A Machine Hearing Framework for Real-Time Streaming Analytics Using Lambda Architecture

Konstantinos Demertzis[1(✉)], Lazaros Iliadis[1], and Vardis-Dimitris Anezakis[2]

[1] School of Engineering, Department of Civil Engineering, Faculty of Mathematics Programming and General Courses, Democritus University of Thrace, Kimmeria, Xanthi, Greece
kdemertz@fmenr.duth.gr, liliadis@civil.duth.gr

[2] School of Agriculture and Forestry, Department of Forestry and Management of the Environment and Natural Resources, Democritus University of Thrace, Orestiada, Greece
danezaki@fmenr.duth.gr

Abstract. Disruptions to the earth's biosphere and to the natural environment stemming from the indiscreet human activity, have caused serious environmental problems which are tantamount to an extended and prolonged ecological crisis. Climate change is clearly reflected in the increase of the global average air and ocean temperatures, in the excessive melting of snow-ice, and in the rise of the global average sea level. One of the most serious impacts of climate change is the complex interaction of species in relation to their corresponding climatic survival factors, which favors the spread of invasive species (INSP). These species constitute a very serious and rapidly deteriorating threat to the natural biodiversity of the native environment, but also to the flora, fauna, and even to the local human population. This research proposes a Machine Hearing (MH) framework for real-time streaming analytics, employing Lambda Architecture (LARC). The hybrid modeling effort is based on timely and advanced Computational Intelligence (COIN) approaches. The Framework for Lambda Architecture Machine Hearing (FLAME_H) uses a combination of batch and streaming data. The FLAME_H applies the EL_GROSEMMARI (Extreme Learning Graph Regularized Online Sequential Multilayer Multiencoder Algorithm) to classify the batch data and the Adaptive Random Forest (ARF) in order to control the data streams in real time. The aim of the proposed framework is the intelligent identification and classification of invasive alien species, based on the sounds they produce. This would contribute to the protection of biodiversity and biosecurity in a certain area.

Keywords: Invasive species · Machine Hearing · Lambda Architecture · Adaptive random forests · Deep learning · Extreme Learning Machine · Streaming data

J. Macintyre et al. (Eds.): EANN 2019, CCIS 1000, pp. 246–261, 2019.
https://doi.org/10.1007/978-3-030-20257-6_21

1 Introduction

Endemism can be sustained by natural hurdles such as rivers, oceans, mountains, deserts and climatic conditions [1]. Due to increasing and continuous climate change, coupled with globalization and the development of international trade and tourism, these natural barriers are becoming increasingly inefficient. As a result, species, particularly marine ones, are able to travel long distances to other biotopes where they become alien or, in many cases, even expansive species [2]. The process of recognizing INSP is a critical step for the adoption and implementation of a specific policy to tackle, eradicate, control and/or contain these species. Given that these species are usually unknown in their new environment, it is extremely difficult, complex and dangerous to identify and securely isolate them. It should be emphasized that neither the large differences in morphology nor the significant similarities reflect the affinity of biological organisms [3]. The need for thorough and fully valid identification of these species is very serious in the case of planning their response programs, as the recognition process depends on a multitude of required information and on a continuous monitoring of the current situation. Searching for novel methods of resolving or analyzing phenomena related to the potential impacts of climate change, such as methods of identifying invasive species, are key research priorities of high importance.

On the other hand, Machine Hearing (MAHE) is a scientific branch of artificial intelligence that attempts to reproduce the sense of hearing algorithmically [4]. It is related (in theoretical and practical level) to the design and development of data analysis systems. MAHE data are obtained from digital sound recorders or by appropriate sensors. Audio signal analysis is related to knowledge mining and it aims in the classification, segmentation, or automated retrieval [5]. In general, the process initially involves the extraction of certain features which must be able to differentiate their values according to the content and structure of the respective signals. After the determination of the audio features that characterize the sound signal, a pattern recognition approach is employed [6]. The algorithmic resolution of a MAHE classification problem requires a high availability of resources. In this case, we have to examine the temporal complexity of the algorithm, the memory availability as a function of its input data, as well as individual analysis should be performed related to other resources as appropriate (e.g. how many parallel processors are needed for a parallel solution of the problem). This is a *Big Data* (BDA) problem, as data extracted from audio clips, require big storage space for their clear computer comprehension.

