Noname manuscript No.
(will be inserted by the editor)

One-Class Classification based on Extreme Learning and Geometric Class Information

Alexandros Iosifidis · Vasileios Mygdalis · Anastasios Tefas · Ioannis Pitas

Received: date / Accepted: date

Abstract In this paper, we propose an Extreme Learning Machine-based one-class classification method that exploits geometric class information. We formulate the proposed method to exploit data representations in the feature space determined by the network hidden layer outputs, as well as in ELM spaces of arbitrary dimensions. We show that the exploitation of geometric class information enhances performance. We evaluate the proposed approach in publicly available datasets and compare its performance with the recently proposed One-Class Extreme Learning Machine algorithm, as well as with standard and recently proposed one-class classifiers. Experimental results show that the proposed method consistently outperforms the remaining approaches.

Keywords One-class classification · Novelty detection · Big Data · Extreme Learning Machine

1 Introduction

Extreme Learning Machine (ELM) is an algorithm for Single-hidden Layer Feedforward Neural (SLFN) networks training [8, 9] that leads to fast network training requiring low human supervision. In ELM networks, the network hidden layer parameters are randomly assigned. By using a non-linear activation function for the

Alexandros Iosifidis
Department of Signal Processing, Tampere University of Technology, FI-33101 Tampere, Finland
Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki, 54124, Greece E-mail: alexandros.iosifidis@tut.fi

Vasileios Mygdalis · Anastasios Tefas
Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki, 54124, Greece

Ioannis Pitas
Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki, 54124, Greece
Department of Electrical and Electronic Engineering, University of Bristol, UK
E-mail: {tefas,pitas}@aiia.csd.auth.gr

network hidden layer, the random assignment of the hidden layer weights and bias values results in a random nonlinear mapping of the training data to the so-called ELM space, having dimensionality equal to the number of the network hidden neurons. After obtaining the training data representations in the ELM space, the network output parameters can be analytically calculated using an optimization scheme involving the network target vectors, in order to minimize the training error of the network. Despite the fact that the determination of the hidden layer network outputs is based on randomly assigned input weights, it has been shown that SLFN networks trained by using the ELM algorithm have the properties of global approximators [9].

ELM has been proposed for supervised classification [8]. Since it's first proposal, several optimization schemes have been proposed in the literature for the calculation of the network output parameters, each highlighting different properties of the ELM networks [9, 22, 23, 7, 3, 29, 37, 10, 13, 1, 12]. It has been shown that ELM networks are able to outperform other sophisticated classification schemes, like the Support Vector Machine classifier, [10, 1]. Extensions of the ELM algorithm targeting to semi-supervised [25, 15] and unsupervised [6] SLFN network training have been recently proposed, where it is shown that ELM-based approaches are able to provide state-of-the-art performance, while requiring low human supervision.

While ELM networks have been successfully exploited in (semi-)supervised and unsupervised learning, their application to one-class classification problems has only recently been investigated [21]. One-class classification (sometimes also called novelty detection, outlier detection, or anomaly detection) refers to the classification case where the available training data come from only one class, which needs to be modeled. After learning a model for the class under consideration, unknown (test) samples are compared to that model and classified as class data or outliers (i.e., data not belonging to the class). It is an important and challenging task, which has found application in numerous problems where a large amount of data belonging to the class under consideration can be acquired, while the description of the remaining world, i.e., all other possibilities, is impractical. As an example application where one-class classification is important, let us consider failure detection in industrial systems [35]. In failure detection tasks, the available class to be modeled is the *working properly* case, while every other possibility can be considered as *failure*. In this case, one can easily obtain data describing class *working properly*. However, obtaining data describing class *failure* is expensive, since one should cause failures to hundreds or even thousands of machines. In addition, the caused failures should contain examples of all types of failures that may appear in real situations, which is difficult. For this reason, modeling the class *working properly* by an one-class classifier is the only reasonable choice. Other application scenarios include and medical diagnosis problems [32], sensor networks [39], video surveillance/summarization [28, 30] and mobile robotics [34]. An up-to-date review of one-class classification approaches can be found in [31].

Among the one-class classification methods proposed in the literature, the One-Class Support Vector Machine (OC-SVM) [33] and the Support Vector Data Description (SVDD) [36] algorithms have been found to be effective. SVDD creates the smallest possible boundary by calculating a center and a radius to define a hypershere that can model the class of interest. Samples that appear outside of the hypershere are

classified as novelties [36]. OC-SVM defines a hyperplane that can separate the class of interest from the origin with the maximum possible margin [33]. Kernel Principal Component Analysis (KPCA) based methods have also been proposed for novelty detection [5, 2], where eigen-analysis is applied on the data forming the class to be modeled. A novelty score can be computed using the reconstruction error [5] in the feature space. Novelty score can also be calculated by projecting the training data in the so-called null space, where the training data variance is zero, then calculate distances of the test data in the null space [2]. OC-ELM [21] has been found to achieve performance comparable with several other state-of-art choices [21].

While the OC-ELM algorithm proposed in [21] has been found to be in line with other one-class classification techniques, in terms of performance, in this paper we show that geometric information concerning the class under consideration in the ELM space is important for one-class classification problems. Such information can be exploited for performance enhancement. We propose an One-Class Extreme Learning Machine classifier that is able to exploit such geometric class information. In more detail, the proposed classifier performs a nonlinear mapping of the training data to the ELM space, where the class under consideration is modeled. Geometric class data relationships are described by using graph structures describing the scatter of the class under consideration, or by exploiting pair-wise data relationships in the ELM space. The proposed method is also extended in order to operate in ELM spaces of arbitrary dimensions [10, 17]. We evaluate the proposed approach in publicly available datasets, where we compare its performance with that of standard one-class classifiers, i.e., OC-SVM [33] and SVDD [36], as well as recently proposed state-of-the-art methods, i.e., Kernel Null Space Methods for Novelty Detection (KNFST) [2], Kernel PCA for novelty detection (KPCS) [5] and OC-ELM [21]. Experimental results show that the proposed algorithms are able to consistently outperform the above mentioned methods.

The rest of the paper is structured as follows. Section 2 provides an overview of previous related work in ELM-based classification. The proposed one-class classification method is described in detail in Section 3. An extension of the proposed method that operates in arbitrary-dimensional ELM spaces is described in Section 4. Experiments evaluating the proposed method are provided in Section 5. Finally, conclusions are drawn in Section 6.

2 Overview of Extreme Learning Machine

Let us denote by $\mathbf{x}_i \in \mathbb{R}^D$ a set of N vectors consisting our training set. We would like to employ $\{\mathbf{x}_i\}_{i=1,\ldots,N}$ in order to train a SLFN network using the ELM algorithm [8]. Such a network consists of D input (equal to the dimensionality of $\mathbf{x}_i$), L hidden and C output (equal to the number of classes involved in the classification problem) neurons. The number of hidden layer neurons L is a parameter of the ELM algorithm and is usually set to be much greater than the number of classes C, i.e., $L \gg C$. In ELMs, the network input weights $\mathbf{W}_{in} \in \mathbb{R}^{D\times L}$ and the hidden layer bias values $\mathbf{b} \in \mathbb{R}^L$ are randomly assigned, while the network output weights $\mathbf{W}_{out} \in \mathbb{R}^{L\times C}$ are analytically calculated, as subsequently described.

