Neural Comput & Applic
DOI 10.1007/s00521-014-1558-4

Absent extreme learning machine algorithm with application to packed executable identification

Peidai Xie · Xinwang Liu · Jianping Yin · Yongjun Wang

Received: 23 November 2013 / Accepted: 7 February 2014

Abstract Extreme learning machine (ELM) has been an important research topic over the last decade due to its high efficiency, easy-implementation, unification of classification and regression, and unification of binary and multi-class learning tasks. Though integrating these advantages, existing ELM algorithms cannot directly handle the case where some features of the samples are missing or unobserved, which is usually very common in practical applications. The work in this paper fills this gap by proposing an absent ELM (A-ELM) algorithm to address the above issue. By observing the fact that some structural characteristics of a part of packed malware instances hold unreasonable values, we cast the packed executable identification tasks into an absence learning problem, which can be efficiently addressed via the proposed A-ELM algorithm. Extensive experiments have been conducted on six UCI data sets and a packed data set to evaluate the performance of the proposed algorithm. As indicated, the proposed A-ELM algorithm is superior to other imputation algorithms and existing state-of-the-art ones.

Keywords Extreme learning machine · Packed executable identification · Absent learning · Malware

P. Xie (✉) · X. Liu · J. Yin · Y. Wang
College of Computer, National University of Defense Technology, Changsha 410073, China
e-mail: peidaixie@gmail.com

X. Liu
e-mail: 1022xinwang.liu@gmail.com

J. Yin
e-mail: jpyin@nudt.edu.cn

Y. Wang
e-mail: yjwang@nudt.edu.cn

1 Introduction

Extreme learning machine (ELM) was first designed for single hidden layer feedforward neural networks [1–3] and then extended to generalized single hidden layer feedforward networks (SLFN) which did not necessarily resemble neurons [4, 5]. Different from traditional neural SLFN learning algorithms, ELM aims to minimize both training error and the norm of output weights [3, 6]. Due to its (1) *high efficiency*, (2) *easy-implementation*, (3) *unification of classification and regression* and (4) *unification of binary and multi-class classification* [6], ELM has been an active research topic over past a few years [3, 6–12]. In addition, the ELM has also been successfully applied to many applications such as imbalance learning [13], missing data learning [14] and activity recognition [15], to name just a few. More recent advances in ELM can be found in [10–12].

Although researchers have made great progress from both a theoretical and a practical point of view, ELM has still not well considered the issue of incomplete data learning. Specifically, ELM cannot directly handle the learning tasks when there are some features of the samples are missing due to corruption or noise. A direct remedy is to first impute the missing features with zeros, mean value or via EM algorithm [16], and then perform a standard ELM algorithm with the imputed data. However, the above approaches may not produce accurate imputed values, especially in the presence of extensive missing. Worse is that the inaccurate imputation may adversely affect the sequential classification tasks, leading to poor classification performance.

In this paper, we propose a novel algorithm, termed as absent ELM (A-ELM), to solve the above issue. Different from the above imputation approaches, our algorithm

Published online: 26 February 2014

classifies the samples with absent features directly without imputation. Specifically, each sample is classified by its own normal vector, termed as sample-based normal vector, in its own space. To guarantee the generalization performance of the learned classifier, we minimize the maximum of the norm of all sample-based normal vectors, learning to a difficult optimization problem. We then reformulate it as a convex one, enabling it be efficiently solved via existing convex optimization packages. Comprehensive experiments on six UCI data sets and the packed executable identification data set validate the superior performance of the proposed algorithm.

2 Related work

According to the ELM theory [6, 10], ELM aims to simultaneously minimize the training errors and the norm of output weights. This objective function, for both binary and multi-class classification tasks, can be expressed as follows,

$$\begin{aligned} \min_{\boldsymbol{\beta},\boldsymbol{\xi}} \quad & \frac{1}{2}\|\boldsymbol{\beta}\|_{\mathbf{F}}^2 + \frac{C}{2}\sum_{i=1}^{n}\|\boldsymbol{\xi}_{\cdot i}\|^2 \\ \text{s.t.} \quad & \boldsymbol{\beta}^\top \phi(\mathbf{x}_i) = \mathbf{y}_i - \boldsymbol{\xi}_{\cdot i}, \quad \forall i = 1,\ldots,n. \end{aligned} \tag{1}$$

