Chemometrics and Intelligent Laboratory Systems 214 (2021) 104329

Contents lists available at ScienceDirect

Chemometrics and Intelligent Laboratory Systems

journal homepage: www.elsevier.com/locate/chemometrics

A novel approach for water quality classification based on the integration of deep learning and feature extraction techniques

Smail Dilmi [a,b,*], Mohamed Ladjal [a,c]

[a] LASS, Laboratory of Analysis of Signals and Systems, Algeria
[b] Department of Electrical Engineering, Faculty of Technology, University of M'sila, 28000, M'sila, Algeria
[c] Department of Electronics, Faculty of Technology, University of M'sila, 28000, M'sila, Algeria

ARTICLE INFO

Keywords:
Deep learning
Long short term memory recurrent neural networks
Support vector machines
Dimensionality reduction
Time series prediction
Water quality classification

ABSTRACT

Water quality monitoring plays a vital role in the protection of water resources, environmental management, and decision-making. Artificial intelligence (AI) based on machine learning techniques has been widely used to evaluate and classify water quality for the last two decades. However, traditional machine learning techniques face many limitations, the most important of which is the inability to apply these techniques with big data generated by smart water quality monitoring stations to improve the prediction. Real-time water quality monitoring with high accuracy and efficiency for intelligent water quality monitoring stations requires new and sophisticated techniques based on machine and deep learning techniques. For this purpose, we propose a novel approach based on the integration of deep learning and feature extraction techniques to improve water quality classification. In this paper, was chosen the Tilesdit dam in Bouira (Algeria) as a case study. Moreover, we implemented the advanced deep learning method - Long Short Term Memory Recurrent Neural Networks (LSTM RNNs) to construct an intelligent model for drinking water quality classification. Furthermore, principal component analysis (PCA), linear discriminant analysis (LDA) and independent component analysis (ICA) techniques were used for features extraction and data reduction from original features. Additionally, we used three methods of cross-validation and two methods of the out-of-sample test to estimate the performance of LSTM RNNs model. From the results we found that the integration of LSTM RNNs with LDA, and LSTM RNNs with ICA yields an accuracy of 99.72%, using Random-Holdout technique.

1. Introduction

Water is one of the key elements required for the survival of living organisms since in both plant and animal species, water constitutes a large proportion of the cell's living matter content. In fact the amount of water on the Earth's surface is constant. It must be emphasized that the water problem is not a problem of quantity but of flow. Water covers more than 70% of the Earth's surface, yet only 1% of it is useable freshwater [1]. Despite this small amount of freshwater, however, it suffers from many pollution risks such as agricultural runoff, domestic and industrial pollution, etc. [2]. The real catastrophe is that more than 800 million people have insufficient access to safe drinking water [3], and nearly 2 million, mostly babies, die each year due to a lack of access to safe drinking water [4]. Today, the degradation of freshwater quality is one of the greatest environmental impendences [5,6], in addition to its negative effects on human health and water sustainability, because water-borne diseases represent the top 10 causes of death worldwide [2], among these diseases, the most deadly are diarrhea, malaria, trypanosomiasis, intestinal worms infections, dengue and bilharzia. Water quality monitoring plays an important role in environmental management and protection of water resources. Many developing countries, still rely on traditional methods to monitor water quality in most drinking water production stations, although they are equipped with modern and advanced equipment. The traditional methods are based on the knowledge of different parameters of the raw water by chemical analyzes carried out in the laboratory, to decide later on its state and to look for the appropriate methods to make it drinkable. The disadvantages of these methods are that they require the intervention of a human expert and a long enough time. In addition to the disadvantage of having a relatively long delay time, they do not allow to follow finely the evolution of the quality of raw water.

In recent years, several techniques based on machine learning have

* Corresponding author. LASS, Laboratory of Analysis of Signals and Systems, Algeria.
E-mail addresses: smail.dilmi@univ-msila.dz (S. Dilmi), mohamed.ladjal@univ-msila.dz (M. Ladjal).

https://doi.org/10.1016/j.chemolab.2021.104329
Received 6 December 2020; Received in revised form 29 March 2021; Accepted 22 April 2021
Available online 1 May 2021

been applied to assess and classify water quality [7,8], such as Support Vector Machines (SVMs), Artificial Neural Network (ANN), k-Nearest Neighbors (kNN) and Decision Tree. Among these techniques, SVM over the past few years has gained great popularity in water quality monitoring field [7,9], and has been widely used in water quality classification [9–14], this is because of SVM has a good generalization ability, high classification accuracy and speed of execution, which makes it a good water quality classification tool. However, in sensors-based drinking water quality monitoring stations, especially those based on remote sensing (RS) and Internet of Things (IoT), more and more big data are produced at high speeds and irregular, which has made water quality data complicated [15]. This has led to the emergence of many shortcomings with respect to traditional machine learning techniques when dealing with this big data. The most important of them is that these techniques cannot be applied with big data to improve the prediction. Therefore must use new and sophisticated algorithms based on machine learning techniques to process data in real-time with high accuracy and efficiency [16]. The advanced deep learning techniques are the most machine learning techniques that the big data applications benefited from them in many fields such as Energy, Intelligent Transportation Systems (ITS), Agriculture, etc. [17]. In recent years, a novel technique from the Artificial Intelligence (AI) domain called Long Short Term Memory Recurrent Neural Networks (LSTM RNNs) for time series prediction has gained very popularity in the deep learning field [17,18], and it has proven its highly efficient in dealing with big data [18]. LSTM RNNs is a very good tool for time series prediction due to its ability to model non-linear relationships and its ability to process multi-dimensional data in a non-linear manner. LSTM RNNs has been effectively implemented to water quality forecasting (regression) and outperform both Support Vector Regression (SVR) and Autoregressive Integrated Moving Average (ARIMA) [15]. However, its effectiveness has not been tested and compared with traditional machine learning techniques in water quality classification till now.

In this paper, we propose, in a detailed and in-depth study, how to use the advanced deep learning method - LSTM RNNs to construct an intelligent model for drinking water quality classification and compare it with the robust SVMs model. These models are built based on the data of the physicochemical parameters collected in 4 seasons during the period of three years (2016–2018) from the drinking water quality monitoring station of the Tilesdit dam in Bouira (Algeria). In order to have a good performance, the preparation of data inputs needs a special treatment to guarantee a well decision of the classifier. Dimensionality reduction has a major impact on machine learner's performance. The use of useful features (variables) can result in high performance, even if the machine learner is simple, while the use of unhelpful features (variables) with advance complex machine learner might lead to decreased performance [16]. Therefore, the number of features must be reduced by extracting or selecting only relevant and useful features. Dimensionality reduction can be divided into feature extraction and feature selection. In this study, we used principal component analysis (PCA), linear discriminant analysis (LDA) and independent component analysis (ICA) techniques for feature extraction, and used the correlation coefficient between variables method for feature selection. Dimensionality reduction in this study is used for two purposes: the first to increase the accuracy and reduce the computational time of classification; and the second to cost-savings. Our problem is how to estimate the performance of LSTM RNNs model. Because LSTM RNNs models are specifically used for time series prediction. In this study, we dealing with one of the most complex types of time series which is water quality monitoring data. In fact, there is no specific method to estimate the performance of these models, because they depend mainly on the type of data used to construct the model. In addition, there is significant controversy in the literature over the use of cross-validation (CV) methods and the out-of-sample (OOS) test methods to estimating the performance of time series prediction models. For this, we used three methods of CV namely standard *k*-fold CV, Multiple Train-Test splits and Blocked cross validation (CV-Bl); and two methods of the OOS test namely simple Holdout (OOS.H) and Random-Holdout.

In this paper, the integration of feature extraction techniques with deep learning method - LSTM RNNs or SVM is proposed to perform the water quality classification. The used techniques are employed to show the performance of water quality classification. Various scenarios are examined using real dataset of water descriptors, and the results are compared to get the best performance of classification process. Nevertheless, there were no references which studied our approach in water quality monitoring.

2. Study area and data description

The Tilesdit dam is located geographically in the town of Bechloul, at 20 km southeast of Bouira Department. This dam is located between the following cartographic and Lambert coordinates: Latitude: 35° 13′ 22″ North; Longitude: 4° 14′ 23″ East. It's characterized by a semi-arid climate and an average downpour of about 440–660 mm/year. The volume of reservoir of this dam at 167 million m^3, it is designed to control the tension that persists in the distribution of water in 12 communes. According to the hydraulics department of the wilaya of Bouira, the dam of Tilesdit is characterized by the following technical data: Length: 425 m, Width in crest: 10 m and Height: 65 m.

In this paper, we seek to effect our approach for surface water quality monitoring using several physicochemical parameters provided by some measurement sensors installed in the station. These parameters were collected from Tilesdit water production station during three years (2016–2018). Our knowledge of the treatment process is limited to the recorded data from this station. More quality parameters of the surface water are every day measured, in addition to laboratory tests which are carried out every week at all treatment levels. Directly after sampling, *Temperature* (T°), *pH*, *Electrical Conductivity* (EC) and *Turbidity* (TU) are measured in the field. These parameters are measured continuously at 3 times/day and at any level of the treatment process. Thereafter, the samples are analyzed in the laboratory every week for their chemical constituents such as *Magnesium* (Mg), *Bicarbonate* (B), *Permanente Hardness* (H) and *Full Title Alkaline* (FTA). After taking the measurements, they are compared with the drinking standards specified by the National Water Resources Agency (NWRA) by an expert to determine the water quality used. In the Tilesdit dam water production station, water quality is classified into three categories according to the drinking standards specified by the NWRA (Class I: upper, Class II: middle, Class III: lower). In this study, the above measured data is used to analyze the relationship among these parameters and to verify the water quality monitoring model. Descriptive statistics for these selected parameters (physicochemical parameters) are given in Table 1.

3. Proposed multi-sensor monitoring system of water quality

In this study, the control, measurement and monitoring of water quality may be considered as a pattern recognition problem (for LSTM RNNs technique, this is called pattern recognition in time series). Generally, it consists of data acquisition, signal processing, feature extraction and selection including feature reduction, decision making of water quality and process control. Our approach proposed for water

Table 1
Descriptive statistics of the analyzed physicochemical parameters.

Variables	Min	Max	Mean	Standard deviation
pH	7,15	8,30	7567	0,25
EC	414,00	624,00	585,393	36,278
T°	9,70	24,20	16,13	3483
TU	1320	23,81	3835	2392
Mg	7290	47,628	22,268	4931
B	158,620	289,14	222,497	23,213
H	0,00	168,00	32,287	23,029
FTA	130,00	237,00	181,845	18,703

quality monitoring used in this study which is based first by preparation of database before inputting into classifiers. The control process is seen as a pattern recognition problem, where classes correspond to different states of water, and patterns represent the ensemble of the observations or measurements of the parameters related to their characteristics. The role of pattern classification techniques is to separate the data into distinct classes. The classification process of water quality is carried out by using LSTM RNNs and multi-class SVMs for a possible integration as a decision-making tool in a multi-sensor water quality monitoring system. The architecture of proposed system is presented in Fig. 1.

At the system level, it can be assumed that the different physic-chemical parameters used are transformed into electrical signals starting from the sensors, and transmitted to a processing and control unit that provides acquisition, processing and analysis.

4. Proposed methods

In this paper, we present a performance assessment of the two artificial intelligence methods namely, LSTM RNNs and multi-class SVM. Such approaches demonstrate maximal training efficiency and generalization in many application fields. The purpose of this work is to classify water quality from independent variables, which in our case are physicochemical parameters, into three distinct groups (Class I, Class II and Class III). Our proposed approach is based first on the preparation of the dataset using feature extraction techniques (PCA, LDA and ICA) and feature selection prior to input into classifiers. This step is performed to remove the irrelevant features which are redundant. Then the data is entered into the classifiers to perform the classification process.

4.1. Dimensionality reduction methods

In statistics and machine learning, dimensionality reduction is the operation of reducing the number of variables through obtaining a set of major variables that represent the original variables [19,20]. It can be split into feature extraction and feature selection [21]. In machine learning, dimensionality reduction technique is used to improve classifier performance (predictive accuracy and/or reduce the computational time of classification algorithms) [20,22]. A summary description of the feature extraction and selection techniques used in this work is presented.

