ESANN 2015 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 22-24 April 2015, i6doc.com publ., ISBN 978-287587014-8. Available from http://www.i6doc.com/en/.

Model Selection for Big Data: Algorithmic Stability and Bag of Little Bootstraps on GPUs

Luca Oneto[1], Bernardo Pilarz[1], Alessandro Ghio[2], and Davide Anguita[2]

1 - DITEN - University of Genova
Via Opera Pia 11A, I-16145 Genova - Italy

2 - DIBRIS - University of Genova
Via Opera Pia 13, I-16145 Genova - Italy

Abstract. Model selection is a key step in learning from data, because it allows to select optimal models, by avoiding both under- and over-fitting. However, in the Big Data framework, the effectiveness of a model selection approach is assessed not only through the accuracy of the learned model but also through the time and computational resources needed to complete the procedure. In this paper, we propose two model selection approaches for Least Squares Support Vector Machine (LS-SVM) classifiers, based on Fully-empirical Algorithmic Stability (FAS) and Bag of Little Bootstraps (BLB). The two methods scale sub-linearly respect to the size of the learning set and, therefore, are well suited for big data applications. Experiments are performed on a Graphical Processing Unit (GPU), showing up to 30x speed-ups with respect to conventional CPU-based implementations.

1 Introduction

In the Big Data Era [1], transforming large amounts of data into actionable knowledge in a feasible time frame is a key task to map large investments in database storage into an actual advantage for final users. Learning algorithms must then be able to handle big data by optimizing economic sustainability aspects, which result in resource, time, and accuracy constraints [2, 3, 4, 5]. Two challenges consequently arise: (i) to train and select accurate models (i.e. to choose an effective model selection strategy); (ii) to deploy such strategy onto computing systems, which allow optimizing cost-to-performance ratio [3].

Concerning challenge (i), in the supervised binary classification learning framework, model selection addresses the problem of choosing the most suitable classifier given the available data, by properly tuning one or more hyperparameters in order to avoid either under- or overfitting [6]. For this purpose, in this paper we exploit two recent theoretical results, namely Bag of Little Bootstraps (BLB) [7, 8] and Fully-empirical Algorithmic Stability (FAS) [9, 10, 11]. They both allow to effectively implement model selection strategies with memory requirements and computational complexity proportional to $\sqrt{n}$, where n is the number of available samples, so ensuring sub-linear scalability.

Concerning challenge (ii), distributing the learning effort on different machines is fundamental to allow limiting the computational burden related to the analysis of large data volumes. Nevertheless, costs could be remarkably affected

ESANN 2015 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 22-24 April 2015, i6doc.com publ., ISBN 978-287587014-8. Available from http://www.i6doc.com/en/.

by the exploitation of several parallel workstations. In order to avoid giving up parallelism while limiting costs, in the last years Graphical Processing Units (GPUs) have been exploited to speed-up computations [12, 13, 14, 15, 16], as they allow to optimize the cost-to-performance ratio with respect to conventional CPUs.

In this paper, we deal with both challenges. In particular, we consider one state-of-the-art classification algorithm, namely the Least Squares Support Vector Machine (LS-SVM) [17], and we propose an implementation strategy for BLB and FAS model selection approaches on GPUs. Comparative benchmarks on real world datasets, performed on both GPUs and conventional CPUs, show the effectiveness of the proposed methods: GPU-based implementations can achieve a 30x speed-up with respect to their CPU-based counterparts. In particular, FAS results to require less resources than BLB, without affecting the performance of the final classifier.