where $\{(\mathbf{x}_i,\mathbf{y}_i)\}_{i=1}^n$ is a training set, $\phi(\mathbf{x}_i)(i=1,\ldots,n)$ is the hidden-layer output (feature mapping) corresponding to $\mathbf{x}_i$, $\boldsymbol{\beta} \in \mathbb{R}^{|\phi(\cdot)|\times T}$ is the output weights, $\boldsymbol{\xi} \in \mathbb{R}^{T\times n}$ is the training error matrix on training data, $\boldsymbol{\xi}_{\cdot i} = [\xi_{1i}, \xi_{2i}, \ldots, \xi_{Ti}]^\top (1 \le i \le n)$ is the ith column of $\boldsymbol{\xi}, \mathbf{y}_i = [0,\ldots,0,\underset{t}{1},0,\ldots,0]^\top \in \{0,1\}^T$ if $\mathbf{x}_i$ belongs to the tth $(1 \le t \le T)$ class, n and T are the number of training samples and classes, and C is a regularization parameter which trades off the norm of output weights and training errors. $\|\cdot\|_{\mathbf{F}}$ is the Frobenius norm.

The optimization problem in Eq. (1) can be efficiently solved. According to [6], the optimal $\boldsymbol{\beta}^\star$ which minimizes Eq. (1) can be analytically obtained as

$$\boldsymbol{\beta}^\star = \boldsymbol{\Phi}^\top\left(\frac{\mathbf{I}}{C} + \boldsymbol{\Phi}\boldsymbol{\Phi}^\top\right)^{-1}\mathbf{Y}^\top, \tag{2}$$

where $\boldsymbol{\Phi} = [\mathbf{x}_1,\ldots,\mathbf{x}_n]^\top \in \mathbb{R}^{n\times d}$, d is the dimensionality of features, $\mathbf{Y} = [\mathbf{y}_1,\ldots,\mathbf{y}_n] \in \mathbb{R}^{T\times n}$ and $\mathbf{I}$ is an identity matrix.

As can be seen from the above, both the binary and multi-class classification tasks in ELM can be handled via an unified formula Eq. (1). Moreover, Eq. (1) can be analytically solved by a matrix inverse operation, while a constrained quadratic programming problem is required in SVM. This makes the ELM easy and efficient to implement due to the fact that solving a matrix inverse problem is usually much more computationally efficient than solving the same-size constrained QP problem. In addition, it is worth mentioning that though both ELM and least square SVM (LSSVM) [17] share the same objective function as far as the optimization is concerned, there is no bias term deployed in ELM, as in Eq. (1). Such a subtle difference makes ELM have milder optimization constraint than LSSVM. These advantages help ELM achieve better classification performance while incurring less computational cost, as demonstrated by the experimental results in [6].

After obtaining the optimal $\boldsymbol{\beta}^\star$, the decision score of the ELM on test point $\mathbf{x}$ is determined by

$$f(\mathbf{x}) = \boldsymbol{\beta}^{\star\top}\mathbf{x}, \tag{3}$$

and the index corresponding to the highest value of $f(\mathbf{x}) \in \mathbb{R}^T$ is considered as the label of $\mathbf{x}$.

As can be seen in Sect. 2, existing ELM algorithms can not handle the applications in the presence of absent features. A straightforward remedy may firstly impute the absent features with zero (known as zero-filling in the literature), mean value (mean-filling) or by more advanced expectation maximization (EM-filling) [16] approaches, and then deploy a standard ELM algorithm on the imputed data. Such imputation methods usually work well when the missing ratio is small. However, these methods may produce inaccurate imputation when the percentage of absence is relatively large. Worse is that such inaccurate imputation could even deteriorates the subsequent classification performance.

3 Proposed absent ELM (A-ELM)

In this section, we propose an absent ELM (A-ELM) algorithm to handle the above issue. Specifically, our algorithm classifies each sample with its observed features directly without any imputation.

3.1 The proposed formulation: sample-based ELM

Existing ELM algorithms assume that the features for all samples are available to train an ELM. However, this assumption will not hold anymore when some features of samples are missing or unobserved, which is very common in practical applications. However, this issue has not been well studied in current literature. In this paper, we propose the concept of sample-based ELM to enable the ELM have the ability to classify samples with absent features. To the best of our knowledge, this is the first time that the concept is proposed in the field.