Given an activation function $\Phi(\cdot)$ for the network hidden layer and using a linear activation function for the network output layer, the response $\mathbf{o}_i = [o_{i1}, \ldots, o_{iC}]^T$ of the network corresponding to $\mathbf{x}_i$ is calculated by:

$$o_{ik} = \sum_{j=1}^{L} w_{kj}\, \Phi(\mathbf{v}_j, b_j, \mathbf{x}_i),\ k = 1, ..., C, \tag{1}$$

where $\mathbf{v}_j$ is the j-th column of $\mathbf{W}_{in}$, $\mathbf{w}_k$ is the k-th column of $\mathbf{W}_{out}$ and w_{kj} is the j-th element of $\mathbf{w}_k$. It has been shown that almost any non-linear piecewise continuous activation functions $\Phi(\cdot)$ can be used for the calculation of the network hidden layer outputs, e.g., the sigmoid, polynomial, Radial Basis Function (RBF), RBF-χ^2, Fourier series, etc [9, 7, 10, 13]. It has been also recently proven that ELM networks using polynomials, Nadaraya-Watson and sigmoid functions attain the theoretical generalization bound of feedforward neural networks. For the remaining activation function choices, the Tikhonov regularization can be applied to guarantee the weak regularity of the hidden layer output matrix, while not sacrificing the network's generalization capability [26].

By storing the network hidden layer outputs $\boldsymbol{\phi}_i \in \mathbb{R}^L$ corresponding to all the training vectors $\mathbf{x}_i,\ i = 1, \ldots, N$ in a matrix $\boldsymbol{\Phi} = [\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_N]$, the network response for the entire training set $\mathbf{O} \in \mathbb{R}^{C \times N}$ can be expressed in a matrix form as:

$$\mathbf{O} = \mathbf{W}_{out}^T \boldsymbol{\Phi}. \tag{2}$$

For multi-class classification problems, the network target vectors $\mathbf{t}_i \in \mathbb{R}^C,\ i = 1, \ldots, N$ are set to $t_{ik} = 1$, when $\mathbf{x}_i$ belongs to class k, and $t_{ik} = -1$, otherwise. The original ELM algorithm [8] assumes zero training error. That is, it is assumed that $\mathbf{o}_i = \mathbf{t}_i, i = 1, \ldots, N$, or by using a matrix notation $\mathbf{O} = \mathbf{T}$, where $\mathbf{T} = [\mathbf{t}_1, \ldots, \mathbf{t}_N]$ is a matrix containing the network target vectors. By using (3), the network output weights $\mathbf{W}_{out}$ can be analytically calculated by:

$$\mathbf{W}_{out} = \left(\boldsymbol{\Phi}\boldsymbol{\Phi}^T\right)^{-1} \boldsymbol{\Phi}\, \mathbf{T}^T = \boldsymbol{\Phi}^{\dagger}\, \mathbf{T}^T. \tag{3}$$

In the case where $L > N$, the calculation of the network output weights $\mathbf{W}_{out}$ through (3) is inaccurate, since the matrix $\boldsymbol{\Phi}\boldsymbol{\Phi}^T$ is singular. A regularized version of the ELM algorithm that allows small training errors and tries to minimize the norm of the network output weights $\mathbf{W}_{out}$ has been proposed in [10]. In this case, the network output weights are calculated by solving the following optimization problem:

$$\textbf{Minimize: } \mathcal{J}_{RELM} = \frac{1}{2}\|\mathbf{W}_{out}\|_F^2 + \frac{c}{2}\sum_{i=1}^{N} \|\boldsymbol{\xi}_i\|_2^2, \tag{4}$$

$$\textbf{Subject to: } \mathbf{W}_{out}^T \boldsymbol{\phi}_i = \mathbf{t}_i - \boldsymbol{\xi}_i, \quad i = 1, ..., N, \tag{5}$$

where $\boldsymbol{\xi}_i \in \mathbb{R}^C$ is the error vector corresponding to $\mathbf{x}_i$ and c is a parameter denoting the importance of the training error in the optimization problem, satisfying $c > 0$.

By substituting the constraints (5) in (4) and determining the saddle point of $\mathcal{J}_{RELM}$ with respect to $\mathbf{W}_{out}$, the network output weights are obtained by:

$$\mathbf{W}_{out} = \left(\mathbf{\Phi}\mathbf{\Phi}^T + \frac{1}{c}\mathbf{I}\right)^{-1} \mathbf{\Phi}\mathbf{T}^T, \tag{6}$$

or

$$\mathbf{W}_{out} = \mathbf{\Phi}\left(\mathbf{\Phi}^T\mathbf{\Phi} + \frac{1}{c}\mathbf{I}\right)^{-1} \mathbf{T}^T = \mathbf{\Phi}\left(\mathbf{K} + \frac{1}{\lambda}\mathbf{I}\right)^{-1} \mathbf{T}^T, \tag{7}$$

where $\mathbf{K} \in \mathbb{R}^{N\times N}$ is the so-called ELM kernel matrix [10, 17] and $\mathbf{I}$ is the identity matrix of appropriate dimensions.

In the latter case, after the calculation of the network output weights $\mathbf{W}_{out}$, the network response for a given vector $\mathbf{x}_l \in \mathbb{R}^D$ is given by:

$$\mathbf{o}_l = \mathbf{W}_{out}^T\boldsymbol{\phi}_l = \mathbf{T}\left(\mathbf{K} + \frac{1}{\lambda}\mathbf{I}\right)^{-1} \mathbf{\Phi}^T\boldsymbol{\phi}_l = \mathbf{T}\left(\mathbf{K} + \frac{1}{\lambda}\mathbf{I}\right)^{-1} \mathbf{k}_l, \tag{8}$$

where $\mathbf{k}_l$ is the ELM kernel vector for $\mathbf{x}_l$.