4.1.1. Feature extraction

Feature extraction is transformation the data in the high-dimensional space to a space of fewer dimensions that includes most of the valuable information [20]. There are many well-known methods of feature extraction, the most popular of which are PCA, LDA [19,20,23,24] and

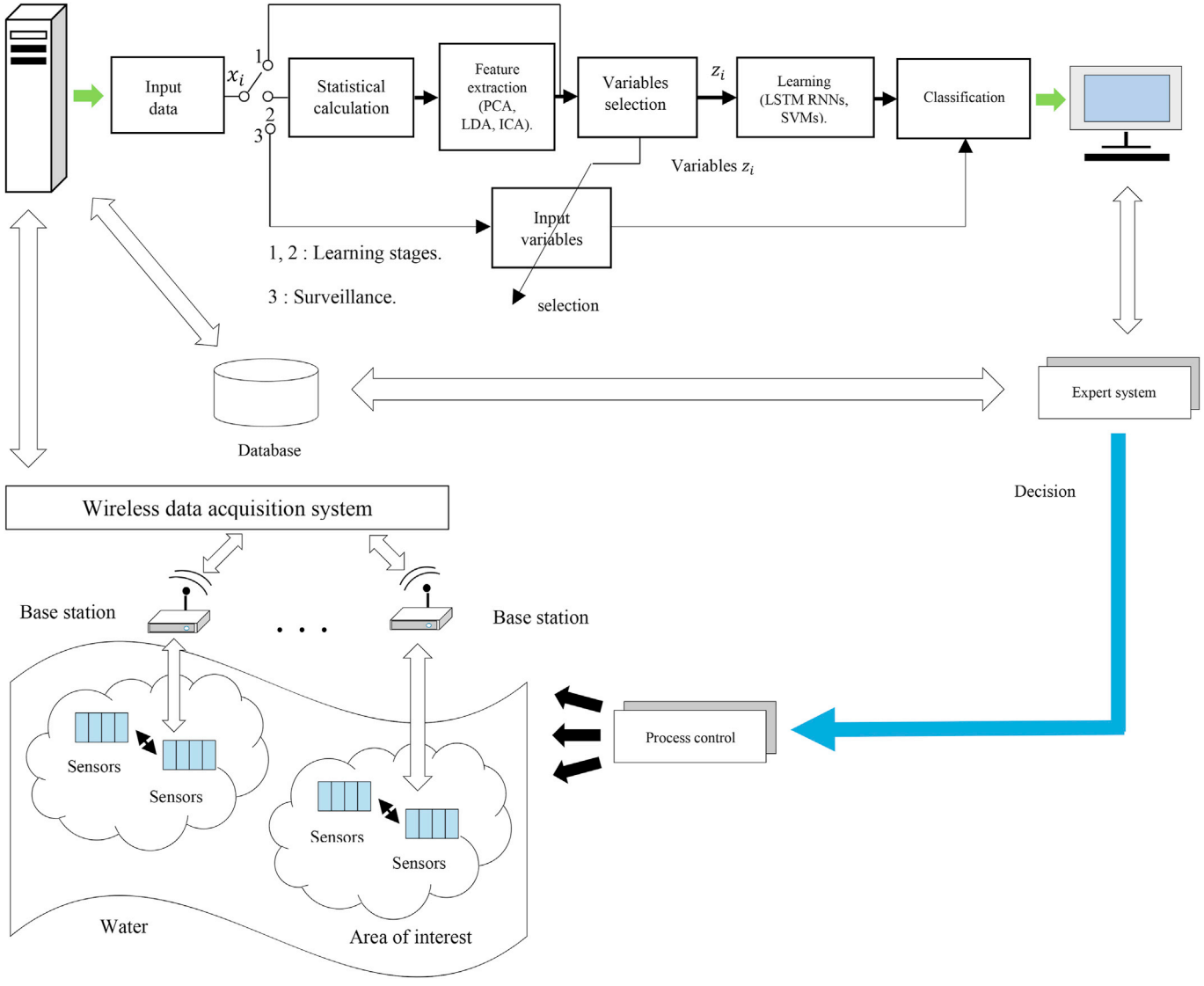

Fig. 1. Multi-sensor monitoring system of water quality.

ICA [19,20,23]. Feature extraction is often based on the calculation of cumulative variance [25,26].

4.1.1.1. Principal component analysis. Principal Component Analysis (PCA) is a mathematical process that belongs to the Data Analysis division [26,27]. This method is used to convert the set of correlated original variables into a smaller number of uncorrelated variables [28]. Variables resulting from the conversion process are called principal components (PCs). The purpose of this process is to facilitate the interpretation of complex data [29] by enabling the researcher and statistician to achieve the optimal compatibility between reducing the number of variables and the loss of the original information (variance) resulting from the reduction of the original dimensions. The principle of PCA is as follows [28,30]:

Given a data matrix $(X = [x_1, x_2, ..., x_n]^T)$, where n represents the total number of samples and x_i represents the ith sample. Estimate the mean value of x_i, denoted as μ, and thus, the mean value vector of X can be written as:

$$\mu = E[X] = \frac{1}{n}\sum_{i=1}^{n} x_i = [\mu_1, \mu_2, ..., \mu_n]^T \tag{1}$$

Subtract μ from X to centralize the matrix X, then the centralized matrix x_t is achieved:

$$x_t = \sum_{i=1}^{n} (x_i - \mu) = [x_1 - \mu_1, x_2 - \mu_2, ..., x_n - \mu_n]^T \tag{2}$$

The standardization is generally used in this step instead of the centering only.

Given a set of centered input vectors $x_t (t = 1, ..., n \text{ and } \sum_{t=1}^{n} x_t = 0)$, each of which has m dimension $x_t = (x_t(1), x_t(2), ..., x_t(m))^T$ (usually $m < n$). PCA linearly transforms each vector x_t into a new one s_t by:

$$s_t = U^T x_t \tag{3}$$

where U is the $m \times m$ orthogonal matrix whose ith column u_i is the ith eigenvector of the sample covariance matrix:

$$C = \frac{1}{n}\sum_{t=1}^{n} x_t x_t^T \tag{4}$$

In other words, PCA firstly solves the eigenvalue problem.

$$\lambda_i u_i = C u_i, \ i = 1, ..., m \tag{5}$$

where λ_i is one of the eigenvalues of C. u_i is the corresponding eigenvector. Based on the estimated u_i, the components of s_t are then calculated as the orthogonal transformations of x_t:

$$s_t = u_i^T x_t, \ i = 1, ..., m \tag{6}$$

The number of PCs in s_t can be reduced by using the first eigenvectors sorted in downward order of the eigenvalues.

4.1.1.2. Linear discriminant analysis. Linear Discriminant Analysis (LDA) is a generalization of Fisher Discriminant Analysis (FDA), a method utilized in pattern recognition and machine learning to find a linear combination of features that classify or separate two or more of groups. The resulting composition can be used as a linear classifier or to reduce the dimensions before the classification process. Thus, similar to PCA, it can be classified as one of the traditional approaches for dimensionality reduction [26]. There are two types of LDA technique to deal with classes: class-independent and class-dependent [31,32]. The type which deals with class-dependent was applied in this study. The theory of this technique is as follows [32]:

Given a set of N samples $[x_i]_{i=1}^{N}$, each of which is represented as a M-length row, and $X(N \times M)$ is given as follows:

$$X = \begin{bmatrix} x(1,1) & x(1,2) & \cdots & x(1,M) \\ x(2,1) & x(2,2) & \cdots & x(2,M) \\ \vdots & \vdots & \vdots & \vdots \\ x(N,1) & x(N,2) & \cdots & x(N,M) \end{bmatrix} \tag{7}$$

Compute the mean of each class $\mu_i (1 \times M)$:

$$\mu_i = \frac{1}{n_i}\sum_{x_i \in \omega_i} x_i \tag{8}$$

where n_i represents the number of samples of the ith class and ω_i represents ith class.

Compute the total mean of all data $(1 \times M)$:

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i = \sum_{i=1}^{c} \frac{n_i}{N}\mu_i \tag{9}$$

where c represents the total number of classes, and N represents the total number of samples is calculated as follows:

$$N = \sum_{i=1}^{c} n_i \tag{10}$$

Calculate between-class matrix $S_B (M \times M)$:

$$S_B = \sum_{i=1}^{c} n_i (\mu_i - \mu)(\mu_i - \mu)^T \tag{11}$$

➢ For each class, separate lower-dimensional space is calculated to project its data on it through the following operations:

Compute within-class matrix of each class $S_{W_i} (M \times M)$:

$$S_{W_i} = \sum_{x_i \in \omega_i} (x_i - \mu_i)(x_i - \mu_i)^T \tag{12}$$

Construct a transformation matrix for each class (W_i) as follows:

From (eq. (11) and (12)), the matrix W_i that maximizing Fisher's formula defined in (eq. (13)) is calculated as follows, $W_i = S_{W_i}^{-1} S_B$.

$$\arg \max_W \frac{W^T S_B W}{W^T S_W W} \tag{13}$$

This formula can be reformulated as in (eq. (14)):

$$S_W W = \lambda S_B W \tag{14}$$

The eigenvalues (λ^i) and eigenvectors (V^i) of each transformation matrix (W_i) are then calculated, where λ^i and V^i represent the calculated eigenvalues and eigenvectors of the ith class, respectively.

Sorting the eigenvectors in descending order according to their corresponding eigenvalues. The first k eigenvectors are then used to construct a lower dimensional space for each class V_k^i.

Project the samples of each class (ω_i) onto their lower dimensional space (V_k^i), as follows:

$$Y_i = x_i V_k^i, \ x_i \varepsilon \omega_i \tag{15}$$

where Y_i represents the projected samples of the class ω_i.

4.1.1.3. Independent component analysis. Independent Component Analysis (ICA) is a method of data analysis that identifies statistics, neural networks and the processing of signals. It is notoriously and historically known as a blind source separation method [30], but has subsequently been applied to various problems. One of these issues is the dimensionality reduction problem, to build a good data classifiers. Thanks to the wonderful results it gives, ICA has been used in many fields including

medical signals, biological assays [33], and chemistry [34]. Also, the ICA technique in data reduction is better than PCA [35], since PCA gives uncorrelated components and ICA gives independent components (ICs), it should be noted that independence is a much stronger property than uncorrelatedness [36]. Independence here means that the information carried by one component cannot be deduced from other components [28]. The general ICA model is given by:

$$x = As \tag{16}$$

where $x = [x_1, x_2, \ldots, x_n]^T$ is known to be the matrix of observed signals, $s = [s_1, s_2, \ldots, s_n]^T$ is the matrix of unknown source signals (ICs), and A is an unknown constant matrix, called the mixing matrix. If matrix A columns are denoted by a_j the model can be therefore presented as:

$$x = \sum_{i=1}^{n} a_i s_i \tag{17}$$

The ICA goal is to estimate s by:

$$s = Wx \tag{18}$$

so that the components in the s matrix are as statistically independent as possible. W is called the matrix $m \times m$ unmixing.

The ICs of a data set are find by minimizing the mutual information, maximizing the non-gaussianity or using maximum likelihood estimation method [37,38]. Among these three methods, maximizing the non-gaussianity is the best and the most widely used method [39], and therefore it was chosen for use in the current study. There are many ICA algorithms [38]. One of the best methods is the fixed-point-FastICA algorithm [30], since FastICA is much faster than Gradient-based algorithms that have linear convergence [37]. This algorithm will now be described briefly.

The first step is to preprocessing the measured data vector x because it is useful before attempting to estimate W [40]. This phase has two main steps:

First, the obtained signals should be centered by subtracting their mean value μ:

$$\widehat{x} = x - \mu \tag{19}$$

where $\widehat{x}$ is the mixture signals after the centering step.

Second, they are whitened, which means they are linearly transformed so that the components are uncorrelated and have unit variance [37,40]. Whitening can be performed via eigenvalue decomposition of the covariance matrix as following, $V\lambda V^T = E\{\widehat{x}\widehat{x}^T\}$. Where $\widehat{x}$ is the centered data, V is the matrix of orthogonal eigenvectors and λ is a diagonal matrix with the corresponding eigenvalues. This step is called decorrelation. The whitening is done by multiplication with the transformation matrix:

$$P = V\lambda^{-\frac{1}{2}}V^T \tag{20}$$

Thus:

$$\tilde{x} = P\widehat{x} \tag{21}$$

where $\tilde{x}$ is the whitened or sphered data and $\lambda^{-\frac{1}{2}}$ is calculated by simple component wise operation as follows, $\lambda^{-\frac{1}{2}} = \left\{\lambda_1^{-\frac{1}{2}}, \lambda_2^{-\frac{1}{2}}, \ldots, \lambda_n^{-\frac{1}{2}}\right\}$, this step is called scaling. The whitening phase is as well called sphering, since the data becomes rotationally symmetrical like a sphere after the scaling step [37].

The covariance of the whitened data $E\{\widehat{x}\widehat{x}^T\}$ equals the identity matrix, and the mixing matrix $\tilde{A} = PA$ is orthogonal $(\tilde{A}^T = \tilde{A}^{-1})$. The matrix for extracting the ICs from $\tilde{x}$ is now denoted $\tilde{W}$, so $W = \tilde{W}^T P$.

Thus:

$$s = Wx = \tilde{W}^T Px \tag{22}$$

To calculate $\tilde{W}$, each column vector w_i is initialized and then updated so that ith independent component $(y_i \approx s_i = w_i^T\tilde{x})$ may have great non-gaussianity. One method of measuring non gaussianity is by maximizing negentropy and it is the most used method [40,41]. Negentropy is depend on the information theoretic quantity of differential entropy [36]. The (differential) entropy H of a random vector y with density $f(y)$ is defined as:

$$H(y) = -\int f(y)\log f(y)dy \tag{23}$$

A gaussian variable has the largest entropy among all random variables of equal variance [36]. To obtain a measure of non gaussianity that is zero for a gaussian variable, the negentropy J is defined as follows:

$$J(y) = H\left(y_{gauss}\right) - H(y) \tag{24}$$

where y_{gauss} is a gaussian random vector of the same correlation (and covariance) matrix as y. The negentropy is estimated as follows:

$$J(y) \propto [E\{G(y)\} - E\{G(v)\}] \tag{25}$$

$\propto$ denotes proportionality, but as we are only interested in the w_i that maximizes $J(w_i^T\tilde{x})$, the actual value at the maximum is not important [40]. y is supposed to be of unit variance and zero mean, and v is a gaussian variable of unit variance and zero mean, thus the term $E\{G(v)\}$ is a constant. $G(y)$ is a non-quadratic function. The best option of $G(y)$ depends on the problem, but common used functions are:

$$G_1(y) = \frac{1}{a_1}\log(\cosh(a_1 y)) \tag{26}$$

$$G_2(y) = -\frac{1}{a_2}exp\left(-a_2 y^2/2\right) \tag{27}$$

$$G_3(y) = \frac{1}{4}y^4 \tag{28}$$

where $1 \le a_1 \le 2$, $a_2 \approx 1$ are constants. Among these three functions, G_2 is a good general-purpose contrast function and was therefore selected for use in the present study.