2 FAS and BLB Model Selection Strategies

Let $\mathcal{S}_n : \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n\}$ be a set of n i.i.d. patterns $\boldsymbol{z}_i{=}(\boldsymbol{x}_i, y_i)$, where $\boldsymbol{x}_i \in \mathbb{R}^d$ and $y_i \in \{\pm 1\}$, sampled from an unknown distribution μ. A learning algorithm $\mathcal{A}$, characterized by a set of hyperparameters $\mathcal{H}$, allows training a model $f = \mathcal{A}_{(\mathcal{S}_n,\mathcal{H})}$ from the available data. The objective of a model selection procedure is to identify the best configuration $\mathcal{H}^*$ for the model hyperparameters. This task can be accomplished by finding the model that minimizes the generalization error of f, namely the error that f will perform on all data generated by μ. Unfortunately, the generalization error cannot be computed in practice, since μ is unknown: different approaches have been thus proposed to estimate the performance of a model based on a finite dataset [6].

One recently proposed procedure, the Fully-empirical Algorithmic Stability (FAS), relies on measuring the ability of an algorithm to select similar models, even if the training data are (slightly) modified: this ensures that the algorithm is actually learning from data, without overfitting them. Let $\mathcal{S}_n^{\setminus i} = \mathcal{S}_n \setminus \{\boldsymbol{z}_i\}$ be the set, where the i-th pattern is removed. Let also $\hat{L}_n^{\text{loo}}\left(\mathcal{A}_{(\mathcal{S}_n,\mathcal{H})}, \mathcal{S}_n\right) = {}^1\!/_n \sum_{i=1}^n \ell(\mathcal{A}_{(\mathcal{S}_n^{\setminus i},\mathcal{H})}, \boldsymbol{z}_i)$ be the Leave-One-Out (LOO) error, where $\ell(.,.)$ is a suitable loss function [9]. Then, the following model selection procedure can be defined [9]:

$$\mathcal{H}^* : \arg\min_{\mathcal{H}\in\mathcal{G}} \left\{ \hat{L}_n^{\text{loo}}(\mathcal{A}_{(\mathcal{S}_n,\mathcal{H})}, \mathcal{S}_n) + \sqrt{\tfrac{2}{\delta}\left[\tfrac{1}{\sqrt{n}} + 3\left(\hat{H}_{\text{loo}}(\mathcal{A}_{(\mathcal{S}_{\sqrt{n}/2},\mathcal{H})}, \mathcal{S}_{\sqrt{n}/2}) + \sqrt{\tfrac{\log(2/\delta)}{\sqrt{n}}}\right)\right]} \right\} \quad (1)$$

where $\hat{H}_{\text{loo}}\left(\mathcal{A}_{(\mathcal{S}_{\sqrt{n}/2},\mathcal{H})}, \mathcal{S}_{\sqrt{n}/2}\right)$ is the Empirical Hypothesis Stability:

$$\hat{H}_{\text{loo}}(\mathcal{A}_{(\mathcal{S}_{\sqrt{n}/2},\mathcal{H})}, \mathcal{S}_{\sqrt{n}/2}) = \tfrac{8}{n\sqrt{n}} \sum_{i,j,k=1}^{\sqrt{n}/2} \left|\ell(\mathcal{A}_{(\breve{\mathcal{S}}^k_{\sqrt{n}/2},\mathcal{H})}, \breve{\boldsymbol{z}}_j^k) - \ell(\mathcal{A}_{(\breve{\mathcal{S}}^{k\setminus i}_{\sqrt{n}/2},\mathcal{H})}, \breve{\boldsymbol{z}}_j^k)\right| \quad (2)$$

In Eq. (2), $\breve{\mathcal{S}}^k_{\sqrt{n}/2} : \{\boldsymbol{z}_{(k-1)\sqrt{n}+1}, \ldots, \boldsymbol{z}_{(k-1)\sqrt{n}+\sqrt{n}/2}\}$, $\breve{\boldsymbol{z}}_j^k : \boldsymbol{z}_{(k-1)\sqrt{n}+\sqrt{n}/2+j}$, and $k \in \{1, \ldots, \sqrt{n}/2\}$. Every quantity involved in the bound can be computed from

ESANN 2015 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 22-24 April 2015, i6doc.com publ., ISBN 978-287587014-8. Available from http://www.i6doc.com/en/.

the available data [10, 9], and sets of smaller cardinality are involved in the derivation of the bound: this is particularly appealing for big data applications. Note also that $\hat{H}_{\text{loo}}(\mathcal{A}_{(\mathcal{S}_{\sqrt{n}/2},\mathcal{H})}, \mathcal{S}_{\sqrt{n}/2})$ can be effectively estimated via a Monte Carlo procedure: this enables computing a subset s_{MC} of the required steps, i.e. $s_{MC} \ll \frac{n\sqrt{n}}{8}$.