Recently, it has been shown that the performance of the ELM network can be enhanced by exploiting geometric (class) data relationships in the ELM space [13, 15]. By trying to minimize both the network training error and the within-class dispersion of the training data in the ELM space, the network output weights $\mathbf{W}_{out}$ are calculated by solving the following optimization problem [13]:

$$\textbf{Minimize: } \mathcal{J}_{GELM} = \frac{1}{2}\|\mathbf{S}^{\frac{1}{2}}\mathbf{W}_{out}\|_F^2 + \frac{c}{2}\sum_{i=1}^{N}\|\boldsymbol{\xi}_i\|_2^2, \tag{9}$$

$$\textbf{Subject to: } \mathbf{W}_{out}^T\boldsymbol{\phi}_i = \mathbf{t}_i - \boldsymbol{\xi}_i, \quad i = 1, ..., N, \tag{10}$$

where $\mathbf{S} \in \mathbb{R}^{L\times L}$ is the within-class scatter matrix calculated using $\boldsymbol{\phi}_i$. $\mathbf{S}$ can also express general geometric data relationships expressed within the context of the Graph Embedded framework [38, 15]. By substituting the constraints (10) in (9) and determining the saddle point of $\mathcal{J}_{GELM}$ with respect to $\mathbf{W}_{out}$, the network output weights are obtained by:

$$\mathbf{W}_{out} = \left(\mathbf{\Phi}\mathbf{\Phi}^T + \frac{1}{c}\mathbf{S}\right)^{-1} \mathbf{\Phi}\mathbf{T}^T. \tag{11}$$

After the calculation of the network output weights $\mathbf{W}_{out}$, the network response for a given vector $\mathbf{x}_l \in \mathbb{R}^D$ is given by:

$$\mathbf{o}_l = \mathbf{W}_{out}^T\boldsymbol{\phi}_l. \tag{12}$$

$\mathbf{x}_l$ is finally classified to the class corresponding to the network output neuron providing the maximal network output.

For one-class classification problems, the following optimization problem has been recently proposed for the calculation of the network output weight vector $\mathbf{w} \in \mathbb{R}^L$ [21]:

$$\textbf{Minimize: } \mathcal{J}_{OC-ELM} = \frac{1}{2}\|\mathbf{w}\|_2^2 + \frac{c}{2}\sum_{i=1}^{N}\xi_i^2, \tag{13}$$

$$\textbf{Subject to: } \mathbf{w}^T\boldsymbol{\phi}_i = 1 - \xi_i, \quad i = 1, ..., N, \tag{14}$$

leading to the solution:

$$\mathbf{w} = \left(\boldsymbol{\Phi}\boldsymbol{\Phi}^T + \frac{1}{c}\mathbf{I}\right)^{-1}\boldsymbol{\Phi}\mathbf{1}, \tag{15}$$

where $\mathbf{1} \in \mathbb{R}^N$ is a vector of ones. After the calculation of the network output weight $\mathbf{w}$, the network response for a given vector $\mathbf{x}_l \in \mathbb{R}^D$ is given by:

$$o_l = \mathbf{w}^T\boldsymbol{\phi}_l \tag{16}$$

and $\mathbf{x}_l$ is classified to the class under consideration if $(o_l - 1)^2 \leq \epsilon$, or is characterized as an outlier, if $(o_l - 1)^2 > \epsilon$, where $\epsilon \geq 0$ is a threshold determined by using the network responses for the training data.

3 One-Class Extreme Learning Machine exploiting geometric data relationships

In this Section, we describe in detail the proposed one-class classification method that exploits geometric data relationships in the ELM space in order to incorporate geometric class information in the one-class classification model.

In order to describe pairwise data relationships, we assume that the training data representations in the ELM space are used in order to form the vertex set of a graph $\mathcal{G} = \{\boldsymbol{\Phi}, \mathbf{V}\}$, where $\mathbf{V} \in \mathbb{R}^{N\times N}$ is a matrix expressing pair-wise similarities between the graph vertices $\boldsymbol{\phi}_i$. $\mathbf{S}$ can be defined as follows:

$$\mathbf{S} = \boldsymbol{\Phi}\mathbf{L}\boldsymbol{\Phi}^T, \tag{17}$$

where $\mathbf{L} \in \mathbb{R}^{N\times N}$ is the graph Laplacian matrix defined by $\mathbf{L} = \mathbf{D} - \mathbf{V}$, $\mathbf{D}$ being the diagonal degree matrix of $\mathcal{G}$ having elements $D_{ii} = \sum_{j=1}^{N} V_{ij}$ [38].

The dispersion of the training data in the ELM space from their mean can be expressed as follows:

$$\mathbf{S}_w = \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{\phi}_i - \bar{\boldsymbol{\phi}}\right)\left(\boldsymbol{\phi}_i - \bar{\boldsymbol{\phi}}\right)^T = \frac{1}{N}\boldsymbol{\Phi}\left(\mathbf{I} - \frac{1}{N}\mathbf{1}\mathbf{1}^T\right)\boldsymbol{\Phi}^T = \boldsymbol{\Phi}\mathbf{L}_w\boldsymbol{\Phi}^T, \tag{18}$$

where $\bar{\boldsymbol{\phi}}$ expresses the mean training vector in the ELM space, $\mathbf{1}$ is a vector of ones and $\mathbf{I}$ is an identity matrix of appropriate dimensions. Clearly, $\mathbf{S}_w$ can be expressed within the Graph Embedded framework, using the matrices $V = \frac{1}{N^2}\mathbf{1}\mathbf{1}^T$ and $\mathbf{D} = \frac{1}{N}\mathbf{I}$.

Global pair-wise geometric information in the ELM space can be expressed by employing a fully-connected graph in which the weight matrix $\mathbf{V}$ is defined by:

$$v_{ij} = exp\left(-\frac{\|\boldsymbol{\phi}_i - \boldsymbol{\phi}_j\|_2^2}{2\sigma^2}\right), \tag{19}$$

where σ can be used as a scaling parameter of the Euclidean distances between $\boldsymbol{\phi}_i$ and $\boldsymbol{\phi}_j$.

Finally, kNN graphs describing local class data relationships can be exploited by defining the graph weight matrix $\mathbf{V}$ elements as follows:

$$V_{ij} = \begin{cases} v_{ij}, \text{ if } \boldsymbol{\phi}_j \in \mathcal{N}_i, \\ 0, \quad \text{otherwise,} \end{cases} \tag{20}$$

where $\mathcal{N}_i$ denotes the neighborhood of $\boldsymbol{\phi}_i$. Here it is worth noting that the proposed approach can exploit several types of graphs, each describing different pairwise relationships between the training data.

In order to incorporate information related to the geometry of the class in the calculation of the network output weight vector $\mathbf{w}$, the following optimization problem is solved:

$$\textbf{Minimize: } \mathcal{J}_{GOC-ELM} = \frac{1}{2}\mathbf{w}^T\mathbf{S}\mathbf{w} + \frac{c}{2}\sum_{i=1}^{N}\xi_i^2, \tag{21}$$

$$\textbf{Subject to: } \mathbf{w}^T\boldsymbol{\phi}_i = 1 - \xi_i, \quad i = 1, ..., N, \tag{22}$$

where $\mathbf{S}$ is the matrix encoding data relationships of interest, as described above. By observing (13) and (21), it can be seen that the second term of the optimization problems solved by the OC-ELM and the proposed method expresses the training error, i.e. $\mathbf{w}$ should be selected so as to map most of the training data to a value close to one. The first term in both (13) and (21) is a regularizer. In the original OC-ELM method (13), the adopted regularizer tries to minimize the l_2 norm of $\mathbf{w}$. The regularizer used in (21) can be considered as a data projection minimizing the geometric properties expressed in $\mathbf{S}_w$.