Hyvärinen [39,42] introduced a very simple and highly efficient fixed-point algorithm for ICA, also gave a detailed explanation on estimating w_i and extracting ICs, using the fixed-point-FastICA algorithm.

Note: the percentage of the total variation in the data can be determined as:

$$t_m = \frac{Total\ Variance\ of\ H}{Total\ Variance} = 100 \times \frac{\sum_{i=1}^{m}\lambda_i}{\sum_{i=1}^{n}\lambda_i} \tag{29}$$

where m is number of components that contribute a given percentage of the total variation in the data, and $H = \{v_1, \ldots, v_m\}$ is the lower dimensional space.

Choosing the typical values for t_m range between 70% and 95% [26,43].

4.1.2. Feature selection

The name of the feature selection in machine learning, known as variable subset selection, is the selection of a subset of relevant variables [44,45] for use in model construction (classification or regression). The correlation coefficients between variables are often used for selecting the appropriate variables in water quality monitoring [46–48].

4.2. Classification methods

4.2.1. Deep learning using Long Short Term Memory Recurrent Neural Networks

Recurrent neural networks (RNNs) are a type of Deep Learning networks [49,50], where they contain rings (repetitions) within networks giving it is memory effect processes (the previous information is used to estimate the value that follows). These networks are very useful in identifying the subsequent sequence of certain data (predicting the next value of the string), where they retain some important features of sequential data. RNNs can also perform the classification task for every element of a time sequence, with the output being depended on the previous computations [51]. Fig. 2, illustrates the standard RNN architecture and an unfolded structure with t time steps.

As shown in Fig. 2 is the structure of RNN which includes input layer, hidden layer and output layer. The nodes in hidden layer are fully connected, the output of the hidden layer also becomes the input of the hidden layer at the next time. X_t is the input at t^{th} time. Y_t is the output at t^{th} time. h_t is the state of hidden layer at t^{th} time. W_{xh} is the weight between input layer and hidden layer, W_{yh} is the weight between hidden layer and output layer, W_{hh} is the weight between current hidden layer and hidden layer at next time.

In theory, RNNs are supposed to carry information until a long time. But in practice they have difficulties learning long-range dependencies, because it is very difficult to propagated all this information when the time step is too long [52]. When the network contains a large number of deep layers, it becomes untrainable (loss of information). This problem is called: vanishing gradient problem [53]. The neural network updates the weights using the gradient descent algorithm [49,52]. The gradients become smaller as the network progresses to the lower layers. If the gradient stays constant, means there is no improvement. The model learns from the change in the gradient, this change affects the output of the network. However, if the difference in the gradient is very small (i.e., the weights change a little), the network will not be able to learn anything [52].

To overcome this problem, Sepp and Schmidhuber [54], improved RNNs by creating an architecture called Long Short Term Memory (LSTM). LSTM provides the network with past information relevant to a more recent time [55]. The network uses a better structure to identify and transmit information to later. A schematic of a unit of the LSTM RNNs used in this work can be seen in Fig. 3.

$$f_t = \sigma\left(W_f.[C_{t-1}, y_{t-1},\ x_t] + b_f\right) \tag{30}$$

$$i_t = \sigma(W_i.[C_{t-1}, y_{t-1},\ x_t] + b_i) \tag{31}$$

$$O_t = \sigma(W_O.[C_{t-1}, y_{t-1},\ x_t] + b_O) \tag{32}$$

$$C_t = f_t{*}C_{t-1} + i_t{*}Z \tag{33}$$

$$Z = \tanh(W_C.[y_{t-1},\ x_t] + b_C) \tag{34}$$

$$y_t = O_t{*}\tanh(C_t) \tag{35}$$

where x_t is input vector, y_t is state of the output units, C_t is cell state vector, W and b are parameter matrices and vector, f_t, i_t and O_t are forget gate, input gate and output gate vectors. σ is sigmoid function and tanh is a rescale logistic sigmoid function between - 1 and 1 [18].

4.2.2. Support Vector Machines (SVMs)

SVMs are a relatively new statistical learning technique, proposed by V. Vapnik in 1995. It allows to address very different problems like the classification, the regression and the density estimation [57,58]. But most of SVMs are used for classification. The essential idea for SVM is to project non-linearly separable input space data (belonging to different classes) into a high-dimensional space called feature space, so that the data becomes linearly separable. In this space, the optimal hyperplane construction technique is used to calculate the classification function that separates the classes. Training the SVM is a quadratic optimization problem [59].

4.2.2.1. Basic SVM (binary classifier). Suppose given the training sample: $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$, where $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$. It is assumed at first that there is a linear separator to distinguish positive examples $+1$ from negative examples -1. We know that the search for such a separator in the input space χ returns to look for a hypothesis function $f(x) = w^T x_i + b$ such as [60]:

$$w^T x_i + b \begin{cases} > 0 \\ < 0 \end{cases} \Rightarrow y_i = \begin{cases} +1 \\ -1 \end{cases} \tag{36}$$

where b is a constant and w is a vector with M-dimensions. The constant b and the vector w are utilized to determine the position of separating hyperplane.

This separator is valid on the learning sample if: $\forall\ 1 \le i \le m,\ y_i f(x) > 0 \Rightarrow \forall\ 1 \le i \le m,\ y_i\ (w^T x_i + b) > 0$.

The function $f(x)$ corresponds to the equation of a hyperplane in the input space χ of normal vector w. The distance from a point x to the hyperplane of equation $f(x) = w^T x + b$ is equal to: $\frac{f(x)}{\|w\|}$, where $\| w \|$ is the Euclidean norm of the vector w. When there is a linear separator between the training points, there is usually an infinity. We can then search among these separators for the one that is (in the middle) of the two examples and counter-examples points clouds.

This optimal hyperplane is defined by:

$$\underset{w,\ b}{\text{Argmax}}\ \min\{\ \| x - x_i \| : x \in \mathbb{R}^m,\ (w^T x + b) = 0,\ i = 0,\ \ldots,\ m\} \tag{37}$$

that is, the hyperplane that increases the minimum distance to the

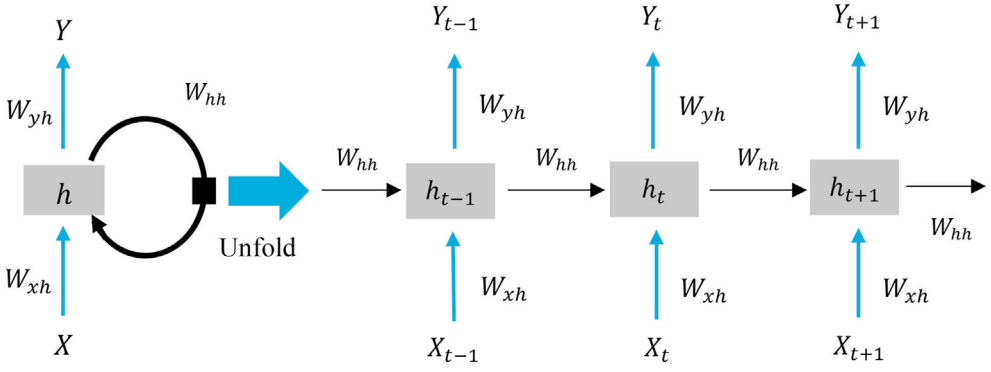

Fig. 2. Standard RNN architecture and an unfolded structure with t time steps.

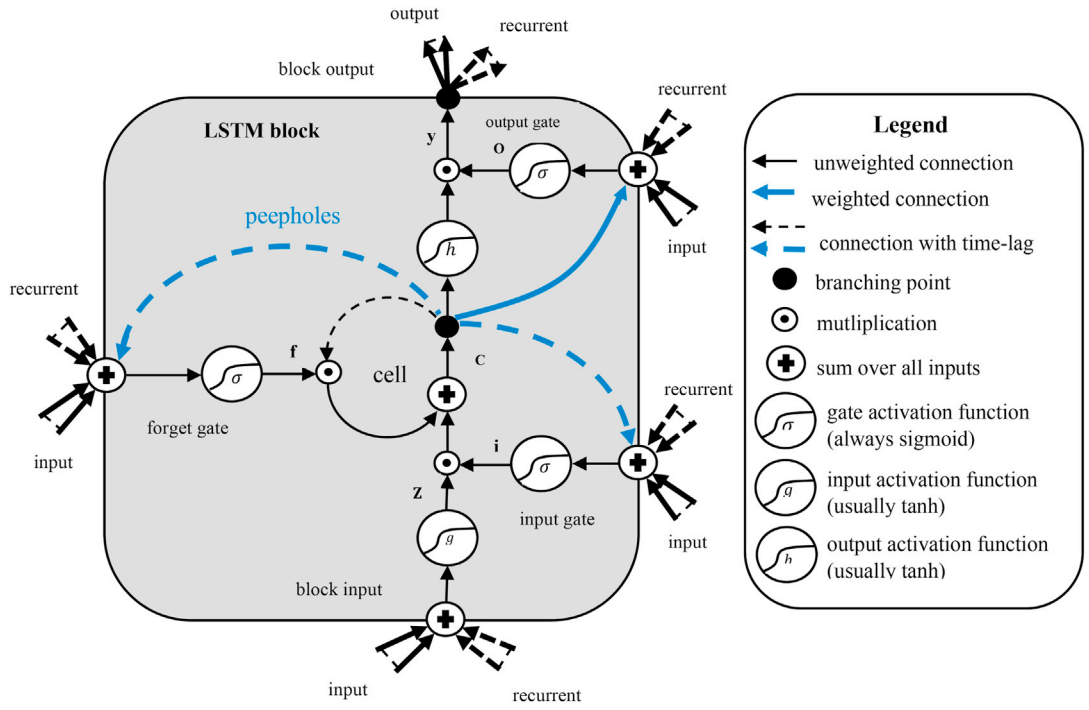

Fig. 3. Long Short-Term Memory recurrent neural network architecture [56].

training examples (Fig. 4).

The objective of the SVM is to find a separator that minimizes the classification error on the learning set, this means increasing the margin of the hyperplane. In other words, we have to solve the problem of a quadratic optimization according to which relates the parameters w, b [59]:

$$\begin{cases} \underset{w}{\text{Min}} \frac{1}{2} \| w \|^2 + C \sum_{i=1}^{M} \xi_i \\ s.t. \quad y_i(wx + b) \geq 1 - \xi_i \\ \xi_i \geq 0, \ i = 1, \ldots, n \end{cases} \tag{38}$$

where ξ_i is slack variable, measuring the distance between the margin and the examples x_i that lying on the wrong side of the margin [28], and

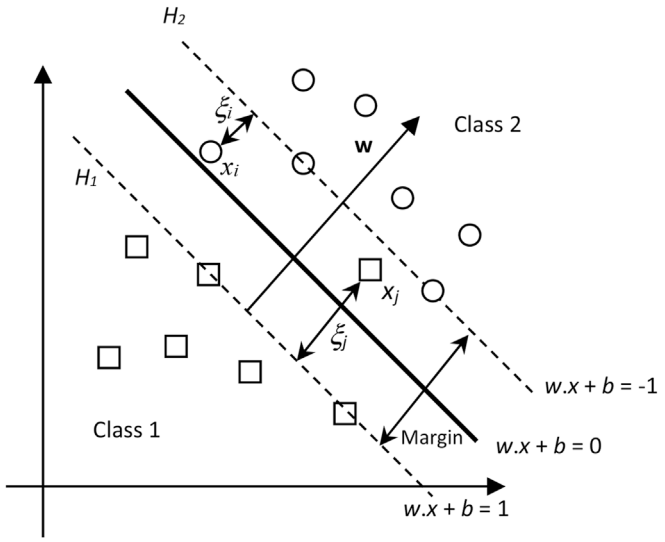

Fig. 4. The structure of a basic SVM.

C is a strictly positive constant that allows the compromise between the number of classification errors and the width of the margin.

By using the multipliers of Lagrange α_i, we get the following dual problem [59]:

$$\begin{cases} \underset{\alpha_i}{\text{Max}} L(\alpha) = \sum_{i=1}^{M} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{M} \alpha_i \alpha_j y_i y_j (x_i . x_j) \\ s.t. \quad \sum_{i=1}^{n} \alpha_i y_i = 0, \ 0 \leq \alpha_i \leq C \end{cases} \tag{39}$$

where α_i are solution of (eq. (39)).

On the other hand, SVMs can also be used in non-linear classification tasks with application of kernel functions $K(x_i.x_j)$ that define a nonlinear mapping from the input space to higher dimensional feature space [61]. The dual problem becomes as following [62]:

$$\begin{cases} \underset{\alpha_i}{\text{Max}} L(\alpha) = \sum_{i=1}^{M} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{M} \alpha_i \alpha_j y_i y_j K(x_i . x_j) \\ s.t. \quad \sum_{i=1}^{n} \alpha_i y_i = 0, \ 0 \leq \alpha_i \leq C \end{cases} \tag{40}$$

where α_i are solution of (eq. (40)).