The Bag of Little Bootstraps (BLB) approach [8, 7] represents an alternative to FAS, which builds on the conventional Bootstrap procedure [18] by considering in turn only $b = n^\gamma$ data, with $\gamma \in [1/2, 1]$, in place of the whole dataset. In particular, BLB consists in sampling b_s times $\mathcal{S}_n$ without replacement, so to create couples of datasets $\mathcal{L}_b^j$ and $\mathcal{T}_b^j$ ($j \in \{1, \dots, b_s\}$), each consisting of $b \in [\sqrt{n}, n]$ data. Then, each $\mathcal{L}_b^j$ is sampled with replacement b_b times, so to derive $\mathcal{B}_n^{j,k}$ datasets ($k \in \{1, \dots, b_b\}$), each consisting of b samples. Finally, models are trained on the sets $\mathcal{B}_n^{j,k}$ and tested on the corresponding $\mathcal{T}_b^j$, so to define the following model selection procedure:

$$\mathcal{H}^* : \arg\min_{\mathcal{H}\in\mathcal{G}} \frac{1}{b_s b_b b} \sum_{j=1}^{b_s} \sum_{k=1}^{b_b} \sum_{\boldsymbol{z}\in\mathcal{T}_b^j} \ell(\mathcal{A}_{(\mathcal{B}_n^{j,k},\mathcal{H})}, \boldsymbol{z}). \tag{3}$$

3 CPU-based and GPU-based LS-SVM Model Selection

Least Squares Support Vector Machines (LS-SVM) [17] is a state-of-the-art algorithms for classification. LS-SVM is preferred to other approaches since its training phase can be easily parallelized on different architectures [19, 12], resulting in effective implementations especially when the input space dimensions are small with respect to the number of samples.

We focus in this paper on linear classifiers $f(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + b$, where $\boldsymbol{w} \in \mathbb{R}^d$ and $b \in \mathbb{R}$, since they are suitable for big data purposes [20]. The LS-SVM classifier is trained by solving the following linear system:

$$([X^{\mathbf{1}}]^T X^{\mathbf{1}} + \lambda I^0)[\boldsymbol{w}^T, b]^T = [X^{\mathbf{1}}]^T \boldsymbol{y} \tag{4}$$

where $X = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_n]^T$, $X^{\mathbf{1}} = [X, \mathbf{1}]$, $\boldsymbol{y} = [y_1, \dots, y_n]^T$, $\mathbf{1} = \{1, ..., 1\}$ is a n-dimensional array, and I^0 is a $(d+1)\times(d+1)$ diagonal matrix with $I_{0,0} = 0$. Moreover, $\lambda > 0$ is a hyperparameter that balances the tradeoff between over and under-fitting. Then, in this framework, $\mathcal{A} = \text{LS-SVM}$ and $\mathcal{H} = \{\lambda\}$.

The model selection strategies introduced in Section 2 require that some LS-SVM models are trained on sets of different cardinalities. BLB relies on $b_s \cdot b_b$ sets of $b \in [\sqrt{n}, n]$ patterns: in big data applications, usually choosing $b = \sqrt{n}$ (i.e. $\gamma = 1/2$) is sufficient to guarantee a good trade-off between computational time and accuracy . FAS works on sets consisting of $\sqrt{n}/2$ samples; it also requires to compute the LOO error, which can be derived with a small effort through a decremental unlearning algorithm [21, 12]. As a consequence, when n is large, the computational burden is remarkably reduced with respect to conventional approaches, like standard Bootstrap or Cross Validation [6].