By supposing the full feature space of samples is $\mathbb{R}^d$, each sample $\mathbf{x}_i$ $(1 \leq i \leq n)$ can be treated as residing in a subspace of that full space, i.e., $\mathbb{R}^{|\mathbf{x}_i|} \subseteq \mathbb{R}^d$. Based on this observation, a natural way to deal with this case is to classify each sample $\mathbf{x}_i$ $(1 \leq i \leq n)$ in its own space, which is the core idea of the sample-based ELM. This idea can be fulfilled as follows,

$$\begin{aligned} \min_{\boldsymbol{\beta}^{(i)}, \xi_i} \quad & \frac{1}{2}\|\boldsymbol{\beta}^{(i)}\|_{\mathbf{F}}^2 + \frac{C}{2}\|\boldsymbol{\xi}_{\cdot i}\|^2 \\ \text{s.t.} \quad & {\boldsymbol{\beta}^{(i)}}^\top \mathbf{x}_i = \mathbf{y}_i - \boldsymbol{\xi}_{\cdot i}, \end{aligned} \tag{4}$$

where $\boldsymbol{\beta}^{(i)}$ is the normal vector in a subspace corresponding to $\mathbf{x}_i$ and $\boldsymbol{\xi}_{\cdot i}$ is the slack variable. Equation (4) is called the ith sample-based ELM. It is worth pointing out that here we only adopt $\phi(\mathbf{x}_i) = \mathbf{x}_i (i = 1, \ldots, n)$ for the safety of illustration. Actually, this approach can be readily extended to nonlinear hidden layer output by the widely used kernel trick.

In order to train a classifier on the absent training set $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$, we assume that a normal vector $\boldsymbol{\beta}$ should be consistent across samples, no matter whether they have the same observed features or not. To this end, $\boldsymbol{\beta}^{(i)}$ is a vector by taking the component of $\boldsymbol{\beta}$ that are relevant for the ith sample, namely those for which $\mathbf{x}_i$ has observed features. As can be seen, different absence among samples may induce different $\|\boldsymbol{\beta}^{(i)}\|_{\mathbf{F}}^2$, which is related to the generalization performance of the learned classifier. Therefore, the maximum of all $\|\boldsymbol{\beta}^{(i)}\|_{\mathbf{F}}^2 (i = 1, \ldots, n)$ should be minimized in order to guarantee better generalization performance. Consequently, the optimization problem of our proposed A-ELM is formulated as follows,

$$\begin{aligned} \min_{\boldsymbol{\beta}, \boldsymbol{\xi}} \quad & \left(\max_{1 \leq i \leq n} \frac{1}{2}\|\boldsymbol{\beta}^{(i)}\|_{\mathbf{F}}^2 \right) + \frac{C}{2}\sum_{i=1}^n \|\boldsymbol{\xi}_{\cdot i}\|^2 \\ \text{s.t.} \quad & {\boldsymbol{\beta}^{(i)}}^\top \mathbf{x}_i = \mathbf{y}_i - \boldsymbol{\xi}_{\cdot i}, \quad i = 1, \ldots, n. \end{aligned} \tag{5}$$

By introducing an absent matrix $\mathbf{S} \in \{0, 1\}^{n \times d}$ which indicates the absence of each sample, Eq. (5) can be equivalently rewritten as,

$$\begin{aligned} \min_{\boldsymbol{\beta}, \boldsymbol{\xi}} \quad & \left(\max_{1 \leq i \leq n} \frac{1}{2}\sum_{p=1}^d S(i,p)\beta_p^2 \right) + \frac{C}{2}\sum_{i=1}^n \|\boldsymbol{\xi}_{\cdot i}\|^2 \\ \text{s.t.} \quad & \sum_{p=1}^d S(i,p)\beta_p x_{ip} = \mathbf{y}_i - \boldsymbol{\xi}_{\cdot i}, \quad i = 1, \ldots, n, \end{aligned} \tag{6}$$

where $\|\boldsymbol{\beta}^{(i)}\|_{\mathbf{F}}^2 = \sum_{p=1}^d S(i,p)\beta_p^2$ and ${\boldsymbol{\beta}^{(i)}}^\top \mathbf{x}_i = \sum_{p=1}^d S(i,p)\beta_p x_{ip}$.

