.19 (0.19)	0.12±0.11 (0.16)	0.08±0.11 (0.13)	0.08±0.11 (0.14)	0.03±0.12 (0.12)
p = 1	n = 10	0.42±0.07	0.03±0.11 (0.11)	-0.02±0.14 (0.14)	0.06±0.09 (0.11)	0.04±0.10 (0.10)	0.03±0.10 (0.11)	0.01±0.10 (0.10)
p = 1	n = 20	0.40±0.05	-0.01±0.09 (0.09)	-0.03±0.10 (0.10)	0.01±0.08 (0.08)	-0.00±0.09 (0.09)	-0.01±0.09 (0.09)	-0.02±0.09 (0.09)
p = 2	n = 5	0.44±0.07	0.14±0.13 (0.19)	-0.02±0.19 (0.19)	0.22±0.09 (0.24)	0.10±0.10 (0.14)	0.13±0.11 (0.17)	0.09±0.12 (0.15)
p = 2	n = 10	0.42±0.06	0.08±0.11 (0.13)	-0.02±0.15 (0.15)	0.12±0.08 (0.15)	0.06±0.09 (0.11)	0.07±0.10 (0.12)	0.03±0.10 (0.11)
p = 2	n = 20	0.40±0.04	0.03±0.10 (0.10)	-0.01±0.11 (0.11)	0.06±0.08 (0.10)	0.03±0.08 (0.09)	0.03±0.09 (0.09)	0.00±0.09 (0.09)
p = 3	n = 5	0.44±0.07	0.19±0.14 (0.24)	-0.02±0.20 (0.21)	0.29±0.09 (0.30)	0.13±0.10 (0.16)	0.18±0.11 (0.22)	0.14±0.13 (0.19)
p = 3	n = 10	0.41±0.06	0.11±0.11 (0.15)	-0.02±0.15 (0.15)	0.17±0.08 (0.19)	0.08±0.08 (0.12)	0.10±0.10 (0.14)	0.07±0.10 (0.12)
p = 3	n = 20	0.40±0.05	0.05±0.10 (0.11)	-0.02±0.11 (0.12)	0.09±0.08 (0.12)	0.04±0.08 (0.09)	0.05±0.09 (0.10)	0.02±0.09 (0.09)
p = 5	n = 5	0.45±0.07	0.29±0.13 (0.32)	-0.02±0.21 (0.21)	0.37±0.08 (0.38)	0.16±0.09 (0.18)	0.27±0.11 (0.29)	0.24±0.12 (0.27)
p = 5	n = 10	0.42±0.06	0.18±0.11 (0.21)	-0.02±0.16 (0.16)	0.26±0.08 (0.27)	0.11±0.08 (0.13)	0.17±0.09 (0.19)	0.13±0.10 (0.17)
p = 5	n = 20	0.39±0.06	0.09±0.10 (0.13)	-0.01±0.12 (0.12)	0.10±0.08 (0.13)	0.06±0.08 (0.10)	0.09±0.09 (0.13)	0.06±0.09 (0.11)

Complete raw data for both experiments are available at http://www.molbiol.sci.am/big/jbc/Arakelyan_JBC_src.zip (Performance_GSE13255.xls and Performance_GSE26712.xls in archive file).

Discussion

In this paper we have compared the performance of our geometrical bolstered and semi-bolstered error estimation algorithms with their original counterparts [Braga-Neto, 2004], as well as several popular error estimation techniques [Devroye, 1996; Lachenbrucha, 1968; Efron, 1983]. Bolstered error estimation is used for estimation of classification true error and has several advantages over other contemporary algorithms in terms of accuracy and speed. For general cases of classification, the bolstered error is estimated by Monte-Carlo integration, but in specific cases it can be computed exactly by solving the integral-containing equations described in [Braga-Neto, 2004]. Here, we propose a geometric solution to the bolstered error estimation, namely gBOL and gsBOL algorithms, specifically designed for the case of linear classification and spherical bolstering.

There are two key features that are characteristic to a good error estimator: accuracy and speed. In terms of accuracy, both gBOL and gsBOL perform comparable to other commonly used error estimation techniques, such as LOO and BST, as well as the mBOL algorithm, proposed in [Braga-Neto, 2004]. Moreover, gBOL and gsBOL are at least 10-350 times faster than LOO and BST and 5-10 times faster than mBOL depending on the sample size. The speed issue is very important when performing feature selection from extremely high numbers of features. For example, error estimation with LOO for 5-gene feature set takes almost 1 second, while gBOL or gsBOL do the same with comparable accuracy in 0.05 second. This means that for 10000 feature sets (which is a quite common situation, e.g. in microarray gene expression data analysis), a user will spend about 3 hours with LOO, while with gBOL or gsBOL – only about 10 minutes. This advantage will allow for saving time and computing resources, and concentrating more on data analysis results interpretation.