The CPU-based implementations of BLB and FAS are straightforward to deploy. However, modern GPU systems outperform CPU architectures in terms of

ESANN 2015 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 22-24 April 2015, i6doc.com publ., ISBN 978-287587014-8. Available from http://www.i6doc.com/en/.

cost-to-performance ratio for highly-parallel and computational-intensive workloads [16]: this is enabled by larger memory bandwidth and FLoating point Operations Per Second (FLOPS) values, and by the possibility of exploiting several parallel pipelines to run programs in Single Instruction Multiple Data (SIMD) mode. In particular, when dealing with BLB and FAS model selection, we exploit the main GPU features to:

- Solve Eq. (4) in a parallel fashion, through the use of cuBLAS libraries[1];
- Train the different models for FAS and BLB model selection, so to saturate the intrinsic parallelism capabilities of GPUs;
- Find the LOO error for FAS, through a parallel decremental unlearning procedure for LS-SVM [12].

4 Experimental Results and Discussion

We test BLB and FAS model selection strategies on two well known real-world datasets: Mnist [22] (10 digit recognition task, 28×28 pixels images, 60000 samples), and NotMnist [23] (A-to-J characters recognition task, 28×28 pixels images, 550000 samples). Since we are dealing with binary classification, in case of multi-class datasets we adopt the One Vs. One (OVO) procedure [9] in order to derive $m(m-1)/2$ binary classification problems, where m is the number of classes. We use $n = \{10^2, 10^3, 10^4\}$ training samples for both datasets, while we also performed experiments with $n = 10^5$ on NotMnist; the unselected data are used as reference set for computing the error of the selected model. We search for λ among 20 values in the range $[10^{-5}, 10^2]$, equally spaced in logarithmic scale [9]. Concerning the experimental setup for FAS and BLB, we tested $s_{MC} \in \{50, 100, 200\}$ and $b_s = b_b = \{7, 10, 14\}$: for each value, experiments are replicated 10 times to generate statistically relevant results. Tests have been performed on a PC equipped with Windows 8.1 x64, mounting an Intel i7 3820 3.6 GHz CPU, 16 GB @1.6GHz RAM, 1TB 7200rpm @6Gb/s Hard Disk, and a GeForce GTX 690 (2x GK104-355-A2 @1 GHz) GPU board.

Table 1 presents the results. In particular, we report the average error rate on the reference sets: since we verified that this quantity is not remarkably influenced by the variations of s_{MC}, b_s, and b_b, due to space constraints we only report results for $s_{MC} = 100$ and $b_s = b_b = 10$. Table 1 also shows the computational time (in seconds) needed by FAS and BLB to complete model selection, as s_{MC}, b_s, b_b, and n are varied, on CPU-based and GPU-based architectures (T_{CPU} and T_{GPU}, respectively): U is the relative speed-up obtained by exploiting GPUs. The following conclusions can be drawn:

- FAS and BLB allow choosing models, characterized by similar errors;
- GPU-based model selection procedures are much faster than CPU-based ones (up to 30x speed-up);
- On average, FAS can be parallelized to higher extents than BLB: as a consequence, FAS is faster and requires less resources overall.

[1] `https://developer.nvidia.com/cublas`.

ESANN 2015 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 22-24 April 2015, i6doc.com publ., ISBN 978-287587014-8.
Available from http://www.i6doc.com/en/.