Let us consider the within-class scatter expressed in (18). The regularizer employing $\mathbf{S}_w$ can be expressed as follows:

$$\begin{aligned} \mathbf{w}^T\mathbf{S}_w\mathbf{w} = \mathbf{w}^T\boldsymbol{\Phi}\mathbf{L}_w\boldsymbol{\Phi}\mathbf{w} &= \mathbf{w}^T\left(\frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{\phi}_i - \bar{\boldsymbol{\phi}}\right)\left(\boldsymbol{\phi}_i - \bar{\boldsymbol{\phi}}\right)^T\right)\mathbf{w} \\ &= \frac{1}{N}\sum_{i=1}^{N}\left(\mathbf{w}^T\boldsymbol{\phi}_i - \mathbf{w}^T\bar{\boldsymbol{\phi}}\right)\left(\mathbf{w}^T\boldsymbol{\phi}_i - \mathbf{w}^T\bar{\boldsymbol{\phi}}\right)^T \\ &= \sum_{i=1}^{N}\left(o_i - \bar{o}\right)\left(o_i - \bar{o}\right)^T = \tilde{\mathbf{S}}_w. \end{aligned} \tag{23}$$

That is, by exploiting $\mathbf{S}_w$ in its regularization term, the proposed method tries to find the vector $\mathbf{w}$ which minimizes the training error and at the same time minimizes the

dispersion of the data in the network's output layer. It can be easily shown that the exploitation of the graphs in (19) and (20) lead to regularization terms minimizing pair-wise distances of the data in the network's output layer.

In order to obtain the solution of (21), we use its dual optimization problem which is given by:

$$\mathcal{L}_{GOC-ELM} = \frac{1}{2}\mathbf{w}^T\mathbf{S}\mathbf{w} + \frac{c}{2}\sum_{i=1}^{N}\xi_i^2 - \sum_{i=1}^{N}\alpha_i(\mathbf{w}^T\boldsymbol{\phi}_i - 1 + \xi), \tag{24}$$

where α_i, $i = 1, \ldots, N$ are the Lagrange multipliers corresponding to the constraints in (22). By determining the saddle points of $\mathcal{L}_{GOC-ELM}$ with respect to $\mathbf{w}$, ξ and α_i, we obtain:

$$\frac{\vartheta\mathcal{L}_{GOC-ELM}}{\vartheta\mathbf{w}} = 0 \Rightarrow \mathbf{S}\mathbf{w} = \boldsymbol{\Phi}\boldsymbol{\alpha}, \tag{25}$$

$$\frac{\vartheta\mathcal{L}_{GOC-ELM}}{\vartheta\xi_i} = 0 \Rightarrow \boldsymbol{\xi} = \frac{1}{c}\boldsymbol{\alpha}, \tag{26}$$

$$\frac{\vartheta\mathcal{L}_{GOC-ELM}}{\vartheta\alpha_i} = 0 \Rightarrow \boldsymbol{\Phi}^T\mathbf{w} = \mathbf{1} - \boldsymbol{\xi}. \tag{27}$$

In order to avoid singularity issues, we adopt a regularized version of $\mathbf{S}$, i.e.:

$$\tilde{\mathbf{S}} = \mathbf{S} + \lambda\mathbf{I}, \tag{28}$$

where $\lambda > 0$ is a regularization parameter used in order to exploit the diagonally dominant property of non-singular matrices. Thus, from (25) and (28) we obtain:

$$\mathbf{w} = \tilde{\mathbf{S}}^{-1}\boldsymbol{\Phi}\boldsymbol{\alpha}. \tag{29}$$

By substituting (26) and (27) to (29), the network output weight vector $\mathbf{w}$ is given by:

$$\begin{aligned}\mathbf{w} &= \left(\boldsymbol{\Phi}\boldsymbol{\Phi}^T + \frac{1}{c}\tilde{\mathbf{S}}\right)^{-1}\boldsymbol{\Phi}\mathbf{1} \\ &= \left(\boldsymbol{\Phi}\boldsymbol{\Phi}^T + \frac{1}{c}\mathbf{S} + \frac{\lambda}{c}\mathbf{I}\right)^{-1}\boldsymbol{\Phi}\mathbf{1}.\end{aligned} \tag{30}$$

After the calculation of the network output weight $\mathbf{w}$, the network response for a given vector $\mathbf{x}_l \in \mathbb{R}^D$ is given by:

$$o_l = \mathbf{w}^T\boldsymbol{\phi}_l \tag{31}$$

and $\mathbf{x}_l$ is classified to the class under consideration if $(o_l - 1)^2 \leq \epsilon$, or is characterized as an outlier, if $(o_l - 1)^2 > \epsilon$, where $\epsilon \geq 0$ is a threshold determined by using the network responses for the training data. Clearly, by describing (15) and (30), one can observe that the proposed GOC-ELM is an extension of the OC-ELM classifier that exploits geometric class relationships for the calculation of the network output weight vector.

4 Kernel One-Class Extreme Learning Machine exploiting geometric data relationships

The above-given analysis exploits random network hidden layer parameters and determines a network output weight vector $\mathbf{w}$ that is defined in the ELM space $\mathbb{R}^L$ of finite dimensions. The same applies to the OC-ELM algorithm [21] described in Section 2. In multi-class classification problems, however, ELM algorithms exploiting kernel formulations have been found to outperform ELM networks exploiting random hidden layer parameters [10, 17]. The kernel formulation for the proposed GOC-ELM algorithm, requires a mapping of the data from the input space to an ELM space of arbitrary dimensionality, by employing any non-linear activation function $\Phi(\cdot) \mapsto \mathcal{F}$, i.e., the RBF kernel function. By exploiting the Representer Theorem, the network output weight vector $\mathbf{w}$ in the ELM space can be expressed as a linear combination of the data representations in the ELM space of and a reconstruction vector i.e.:

$$\mathbf{w} = \mathbf{\Phi}\boldsymbol{\beta}, \tag{32}$$

where $\mathbf{\Phi} \in \mathbb{R}^{|\mathcal{F}|\times N}$ is a matrix that contains the data representations in the ELM space and $\boldsymbol{\beta} \in \mathbb{R}^N$ is a vector containing the reconstruction weights of $\mathbf{w}$ with respect to $\mathbf{\Phi}$. By substituting $\mathbf{w}$ using (32) in (24), and by using a regularized version of $\mathbf{S}$ as in (28), the Lagrangian function $\mathcal{L}_{GOC-ELM}$ gets the following form:

$$\begin{aligned}\mathcal{L}_{GOC-ELM} = &\frac{1}{2}\boldsymbol{\beta}^T\left(\mathbf{KLK} + \lambda\mathbf{K}\right)\boldsymbol{\beta} + \frac{c}{2}\sum_{i=1}^{N}\xi_i^2 \\ &- \sum_{i=1}^{N}\alpha_i(\boldsymbol{\beta}^T\mathbf{k}_i - 1 + \xi),\end{aligned} \tag{33}$$

where $\mathbf{K} \in \mathbb{R}^{N\times N}$ is the so-called kernel ELM matrix defined by $\mathbf{K} = \mathbf{\Phi}^T\mathbf{\Phi}$ and $\mathbf{k}_i \in \mathbb{R}^N$ is a vector having elements equal to $\mathbf{k}_{i,j} = \boldsymbol{\phi}_i^T\boldsymbol{\phi}_j,\ j = 1,\ldots,N$.