We acknowledge the fact that for more complex classifiers and other kernel types, our approach might not be fully applicable; however, we also recognize that in biomedical research, linear classification is among the most frequently used classification techniques [Schölkopf, 2004; Tarca, 2007; Ben-Hur, 2008]. Thus, gBOL and gsBOL implementation of bolstered error estimation is an optimal choice in the case of linear classifiers. We believe that gBOL and gsBOL will find their users at least in the fields of genomics and structural bioinformatics, where linear classification is being actively used [Lee, 2011; Holloway, 2007; Yang, 2004].

Finally, gBol and gsBOL algorithms do not include any special MATLAB commands or functions and can be easily implemented in any other programming language, like C/C++, R, FORTRAN, and Java.

Conclusions

Taken together, gBOL and gsBOL are very fast error estimation techniques characterized by accuracy comparable with LOO and with lower variance. These algorithms are useful for analyzing extremely large numbers of features and may find their application in various fields of - *omics* data analysis.

Acknowledgments

This study was supported by Armenian National Science and Education foundation (NS molbio-3507 to A.A.).

Bibliography

[Allison, 2006] D. B. Allison, X. Cui, G.P. Page, and M. Sabripour, "Microarray Data Analysis: from Disarray to Consolidation and Consensus," Nature Reviews Genetics, vol. 7, no. 1, pp. 55–65, Jan 2006.

[Ben-Hur, 2008] A. Ben-Hur, C.S. Ong, S. Sonnenburg, B. Schölkopf, and G. Rätsch, "Support Vector Machines and Kernels for Computational Biology," PLoS Computational Biology, vol. 4, no. 10, e1000173, Oct 2008.

[Bonome, 2008] T. Bonome, D.A. Levine, J. Shih, M. Randonovich, C.A. Pise-Masison, F. Bogomolniy, L. Ozbun, J. Brady, J.C. Barrett, J. Boyd, and M.J. Birrer, "A Gene Signature Predicting for Survival in Suboptimally Debulked Patients with Ovarian Cancer," Cancer Research, vol. 68, no. 13, pp. 5478–5486, Jul 2008.

[Braga-Neto, 2004] U. M. Braga-Neto, and E.R. Dougherty, "Bolstered Error Estimation," Pattern Recognition, vol. 37, no. 6, pp. 1267-1281, June 2004.

[Cai, 2003] C. Z. Cai, W.L. Wang, L.Z. Sun, and Y.Z Chen, "Protein Func-tion Classification via Support Vector Machine Approach," Mathematical Biosciences, vol. 185, no. 2, pp. 111-22, Oct 2003.

[Choi, 2011] S. Choi, and K. Han, "Prediction of RNA-binding Amino Acids from Protein and RNA Sequences," BMC Bioinformatics, 12 Supple 13:S7, Nov 2011, doi: 10.1186/1471-2105-12-S13-S7.

[Cui, 2012] G. Cui, C. Fang, and K. Han, "Prediction of Protein-protein Interactions between Viruses and Human by an SVM Model," BMC Bioinformatics, 13 Suppl 7:S5, May 2012, doi: 10.1186/1471-2105-13-S7-S5.

[Devroye, 1996] L. Devroye, L. Györfi, and G. Lugosi, "A Probabilistic Theory of Pattern Recognition", Springer, New-York, 1996.

[Dougherty, 2010] E. R. Dougherty, C. Sima, J. Hua, B. Hanczar, and U.M. Braga-Neto, "Performance of Error Estimators for Classification," Current Bioinformatics, vol. 5, no. 1, 53-67, March, 2010.

[Efron, 1994] B. Efron, and R.Tibshirani, "An Introduction to the Bootstrap", Chapman & Hall, New York, 1994.

[Efron, 1983] B. Efron, "Estimating the Error Rate of a Prediction Rule: Improvement on Cross-Validation," Journal of the American Statistical Association, vol. 78, no. 382, pp. 316-333, 1983.

[Golub, 1999] T. R. Golub, D.K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J.P. Mesirov, H. Coller, M.L. Loh, J.R. Downing, M.A. Caligiuri, C.D. Bloomfield, and E.S. Lander, "Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring,"Science, vol. 286, no. 5439, pp. 531-7, Oct 1999.

[Hastie, 2009] T. Hastie, R.Tibshirani, and J. Friedman, "The Elements of Statistical Learning", Second Edition: Data Mining, Inference, and Prediction, Springer, New York, 2009.

[Holloway, 2007] D.T. Holloway, M. Kon, and C. Delisi, "Machine Learning for Regulatory Analysis and Transcription Factor Target Prediction in Yeast," Systems and Synthetic Biology, vol. 1, no. 1, pp. 25-46, Mar 2007.