Error on the reference set with $s_{MC} \in 100$ and $b_s = b_b = 10$															
n	10^2		10^3		10^4		n	10^2		10^3		10^4		10^5	
OVO	FAS	BLB	FAS	BLB	FAS	BLB	OVO	FAS	BLB	FAS	BLB	FAS	BLB	FAS	BLB
0vs1	**0.49**	**0.47**	0.50	**0.45**	0.52	**0.31**	AvsB	**10.38**	**10.35**	**11.87**	**11.23**	**7.48**	**7.44**	**6.50**	**6.50**
0vs2	3.91	**2.82**	**1.98**	**2.17**	**1.54**	1.83	AvsC	**8.57**	**8.51**	**9.19**	**8.34**	**5.85**	**5.83**	**5.10**	**5.10**
0vs3	**2.49**	**2.29**	**1.23**	**1.25**	**0.89**	1.16	AvsD	**10.88**	**10.78**	**11.07**	**10.24**	**7.18**	**7.16**	**6.01**	**6.01**
0vs4	**1.47**	1.67	**0.66**	1.03	**0.57**	**0.57**	AvsE	**9.63**	**9.57**	**10.83**	**10.34**	**6.84**	**6.80**	**5.74**	**5.74**
0vs5	**3.86**	**4.12**	**2.25**	**2.22**	**1.38**	1.96	AvsF	**9.65**	**9.57**	**10.68**	**9.20**	**6.05**	**6.00**	**5.09**	**5.09**
0vs6	3.08	**2.76**	**1.52**	**1.62**	**1.13**	1.35	AvsG	**11.21**	**11.07**	**11.22**	**11.41**	**7.35**	**7.31**	**6.30**	**6.30**
0vs7	**1.92**	**1.85**	**0.58**	1.00	**0.47**	0.73	AvsH	**12.92**	**12.91**	**14.12**	**12.81**	**9.52**	**9.49**	**8.50**	**8.50**
0vs8	**2.29**	**2.20**	1.77	**1.56**	**1.32**	**1.32**	AvsI	**11.97**	**11.88**	**12.15**	**11.33**	**8.43**	**8.44**	**7.46**	**7.46**
0vs9	**1.71**	2.08	**1.03**	1.35	**0.77**	1.24	AvsJ	**11.39**	**11.17**	**11.20**	**10.26**	**7.47**	**7.44**	**6.87**	**6.88**
1vs2	4.62	**3.71**	**2.22**	**2.37**	**1.77**	2.22	BvsC	**10.86**	**10.76**	**10.39**	**9.36**	**6.42**	**6.40**	**5.35**	**5.34**
1vs3	**3.03**	**3.07**	**1.76**	3.10	**1.54**	**1.67**	BvsD	**13.60**	**13.49**	**13.04**	**12.52**	**9.40**	**9.37**	**8.06**	**8.06**
1vs4	**1.39**	**1.51**	**0.68**	**0.74**	**0.41**	0.53	BvsE	**12.91**	**12.83**	**12.61**	**11.77**	**8.20**	**8.15**	**7.09**	**7.10**
1vs5	**2.04**	**1.94**	**1.13**	1.32	**1.06**	**1.12**	BvsF	**10.03**	**9.98**	**10.00**	**9.67**	**6.79**	**6.75**	**5.77**	**5.77**
1vs6	**1.12**	**1.17**	0.80	**0.70**	**0.47**	**0.51**	BvsG	**12.62**	**12.59**	**13.56**	**12.03**	**8.17**	**8.12**	**6.86**	**6.87**
1vs7	**2.15**	**2.24**	**1.11**	1.59	**0.89**	1.15	BvsH	**10.92**	**10.90**	**11.24**	**11.27**	**7.85**	**7.78**	**6.69**	**6.69**
1vs8	6.61	**5.33**	**4.39**	**4.06**	**3.58**	**3.81**	BvsI	**13.03**	**12.92**	**12.46**	**13.19**	**9.21**	**9.20**	**8.34**	**8.34**