By determining the saddle points of $\mathcal{L}_{GOC-ELM}$ with respect to $\boldsymbol{\beta}$, ξ and α_i, we obtain:

$$\frac{\vartheta\mathcal{L}_{GOC-ELM}}{\vartheta\boldsymbol{\beta}} = 0 \Rightarrow \boldsymbol{\beta} = \left(\mathbf{LK} + \lambda\mathbf{I}\right)^{-1}\boldsymbol{\alpha}, \tag{34}$$

$$\frac{\vartheta\mathcal{L}_{GOC-ELM}}{\vartheta\xi_i} = 0 \Rightarrow \boldsymbol{\xi} = \frac{1}{c}\boldsymbol{\alpha}, \tag{35}$$

$$\frac{\vartheta\mathcal{L}_{GOC-ELM}}{\vartheta\alpha_i} = 0 \Rightarrow \mathbf{K}\boldsymbol{\beta} = \mathbf{1} - \boldsymbol{\xi}. \tag{36}$$

By substituting (35) and (36) in (34), we obtain:

$$\boldsymbol{\beta} = \left(\mathbf{K} + \frac{1}{c}\mathbf{LK} + \frac{\lambda}{c}\mathbf{I}\right)^{-1}\mathbf{1}. \tag{37}$$

Thus, the network output weight $\mathbf{w}$ is given by combining (32) and (37) by:

$$\mathbf{w} = \mathbf{\Phi}\left(\mathbf{K} + \frac{1}{c}\mathbf{LK} + \frac{\lambda}{c}\mathbf{I}\right)^{-1}\mathbf{1}. \tag{38}$$

Table 1 Training set cardinalities and number of features for each class

Name	Dataset Size	Target Class Cardinality (N)	Data dimension (D)
IMPART [18]	1204	86	500
I3DPost [4]	832	104	100
Arrhythmia [24]	420	237 and 183	278
Wisconsin Breast Cancer [24, 27]	699	241 and 458	9

After the calculation of the network output weight $\mathbf{w}$, the network response for a given vector $\mathbf{x}_l \in \mathbb{R}^D$ is given by:

$$o_l = \mathbf{w}^T \boldsymbol{\phi}_l = \boldsymbol{\beta}^T \boldsymbol{\Phi}^T \boldsymbol{\phi}_l = \boldsymbol{\beta}^T \mathbf{k}_l. \tag{39}$$

$\mathbf{x}_l$ is finally classified to the class under consideration if $(o_l - 1)^2 \leq \epsilon$, or is characterized as an outlier, if $(o_l - 1)^2 > \epsilon$, where $\epsilon \geq 0$ is a threshold determined by using the network responses for the training data.

5 Experiments

In this section, we describe experiments conducted in order to evaluate the performance of the proposed GOC-ELM classifier. Since several types of graphs describing different types of class data relationships can be defined, we report the performance of the proposed GOC-ELM classifier for several types of graphs. We chose four different graph types, which have been shown to provide good performance in multi-class classification problems [14, 16]. Specifically, in our experiments we employ graphs defined for the Laplacian Eigenmap (LE), the Locally Linear Embedding (LLE), the within-class scatter in Linear Discriminant Analysis (LDA) and the within-class scatter in Clustering-based Disriminant Analysis (CDA) methods. In what follows, each variant of the proposed method is expressed by using the acronym (GOC-ELM-X), where X denotes the type of graph that is exploited (i.e., LE, LLE, LDA or CDA).

In order to directly compare the performance of the proposed method with that of other one-class classification choices, we include in our experiments standard one-class classifiers, i.e., OC-SVM [33] and SVDD [36], as well as recently proposed state-of-the-art one-class classifiers, i.e., KPCS [5] and KNFST [2]. We have employed publicly available datasets that are briefly described in subsection 5.1. Finally, experimental results are provided in Subsection 5.2.

5.1 Data sets

In this section, we present the publicly available datasets employed in our experiments. These datasets were selected in order to evaluate the performance of the proposed GOC-ELM in one-class classification problems related to video summarization (e.g., viewing angle selection) and problems of generic interest. Training set cardinalities and number of features for each class are summarized in Table 1.

5.1.1 IMPART Dataset [18]

The IMPART Multi-modal/Multi-view Dataset consists of five recording sections including multi-view and multi-modal captures. The classification scenario used in our experiments refers to distinguishing one (a priori chosen) viewing angle (e.g., frontal with respect to the human body orientation) from all the remaining ones. Fourteen cameras were placed in a circle, in order to capture a scene with 360^0 coverage around each subject. Long videos depicting multiple human actions were automatically temporally segmented to shorter ones depicting single actions by using the method in [19]. This process led to the creation of 1204 action videos. We obtained vectorial action video representations following a Bag-Of-Features-based approach [11, 14], which resulted into 1204 500-dimensional vectors. For each of the 14 target (view) classes, there are 86 available vectors in total. Example frames from the IMPART dataset can be seen in Figure 5.1.1.

Fig. 1 Example frames from the IMPART dataset

5.1.2 i3DPost Dataset [4]

The i3DPost Multi-view Human Action Dataset contains 832 high-resolution ($1080\times$ 1920 pixel) videos depicting eight persons performing different activities. The database camera setup consists of eight cameras placed in the perimeter of a ring at a height of 2 meters above the studio floor. Similar to the IMPART dataset, the classification scenario used in our experiments refers to distinguishing one viewing angle (e.g., frontal with respect to the human body orientation), from all the remaining ones. We obtained vectorial action video representations by following a Bag-Of-Features-based approach [11, 14], which resulted into 832 100-dimensional vectors. For each of the 8 target (view) classes, there are 104 available vectors in total. Example frames from the I3DPost dataset can be seen in Figure 5.1.2.

5.1.3 Arrhythmia Dataset [24]

Arrhythmia dataset is part of the UCI machine learning repository [24]. It contains 2 (healthy and not healthy) classes of cardinality of 237 and 183 samples each, leading to a total dataset cardinality of 420 samples. Each sample is represented by a 278-dimensional feature vector.