The need for Big Data management and analysis such as the MAHE problems, has re-established the prototyping architectures of BDA [7]. Lambda architecture (LAR) is the most important one. It has been designed to handle massive amounts of data using the batch and streaming processing methods. This approach attempts to balance latency, throughput, and fault tolerance using the batch process to provide complete and accurate views of historical data. At the same time, it uses real time data stream processing to provide views of new inputs [8]. The two projection outputs can be joined before the final data presentation or the final decision.

2 The Proposed FLAME_H

This paper proposes the development of the FLAME_H (Framework of Lambda Architecture for Machine Hearing) which performs real-time streaming analytics. It is an advanced hybrid computational intelligence approach. The FLAME_H, employs an innovative version of the Lambda architecture, which optimally combines batch and streaming data, to safely perform classification. FLAME_H is a MAHE system, that performs real-time audio streaming analysis and classifies audio data sets to identify patterns. At the same time, it adopts different biosecurity policies for each resulting category. It is a very important method of locating invasive species, and an innovative approach of recording biofouling. At the same time, it can be considered a key tool in security policy mechanisms as it allows for safe and cost-effective assessment of their behavior and disclosure of the damage caused by their activities.

Batch data (BADA) is usually a set of data that is collected during some processes for a specific time period and it characterizes and identifies species. Their processing by conventional data mining (DAM) or Machine Learning (ML) methods, considers that they are available and can be accessed simultaneously without any limitation in terms of their processing or analysis time. It should be noted that this data is susceptible to noise, their classification process has a significant cost, and they require serious hardware infrastructure for safe storage and general handling. On the contrary, due to their reliance on strict time constraints and their more general availability, they are selected for detailed and specialized data processing techniques that can lead to multiple levels of revelation of the hidden knowledge they may contain. The growing field of real-world applications produces streaming data (SDA) at an increasing rate, requiring large-scale and real-time processing. SDA such as audio data analysis and data generated in dynamic environments, leads to one of the most robust research areas of DAM. It is the ML application on data streams for pattern recognition under dynamic displacement and feedback environments. In general, FLAME_H is an intelligent hybrid ML system, that employs a special version of the LARC architecture and the Deep Learning EL_GROSEMMARI algorithm (for the batch data classification) combined with the ARF approach in order to control data streams in real time.

Figure 1 presents an overall description of the algorithm.

In the first phase of the algorithm, the appropriate features are derived from the *Sea Audio* data stream (*audio feature extraction*). The data are then provided as input and they are controlled in parallel by the two learning algorithms. This is done aiming to minimize the likelihood of misleading and to achieve high reliability classification. The decision process merges the results of the two algorithms offering advantage to the ones obtained by the EL_GROSEMMARI. The decision concerns whether it is a "sound" coming from an invasive species fish. If the sound is described as noise that comes from a usually human sea-related process, then it is rejected and there is no further development in the process. If the sound comes from a species of fish or mammal and once this species is identified, the coordinates are taken from the GPS and assigned to the country where they belong. Then a check is made on whether the species identified is native to this country, otherwise it is recorded as an invasive species. Listings with indigenous and invasive species were extracted from the Invasive Species Compendium