1vs9	**1.41**	1.77	**0.82**	1.03	**0.44**	0.72	BvsJ	**10.19**	**10.21**	**10.62**	**10.03**	**7.15**	**7.11**	**6.30**	**6.30**
2vs3	**7.14**	**6.57**	4.77	**4.08**	**3.19**	3.66	CvsD	**8.49**	**8.47**	**9.46**	**8.66**	**6.11**	**6.08**	**5.19**	**5.19**
2vs4	4.11	**3.51**	**2.60**	**2.38**	**1.72**	2.51	CvsE	**13.16**	**13.09**	**13.79**	**13.22**	**9.63**	**9.61**	**8.66**	**8.66**
2vs5	6.85	**5.73**	3.65	**3.21**	**2.37**	2.76	CvsF	**8.11**	**8.06**	**8.55**	**8.14**	**5.77**	**5.73**	**5.06**	**5.07**
2vs6	6.62	**5.49**	**3.02**	3.39	**1.96**	2.93	CvsG	**13.57**	**13.52**	**15.01**	**13.13**	**9.27**	**9.25**	**7.93**	**7.93**
2vs7	5.07	**4.03**	3.16	**2.82**	**1.58**	2.69	CvsH	**7.98**	**7.94**	**8.82**	**8.39**	**5.94**	**5.89**	**5.05**	**5.05**
2vs8	8.65	**6.57**	4.83	**3.98**	**3.17**	3.71	CvsI	**10.26**	**10.14**	**10.62**	**9.43**	**7.12**	**7.10**	**6.12**	**6.12**
2vs9	**4.28**	**4.16**	**2.19**	2.71	**1.44**	2.06	CvsJ	**8.53**	**8.40**	**9.53**	**8.73**	**6.25**	**6.21**	**5.05**	**5.05**
3vs4	2.64	**1.87**	**1.35**	1.54	**0.90**	1.16	DvsE	**10.62**	**10.57**	**10.62**	**9.80**	**6.70**	**6.66**	**5.72**	**5.72**
3vs5	**11.45**	**12.24**	7.20	**5.97**	**4.67**	5.84	DvsF	**9.20**	**9.12**	**9.74**	**9.08**	**6.24**	**6.20**	**5.21**	**5.21**
3vs6	3.16	**2.09**	1.56	**1.39**	**0.85**	1.11	DvsG	**11.26**	**11.18**	**11.11**	**10.69**	**7.40**	**7.38**	**6.15**	**6.15**
3vs7	4.05	**3.46**	**2.36**	**2.51**	**1.69**	2.25	DvsH	**11.62**	**11.42**	**10.72**	**10.23**	**7.25**	**7.19**	**6.16**	**6.16**
3vs8	**10.85**	**9.83**	5.65	**4.82**	**3.93**	4.57	DvsI	**11.33**	**11.26**	**11.80**	**10.79**	**8.52**	**8.50**	**7.49**	**7.48**
3vs9	**4.37**	**4.50**	**3.06**	3.57	**2.24**	2.87	DvsJ	**11.28**	**11.14**	**10.79**	**10.25**	**7.36**	**7.35**	**6.17**	**6.18**
4vs5	3.32	**2.92**	**1.85**	**1.70**	**1.34**	**1.41**	EvsF	**10.64**	**10.53**	**12.42**	**10.96**	**7.68**	**7.64**	**6.52**	**6.52**
4vs6	**2.19**	**2.00**	1.76	**1.45**	**0.98**	1.21	EvsG	**12.28**	**12.25**	**13.55**	**11.58**	**8.48**	**8.46**	**7.51**	**7.51**
4vs7	4.28	**3.48**	3.18	**2.11**	**1.58**	2.06	EvsH	**11.00**	**10.91**	**11.30**	**10.59**	**7.37**	**7.34**	**6.17**	**6.18**
4vs8	**2.70**	**2.58**	**1.72**	**1.56**	**1.01**	1.37	EvsI	**13.88**	**13.80**	**13.49**	**12.77**	**9.66**	**9.64**	**8.74**	**8.74**