The necessary and sufficient condition for an optimal α_i according to Karush-Kuhn-Tucker (KKT) theorem is [59,63,64]:

$$\alpha_i (y_i (w.x_i + b)) = 0, \ \forall i \ \in M \tag{41}$$

Solve (eq. (41)) is $\alpha_i = 0$ or $y_i(w.x_i + b) = 1$. The last term corresponds to the Support Vectors (SVs), which is equivalent to Ref. [59]:

$$SVs = \{x_i \text{ that } \quad \alpha_i > 0\} \tag{42}$$

The decision function is given by Ref. [65]:

$$f(x) = sign\left(\sum_{i,j=1}^{M} \alpha_i y_i K(x_i, x_j) + b \right) \tag{43}$$

Any function that satisfies Mercer's conditions can be utilized as a

kernel function to calculate a dot (scalar) product in feature space [28]. There are different kernel functions used in SVM such as [65,66]:

Linear kernel:

$$K(x_i, x_j) = x_i^T . x_j \tag{44}$$

Polynomial function:

$$K(x_i, x_j) = (\gamma x_i^T . x_j + r)^d \tag{45}$$

Radial Basis Function (RBF):

$$K(x_i, x_j) = exp\left(- \frac{\| x_i - x_j \|^2}{2\gamma^2} \right) \tag{46}$$

Sigmoid kernel:

$$K(x_i, x_j) = \tanh\{\gamma x_i^T . x_j + r\} \tag{47}$$

where d, $r \in \mathbb{N}$ and $\gamma \in \mathbb{R}^+$ are constants.

The most used kernel functions are RBF and polynomial [59].

4.2.2.2. Multi-class SVMs. SVMs are in their binary origin. However, real-world problems are in most cases multiclass. The term multi-class means a classification involving more than two classes. Multiclass support vector machine methods reduce the multiclass problem to a composition of several two-class hyperplanes making it possible to draw the decision boundaries between the different classes. These methods break down the set of examples into several subsets, each representing a binary classification problem. For each problem a separation hyperplane is determined by the binary SVM method. During the classification, a hierarchy of binary hyperplanes is constructed which is traversed from the root to a leaf to decide on the class of a new example. Many proposals have been made to reformulate the SVM in a multi-class framework. The two best known methods are: "*one-against-all*" and "*one-against-one*" approaches [61,67].

- **One-against-All (OAA) approach**

It is the simplest and oldest method. According to Vapnik's formulation [68], it consists in determining for i^{th} class a hyperplane $f(w_i, b_i)$ separating it from all the other classes, where $i = 1,...,N$. This class i is considered to be the positive class $(+1)$ and the other classes to be the negative class (-1), which results, for a problem with N classes, in N binary SVM. The SVM of the i^{th} class solves the following problem [61, 67]:

$$\underset{w^i, b^i, \xi^i}{Min} \frac{1}{2} \| w^i \|^2 + C \sum_{i=1}^{N} \xi_j^i (w^i)^T \tag{48}$$

subject to:

$$\begin{cases} (w^i)^T \varphi(x_j) + b^i \geq 1 - \xi_j^i, \ if \ y_j = i, \\ (w^i)^T \varphi(x_j) + b^i \leq 1 - \xi_j^i, \ if \ y_j \neq i, \\ \xi_j^i \geq 0, \ j = 1, \ ..., \ N, \end{cases} \tag{49}$$

The function of decision is given by Refs. [60,61,67]:

$$f_i(x) = \arg \max_{i=1, \ ..., \ N} \left(w_i^T \varphi(x) + b_i \right) \tag{50}$$

- **One-against-One (OAO) approach**

OAO approach consists in using a classifier for each pair of classes. Instead of learning N decision functions, the OAO method discriminates each class from each other class, so $N(N-1)/2$ decision functions are learned. For training data from the i^{th} and the j^{th} classes, we solve the following binary classification problem [61,67]:

$$\underset{w^{ij}, b^{ij}, \xi^{ij}}{Min} \frac{1}{2} \| w^{ij} \|^2 + C \sum_t \xi_t^{ij} (w^{ij})^T \tag{51}$$

subject to:

$$\begin{cases} (w^{ij})^T \varphi(x_t) + b^{ij} \geq 1 - \xi_t^{ij}, \quad if \ y_t = i, \\ (w^{ij})^T \varphi(x_t) + b^{ij} \leq -1 + \xi_t^{ij}, \quad if \ y_t \neq i, \\ \xi_t^{ij} \geq 0, \quad j = 1, \ ..., \ N, \end{cases} \tag{52}$$

During classification, an input vector x is presented to all classifiers constructed. The output of each SVM provides a partial vote concerning only the couple of classes (w_i, w_j). Considering that each SVM calculates an estimate of the probability, the simplest classification rule can be written [69]:

$$f^{ij}(x) = (w^{ij} \bullet \varphi(x_t)) + b^{ij}, \ (i < j \leq N) \tag{53}$$

Voting strategy is used for classification. If $sgn(f^{ij}(x)) = 1$, then $vote(i)_{i=1, \ ..., N}$ of class i add 1, otherwise $vote(j)$ of class j plus 1. The new sample x is classify using the $N(N-1)/2$ standard SVM decision function $f_{ij}(\bullet)$, and the corresponding votes are recorded. The decision function is gained as follows [69]:

$$f(x) = \arg \max_{i=1, \ ..., \ N} \{vote(i)\} \tag{54}$$

5. Results and discussion

5.1. Variables selection

In this section, we will reduce the dimensions of the input variables and determine the appropriate variables to build a good classifier by features extraction based on cumulative variance and features selection based on the correlation between variables. From features extraction using PCA, LDA and ICA, we can know that there is a change from data features. The selection of features will be based on the following two conditions:

- Variables are selected by selecting one variable in each factor and has the highest correlation value (positive or negative) with this factor.
- Will be the selection priority to the parameters that have physical sensors, since these parameters are measured directly from the sensors installed in the station, and measured continuously and at any level of the all treatment process such as pH, EC, T° and TU, this allows us to create an intelligent monitoring system that works permanently and continuously. Unlike chemical parameters that are measured once a week and measured by complex and very expensive means in laboratories such as Mg, B, H and FAT, this does not allow us to create an intelligent control system that works permanently and continuously.

To implement this phase a total of 122 samples are obtained from 8 variables of water quality data. Fig. 5 represents the temporal evolution of the descriptor parameters of raw water quality (Station Tilesdit).

5.1.1. Variables selection with PCA

The first step in PCA is to standardize the data (z-score). "Standardization" means subtracting the sample mean from each observation, then dividing by the sample standard deviation. This centers and scales the data. Fig. 6 represents z-score variability of data.

The vertical line (red) within each box displays the average of the normalized values for the parameter. The vertical lines (black) which in lie beyond the boxes represent the minimum and maximum values. For example, in case of Permanent hardness (H), the median z-score value is

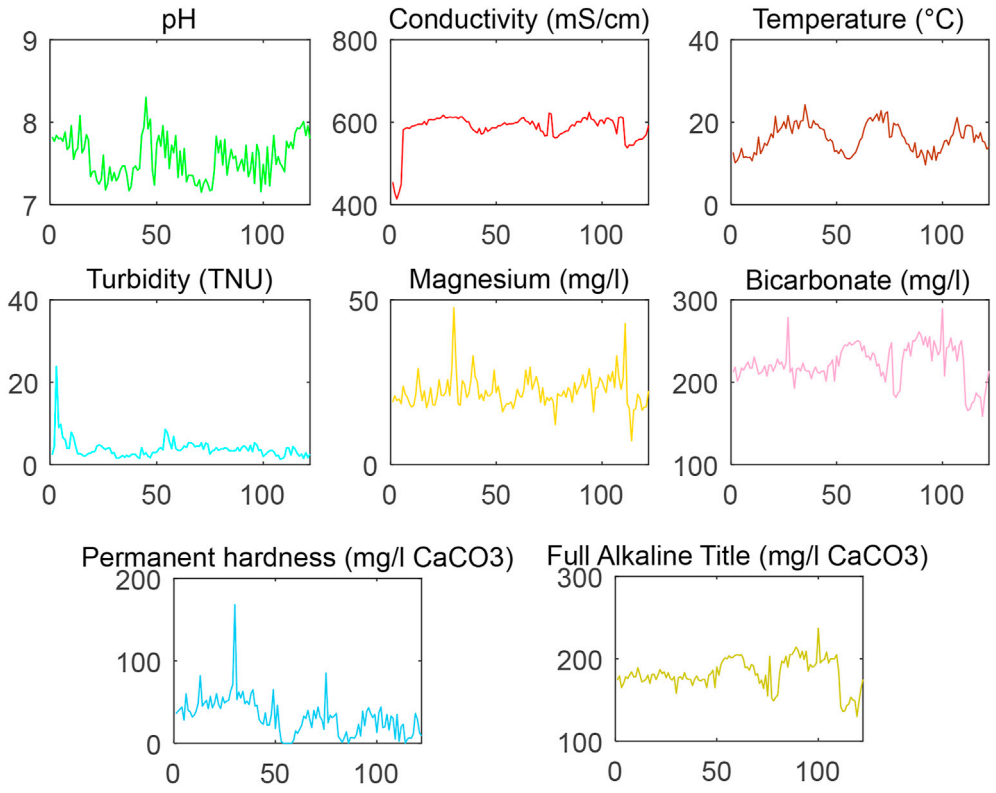

Fig. 5. Temporary evolution of descriptor parameters for the raw water quality (Tilesdit station).

0, whereas, the minimum and maximum z-score values are approx. −1.4 and 2.4 respectively. The '+' symbol represents the outliers values, and the whiskers expand to the most extreme data points that are not considered outliers.

The goal of estimate z-scores are obtaining a set of linearly transformed scores. After estimation of z-scores, the parameters are then subjected to PCA. PCA method was used to reduce the feature dimensionality that contains 75% variation of eigenvalues. Also, we can understand that there is a change from data features to components which are uncorrelated. The original data and the first three PCs are plotted in Figs. 7 and 8 respectively. From Fig. 8, it can be observed that the data cluster for three classes are separated.

Table 2 and histogram data depicted in Fig. 9 represent PCA application for the total dataset. From Fig. 9 can be seen a fast decrease in eigenvalues. We can also note that the first four components in Table 2 represent 84.682% of total variance proportion.

Selecting 75% of the total variance means choosing the first four factors:

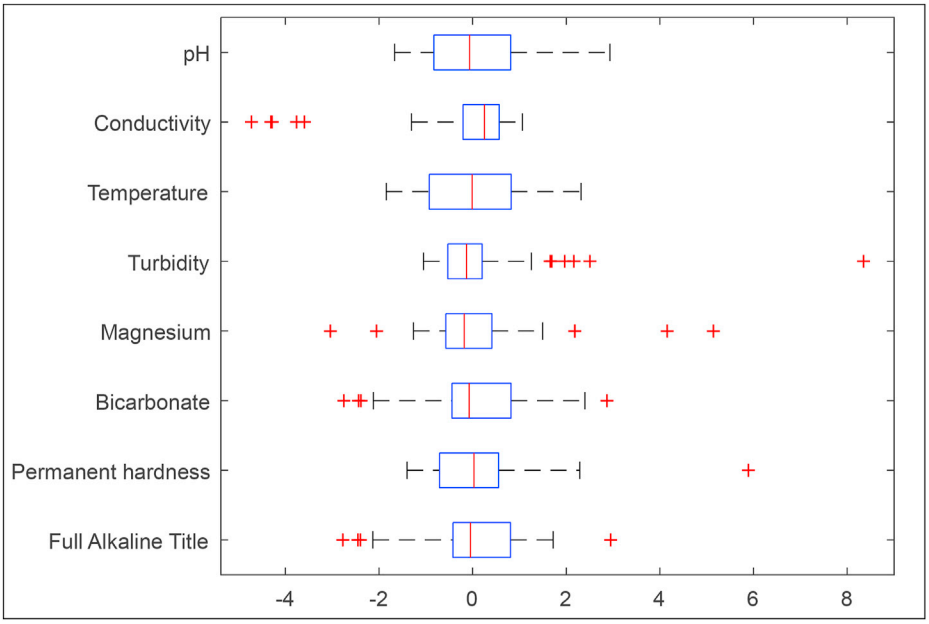

Fig. 6. z-score variability of data.

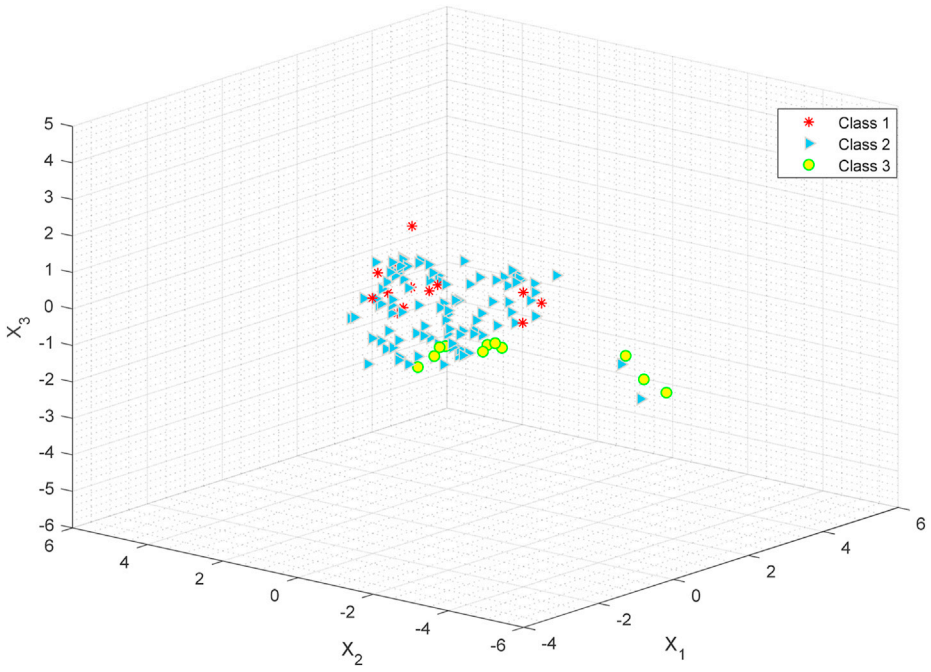

Fig. 7. Original data.