[Lachenbrucha, 1968] P.A. Lachenbrucha, and M.R. Mickey, "Estimation of Error Rates in Discriminant Analysis", Technometrics, vol. 10, no. 1, pp. 1-11, 1968

[Lee, 2011] D. Lee, R. Karchin, and M.A. Beer, "Discriminative Prediction of Mammalian Enhancers from DNA Sequence," Genome Research, vol. 21, no. 12, pp. 2167-80, Dec 2011.

[Li, 2011] S. Li, "Concise Formulas for the Area and Volume of a Hyperspherical Cap," Journal of Mathematics & Statistics, vol. 4, no. 1, pp. 66-70, 2011.

[Maulik, 2013] U. Maulik, A. Mukhopadhyay, and D. Chakraborty, "Gene-Expression-Based Cancer Subtypes Prediction Through Feature Selection and Transductive SVM," IEEE Transactions on Bio-Medical Engineering, vol. 60, no. 4, pp. 1111-7, Apr 2013.

[Nguyen, 2011] M.N. Nguyen, J.M. Zurada, and J.C. Rajapakse, "Toward Better Understanding of Protein Secondary Structure: Extracting Prediction Rules," IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 8, no. 3, pp. 858-64, May 2011.

[Nolan, 1997] J.R. Nolan, "Estimating the True Performance Of Classifica-tion-Based NLP Technology," Proc. From Research to Commercial Applications: Making NLP Work in Practice, pp.23-28, 1997.

[Nolan, 1997] T.R. Nolan, "DISXPERT: A Rule-Based Vocational Rehabilitation Risk Assessment System," Expert Systems with Applications, vol. 12, no. 4, pp. 465-472, May 1997.

[Okun, 2011] O. Okun, Feature Selection and Ensemble Methods for Bioinformatics: Algorithmic Classification and Implementations. Medical Information Science Reference, Hershey, 2011.

[Sayes, 2007] Y.Saeys, I. Inza, and P. Larrañaga, "A Review of Feature Selection Techniques in Bioinformatics," Bioinformatics, vol. 23, no. 19, pp. 2507-17, Oct 2007.

[Schölkopf, 2004] B. Schölkopf, K.Tsuda, and J.P. Vert, Kernel methods in computational biology. MIT Press, Cambridge, 2004.

[Showe, 2009] M.K. Showe, A. Vachani, A.V. Kossenkov, M. Yousef, C. Nichols, E.V. Nikonova, C. Chang, J. Kucharczuk, B. Tran, E. Wakeam, T.A. Yie, D. Speicher, W.N. Rom, S. Albelda, and L.C. Showe, "Gene Expression Profiles in Peripheral Blood Mononuclear Cells Can Distinguish Patients with Non-small Cell Lung Cancer from Patients with Nonmalignant Lung Disease," Cancer Research, vol. 69, no. 24, pp. 9202-10, Dec 2009.

[Tarca, 2007] A. L. Tarca, V.J. Carey, X.W. Chen, R. Romero, and S. Drăghici, "Machine Learning and its Applications to Biology," PLoS Computational Biology, vol. 3, no. 6, pp. e116, Jun 2007.

[Vapnik, 1995] V. Vapnik, "The Nature of Statistical Learning Theory", Springer, New York, 1995.

[Yang, 2004] Z. R. Yang, "Biological Applications of Support Vector Ma-chines," Briefings in bioinformatics, vol. 5, no. 4, pp. 328-38, Dec 2004.

Authors' Information

Arsen Arakelyan *– Bioinformatics group of the Institute of Molecular Biology NAS RA, Laboratory of Information Biology of the the Institute of Molecular Biology and Institute for Informatics and Automation Problems NAS RA, 7 Hasratyan St, 0014, Yerevan, Armenia, email: aarakelyan@sci.am*

Major Fields of Scientific Research: Algorithm development foe gene expression analysis, gene network analysis and modeling

Lilit Nersisyan *- Bioinformatics group of the Institute of Molecular Biology NAS RA, 7 Hasratyan St, 0014, Yerevan, Armenia, email: l_nersisyan@mb.sci.am*

Major Fields of Scientific Research: Algorithm development foe gene expression analysis, gene network analysis and modeling

Aram Gevorgyan *- Bioinformatics group of the Institute of Molecular Biology NAS RA, 7 Hasratyan St, 0014, Yerevan, Armenia, email: imb@sci.am*

Major Fields of Scientific Research: Algorithm development foe gene expression analysis, gene network analysis and modeling

Anna Boyajyan *– Department of Applied Molecular Biology of the Institute of Molecular Biology NAS RA, Laboratory of Information Biology of the Institute of Molecular Biology and Institute for Informatics and Automation Problems NAS RA 7 Hasratyan St, 0014, Yerevan, Armenia, email: aboyajyan@sci.am*

Major Fields of Scientific Research: Genomics & Immunomics