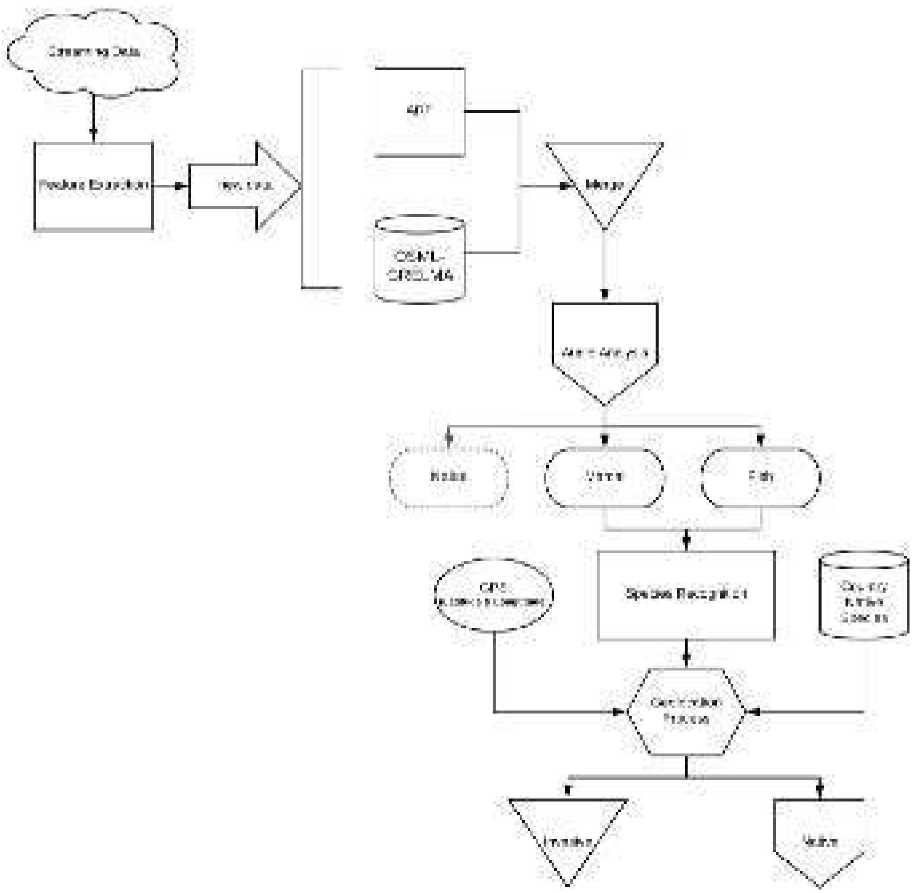

Fig. 1. Structure of the FLAME_H

(http://www.cabi.org/isc/) [9], the most valid and comprehensive database on the issue, world-wide. The Geolocation process is presented below:

Algorithm 1. Geolocation Process

```
Input:
   Recognized_Species;
   Country;
   Country_Native_Species;
Read Recognized_Species, Country, Country_Native_Species;
     for i=1 to Country_Native_Species [max] do
          if Country_Native_Species [i]= Recognized_Species then
             Recognized Species=Native_Species
          else
             Recognized Species=Invasive_Species
          end if
     end
Output:
   Species Identity;
```

Lambda architecture was employed, as in multifactorial problems of high complexity of large data sets such as the one under consideration, the results of the forecast are multi-variable, especially with respect to analysis and integration of data flows.v. This architecture employs a serious *Reactive* strategy to deal with invasive species. The combination of two different algorithms facilitates the sorting process, making each classifier more robust, and it accelerates the convergence of the generic multiple model, which is less noisy than any single one [10]. Thus, this approach offers generalization and avoids overfitting which is one of the basic targets in Machine Learning.

3 Literature Review

Invasive alien species are a result of generalized climate change and they constitute a serious and rapidly worsening threat to natural biodiversity in Europe. European Union spends at least 12 billion Euros per year on control of IAS and disasters they cause. Also, the risk to public health should not be overlooked as these species may be toxic, such as "*Lagocefalus*" fish, which contains "*Tetrodotoxin*" a very dangerous substance, capable of causing serious health problems, even death in the consumer [11, 12]. The significance of the hybrid innovative intelligent approaches (Machine Learning Algorithms) for identifying IAS and their separation from indigenous ones has been developed by recent researches [13, 14]. Soft computing techniques are capable to model and detect cyber security threats [15, 16] and they also offer optimization mechanisms in order to produce reliable results.

Hinton et al. [17] had proposed methods and applications of DL. Through a series of new learning architectures and algorithms, domains such as object recognition [18] and machine translation [19, 20] have been transformed; deep learning methods are now the state-of-the-art in object, speech and audio recognition. In particular, deep learning has been the driving force behind large leaps in accuracy and model robustness in audio related domains like audio sensing [21]. Alom et al. [22] applied the Cellular Simultaneous Recurrent Networks (CSRNs) to generate initial filters of CNNs for features extraction and Regularized Extreme Learning Machines (RELM) for classification. Experiments were conducted on three popular datasets for object recognition (such as face, pedestrian, and car) to evaluate the performance of the proposed system. Zhang, et al. [23], proposed an object recognition algorithm which did not depend on human experts to design features for fish species classification, but constructed efficient features automatically. Results from experiments showed that the proposed method obtained an average of 98.9% classification accuracy with a standard deviation of 0.96% with a dataset composed of 8 fish species and a total of 1049 images. Also, DL has been the driving force behind large leaps in accuracy and model robustness in audio related domains like speech recognition. Moreover Han et al. [24] proposed to utilize DNNs to extract high level features from raw data and show that they are effective for speech emotion recognition. Finally, Zhao et al. [25] proposed a new method for automated field recording analysis with improved automated segmentation and robust bird species classification by a Gaussian Mixture Model.