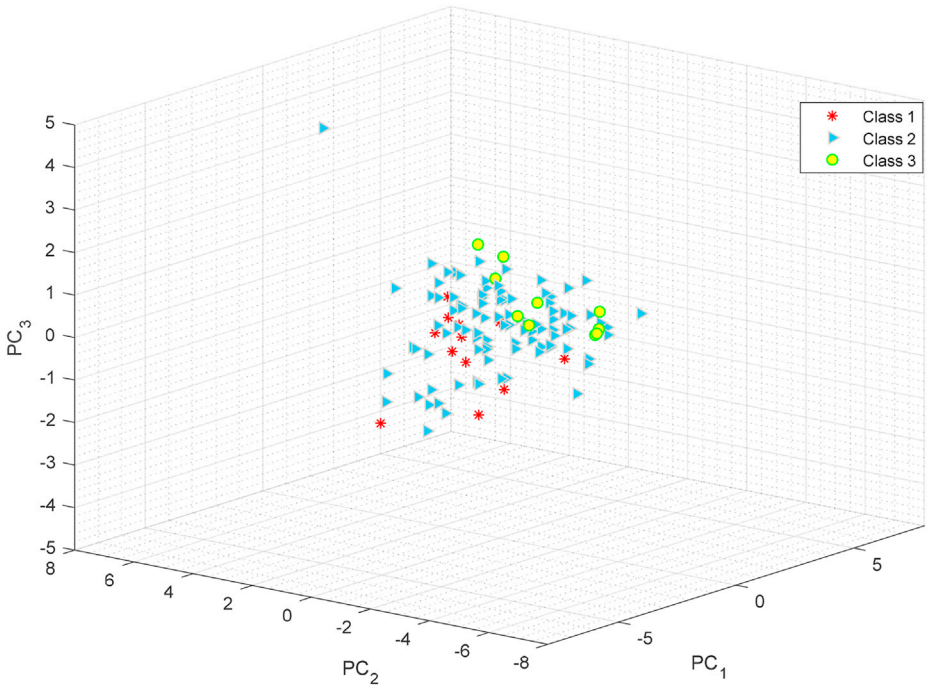

Fig. 8. Feature extraction using PCA.

- Factor one (32.070%), in this factor, EC, B and FAT show strong positive loading.
- Factor two (27.933%), in this factor, T° and H show strong positive loading.
- Factor three (14.048%), in this factor, TU shows strong positive loading.
- Factor four (10.630%), in this factor, pH shows positive loading.

This is illustrated in Table 3. This table, represents the correlation between variables, eigenvalues, variance proportion (%) and the cumulative variance proportion for the four first principal components. We can observe that F1 (PC1) correlates positively with EC (0.736), B (0.867) and FAT (0.854). However, F2 (PC2) correlates positively with T° (0.780) and H (0.704). While F3 (PC3) correlates positively with TU (0.729) only. F4 (PC4) correlates positively with pH (0.516) only.

Finally, we can only maintain the parameters pH, EC, T° and TU to build an intelligent model.

5.1.2. *Variables selection with LDA*

As we mentioned earlier, LDA technique finds a linear combination of features that best separate given classes of original data. Can be utilized as a linear classifier or to reduce the dimensions before the classification

Table 2
Descriptive statistics of the created principal components.

	F1	F2	F3	F4	F5	F6	F7	F8
Eigenvalues								
	2.566	2.235	1.124	0.850	0.572	0.328	0.266	0.059
Percent of total variance proportion								
	32.070	27.933	14.048	10.630	7.156	4.098	3.323	0.742
Cumulative percent of total variance proportion								
	32.070	60.004	74.052	84.682	91.838	95.935	99.258	100
Variables eigenvectors obtained through the PCA application								
pH	−0.572	−0.464	−0.253	0.516	−0.080	0.286	0.196	0.000
EC	0.736	0.300	−0.374	0.081	0.263	0.327	−0.211	0.003
T°	−0.037	0.780	−0.100	−0.446	−0.283	0.224	0.225	0.005
TU	−0.260	−0.473	0.729	−0.275	0.071	0.295	−0.100	0.004
Mg	0.381	0.479	0.430	0.478	−0.442	0.026	−0.127	−0.014
B	0.867	−0.411	0.107	−0.028	0.014	0.006	0.198	−0.167
H	−0.032	0.704	0.419	0.272	0.459	−0.021	0.205	0.019
FAT	0.854	−0.450	0.086	−0.002	−0.071	−0.009	0.156	0.175

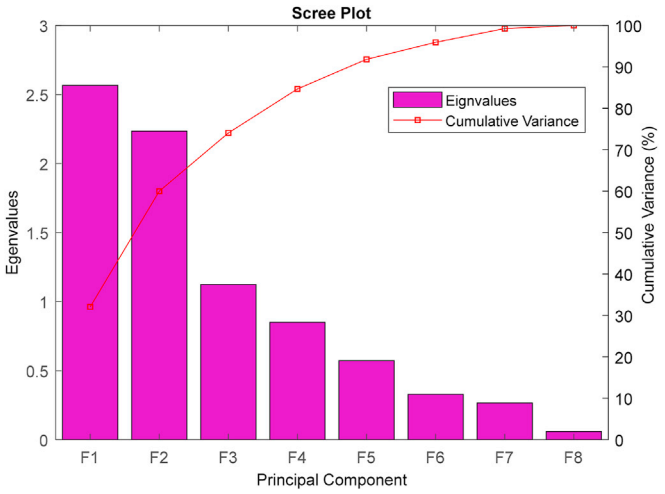

Fig. 9. Histogram of the principal component eigenvalues.

Table 3
Correlation between variables, eigenvalues, variance proportion (%) and the cumulative variance proportion in the four PCs.

Inputs (variables)	Factors (PCs)			
	F1	F2	F3	F4
pH	−0.572	−0.464	−0.253	**0.516**
EC	**0.736**	0.300	−0.374	0.081
T°	−0.037	**0.780**	−0.100	−0.446
TU	−0.260	−0.473	**0.729**	−0.275
Mg	0.381	0.479	0.430	0.478
B	0.867	−0.411	0.107	−0.028
H	−0.032	0.704	0.419	0.272
FTA	0.854	−0.450	0.086	−0.002
Eigenvalues	2.566	2.235	1.124	0.850
Variance proportion (%)	32.070	27.933	14.048	10.630
Cumulative variance proportion (%)	32.070	60.004	74.052	84.682

process. The main purpose of LDA in this work is to implement dimensionality reduction (feature extraction) while preserving as much of the class discriminatory information as possible. The first three discriminant analysis are plotted in Fig. 10, it can be observed that the clusters for three classes are well separated. Nevertheless, the performance of LDA is better than PCA does in clustering of each class.

Table 4 and histogram data shown in Fig. 11 represent LDA application for the total dataset. From Fig. 11, it can be seen that only two discriminant function coefficients have been extracted, since the canonical variants are c-1, if the number of variables is larger than c, in other words, LDA reduces dimensions to c-1. The reason is the transformation of X to Y is done through projecting the samples in X onto a hyperplane with dimension c-1. Where c is the number of classes (in this case we have c = 3).

In Table 4, we listed the correlation between variables, eigenvalues, variance proportion (%) and the cumulative variance proportion for the first two factors. From this table, we can note that the first two factors represent 100% of total variance proportion:

- Factor one (98.483%), in this factor, TU shows strong negative loading, whereas EC and T° show positive loading.
- Factor two (1.517%), in this factor, pH shows positive loading, whereas EC, Mg, B and FAT show negative loading.

We can observe also that F1 correlates positively with EC (0.433) and T° (0.564), and negatively with TU (−0.942). While F2 correlates positively with pH (0.324), and negatively with EC (−0.309), Mg (−0.663), B (−0.556) and FAT (−0.475).

Finally, we can only maintain the parameters TU and EC to build an intelligent model.

5.1.3. *Variables selection with ICA*

The main goal of using ICA in this work is to perform dimensionality reduction (feature extraction). This is done by extracting source signals (ICs) and mixing coefficients from the matrix of mixture signals, by calculating a linear transformation that maximizes a criterion measuring of the statistical independence between the sources. ICs are linear combinations of the initial variables, with maximum non-gaussianity and thus maximum independence.

At first, we normalize the data between 0 and 1, since the FastICA algorithm is very sensitive to outliers. Then we apply PCA which is a preprocessing phase of the ICA algorithm. Fig. 12, displays the scatter plot for the data studied in this work. Fig. 12(a), shows the scatter of the original mixtures forms an ellipse. Fig. 12(b), represents the projection of the mixture signals onto the PCA space, this leads to rotate the principal components to be aligned with the X_1 and X_2 axes and hence the ellipse is also rotated. After the whitening step, we observe that the contour of the mixture signals forms a circle, this is because the signals have unit variance, as shown in Fig. 12(c).

Now we can apply the FastICA algorithm to extract the ICs. ICA was used to reduce the feature dimensionality that contains 75% variation of eigenvalues. The first three ICs are plotted in Fig. 13, it can be seen that the clusters for three classes are well separated. Nevertheless, ICA performed better than PCA in clustering of each class.

Table 5 and histogram data depicted in Fig. 14 represent FastICA algorithm application for the total dataset. From Fig. 14 can be seen a fast decrease in eigenvalues. We can also observe that the first three components in Table 5 represent 79.253% of total variance proportion.

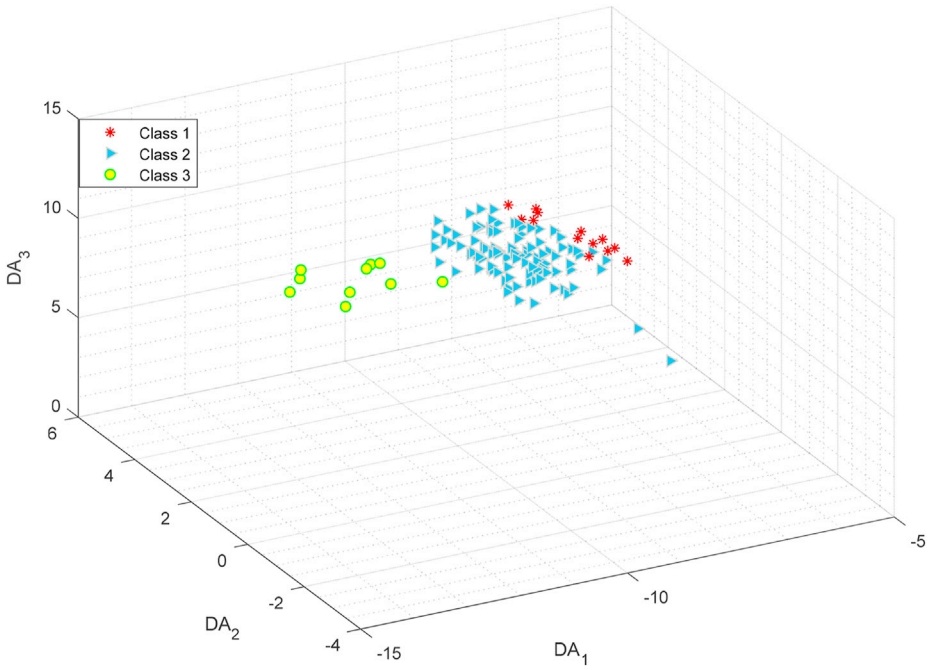

Fig. 10. Feature extraction using LDA.

Table 4
Descriptive statistics of the created discriminant function coefficients.

	F1	F2
Eigenvalues		
	1.392	0.021
Percent of total variance proportion		
	98.483	1.517
Cumulative percent of total variance proportion		
	98.483	100
Variables eigenvectors obtained through the LDA application		
pH	−0.283	0.324
EC	0.433	**−0.309**
T°	0.564	0.115
TU	**−0.942**	−0.049
Mg	0.240	−0.663
B	−0.112	−0.556
H	0.133	0.213
FAT	−0.120	−0.475

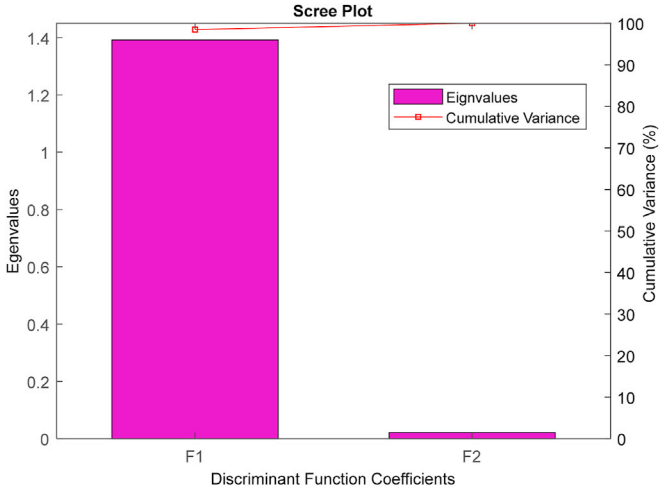

Fig. 11. Histogram of the discriminant function coefficients eigenvalues.

Selecting 75% of the total variance means choosing the first three factors:

- Factor one (35.270%), in this factor, EC shows strong positive loading, whereas TU shows negative loading.
- Factor two (34.025%), in this factor, only TU shows strong positive loading.
- Factor three (9.959%), in this factor, T° shows strong positive loading, whereas B shows negative loading.