The optimization problem in Eq. (6) is more difficult to solve than the traditional ELM algorithms due to the fact that the normal vectors are varied across samples. In the following subsection, we reformulate it as a convex one and solve it with CVX [18].

3.2 Optimization

In order to optimize the problem in Eq. (6), we first define an auxiliary variable μ as follows,

$$\mu = \max_{1 \leq i \leq n} \frac{1}{2}\sum_{p=1}^d S(i,p)\beta_p^2, \tag{7}$$

which represents the maximum of the norm of all sample-based normal vectors.

Based on this definition, the optimization problem in Eq. (6) can be equivalently rewritten as

$$\begin{aligned} \min_{\boldsymbol{\beta}, \boldsymbol{\xi}, \mu} \quad & \mu + \frac{C}{2}\sum_{i=1}^n \|\boldsymbol{\xi}_{\cdot i}\|^2 \\ \text{s.t.} \quad & \sum_{p=1}^d S(i,p)\beta_p x_{ip} = \mathbf{y}_i - \boldsymbol{\xi}_{\cdot i}, \quad i = 1, \ldots, n \\ & \frac{1}{2}\sum_{p=1}^d S(i,p)\beta_p^2 \leq \mu, \quad i = 1, \ldots, n. \end{aligned} \tag{8}$$

As can be observed, there is no such analytical solution for Eq. (8), which is different from existing ELM algorithms where the optimal solution can be obtained analytically, as seen from Eq. (2).

From Eq. (8), we find that: (1) The objective is a quadratic function, which is convex; (2) The first constraint is linear in variables, which is also convex; and (3) The last constraint is a quadratic cone, which is again convex [18]. As a result, the optimization problem in Eq. (8) is a jointly convex one. Actually, Eq. (8) is a quadratic constraint quadratic programming (QCQP) problem, which can be efficiently solved by many existing convex optimization packages. In our paper, we apply the widely used CVX [18] to optimize Eq. (8). More advanced optimization technique such as Nesterov's methods [19] can be adopted to further improve its computation efficiency.

3.3 Discussion

In this section, we briefly discuss the differences among the proposed A-ELM and the widely used zero-filling ELM (ZF-ELM)and mean-filling (MF-ELM). Specifically, the difference between A-ELM and ZF-ELM is the way in calculating the normal vectors. In A-ELM, each sample has its own normal vector in its relevant space. Differently, ZF-ELM firstly imputes the missing features with zeros and then imposes an unified normal vector on all samples. As can be observed, ZF-ELM takes no consideration of the absence across samples. Comparably, our A-ELM considers the absence of different samples by constructing different ELMs in the relevant space of each sample. As a result, the proposed A-ELM is able to achieve superior performance than ZF-ELM, especially when the absence is relatively intensive, as will be validated by our experimental results.

Similarly, the above discussion is also applicable to analysis the difference between A-ELM and MF-ELM.

4 Experiments

4.1 Experimental settings

In this section, we first compare the performance of the proposed A-ELM algorithm with zero-filling ELM (ZF-ELM), mean-filling ELM (MF-ELM) and the approach proposed in [20], termed I-ASVM, on six UCI[1]. After that, we evaluate the performance of the above algorithms on a real world applications, i.e., the packed executable identification. The comparison is evaluated in terms of both classification accuracy and computational cost.

We end this section by showing how to construct the absent matrix [**S** in Eq. (8)] on the training data. To this end, we randomly generate a matrix **S** with size $n \times m$[2], where n and m are the number of training samples and channels, respectively. After that, a threshold $\varepsilon_0 \in [0, 1]$ is pre-defined and all elements in **S** larger than ε_0 are set as one and otherwise zero. By this way, we have constructed an absent matrix $\mathbf{S} \in \{0, 1\}^{n \times m}$ on the training data. The absent matrix on the testing data is generated in the same way. As can be seen, the parameter ε_0, termed missing ratio in this paper, will affect the performance of the above algorithms. Intuitively, the larger the ε_0 is, the worse performance that the algorithms achieve. In order to show this point in depth, we compare the performance of these algorithms with varied ε_0. In specific, the values of missing ratio are set to be 0.1–0.6 with 0.1 interval for the UCI and the packed datasets. The detailed information on UCI and packed data sets used in our experiments is shown in Tables 1 and 3.