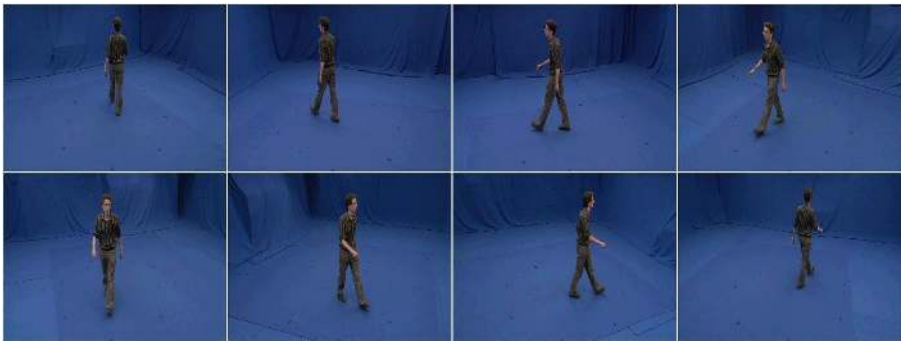

Fig. 2 Example frames from the I3DPost dataset

Table 2 Performance (%) in IMPART Dataset

Algorithm	View 1	View 2	View 3	View 4	View 5	View 6	View 7	View 8	View 9	View 10	View 11	View 12	View 13	View 14	Mean
OC-SVM [33]	55.02 ± 2.84	54.75 ± 1.67	49.80 ± 2.84	50.33 ± 1.87	51.56 ± 3.39	58.10 ± 3.65	53.89 ± 1.42	51.62 ± 1.35	50.22 ±2.69	49.56 ± 1.50	51.52 ±1.63	53.43 ± 0.11	49.63 ± 1.87	55.00 ± 2.49	52.46± 0.18
SVDD [36]	55.80 ± 3.31	54.30 ± 2.15	52.36 ± 4.47	50.35 ± 1.85	52.83 ± 2.92	**57.92±3.39**	54.06 ± 1.54	51.32 ±1.59	49.19 ± 3.11	50.52 ±0.67	51.26 ±1.95	53.03 ±1.12	51.22 ±2.48	55.68 ±3.31	52.85±0.36
KNFST [2]	45.10 ±8.12	53.34 ±1.48	44.70 ± 1.92	52.40 ±0.19	43.55 ±4.95	48.82 ±6.62	53.04 ±3.72	53.83 ±0.56	47.66 ±4.89	53.70 ±0.63	52.81 ±2.15	56.59 ±0.21	50.51 ±2.45	53.80 ±3.36	50.38 ± 0.94
KPCS [5]	45.48 ± 7.91	48.00 ±12.5	53.36 ±5.77	55.33 ±3.51	51.11 ±3.41	50.24 ±0.74	46.58 ±9.56	50.52 ±2.55	49.88 ±7.78	**59.44±2.87**	**57.85± 2.14**	52.90 ±6.67	49.85 ±5.93	41.93 ± 16.7	50.89 ± 4.79
OC-ELM [21]	54.53 ± 1.19	54.89 ±3.37	54.12 ±2.78	53.21 ±2.36	55.71 ±1.21	56.97 ±4.11	55.40 ±3.55	53.04 ±1.07	50.62 ±2.38	52.99 ±1.42	51.57 ±3.70	55.32 ±2.12	51.80 ± 0.96	56.13 ±2.08	54.02±0.78
GOC-ELM-LDA	55.68 ±1.26	54.95 ±3.43	55.05 ±1.98	54.99 ±2.17	56.46 ±1.87	56.97 ±4.11	57.21 ±1.53	54.18 ±1.40	51.53 ±3.04	53.57 ±0.83	52.15 ±3.81	55.74 ±2.28	52.14 ±1.16	56.53 ±2.24	54.80±0.74
GOC-ELM-CDA	**57.53±1.38**	**56.43±2.31**	**56.23±1.88**	**56.48±1.52**	**57.31±1.55**	57.08 ±4.08	**57.93±1.29**	**55.98±1.21**	**53.61±1.88**	55.54 ±0.64	53.59 ±4.00	**56.79±3.07**	53.49 ±1.15	**57.82±1.99**	**56.13±0.72**
GOC-ELM-LE	55.53 ± 1.46	54.95 ±3.43	54.79 ±1.97	54.70 ±2.30	56.50 ±1.92	56.97 ±4.11	56.56 ±2.07	54.02 ±1.35	50.99 ±2.69	53.66 ±0.77	52.29 ±4.16	55.74 ±2.28	52.11 ±1.21	56.59 ±1.75	54.67±0.79
GOC-ELM-LLE	55.02 ±1.48	54.71 ±2.54	54.68 ±.039	55.70 ±4.01	56.48 ±0.93	56.85 ±6.48	54.96 ±4.26	53.93 ±3.84	51.21 ±2.04	56.34 ±3.92	54.22 ±3.91	55.37 ±3.31	52.53±4.51	56.68 ±3.05	54.91 ±2.67

Table 3 Performance (%) in I3DPost Dataset

Algorithm	View 1	View 2	View 3	View 4	View 5	View 6	View 7	View 8	Mean
OC-SVM [33]	64.96 ± 3.54	64.29 ± 4.38	68.92 ± 0.65	69.52 ± 4.64	62.52 ± 2.81	66.20 ± 2.63	65.08 ± 4.79	70.91 ± 1.77	66.55 ± 2.73
SVDD [36]	64.89 ± 3.64	64.79 ± 3.84	68.43 ± 0.88	69.52 ± 4.64	62.41 ± 2.94	66.20 ± 2.29	65.98 ± 4.06	70.91 ± 1.77	66.90 ± 2.62
KNFST [2]	75.74 ± 3.37	67.05 ± 12.60	78.34 ± 0.75	79.45 ± 1.85	66.90 ± 9.67	65.27 ± 11.53	74.52 ± 3.31	78.82 ± 3.85	73.15 ± 5.54
KPCS [5]	75.83 ± 3.75	72.95 ± 1.93	71.91 ± 3.33	70.97 ± 2.24	71.01 ± 5.27	70.95 ± 4.62	68.53 ± 1.06	69.74 ± 1.22	73.75 ± 2.05
OC-ELM [21]	74.15 ± 1.69	75.07 ± 5.34	75.91 ± 1.01	79.14 ± 1.98	69.57 ± 0.58	73.83 ± 1.11	74.34 ± 3.40	76.40 ± 3.81	74.80 ± 2.54
GOC-ELM-LDA	75.25 ± 1.77	77.88 ±3.09	78.73 ± 2.01	80.69 ± 0.21	71.87 ± 2.27	75.58 ± 0.83	75.03 ± 2.80	77.07 ± 2.95	76.51 ±2.52
GOC-ELM-CDA	**78.75 ± 1.53**	**80.53 ± 2.03**	**80.57 ± 1.53**	**82.19 ± 1.09**	**76.45 ± 1.12**	**79.08 ± 0.77**	**79.33 ± 1.19**	**80.47 ± 1.53**	**79.67 ± 1.59**
GOC-ELM-LE	75.86 ± 1.47	77.79 ± 3.15	78.73 ± 2.01	80.79 ± 0.08	71.97 ± 2.28	75.90 ± 1.02	75.03 ± 2.80	77.07 ± 2.95	76.64 ± 2.47
GOC-ELM-LLE	77.65 ± 3.35	79.37 ± 3.46	80.30 ± 2.09	82.01 ± 1.53	71.76 ± 0.83	77.36 ± 0.34	76.30 ± 2.56	80.38 ± 1.51	78.14 ± 2.98

5.1.4 Wisconsin Breast Cancer Dataset [27]

Wisconsin Breast Cancer Dataset is part of the UCI machine learning repository [24]. It was collected between 1989 and 1991 by the University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg [27]. This dataset contains two classes of 241 and 458 samples each, resulting in a total dataset cardinallity of 699 samples. Each sample is represented by a 9-dimensional feature vector.