Table 6, represents the correlation between variables, eigenvalues, variance proportion (%) and the cumulative variance proportion for the three first independent components. We can observe that F1 (IC1) correlates positively with EC (0.896) and negatively with TU (−0.634). However, F2 (IC2) correlates positively with TU (0.756) only. While F3 (IC3) correlates positively with T° (0.880), and negatively with B (−0.529).

Finally, we can only maintain the parameters EC, TU and T° to build an intelligent model.

5.2. *Training and classification*

In this work, we used the two methods explained above namely LSTM RNNs and SVMs multi-class in data training and classification. The hardware and software used to perform our simulation experiments are: an Intel Core TM i3 and 2.40-GHz CPU processor with 4-GB RAM memory. For the LSTM code, we have programmed it ourselves, and for the SVM algorithm, we have used kernel method package [70].

The LSTM RNNs architecture used in this work for training and classification, contains a sequence input layer followed by an LSTM layer. The network ends with a three fully connected layer, a softmax layer, and a classification output layer to predict class labels. Experimentally we used two hidden layers of LSTM RNNs (since the training time would increase with a large number of neurons and iterations, as we use only the CPU).

Concerning to SVMs, the Gaussian RBF and polynomial are used as the basic kernel function. These kernel functions share two basic parameters: the first is the bound on the lagrangian multipliers *C*, and the second is the conditioning parameter for quadratic programming method

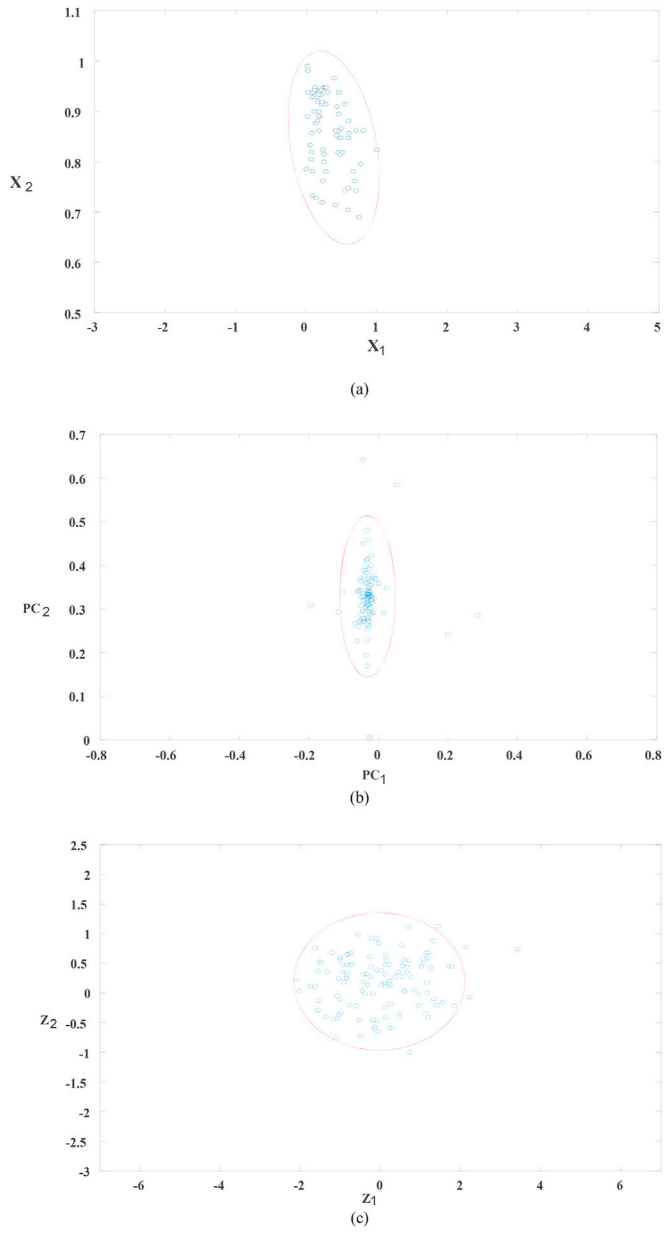

Fig. 12. Visualization for mixture signals of the data studied in this work during the whitening step. (a) Scatter plot for the mixture signals, (b) the projection of mixture signals onto the PCA space, i.e., decorrelation, (c) mixture signals are scaled after the whitening step to have a unit variance.

γ. Additionally, polynomial kernel also contains parameter d related to degree of polynomial. These parameters play a crucial role in performance of SVMs. Therefore, must selection these parameters carefully and correctly. In this study, a standard grid search is used to determine the proper kernel parameter of $d = \{1, 2, 3\}$, $C = \{2^{-3}, 2^{-2}, \ldots, 2^{7}\}$ and $\gamma = \{2^{-13}, 2^{-12}, \ldots, 2^{-3}\}$. In addition to all this, the SVMs-based multi-class classification is applied to perform the classification process using one-against-one and one-against-all approaches.

According to the Environmental Quality Standards of water, we used for this phase a complete real database consisting of 1200 samples. Our problem is how to estimate the performance of LSTM RNNs model. It is known that when we can assume independence and an identical distribution (i.i.d) between observations, the use of standard *k*-fold cross-validation (CV) is typically the most appropriate method. The issue is that LSTM RNNs models are frequently used for time series data in the widest sense, e.g. actual time series, texts, etc. These kinds of dataset are auto-correlated most of the time, i.e. they depend on the order of events. The water quality monitoring data are the time series data and greatly affected by seasons with obvious seasonal diversity [71]. Consequently, the standard *k*-fold CV technique cannot be applied directly to time series data, since it leads to theoretical and practical problems. The reason is that this technique the dependency between observations is not taken into account since the standard *k*-fold CV assumes that the values of the time series are i.i.d [72]. On the other hand, there is considerable controversy in the literature regarding the use of CV to estimate the performance of time series prediction models. Some argue [73–75] that the use of CV methods (including the standard *k*-fold CV) to estimate the performance of time series prediction models is normal and that their performance better than use the out-of-sample (OOS) test methods. Others argue [72,76] that these techniques cannot be applied to estimate the performance of real-world time series prediction models, and these techniques can only be used with stationary and artificial time series, whereas in real-world time series, where time series is more complex than time series in the artificial world, the CV is not recommended used, and the out-of-sample (OOS) test methods and in particular adopt a randomized approach with the OOS test are recommended used. For this, we used three methods of CV and two methods of OOS test:

- Regarding CV methods: the first method is standard *k*-fold CV with k = 5 and 10. In *k*-fold CV generally, the dataset is first divided randomly into subsets of equal size. The *k*-fold CV takes the data sample, leaves a part out for testing and trains the model on the rest (*k*-1 folds). This process is repeated *k*-times until the entire dataset is covered; the second method we have used *v*-fold CV (Multiple Train-Test splits) suitable for time series case [77]. The method involves repeating the process of splitting the time series into train and test sets *v*-times (in this study we used 5-fold CV, i.e. 5 splits). The test size remains fixed while training set size will increase for every fold, with respect the order of time series data; and the third method is the Blocked cross validation (CV-Bl) procedure [76]. This method is similar to the standard CV. But in this procedure, the time series is divided into *k*-blocks of equal size without initial random shuffling, with the natural order of observations is kept within each block. In this study we divided the data into 5-blocks.
- Regarding OOS test methods: the first method is the simple Holdout (OOS.H). In the OOS.H method, the first 70% of the time series data are used for training and the subsequent 30% are used for testing; and the second method is Random-Holdout. This method creates a random non stratified partition for holdout validation on n observations. Random-Holdout method, randomly selects approximately $n \times p$ observations to holdout for the test (evaluation) set. The parameter p must be a scalar, where $0 < p < 1$. In this study, 70% of the data is used for training and 30% of the data for testing.

In order to evaluate the performance of our models, the accuracy (Acc) criterion is used and it is defined as follows:

$$\mathrm{Acc} = \frac{TP + TN}{TP + TN + FP + FN} \tag{55}$$

where TP, TN, FP and FN are true positive, true negative, false positive and false negative respectively.

5.2.1. Classification using LSTM RNNs

In this section, we present and discuss the results of data classification using LSTM RNNs and feature extraction techniques. In addition, we will evaluate and compare the performance of the methods used to estimate the performance of models of the classification. The results of this study can be shown in Tables 7–12. In these tables, we listed the feature extraction technique, classification rate for testing, training and testing time. The results described in these tables were obtained by using single layer LSTM RNNs, the number of neurons in the hidden layer (Hiddnum)

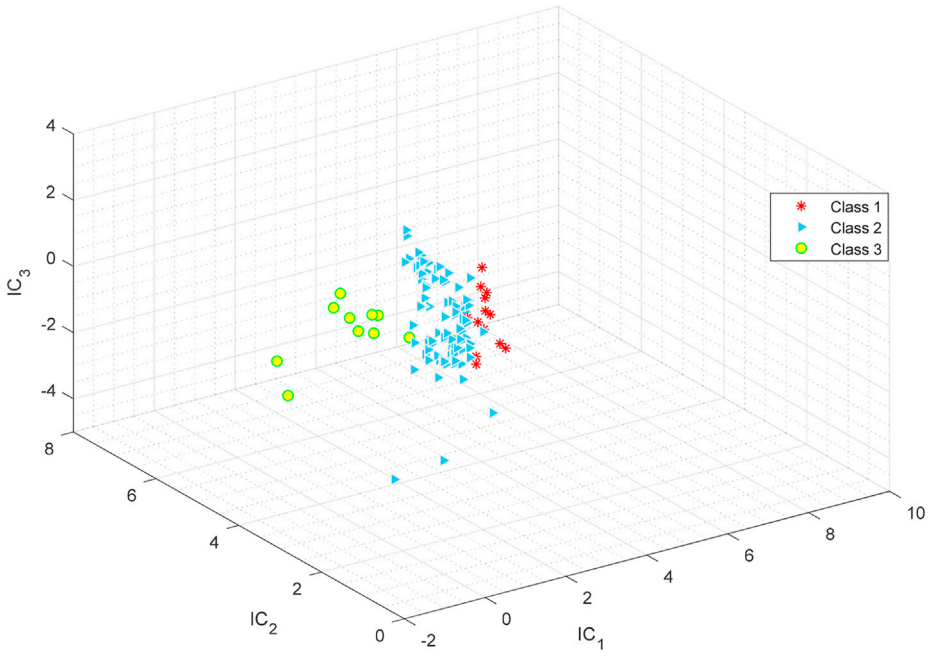

Fig. 13. Feature extraction using ICA.

Table 5
Descriptive statistics of the created independent components.

	F1	F2	F3	F4	F5	F6	F7	F8
Eigenvalues								
	0.085	0.082	0.024	0.021	0.014	0.008	0.005	0.002
Percent of total variance proportion								
	35.270	34.025	9.959	8.714	5.809	3.320	2.075	0.830
Cumulative percent of total variance proportion								
	35.270	69.295	79.253	87.967	93.776	97.095	99.170	100
Variables eigenvectors obtained through the ICA application								
pH	−0.211	0.075	−0.430	0.123	−0.061	−0.323	0.467	0.201
EC	0.896	0.062	−0.144	−0.095	0.036	0.297	−0.359	−0.053
T°	0.275	−0.121	0.880	−0.073	0.248	−0.021	−0.138	−0.385
TU	−0.634	0.756	−0.021	−0.015	−0.143	0.031	−0.151	0.074
Mg	0.123	−0.078	−0.008	0.037	0.914	−0.288	−0.651	−0.889
B	0.109	−0.025	−0.529	−0.223	−0.226	0.501	−0.664	0.118
H	−0.051	−0.046	0.258	−0.177	0.580	0.314	−0.304	−0.156
FAT	0.105	−0.028	−0.499	0.136	−0.204	0.495	−0.678	0.047

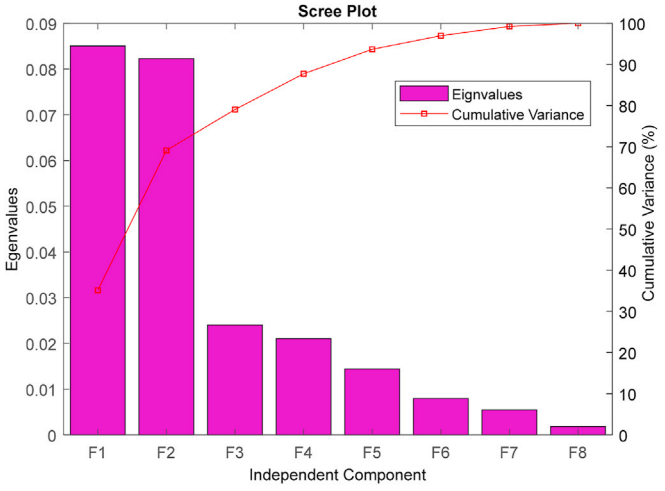

Fig. 14. Histogram of the independent component eigenvalues.

Table 6
Correlation between variables, eigenvalues, variance proportion (%) and the cumulative variance proportion in the three ICs.

Inputs (variables)	Factors (ICs)		
	F1	F2	F3
pH	−0.211	0.075	−0.430
EC	**0.896**	0.062	−0.144
T°	0.275	−0.121	**0.880**
TU	−0.634	**0.756**	−0.021
Mg	0.123	−0.078	−0.008
B	0.109	−0.025	−0.529
H	−0.051	−0.046	0.258
FAT	0.105	−0.028	−0.499
Eigenvalues	0.085	0.082	0.024
Variance proportion (%)	35.270	34.025	9.959
Cumulative variance proportion (%)	35.270	69.295	79.253

= 100, the number of iterations (epochs) = 200, batch size = 27 training examples and the time step = 7 h.