The aggregated classification accuracy is used as a measurement to evaluate the goodness of the above algorithms. It is obtained by averaging the averaged classification accuracy achieved by an algorithm with different ε_0. We repeat this procedure thirty times to eliminate the randomness in generating the absent matrices, and report the averaged aggregated results and standard deviation. Furthermore, to conduct a rigorous comparison, the *paired student's t test* is performed. The p value of the pairwise t test represents the probability that two sets of compared results come from distributions with an equal mean. A p value of 0.05 is considered statistically significant. In our experiments, the regularization parameter C for each algorithm is chosen from an appropriately large $[10^{-2}, 1, \ldots, 10^4]$ by fivefold cross-validation on the training data. All experiments are conducted on a high performance cluster server, where each node has 2.3 GHz CPU and 12 GB memory.

[1] http://archive.ics.uci.edu/ml/.

[2] In our experiments, such a matrix is generated by a Matlab function rand (n, m).

Table 1 The UCI datasets used in our experiments

Dataset	Number of samples	Number of features
Breast-cancer	286	9
Diabetes	N/A	20
Flare-solar	1,389	10
German	1,000	20
Splice	3,190	61
Waveform	5,000	21

4.2 Results on UCI data sets

The classification accuracies of the above algorithms with the variation of the missing ratio on UCI data sets are reported in Fig. 1. As can be observed, the curves corresponding to the proposed A-ELM are always on the top, indicating its superior performance. Taking the result on "splice" data set for example, the classification accuracy of the proposed A-ELM consistently achieve significant improvement over others with the variation of the missing ratio. The aggregated classification accuracy are reported in Table 2. From these results, we observe that:

- The proposed A-ELM gains statistically significant improvements over others on all data sets.
- The computational cost of the proposed A-ELM is much lower than other ones, indicating its computational efficiency.

From the results on the above UCI data sets, we observe that the proposed A-ELM show better performance in terms of both classification accuracy and computational efficiency when compared with existing state-of-the-art algorithms.

4.3 Results on the packed data set

In the following experiment, we apply the proposed algorithm into a real world application: packed executable identification. A malware instance is unwanted software which is used by Internet attackers to perform malicious activities, such as sending spam emails and launching distributed denial of service attacks (DDOS) [21]. Modern malware instances are usually packed with prevalent packers or custom packers in order to make it very hard for