5.2 Experimental Results

In all our experiments, we evaluate the performance of each algorithm by using the *g-mean* metric [20], which have been found to be useful when there are data imbalances. The *g-mean* metric indicates both precision and recall measurements as follows:

$$G = \sqrt{precision * recall} \tag{40}$$

We report the performance obtained by applying the kernel version of the proposed GOC-ELM algorithms, since it consistently outperformed the GOC-ELM algorithm

Table 4 Performance (%) in Wisconsin Breast Cancer Dataset

Algorithm	Benign	Malignant	Mean
OC-SVM [33]	94.51 ± 1.11	86.06 ± 2.45	90.29
SVDD [36]	96.66 ± 0.59	90.88 ± 2.80	93.77
KNFST [2]	94.7 ± 0.92	34.27 ± 0.71	64.49
KPCS [5]	28.01 ± 2.44	12.01 ± 2.78	92.7
OC-ELM [21]	94.70 ± 0.74	72.72 ± 4.81	83.71
GOC-ELM-LDA	96.63 ± 0.75	72.72 ± 4.81	84.68
GOC-ELM-CDA	96.67 ± 0.63	72.72 ± 4.81	84.7
GOC-ELM-LE	96.65 ± 0.74	72.66 ± 4.81	84.66
GOC-ELM-LLE	$\mathbf{96.82 \pm 0.72}$	$\mathbf{97.71 \pm 0.47}$	**97.27**

Table 5 Performance (%) in Arrhythmia Dataset

Algorithm	Healthy	Not healthy	Mean
OC-SVM [33]	70.64 ± 1.79	35.81 ± 5.18	53.23
SVDD [36]	70.69 ± 1.72	27.65 ± 3.12	49.17
KNFST [2]	68.93 ± 1.69	39.59 ± 6.46	54.26
KPCS [5]	54.53 ± 3.43	$\mathbf{67.30 \pm 3.40}$	60.92
OC-ELM [21]	69.79 ± 1.43	52.01 ± 3.16	60.9
GOC-ELM-LDA	69.98 ± 1.36	53.86 ± 2.24	61.92
GOC-ELM-CDA	$\mathbf{71.62 \pm 0.76}$	54.16 ± 2.24	62.89
GOC-ELM-LE	69.96 ± 1.40	53.77 ± 2.35	**63.65**
GOC-ELM-LLE	71.15 ± 1.36	55.42 ± 3.12	63.29

exploiting the ELM space determined by the network hidden layer outputs. The same applies for the OC-ELM algorithm. That is, in the following results we report the performance of the special case of the proposed kernel GOC-ELM algorithm that uses $\mathbf{L} = \mathbf{0}$, since it has been found to outperform the OC-ELM method proposed in [21] in most cases.

Regarding the values of the parameters used in each of the tested algorithms, they have been determined by following a grid search approach. For OC-SVM and SVDD-based one-class classification, we define the regularization parameter and the fraction of the outliers, respectively, by setting the corresponding parameters equal to dN, where the value of parameter d is given by $d = \{0.05, 0.1, 0.3, 0.5, 0.8\}$. In KPCS-based one-class classification, the total PCA energy preserved is set equal to $p = \{0.85, 0.9, 0.95\}$. In KNFST, the reconstruction error boundary is set to the value dN, where $d = \{0.05, 0.1, 0.3, 0.5, 0.8\}$. In OC-ELM and the proposed GOC-ELM classifiers, the regularization parameters c, λ are set equal to $c = 10^l$ and $\lambda = 10^l$, where $l = \{-6, \ldots, 6\}$. The number k of nearest neighbors used in order to define the kNN graph in GOC-ELM-LLE is set to $k = \{2,' \ldots, 10\}$ and the number of clusters used in GOC-ELM-CDA is set to $s = \{2, \ldots, 20\}$. In all our experiments, the parameter σ used for defining the weights of $\mathbf{V}$ in (19) has been set equal to the mean Euclidean distance between the training samples in the ELM space.

In our first set of experiments, we have applied the one-class classification methods in the IMPART and i3DPost action video datasets. In both datasets, we report the average performance of each algorithm by performing a f-fold cross-validation procedure, where $f = 3$ for IMPART and $f = 4$ for i3DPost. We have employed

a RBF kernel function using the χ^2 distance, where the variance is defined to be proportional to the mean χ^2 distance between the training vectors by employing a weight $\sigma = [0.5, 1.0, 5.0, 10.0]$. Multiple experiments have been conducted (one for each view angle) and the obtained performance values are illustrated in Tables 2 and 3 for the IMPART and the i3DPost datasets, respectively. In addition, we report the mean performance and standard deviations of each algorithm, which corresponds to the performance of each algorithm after training multiple one-class classifiers, each used in order to make decisions for a specific view. It is clear that in both cases, the incorporation of geometric class information in OC-ELM-based classification leads to enhanced performance. Specifically, it can be seen that the proposed GOC-ELM algorithms outperform OC-ELM in most cases. The proposed GOC-ELM algorithm exploiting the (clustering-based) within-class variance of the training data provides the best performance in both databases. It is worth noting here that GOC-ELM-CDA outperforms all the remaining choices to a large extend.

In our second set of experiments, we have applied the one-class classification methods in the Wisconsin Breast Cancer and Arrhythmia datasets. Due to the fact that the cardinality of these datasets is rather small, when compared to the IMPART and i3DPost datasets, we employ 70% of the class data in order to form our training set and the remaining 30% in order to form the evaluation set. We apply 10 experiments by randomly partitioning each dataset in training and test sets and we report the mean of the obtained performances, as well as the standard deviations, for each algorithm. Experimental results are illustrated in Tables 4 and 5. The incorporation of geometrical class information in the OC-ELM optimization process clearly enhances the performance of the OC-ELM classifier. The proposed algorithms outperform the OC-ELM algorithm [21] in every case. Finally, it can be seen that the proposed GOC-ELM methods outperforms standard and recently proposed state-of-the-art approaches.

6 Conclusions

In this paper, we have shown that geometric class information described by exploiting graph structures can be incorporated in Extreme Learning Machine-based neural network training for one-class classification. We have proposed two solutions to this end. The first solution employs data representations in the feature space determined by the outputs of the network hidden layer outputs, while the second one employs data representations in arbitrary-dimensional ELM spaces. We have shown that the incorporation of geometric class information in ELMs for one-class classification leads to enhanced performance. In addition, we compared the performance of the proposed one-class classifier with standard, as well as recently proposed methods in public datasets corresponding to real problems. Experimental results show that the proposed method is able to provide satisfactory performance in all tested cases.