4vs9	**9.30**	**8.44**	**5.32**	**5.13**	**3.82**	4.76	EvsJ	**11.12**	**10.88**	**10.57**	**10.17**	**7.01**	**6.96**	**5.98**	**5.99**
5vs6	6.99	**5.38**	**3.49**	**3.28**	**2.59**	**2.85**	FvsG	**9.66**	**9.56**	**10.44**	**9.44**	**6.59**	**6.56**	**5.78**	**5.78**
5vs7	3.16	**2.56**	1.77	**1.44**	**0.93**	1.05	FvsH	**10.33**	**10.19**	**10.87**	**10.01**	**6.89**	**6.80**	**5.56**	**5.56**
5vs8	**8.71**	**8.13**	**6.10**	**5.70**	**4.26**	5.47	FvsI	**10.96**	**10.85**	**12.37**	**10.52**	**8.15**	**8.10**	**7.08**	**7.07**
5vs9	**4.31**	**4.27**	2.82	**2.37**	**1.69**	2.23	FvsJ	**10.98**	**10.95**	**10.67**	**10.28**	**7.38**	**7.33**	**6.67**	**6.68**
6vs7	1.23	**0.99**	0.49	**0.29**	0.25	**0.18**	GvsH	**10.62**	**10.57**	**10.48**	**10.29**	**7.09**	**7.03**	**6.38**	**6.38**
6vs8	3.04	**2.39**	2.12	**1.89**	**1.70**	**1.73**	GvsI	**12.56**	**12.50**	**13.11**	**11.46**	**8.75**	**8.74**	**7.85**	**7.85**
6vs9	0.93	**0.78**	0.65	**0.42**	**0.34**	0.42	GvsJ	**10.44**	**10.40**	**10.82**	**10.21**	**7.20**	**7.16**	**6.50**	**6.50**
7vs8	**3.24**	**3.01**	**1.68**	1.96	**1.13**	1.56	HvsI	**12.62**	**12.54**	**12.51**	**12.27**	**9.22**	**9.23**	**8.26**	**8.27**
7vs9	**10.16**	**9.79**	**6.46**	**5.97**	**4.52**	5.46	HvsJ	**9.81**	**9.62**	**10.42**	**9.89**	**7.00**	**6.98**	**5.99**	**5.99**
8vs9	**5.22**	**4.77**	**3.08**	**3.24**	**2.69**	3.12	IvsJ	**14.93**	**14.87**	**14.11**	**13.36**	**10.57**	**10.53**	**9.76**	**9.76**
$s_{MC} \in 50$ and $b_s = b_b = 7$															
T_{CPU}	**286**	286	**286**	288	**287**	290	T_{CPU}	**285**	287	**286**	289	**287**	292	**293**	301
T_{GPU}	**9**	11	**9**	11	**9**	12	T_{GPU}	**9**	10	**9**	11	**9**	12	**9**	16
U	33	26	33	25	32	23	U	32	28	32	26	32	23	32	19
$s_{MC} \in 100$ and $b_s = b_b = 10$															
T_{CPU}	**286**	286	**286**	288	**287**	290	T_{CPU}	**285**	287	**286**	289	**287**	292	**293**	301
T_{GPU}	**9**	11	**9**	11	**9**	12	T_{GPU}	**9**	10	**9**	11	**9**	12	**9**	16
U	33	26	33	25	32	23	U	32	28	32	26	32	23	32	19
$s_{MC} \in 200$ and $b_s = b_b = 14$															
T_{CPU}	571	**559**	571	**561**	574	**568**	T_{CPU}	571	**563**	573	**565**	581	**569**	**581**	587
T_{GPU}	**20**	**20**	**20**	22	**20**	22	T_{GPU}	**20**	20	**20**	22	**20**	22	**21**	28
U	28	28	28	26	28	26	U	29	28	28	26	28	26	28	21