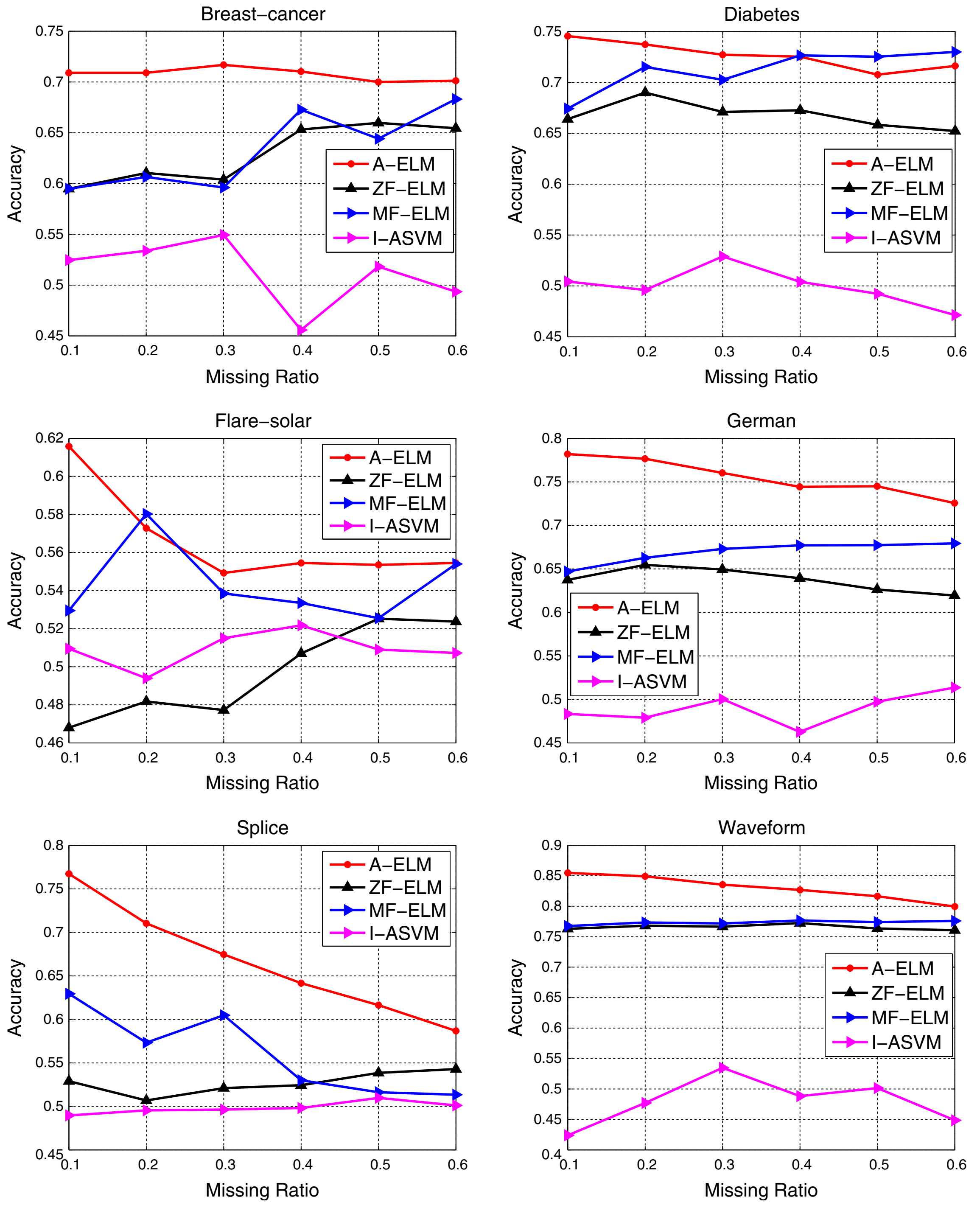

Fig. 1 Classification accuracy comparison of the above algorithms on the UCI data sets

security analysts to perform malware reverse engineering [22]. Packed malware instances pose a significant problem in malware analysis, detection and forensics [23]. If an executable is packed by a runtime packer, the values of some structural characteristics of the executable will be changed. These structural characteristics can be used as features for packed executable identification. Several approaches based on pattern recognition techniques are proposed for packed executable identification [24, 25]. However, all of them don't consider a situation. That is these approaches are also attacked by confusing the features [24]. During research on malware analysis, we obvious a fact that some structural characteristics of a part of packed malware instances hold unreasonable values.

Table 2 Aggregated classification accuracy comparison with statistical test on the UCI data set

	A-ELM Proposed	ZF-ELM	MF-ELM	I-ASVM [20]
Breast-cancer	**70.78** ± **1.03**	62.94 ± 2.10	63.29 ± 2.03	51.26 ± 2.57
	1.00	0.00	0.00	0.00
	0.62	0.14	0.14	0.18
Diabetes	**72.66** ± **0.30**	66.81 ± 1.11	71.24 ± 1.17	49.95 ± 4.49
	1.00	0.00	0.00	0.00
	0.71	0.43	0.43	0.34
Flare-solar	**56.67** ± **0.64**	49.72 ± 0.86	54.35 ± 2.69	50.94 ± 2.04
	1.00	0.00	0.02	0.00
	0.91	1.67	1.09	9.11
German	**75.57** ± **0.55**	63.77 ± 1.15	66.94 ± 1.20	48.94 ± 2.97
	1.00	0.00	0.00	0.00
	0.41	0.77	0.77	0.64
Splice	**66.62** ± **0.42**	52.71 ± 1.23	56.12 ± 3.62	49.85 ± 0.85
	1.00	0.00	0.00	0.00
	0.92	3.90	3.11	1.90
Waveform	**83.03** ± **0.32**	76.56 ± 0.29	77.32 ± 0.20	47.91 ± 4.78
	1.00	0.00	0.00	0.00
	0.50	0.18	0.18	0.11

The three rows of each cell represent mean aggregated accuracy ± standard deviation, p value and the computational cost (in seconds). Boldface means no statistical difference from the best one (p value ≥ 0.05)