4 Algorithms

4.1 Extreme Learning Machines for Batch Data Algorithms

An ELM is a Single-Hidden Layer Feed Forward Neural Network (SLFFNN) [26] with N hidden neurons, randomly selected input weights and random values of bias in the hidden layer, while the weights at its output are calculated with a single multiplication of vector matrix [27]. For an ELM using SLFFNN and random representation of hidden neurons, input data is mapped to a random L-dimensional space with a discrete training set N, where $(x_i, t_i), i \in [\![1, N]\!]$ with $x_i \in R^d$ and $t_i \in R^c$. The specification output of the network is the following:

$$f_L(x) = \sum\nolimits_{i=1}^{L} \beta_i h_i(x) = h(x)\beta \quad i \in [\![1, N]\!] \tag{1}$$

Vector matrix $\beta = [\beta_1, \ldots, \beta_L]^T$ is the output of the weight vector matrix connecting hidden and output nodes. On the other hand, $h(x) = [g_1(x), \ldots, g_L(x)]$ is the output of the hidden nodes for input x, and $g_1(x)$ is the output of the *ith* neuron. Based on a training set $\{(x_i, t_i)\}_{i=1}^{N}$, an ELM can solve the Learning Problem $\mathrm{H}\beta = T$, where $T = [t_1, \ldots, t_N]^T$ are the target labels and the output vector matrix of the Hidden Layer H is the following:

$$\mathrm{H}(\omega_j, b_j, x_i) = \begin{bmatrix} g(\omega_1 x_1 + b_1) & \cdots & g(\omega_l x_1 + b_l) \\ \vdots & \ddots & \vdots \\ g(\omega_1 x_N + b_1) & \cdots & g(\omega_l x_N + b_l) \end{bmatrix}_{N \times l} \tag{2}$$

The input weight vector matrix of the hidden layer ω (before training) and the bias vectors b are created randomly in the interval [−1, 1], with

$$\omega_j = [\omega_{j1}, \omega_{j2}, \ldots, \omega_{jm}]^{\mathrm{T}} \text{ and } \beta_j = [\beta_{j1}, \beta_{j2}, \ldots, \beta_{jm}]^{\mathrm{T}} \tag{3}$$

The output weight vector matrix of the hidden layer H is calculated by the use of the Activation function in the training dataset, based on the following function:

$$H = g(\omega x + b) \tag{4}$$

The output weights β can be estimated by using function:

$$\beta = \left(\frac{\mathrm{I}}{C} + H^T H\right)^{-1} H^T X \tag{5}$$

where $H = [h_1, \ldots, h_N]$ is the output vector matrix of the hidden layer and $X = [x_1, \ldots, x_N]$ the input vector matrix of the hidden layer. Indeed, β can be calculated by the following general relation:

$$\beta = H^{+}T \tag{6}$$

where H^{+} is the generalized inverse vector matrix Moore-Penrose for matrix H. This approach is employing ELM with Gaussian Radial Basis Function kernel $K(u, v) = exp(-\gamma||u - v||^2)$. The size k of the hidden layer are 20 neurons. Subsequently assigned random input weights w_i and biases b_i, i = 1, …, N. To calculate the hidden layer output matrix H we have used the function (7):

$$H = \begin{bmatrix} h(x_1) \\ \vdots \\ h(x_N) \end{bmatrix} = \begin{bmatrix} h_1(x_1) & \cdots & h_L(x_1) \\ \vdots & & \vdots \\ h_1(x_N) & \cdots & h_L(x_N) \end{bmatrix} \tag{7}$$

Where $h(x) = [h_1(x), \ldots, h_L(x)]$ is the output (row) vector of the hidden layer with respect to the input x. $h(x)$ actually maps the data from the D-dimensional input space to the L-dimensional hidden-layer feature space (ELM feature space) H. Thus, $h(x)$ is indeed a feature mapping. ELM is to minimize the training error as well as the norm of the output weights:

$$\text{Minimize: } ||\mathrm{H}\beta - \mathrm{T}||^2 \text{ and } ||\beta|| \tag{8}$$

where H is the hidden-layer output matrix of the function (7).