Table 1: Results on Mnist and NotMnist datasets. Bold face indicates highest statistical significance with respect to Student's t-test.

Future researches will extend the work to the kernel version of the exploited algorithm, thus enabling effective analysis also of high dimensional datasets.

References

[1] V. Mayer-Schönberger and K. Cukier. *Big data: A revolution that will transform how we live, work, and think.* Houghton Mifflin Harcourt, 2013.

[2] O. Bousquet and L. Bottou. The tradeoffs of large scale learning. In *Neural Information Processing Systems*, 2008.

[3] A. Ghio and L. Oneto. Byte the bullet: Learning on real-world computing architectures. In *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2014.

[4] M. I. Jordan. On the computational and statistical interface and big data. In *Conference on Learning Theory*, 2014.

[5] L. Oneto, A. Ghio, S. Ridella, J. L. Reyes-Ortiz, and D. Anguita. Out-of-sample error estimation: the blessing of high dimensionality. In *IEEE International Conference on Data Mining, International Workshop on High Dimensional Data Mining*, 2014.

[6] D. Anguita, A. Ghio, L. Oneto, and S. Ridella. In-sample and out-of-sample model selection and error estimation for support vector machines. *IEEE Transactions on Neural Networks and Learning Systems*, 23(9):1390–1406, 2012.

[7] A. Kleiner, A. Talwalkar, P. Sarkar, and M. I. Jordan. The big data bootstrap. In *International conference on Machine learning*, 2012.

[8] A. Kleiner, A. Talwalkar, P. Sarkar, and M. I. Jordan. A scalable bootstrap for massive data. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(4):795–816, 2014.

[9] L. Oneto, A. Ghio, S. Ridella, and D. Anguita. Fully empirical and data-dependent stability-based bounds. *IEEE Transactions on Cybernetics*, 10.1109/TCYB.2014.2361857:in–press, 2014.

[10] O. Bousquet and A. Elisseeff. Stability and generalization. *The Journal of Machine Learning Research*, 2:499–526, 2002.

[11] T. Poggio, R. Rifkin, S. Mukherjee, and P. Niyogi. General conditions for predictivity in learning theory. *Nature*, 428(6981):419–422, 2004.

[12] T.N. Do, V.H. Nguyen, and F. Poulet. Speed up svm algorithm for massive classification tasks. In *Advanced Data Mining and Applications*, 2008.

[13] M. Heeswijk, Y. Miche, E. Oja, and A. Lendasse. Gpu-accelerated and parallelized elm ensembles for large-scale regression. *Neurocomputing*, 74(16):2430–2437, 2011.

[14] J. D. Owens, M. Houston, D. Luebke, S. Green, J. E. Stone, and J. C. Phillips. Gpu computing. *Proceedings of the IEEE*, 96(5):879–899, 2008.

[15] B. Catanzaro, N. Sundaram, and K. Keutzer. Fast support vector machine training and classification on graphics processors. In *International conference on Machine learning*, 2008.

[16] A. Gieseke, K. L. Polsterer, C. E. Oancea, and C. Igel. Speedy greedy feature selection: Better redshift estimation via massive parallelism. In *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2014.

[17] J. A. K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle. *Least squares support vector machines.* World Scientific, 2002.

[18] B. Efron. *The jackknife, the bootstrap and other resampling plans.* SIAM, 1982.

[19] T.N. Do and F. Poulet. Classifying one billion data with a new distributed svm algorithm. In *IEEE International Conference on Computer Science, Research, Innovation and Vision for the Future*, 2006.

[20] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874, 2008.

[21] P. J. Green and B. W. Silverman. *Nonparametric regression and generalized linear models: a roughness penalty approach.* CRC Press, 1993.

[22] C. Bottou, L.and Cortes, J. S. Denker, H. Drucker, I. Guyon, L. D. Jackel, Y. LeCun, U. A. Muller, E. Sackinger, and P. Simard. Comparison of classifier methods: a case study in handwritten digit recognition. In *International Conference on Pattern Recognition, Conference B: Computer Vision & Image Processing*, 1994.

[23] B. Yaroslav. Notmnist dataset. Technical report, Google (Books/OCR), 2011.