Table 3 The packers and compilers used in the packed dataset

Packers used to generate packed malware instances	Number of instances	Compilers used to generate unpacked malware instances	Number of instances
ASPack	68	Borland Delphi	368
Themida	14	MASM32/TASM32	7
UPX	318	Microsoft Visual	970
Yoda's Protector	741	MingWin32	7
Total	1,141	Total	1,352

For example, the value of the *Number of Symbols* field in *File Header* of a malware instance is 0x0007D717, which is unreasonable. Another one is the value of the *Time Stamp* field is 0. Such a situation will cause an incomplete feature vector. That means we get a dataset with missing features. We use our proposed A-ELM to solve this problem. In this experiment, a set of malware instances in the wild is collected for our A-ELM algorithm evaluation. The label process is as following. Firstly, we scan the dataset with PEiD[3] 0.95 including 7,183 signatures of packers for a rough categorization. And then a security expert confirms every instance manually. If an executable is ambiguous in its structural characteristics, it will be deleted from the dataset. Finally, we pick 2,493 malware instances, of which 1,141 instances are packed by 4 prevalence packers and 1,352 instances are unpacked.

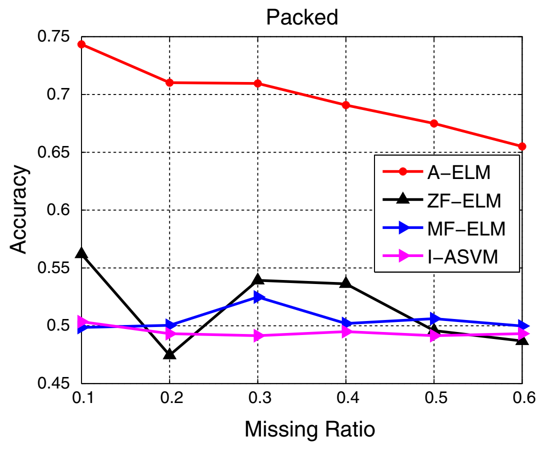

Fig. 2 Classification accuracy comparison of the compared algorithms

We selected a set of 35 structural characteristics from the PE executables, which consist of 4 DOS header characteristics, 10 optional header block characteristics, 12 section characteristics, and 9 section of entry point characteristics. The PEFile[4] is used for features extraction.

The classification accuracy of the above algorithms with the variation of missing ration is plotted in Fig. 2.

[3] http://www.peid.info.

[4] https://code.google.com/p/pefile/.

Table 4 Aggregated classification accuracy comparison with statistical test on the packed data set

	A-ELM proposed	ZF-ELM	MF-ELM	I-ASVM [20]
Packed	**69.74 ± 0.42**	51.57 ± 2.80	50.52 ± 1.09	49.46 ± 0.21
	1.00	0.00	0.00	0.00
	1.05	26.46	26.46	52.84

The three rows of each cell represent mean aggregated accuracy ± standard deviation, p value and the computational cost (in seconds). Boldface means no statistical difference from the best one (p value $\geq$0.05)

As shown, the proposed A-ELM is consistently on the top of the figures, demonstrating its superior performance over others. By avoiding inaccurate imputation, the proposed A-ELM directly classifies each sample in its own space, which attributes to its improvement over others. The aggregated classification accuracy is reported in Table 4. Again, similar observations can also be observed.

Altogether, the above experiment results on UCI data sets and the packed data set validate the effective and efficiency of the proposed algorithm.

5 Conclusion

In this paper, we propose a novel absent ELM algorithm which extends the ELM algorithm to handle the issue of missing data. Specifically, we classify each sample in its own feature space and minimize the maximum of all sample-based normal vectors. The resulting optimization problem is reformulated as a convex one and can be efficiently solved via existing convex packages. The results on six UCI data sets and a real world applications demonstrate its superior performance in both terms of classification accuracy and computational efficiency. Much work is worth exploring. For example, the proposed algorithm in this paper works on original data. We plan to extend it to kernel space via kernelization technique [26] in the future.

Acknowledgments The authors would like to thank Prof. Guang-Bin Huang for the valuable comments. This work was supported by the National Natural Science Foundation of China (Project Nos. 61105050, 61170287, 61303264 and 61271252).