As seen in Table 7, the classification rate using OOS.H ranged from 98.06% to 99.17%. Whereas in Table 8, the classification rate using

Random-Holdout ranged from 99.44% to 99.72%. While in Table 9, the classification rate using Multiple Train-Test splits ranged from 97.60% to 97.80%. Whereas in Table 10, the classification rate using CV-Bl ranged from 97.92% to 98.50%. In Table 11, the classification rate using 5-fold CV ranged from 98.50% to 98.75%. Then, in Table 12, the classification rate using 10-fold CV ranged from 98.58% to 98.75%.

We can observe that Random-Holdout shows the best estimate followed by OOS.H, *k*-fold CV, CV-Bl and Multiple Train-Test splits respectively. It is clear, the use of Multiple Train-Test splits, CV-Bl and *k*-fold CV methods does not suitable for estimating the performance of our models, since they ignore the temporal components inherent in the problem. Obviously, use of the OOS test, and in particular use of the OOS test with adopt a randomized approach (Random-Holdout method) is better option as a method to estimate the performance of our models. In addition, the results of this method are reliable and unbiased.

In case of use Random-Holdout, we note that the classification rate with LDA and ICA is highest 99.72%. The PCA had lowest classification rate 99.44%. ICA performs better than PCA because ICA finds the components not only uncorrelated but independent. ICs are more important than uncorrelated components in classification process, because the negentropy in ICA can take into consideration the higher-order information of the original inputs better than the PCA using the covariance matrix.

The use of two layers LSTM RNNs did not improve the classification performance, as shown in Table 13. The results shown in this table, were obtained using Hiddnum = 100 for each layer, epochs = 200, batch size = 27 training examples and the time step = 7 h. Based on Yang Liu's study [18], this can be explained that the single layer LSTM RNNs can provide accurate prediction for big time interval prediction.

5.2.2. Classification using SVMs multi-class

In this section, we present and discuss the results of data classification using SVMs multi-class and feature extraction techniques, and we introduce the effect of selection of kernel function, namely, Gaussian RBF kernel and polynomial kernel. The selection of kernel function plays a major role in determining the performance of SVMs. The purpose of using the kernel function is to transform a data from input space to a higher dimensional feature space. Incorrect selection of parameters d, C, and γ can cause overfitting or underfitting problem. In this study, a standard grid search is used to determine the proper kernel parameters d, C and γ. For Gaussian RBF kernel we searched the range of parameters $C = \{2^{-3}, 2^{-2}, \ldots, 2^{7}\}$ and $\gamma = \{2^{-13}, 2^{-12}, \ldots, 2^{-3}\}$. For polynomial kernel we evaluated pairs of (d, C, γ) from the range $d = \{1, 2, 3\}$, $C = \{2^{-3}, 2^{-2}, \ldots, 2^{7}\}$ and $\gamma = \{2^{-13}, 2^{-12}, \ldots, 2^{-3}\}$. In order to estimate the performance of our models, we used the standard 10-fold CV (since SVM is a statistical learning technique. Thus, it assumes that the training examples are i.i.d., in this case, it would be good to use the standard *k*-fold CV). The results of this study can be shown in Tables 14–16. In these tables, we listed the kernel function, strategy of multiclass classification with parameters selection, classification rate for testing, training and testing time.

As seen in Table 14, the classification rate with PCA ranged from 95.17% to 99.25%. Whereas in Table 15, the classification rate with LDA ranged from 97.84% to 99.43%. While in Table 16, the classification rate with ICA ranged from 95.17% to 99.42%.

From these Tables we can observe:

Table 7
Classification using single layer LSTM RNNs with OOS.H method.

Feature extraction technique	Classification rate (%)	Training time (s)	Testing time (s)
PCA	98.06	288.63	0.19
LDA	99.17	173.45	0.11
ICA	99.17	212.45	0.16

Table 8
Classification using single layer LSTM RNNs with Random-Holdout method.

Feature extraction technique	Classification rate (%)	Training time (s)	Testing time (s)
PCA	99.44	253.42	0.22
LDA	99.72	137.78	0.08
ICA	99.72	170.58	0.13

Table 9
Classification using single layer LSTM RNNs with Multiple Train-Test splits method.

Feature extraction technique	Classification rate (%)	Training time (s)	Testing time (s)
PCA	97.60	164.20	0.09
LDA	97.80	114.36	0.08
ICA	97.70	142.52	0.08

Table 10
Classification using single layer LSTM RNNs with CV-Bl method.

Feature extraction technique	Classification rate (%)	Training time (s)	Testing time (s)
PCA	97.92	315.93	0.12
LDA	97.92	195.39	0.08
ICA	98.50	251.55	0.1

Table 11
Classification using single layer LSTM RNNs with 5-fold CV method.

Feature extraction technique	Classification rate (%)	Training time (s)	Testing time (s)
PCA	98.50	299.16	0.09
LDA	98.75	152.94	0.07
ICA	98.58	209.98	0.1

Table 12
Classification using single layer LSTM RNNs with 10-fold CV method.

Feature extraction technique	Classification rate (%)	Training time (s)	Testing time (s)
PCA	98.58	356.78	0.08
LDA	98.66	175.16	0.04
ICA	98.75	251.94	0.06

Table 13
Classification using two layers LSTM RNNs with Random-Holdout method.

Feature extraction technique	Classification rate (%)	Training time (s)	Testing time (s)
PCA	97.22	354.92	0.34
LDA	98.61	251.58	0.22
ICA	97.50	307	0.23

- The performance of polynomial kernel is better than Gaussian RBF kernel for both one-against-one and one-against-all strategies.
- When applying polynomial kernel: for one-against-one strategy the classification rate with LDA is highest 99.43% with the proper pair $(1, 2^{6}, 2^{-9})$, and ICA came second 99.42% with the proper pair $(1, 2^{6}, 2^{-10})$. The PCA had lowest classification rate 99.25% with the proper pair $(1, 2^{5}, 2^{-7})$. These results represent the best classification rates using SVM multi-class classification. For one-against-all strategy, the classification rate using ICA and PCA with the proper pairs $(1, 2^{7}, 2^{-11})$ and $(1, 2^{7}, 2^{-9})$ respectively is highest 98%. The LDA had

Table 14
Classification using SVM multi-class and PCA.

Kernel	Multi-class strategy	Classification rate (%)	Training time (s)	Testing time (s)
Polynomial (d, C, γ)	One vs. One $(1, 2^5, 2^{-7})$	99.25	0.6	0
	One vs. All $(1, 2^7, 2^{-9})$	98.00	2.58	0.03
Gaussian RBF (C, γ)	One vs. One $(2^2, 2^{-11})$	95.17	9.86	0
	One vs. All $(2^7, 2^{-6})$	95.17	27.66	0.18

Table 15
Classification using SVM multi-class and LDA.

Kernel	Multi-class strategy	Classification rate (%)	Training time (s)	Testing time (s)
Polynomial (d, C, γ)	One vs. One $(1, 2^6, 2^{-9})$	99.43	0.56	0
	One vs. All $(1, 2^7, 2^{-13})$	97.92	1.98	0.01
Gaussian RBF (C, γ)	One vs. One $(2^7, 2^{-11})$	97.92	2.69	0.01
	One vs. All $(2^5, 2^{-9})$	97.84	6.87	0.1

Table 16
Classification using SVM multi-class and ICA.

Kernel	Multi-class strategy	Classification rate (%)	Training time (s)	Testing time (s)
Polynomial (d, C, γ)	One vs. One $(1, 2^6, 2^{-10})$	99.42	0.55	0.01
	One vs. All $(1, 2^7, 2^{-11})$	98.00	2.11	0.01
Gaussian RBF (C, γ)	One vs. One $(2^6, 2^{-13})$	95.17	8.47	0
	One vs. All $(2^7, 2^{-8})$	95.42	22.4	0.17

lowest classification rate 97.92% with the proper pair $(1, 2^7, 2^{-13})$.

- When applying Gaussian RBF kernel: for one-against-one strategy, the classification rate with LDA is highest 97.92% with the proper pair $(2^7, 2^{-11})$. The classification rate using ICA and PCA with the proper pairs $(2^6, 2^{-13})$ and $(2^2, 2^{-11})$ respectively is lowest 95.17%. For one-against-all strategy, the classification rate with LDA is highest 97.84% with the proper pair $(2^5, 2^{-9})$, and ICA came second 95.42% with the proper pair $(2^7, 2^{-8})$. The PCA had lowest classification rate 95.17% with the proper pair $(2^7, 2^{-6})$.

By comparing the performance of both LSTM RNNs and SVMs multi-class in the classification, obviously, both LSTM RNNs and SVMs multi-class using polynomial kernel and one-against-one strategy performed well in data classification. However, LSTM RNNs' performance better than that of SVMs multi-class.

6. Conclusion and future work

In this paper, we have presented a performance evaluation of new deep learning technique - LSTM RNNs for water quality classification. An appropriate intelligent procedure based on the physicochemical parameters of surface water was proposed. The feature extraction step plays an important role in improving the classification process. In this study, PCA, LDA, and ICA techniques were successfully applied to extract useful and relevant features. However, the clustering feature using LDA and ICA was better than PCA does. After feature extraction, we performed variables selection process. The correlation between variables technique was used due to its reliability and simple.

LSTM was implemented with many methods of estimating the performance of the models. The results indicate that the Random-Holdout technique is a reliable and effective method for estimating the performance of time series prediction models. In this study, the SVM was selected as benchmark for comparison with LSTM model. The results showed that the integration of LSTM with LDA, and LSTM with ICA gave the best performance with 99.72% accuracy. According to this result, the integration of LSTM with LDA or ICA can serve as a promising alternative for intelligent and automated monitoring of water quality in the future.

The future research work we will (1) use soft sensors in the existence of the chemical parameters that cannot be measured continuously, (2) use nonlinear feature extraction techniques such as kernel principal component analysis (KPCA), kernel discriminant analysis (KDA) and kernel independent component analysis (KICA), (3) test the classification performance using new techniques of advanced deep learning such as Gated Recurrent Units (GRU) RNNs and convolutional LSTM (ConvLSTM).

CRediT author statement

Smail Dilmi: Conceptualization, Methodology, Software, Formal analysis, Investigation, Resources, Data Curation, Writing - Original Draft, Writing - Review & Editing, Visualization.

Mohamed Ladjal: Supervision, Project administration, Conceptualization, Data Curation, Validation, Writing - Review & Editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

We thank the editor and the anonymous reviewers for numerous suggestions and their helpful and constructive comments on an earlier draft of this article that significantly improved the quality of the paper. We would like to sincerely thank Tilesdit dam direction for providing the facilities for this investigation. We are also grateful to engineers of Tilesdit dam direction for providing the free access to databases and for their valuable guidance in the field sampling. This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.chemolab.2021.104329.

References

[1] UNESCO International Water Conference, Leveraging inter sectorality for sustainable water security and peace. https://en.unesco.org/events/unesco-international-water-conference-leveraging-intersectorality-sustainable-water-security, 2019. (Accessed 27 December 2019).
[2] M. Tripathi, S.K. Singal, Use of principal component analysis for parameter selection for development of a novel water quality index : a case study of river ganga India, Ecol. Indicat. 96 (2019) 430–436, https://doi.org/10.1016/j.ecolind.2018.09.025.
[3] A.K. Makarigakis, B.E. Jimenez-Cisneros, UNESCO's contribution to face global water challenges, Water 11 (2) (2019) 388, https://doi.org/10.3390/w11020388.
[4] P.J. Vikesland, Nanosensors for water quality monitoring, Nat. nanotech. 13 (2018) 651–660, https://doi.org/10.1038/s41565-018-0209-9.
[5] T. Ma, N. Zhao, Y. Ni, J. Yi, J.P. Wilson, L. He, Y. Du, T. Pei, C. Zhou, C. Song, W. Cheng, China's improving inland surface water quality since 2003, Sci. Adv. 6 (2020), eaau3798, https://doi.org/10.1126/sciadv.aau3798.