7 Acknowledgment

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement number 316564 (IMPART).

References

1. Bai Z, Huang GB, Wang W, Wang H, Westover MB (2014) Sparse Extreme Learning Machine for classification. IEEE Trans Cyber 44(10):1858–1870
2. Bodesheim P, Freytag A, Rodner E, Kemmler M, Denzler J (2013) Kernel null space methods for novelty detection. In: Computer Vision Pattern Recognition (CVPR) pp 3374–3381
3. Feng G, Huang GB, 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
4. Gkalelis N, Kim H, Hilton A, Nikolaidis N, Pitas I (2009) The i3DPost multi-view and 3D human action/interaction database. In: European Conference on Visual Media Production (CVMP) pp 159–168
5. Hoffmann H (2007) Kernel PCA for novelty detection. Pattern Recogn 40(3):863–874
6. Huang G, Song S, Gupta JN, Wu C (2014) Semi-supervised and unsupervised Extreme Learning Machines. IEEE Trans Cybern 44(12):2405–2417
7. Huang GB, Chen L (2007) Convex incremental Extreme Learning Machine. Neurocomputing 70(16):3056–3062
8. Huang GB, Zhu QY, Siew CK (2004) Extreme Learning Machine: a new learning scheme of feedforward neural networks. In: IEEE International Joint Conference on Neural Networks (IJCNN) pp 985–990
9. Huang GB, Chen L, Siew CK (2006) Universal approximation using incremental constructive feedforward networks with random hidden nodes. IEEE Trans Neural Netw 17(4):879–892
10. Huang GB, Zhou H, Ding X, Zhang R (2012) Extreme Learning Machine for regression and multiclass classification. IEEE Trans Syst Man, Cybern B, Cybern 42(2):513–529
11. Huang Y, Wu Z, Wang L, Tan T (2013) Feature coding in image classification: A comprehensive study. IEEE Trans Pattern Anal Mach Intell 36(3):493–506
12. Iosifidis A (2015) Extreme Learning Machine based supervised subspace learning. Pattern Recogn Lett 167:158–164
13. Iosifidis A, Tefas A, Pitas I (2013) Minimum Class Variance Extreme Learning Machine for human action recognition. IEEE Trans Circuits Syst Video Technol 23(11):1968–1979
14. Iosifidis A, Tefas A, Pitas I (2014) Discriminant Bag of Words based representation for human action recognition. Pattern Recogn Lett 49:185–192
15. Iosifidis A, Tefas A, Pitas I (2014) Regularized Extreme Learning Machine for multi-view semi-supervised action recognition. Neurocomputing 145:250–262

16. Iosifidis A, Tefas A, Pitas I (2015) Graph Embedded Extreme Learning Machine. IEEE Trans Cyber 46(1):311–324
17. Iosifidis A, Tefas A, Pitas I (2015) On the kernel Extreme Learning Machine classifier. Pattern Recogn Lett 54:11–17
18. Kim H, Hilton A (2015) Influence of colour and feature geometry on multimodal 3D point clouds data registration. In: International Conference on 3D Vision (3DV) pp 202–209
19. Kourous N, Iosifidis A, Tefas A, Nikolaidis N, Pitas I (2014) Video characterization based on activity clustering. In: IEEE International Conference on Electrical and Computer Engineering (ICECE) pp 266–269
20. Kubat M, Holte RC, Matwin S (1998) Machine learning for the detection of oil spills in satellite radar images. Machine learn 30(2-3):195–215
21. Leng Q, Qi H, Miao J, Zhu W, Su G (2014) One-class classification with Extreme Learning Machine. Mathematical Problems in Engineering pp 1–12
22. Li MB, Huang GB, Saratchandran P, Sundararajan N (2005) Fully complex Extreme Learning Machine. Neurocomputing 68(13):306–314
23. Liang NY, Huang GB, Saratchandran P, Sundararajan N (2006) A fast and accurate on-line sequantial learning algorithm for feedforward networks. IEEE Trans Neural Netw 17(6):1411–1423
24. Lichman M (2013) UCI machine learning repository. URL http://archive.ics.uci.edu/ml
25. Liu J, Chen Y, Liu M, Zhao Z (2011) SELM: Semi-supervised ELM with application in sparse calibrated location estimation. Neurocomputing 74(16):2566–2572
26. Liu X, Lin S, Fang J, Xu Z (2014) Is Extreme Learning Machine feasible? A theoretical assessment (part i). IEEE Trans Neural Netw Learn Syst 26(1):7–20
27. Mangasarian OL, Setiono R, Wolberg W (1990) Pattern recognition via linear programming: Theory and application to medical diagnosis. Large-scale numerical optimization pp 22–31
28. Markou M, Singh S (2006) A neural network-based novelty detector for image sequence analysis. IEEE Trans Pattern Anal Mach Intell 28(10):1664–1677
29. Miche Y, Sorjamaa A, Bas P, Simula O, Jutten C, Lendasse A (2010) OP-ELM: Optimally Pruned Extreme Learning Machine. IEEE Trans Neural Netw 21(1):158–162
30. Mygdalis V, Iosifidis A, Tefas A, Pitas I (2014) Video summarization based on subclass Support Vector Data Description. In: IEEE Symposium Series on Computational Intelligence (SSCI), Computational Intelligence for Engineering Solutions (CIES) pp 183–187
31. Pimentel M, Clifton D, Clifton L, Tarassenko L (2014) A review of novelty detection. Signal Process 99:215–249
32. Quinn J, Williams C (2007) Known unknowns: novelty detection in condition monitoring. Lect notes comput sc 4477:1–6
33. Scholkopf B, Platt JC, Shawe-Taylor J, Smola AJ, Williamson RC (2001) Estimating the support of a high-dimensional distribution. Neural comput 13(7):1443–1471
34. Sofman B, Neuman B, Stentz A, Bagnell J (2011) Anytime online novelty and change detection for mobile robots. Journal of Field Robotics 28(4):589–618

35. Tarassenko L, Clifton D, Bannister P, King S, King D (2009) Novelty detection. John Wiley & Sons, Ltd (Chapter 35) pp 1–22
36. Tax DM, Duin RP (2004) Support Vector Data Description. Machine learn 54(1):45–66
37. Wang Y, Cao F, Yuan Y (2011) A study on effectiveness of Extreme Learning Machine. Neurocomputing 74(16):2483–2490
38. Yan S, Xu D, Zhang B, Zhang H (2007) Graph Embedding and extensions: A general framework for dimensionality reduction. IEEE Trans Pattern Anal Mach Intell 29(1):40–51
39. Zhang Y, Meratnia N, Havinga P (2010) Outlier detection techniques for wireless sensor networks: a survey. IEEE Commun Surv Tut 12(2):159–170