References

1. Huang G-B, Siew CK (2004) Extreme learning machine: RBF network case. In: International conference of the control, automation, robotics and vision, pp 1029–1036
2. Huang G-B, Lei C, Siew CK (2006) Universal approximation using incremental constructive feedforward networks with random hidden nodes. IEEE Trans Neural Netw 17(4):879–892
3. Huang G-B, Zhu Q-Y, Siew C-K (2006) Extreme learning machine: theory and applications. Neurocomputing 70(1C3):489–501
4. Huang G-B, Lei C (2007) Convex incremental extreme learning machine. Neurocomputing 70(16–18):3056–3062
5. Huang G-B, Lei C (2008) Enhanced random search based incremental extreme learning machine. Neurocomputing 71(16–18): 3460–3468
6. Huang G-B, Zhou H, Ding X, Zhang R (2012) Extreme learning machine for regression and multiclass classification. IEEE Trans Syst Man Cybern Part B 42(2):513–529
7. Liu Q, He Q, Shi Z (2008) Extreme support vector machine classifier. In: Proceedings of the advances in knowledge discovery and data mining, pp 222–233
8. Feng G, Huang G-B, Lin Q, Gay R (2009) Error minimized extreme learning machine with growth of hidden nodes and incremental learning. IEEE Trans Neural Netw 20(8):1352–1357
9. Zhang R, Lan Y, Huang G-B, Xu Z-B (2012) Universal approximation of extreme learning machine with adaptive growth of hidden nodes. IEEE Trans Neural Netw Learn Syst 23(2): 365–371
10. Huang G-B, Wang D (2011) Advances in extreme learning machines (ELM2010). Neurocomputing 74(16):2411–2412
11. Huang G-B, Wang D, Lan Y (2011) Extreme learning machines: a survey. Int J Machine Learn Cybern 2:107–122
12. Huang G-B, Wang D (2013) Advances in extreme learning machines (ELM2011). Neurocomputing 102:1–2
13. Zong W, Huang G-B, Chen Y (2013) Weighted extreme learning machine for imbalance learning. Neurocomputing 101:229–242
14. Yu Q, Miche Y, Eirola E, van Heeswijk M, Séverin E, Lendasse A (2013) Regularized extreme learning machine for regression with missing data. Neurocomputing 102:45–51
15. Chen Y, Zhao Z, Wang S, Chen Z (2012) Extreme learning machine-based device displacement free activity recognition model. Soft Comput 16(9):1617–1625
16. Ghahramani Z, Jordan MI (1993) Supervised learning from incomplete data via an em approach. In: Advances in neural information processing systems 6, pp 120–127
17. Suykens JAK, Vandewalle J (1999) Least squares support vector machine classifiers. Neural Process Lett 9:293–300
18. Inc. CVX Research. CVX: Matlab software for disciplined convex programming, version 2.0 beta. http://cvxr.com/cvx, September 2012
19. Nesterov Y (2004) Introductory lectures on convex optimization: a basic course (applied optimization), 1st edn. Springer, Netherlands
20. Chechik G, Heitz G, Elidan G, Abbeel P, Koller D (2008) Max-margin classification of data with absent features. J Mach Learn Res 9:1–21
21. Egele M, Scholte T, Kirda E, Kruegel C (2012) A survey on automated dynamic malware-analysis techniques and tools. ACM Comput Surv (CSUR) 44(2):6
22. Debray S, Patel J (2010) Reverse engineering self-modifying code: unpacker extraction. In: IEEE 17th working conference on reverse engineering (WCRE), 2010, pp 131–140
23. Guo F, Ferrie P, Chiueh T-C (2008) A study of the packer problem and its solutions. In: Recent advances in intrusion detection. Springer, Berlin, pp 98–115
24. Santos I, Xabier U-P, Sanz B, Laorden C, Bringas PG (2011) Collective classification for packed executable identification.

In: Proceedings of the 8th annual collaboration, electronic messaging, anti-abuse and spam conference. ACM, pp 23–30

25. Perdisci R, Lanzi A, Lee W (2008) Classification of packed executables for accurate computer virus detection. Pattern Recogn Lett 29:1941–1946

26. Smola A, Vishwanathan SVN, Hofmann T (2005) Kernel methods for missing variables. In: Cowell RG, Ghahramani Z (eds) AISTATS05, pp 325–332