[6] S. Díaz, et al., Pervasive human-driven decline of life on Earth points to the need for transformative change, Science 366 (2019), eaax3100, https://doi.org/10.1126/science.aax3100.
[7] Tiyasha, T.M. Tung, Z.M. Yaseen, A survey on river water quality modelling using artificial intelligence models: 2000-2020, J. Hydrol. (2020), https://doi.org/10.1016/j.jhydrol.2020.124670.
[8] T. Rajaee, S. Khani, M. Ravansalar, Artificial intelligence-based single and hybrid models for prediction of water quality in rivers: a review, Chemometr. Intell. Lab. Syst. (2020), https://doi.org/10.1016/j.chemolab.2020.103978.
[9] N.S. Raghavendra, P.C. Deka, Support vector machine applications in the field of hydrology: a review, Appl. Soft Comput. J. (2014), https://doi.org/10.1016/j.asoc.2014.02.002.
[10] S. Kar, V.S. Rathore, P.K. Champati ray, R. Sharma, S.K. Swain, Classification of river water pollution using Hyperion data, J. Hydrol. 537 (2016) 221–233, https://doi.org/10.1016/j.jhydrol.2016.03.047.
[11] K.P. Singh, N. Basant, S. Gupt, Support vector machines in water quality management, Anal. Chim. Acta 703 (2011) 152–162, https://doi.org/10.1016/j.aca.2011.07.027.
[12] W. Li, M. Yang, Z. Liang, Y. Zhu, W. Mao, J. Shi, Y. Chen, Assessment for surface water quality in Lake Taihu Tiaoxi River Basin China based on support vector machine, Stoch. Environ. Res. Risk Assess. 27 (2013) 1861–1870, https://doi.org/10.1007/s00477-013-0720-3.
[13] Y. Liao, J. Xu, W. Wang, A method of water quality assessment based on biomonitoring and multiclass support vector machine, in: 2011 3rd International Conference on Environmental Science and Information Application Technology (ESIAT 2011), vol. 10, 2011, pp. 451–457, https://doi.org/10.1016/j.proenv.2011.09.074. Procedia Environ. Sci.
[14] A. Danades, D. Pratama, D. Anggraini, D. Anggriani, Comparison of accuracy level K-nearest neighbor algorithm and support vector machine algorithm in classification water quality status, in: 2016 IEEE 6th International Conference on System Engineering and Technology, ICSET, 2016.
[15] P. Liu, J. Wang, K.A. Sangaiah, Y. Xie, X. Yin, Analysis and prediction of water quality using LSTM deep neural networks in IoT environment, Sustainability 11 (7) (2019) 2058, https://doi.org/10.3390/su11072058.
[16] B. Jan, H. Farman, M. Khan, M. Imran, I. Ul Islam, A. Ahmad, S. Ali, G. Jeon, Deep learning in big data Analytics : a comparative study, Comput. Electr. Eng. 75 (2019) 275–287, https://doi.org/10.1016/j.compeleceng.2017.12.009.
[17] M. Mohammadi, A. Al-Fuqaha, S. Sorour, M. Guizani, Deep learning for IoT big data and streaming analytics: a survey, IEEE Commun. Surv. & Tutor. 20 (4) (2018) 2923–2960, https://doi.org/10.1109/COMST.2018.2844341.
[18] Y. Liu, Novel volatility forecasting using deep learning - long short term memory recurrent neural networks, Expert Syst. Appl. 132 (2019) 99–109, https://doi.org/10.1016/j.eswa.2019.04.038.
[19] L. Xie, M. Yin, X. Yin, Y. Liu, G. Yin, Low-rank sparse preserving projections for dimensionality reduction, IEEE Trans. Image Process. 27 (11) (2018) 5261–5274, https://doi.org/10.1109/TIP.2018.2855426.
[20] A. Subasi, Feature extraction and dimension reduction, in: Practical Guide for Biomedical Signals Analysis Using Machine Learning Techniques, Elsevier, 2019, pp. 193–275, https://doi.org/10.1016/B978-0-12-817444-9.00004-0.
[21] J. Tang, S. Alelyani, H. Liu, Feature selection for classification: a review, in: Data Classification : Algorithms and Applications, CRC Press, 2014, pp. 37–64, https://doi.org/10.1201/b17320.
[22] F. Pourpanaha, Y. Shib, C.P. Limc, Q. Haob, C.J. Tan, Feature selection based on brain storm optimization for data classification, Appl. Soft Comput. 80 (2019) 761–775, https://doi.org/10.1016/j.asoc.2019.04.037.
[23] A. Subasi, M.I. Gursoy, EEG signal classification using PCA, ICA, LDA and support vector machines, Expert Syst. Appl. 37 (2010) 8659–8666, https://doi.org/10.1016/j.eswa.2010.06.065.
[24] I. De Feis, Dimensionality reduction, Encycl. Bioinform. Comput. Biol. 1 (2019) 486–494, https://doi.org/10.1016/B978-0-12-809633-8.20336-1.
[25] W.L. Martinez, M. Cho, STATISTICS IN MATLAB® A PRIMER, CRC Press, 2015.
[26] W.L. Martinez, A.R. Martinez, J.L. Solka, Exploratory Data Analysis with MATLAB®, third ed., CRC Press, 2017.
[27] J. de Leeuw, Principal component analysis of binary data by iterated singular value decomposition, Comput. Stat. Data Anal. 50 (1) (2006) 21–39.
[28] A. Widodo, B.-S. Yang, T. Han, Combination of independent component analysis and support vector machines for intelligent faults diagnosis of induction motors, Expert Syst. Appl. 32 (2007) 299–312, https://doi.org/10.1016/j.eswa.2005.11.031.
[29] A. Widodo, E.Y. Kim, J.-D. Son, B.-S. Yang, A.C.C. Tan, D.-S. Gu, B.-K. Choi, J. Mathew, Fault diagnosis of low speed bearing based on relevance vector machine and support vector machine, Expert Syst. Appl. 36 (2009) 7252–7261, https://doi.org/10.1016/j.eswa.2008.09.033.
[30] L.J. Cao, K.S. Chua, W.K. Chong, H.P. Lee, Q.M. Gu, A comparison of PCA, KPCA and ICA for dimensionality reduction in support vector machine, Neurocomputing 55 (2003) 321–336.
[31] S. Balakrishnama, A. Ganapathiraju, Linear Discriminant Analysis-A Brief Tutorial, Institute for Signal and Information Processing, 1998, pp. 1–8.
[32] A. Tharwat, T. Gaber, A. Ibrahim, A.E. Hassanien, Linear discriminant analysis: a detailed tutorial, AI Commun. 30 (2017) 169–190.
[33] J. Shlens, A Tutorial on Independent Component Analysis, 2014 arXiv:1404.2986.
[34] W. Windig, M.R. Keenan, I.C.A. Homeopathic, A simple approach to expand the use of independent component analysis (ICA), Chemometr. Intell. Lab. Syst. 142 (2015) 54–63, https://doi.org/10.1016/j.chemolab.2015.01.003.
[35] Y.B. Monakhova, R. Godelmann, T. Kuballa, S.P. Mushtakova, D.N. Rutledge, Independent components analysis to increase efficiency of discriminant analysis methods (FDA and LDA): application to NMR fingerprinting of wine, Talanta 141 (2015) 60–65, https://doi.org/10.1016/j.talanta.2015.03.037.
[36] A. Hyvärinen, J. Karhunen, E. Oja, Independent Component Analysis, Wiley, 2001.
[37] A. Tharwat, Independent component analysis: an introduction, Appl. Comput. and Inform. (2018), https://doi.org/10.1016/j.aci.2018.08.006.
[38] J.V. Stone, Independent Component Analysis: A Tutorial Introduction, A bradford book, 2004.
[39] A. Hyvärinen, Fast and robust fixed-point algorithms for independent component analysis, IEEE Trans. Neural Network. 10 (1999) 626–634.
[40] M. Vinther, Independent Component Analysis of Evoked Potentials in EEG, Ørsted, DTU, 2002.
[41] R. Aziz, C.K. Verma, N. Srivastava, A fuzzy based feature selection from independent component subspace for machine learning classification of microarray data, Genom. Data. 8 (2016) 4–15.
[42] A. Hyvärinen, New approximations of differential entropy for independent component analysis and projection pursuit, Adv. Neural Inf. Process. Syst. 10 (1998) 273–279.
[43] W.L. Martinez, A.R. Martinez, Computational Statistics Handbook with MATLAB, third ed., CRC Press, 2016.
[44] Y. Wang, L. Feng, A new hybrid feature selection based on multi-filter weights and multi-feature weights, Appl. Intell. 49 (2019) 4033–4057, https://doi.org/10.1007/s10489-019-01470-z.
[45] Y. Mei, S. Nguyen, B. Xue, M. Zhang, An efficient feature selection algorithm for evolving job shop scheduling rules with genetic programming, IEEE Trans. Emerg. Top. Comput. Intell. 1 (5) (2017) 339–353.
[46] R. Noori, R. Berndtsson, M. Hosseinzadeh, J.F. Adamowski, M.R. Abyaneh, A critical review on the application of the national sanitation foundation water quality index, Environ. Pollut. 244 (2019) 575–587.
[47] E.M. Smeti, S.K. Golfinopoulos, Characterization of the quality of a surface water resource by multivariate statistical analysis, Anal. Lett. 49 (7) (2016) 1032–1039, https://doi.org/10.1080/00032719.2015.1045585.
[48] N. Mazlum, A. Ozer, S. Mazlum, Interpretation of water quality data by principal components analysis, Turk. J. Eng. Environ. Sci. 23 (1999) 19–26.
[49] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (2015) 436–444, https://doi.org/10.1038/nature14539.
[50] J. Schmidhuber, Deep learning in neural networks: an overview, Neural Network. 61 (2015) 85–117.
[51] E. Kanjo, E.M.G. Younis, C.S. Ang, Deep learning analysis of mobile physiological, environmental and location sensor data for emotion detection, Inf. Fusion 49 (2019) 46–56.
[52] RNN (Recurrent Neural Network) Tutorial, TensorFlow example. https://www.guru99.com/rnn-tutorial.html#2. (Accessed 25 January 2020).
[53] R. Jozefowicz, W. Zaremba, I. Sutskever, An empirical exploration of recurrent network architectures, in: Proceedings of the Thirty-Second International Conference on Machine Learning (ICML-15), 2015.
[54] H. Sepp, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[55] G. Swapna, K.P. Soman, R. Vinayakumar, Automated detection of cardiac arrhythmia using deep learning techniques. International Conference on Computational Intelligence and Data Science (ICCIDS 2018), Procedia Comput. Sci. 132 (2018) 1192–1201.
[56] K. Greff, R.K. Srivastava, J. Koutnk, B.R. Steunebrink, J. Schmidhuber, LSTM : a search space odyssey, IEEE Trans. Neural Netw. Learn. Syst. 28 (10) (2017) 2222–2232.
[57] V.N. Vapnik, The Nature of Statistical Learning Theory, second ed., Springer, New York, 2000.
[58] B. Schölkopf, A. Smola, Learning with Kernels, Support Vector Machines, Regularization, Optimization and beyond, MIT Press, Cambridge MA, 2002.
[59] M. Ladjal, M. Bouamar, M. Djerioui, Y. Brik, Performance evaluation of ANN and SVM multiclass models for intelligent water quality classification using Dempster-Shafer Theory, in: 2nd International Conference on Electrical and Information Technologies (ICEIT'2016), 2016.
[60] M.H. Bae, T. Wu, R. Pan, Mix-ratio sampling: classifying multiclass imbalanced mouse brain images using support vector machine, Expert Syst. Appl. 37 (2010) 4955–4965.
[61] M.-H. Horng, Multi-class support vector machine for classification of the ultrasonic images of supraspinatus, Expert Syst. Appl. 36 (2009) 8124–8133.
[62] A. Ebrhimzadeh, H. Azimi, H.M. Naeemi, Classification of communication signals using an optimized classifier and efficient features, Arabian J. Sci. Eng. 35 (1B) (2009) 225–235.
[63] C.J.C. Burges, A tutorial on support vector machines for pattern recognition, Data Min. Knowl. Discov. 2 (1998) 121–167.
[64] C. Junli, J. Licheng, Classification mechanism of support vector machines, WCC 2000 - ICSP 2000, in: 2000 5th International Conference on Signal Processing Proceedings. 16th World Computer Congress, 2000, https://doi.org/10.1109/ICOSP.2000.893396.
[65] J.H. Min, Y.-C. Lee, Bankruptcy prediction using support vector machine with optimal choice of kernel function parameters, Expert Syst. Appl. 28 (2005) 603–614, https://doi.org/10.1016/j.eswa.2004.12.008.
[66] J. Lutsa, F. Ojeda, R.V. de Plas, B.D. Moor, S.V. Huffel, J.A.K. Suykens, A tutorial on support vector machine-based methods for classification problems in chemometrics, Anal. Chim. Acta 665 (2010) 129–145, https://doi.org/10.1016/j.aca.2010.03.030.
[67] C.-W. Hsu, C.-J. Lin, A comparison of methods for multi-class support vector machines, IEEE Trans. Neural Network. 13 (2) (2002) 415–425.
[68] V.N. Vapnik, Statistical Learning Theory, Edition Wiley, 1998.

[69] X. He, W. Wang, X. Liu, Y. Ji, Risk assessment of communication network of power company based on rough set theory and multi-class SVM, International Conference on Applied Physics and Industrial Engineering, Phys. Procedia. 24 (2012) 1226–1231.
[70] S. Canu, Y. Grandvalet, A. Rakotomamonjy, SVM and Kernel Methods MATLAB Toolbox, Perception Systèmes et Information, INSA de Rouen, Rouen, France, 2003. http://asi.insarouen.fr/~arakotom/toolbox/index.
[71] Y. Wang, J. Zhou, K. Chen, Y. Wang, L. Liu, Water quality prediction method based on LSTM neural network, in: 2017 12th International Conference on Intelligent Systems and Knowledge Engineering, ISKE, 2018.
[72] V. Cerqueira, L. Torgo, J. Smailović, I. Mozetič, A comparative study of performance estimation methods for time series forecasting, in: 2017 IEEE International Conference on Data Science and Advanced Analytics, DSAA, 2018.
[73] C. Bergmeir, J.M. Benítez, On the use of cross-validation for time series predictor evaluation, Inf. Sci. 191 (2012) 192–213.
[74] C. Bergmeir, M. Costantini, J.M. Benítez, On the usefulness of cross-validation for directional forecast evaluation, Comput. Stat. Data Anal. (2014).
[75] C. Bergmeir, R.J. Hyndman, B. Koo, A note on the validity of cross-validation for evaluating autoregressive time series prediction, Comput. Stat. Data Anal. (2017).
[76] V. Cerqueira, L. Torgo, I. Mozetič, Evaluating Time Series Forecasting Models : an Empirical Study on Performance Estimation Methods, 2019 arXiv:1905.11744vol. 1.
[77] S. Bouktif, A. Fiaz, A. Ouni, M.A. Serhani, Optimal deep learning LSTM model for electric load forecasting using feature selection and genetic algorithm: comparison with machine learning approaches, Energies 11 (7) (2018) 1636, https://doi.org/10.3390/en11071636.