Minimization of the norm of the output weights $||\beta||$ $||\beta||$ is actually achieved by maximizing the distance of the separating margins of the two different classes in the ELM feature space $2/||\beta||$. To calculate the output weights β we used function (9):

$$\beta = (\frac{\mathrm{I}}{\mathrm{C}} + \mathrm{H}^{\mathrm{T}}\mathrm{H})^{-1}\mathrm{H}^{\mathrm{T}}\mathrm{T} \tag{9}$$

where the value of C (a positive constant) and the value of T are obtained from the *Function Approximation of SLFFNs* with additive neurons:

$$\mathrm{t_i} = [\mathrm{t_{i1}}, \mathrm{t_{i2}}, \ldots, \mathrm{t_{im}}]^{\mathrm{T}} \in \mathrm{R^m}, \mathrm{T} = \begin{bmatrix} \mathrm{t_1^T} \\ \vdots \\ \mathrm{t_N^T} \end{bmatrix} \tag{10}$$

Considering and combining the features of ELM presented above, we introduce and propose a new Deep architecture by creating an Online Learning Multilayer Graph Regularized Extreme Learning Machine Auto-Encoder (OSML-GRELMA). This is a multi-layered neural network model that receives successive OL data streams and uses the unsupervised GRELMA algorithm as a basic building block in which the output of each level is used as inputs to the next one [28].

An autoencoder is an artificial neural network used for unsupervised learning of efficient coding. The aim of an autoencoder is to learn a representation (encoding) for a set of data, but with the output layer having the same number of nodes as the input layer, and with the purpose of reconstructing its own inputs (instead of predicting target value Y given inputs X). The Algorithm 2 is described below [28]:

Algorithm 2. GRELMA Algorithm for Clustering [28]

Input: Data{X}= $\{xi\}_{i=1}^{N}$ the number of hidden neurons n_h, the penalty coefficient κ and λ
Output: The cluster results.
Step 1: Initialize an ELM of n_h hidden neurons with random input weights and biases.
Step 2: If $n_h \leq N$ Compute the output weights β by equation $\beta^* = (I_{nh} + H^T CH + \lambda H^T LH)^{-1} H^T CX$
Else Compute the output weights β by equation $\beta^* = H^T(I_N + CHH^T + \lambda LHH^T)^{-1} CX$
Step 3: $X_{new} = X\beta^T$
Step 4: Treat each row of X_{new} as a point and cluster the N points into K clusters using the k-means algorithm.

The overall function of the OSML-GRELMA is presented in the following algorithm.

Algorithm 3. OSML-GRELMA Algorithm for Classification

Input: A small initial training set $N=\{(x_i, t_i)|x_i \in R^n, t_i \in R^m, i=1, \cdots, \widetilde{N}\}$
The model depth: m;
The number of hidden nodes in the each GRELMA: $n_{h1}, n_{h2}, \ldots, n_{hm}$;
The new activation function: h_{new}.
Output: The classification results of the M data.
Phase 1 (BPh)
Initialize $X_l = X_{train}$
For i =1: m
Assign arbitrary input weight w_i and bias b_i i=1, of the i_{th} layer GRELMA by some random numbers;
Calculate the initial hidden layer output matrix $H_0 = [h_1, \cdots, h_{\widetilde{N}}]^T$, where $h_i = [h_{new}(w_1 \cdot x_i + b_1), \cdots, h_{new}(w_{\widetilde{N}} \cdot x_i + b_{\widetilde{N}})]^T$, $i = 1, \cdots, \widetilde{N}$, where h_{new} activation function.
Train the output weights β_i of the i_{th} layer GRELMA;
Estimate the initial output weight $\beta^{(0)} = M_0 H_0^T T_0$, where $M_0 = (H_0^T H_0)^{-1}$ and $T_0 = [t_1, \ldots, t_{\widetilde{N}}]^T$.
Set $k = 0$.
Compute the outputs $X_i+1 = h_{new}(X_i\beta_i^T)$.
Phase 2 (SLPh)
The essentials step of this phase for each further coming observation (x_i, t_1), where $x_i \in R^n$, $t_i \in R^m$ and $i = \widetilde{N} + 1, \widetilde{N} + 2, \widetilde{N} + 3$, described as follow:
Calculate the hidden layer output vector $h_{(k+1)} = [h_{new}(w_1 \cdot x_i + b_1), \cdots, h_{new}(w_{\widetilde{N}} \cdot x_i + b_{\widetilde{N}})]^T$
Calculate latest output weight $\beta^{(k+1)}$ by the algorithm $\hat{\beta} = (H^T H)^{-1} H^T T$ which is called the Recursive Least-Squares (RLS) algorithm.
Set $k = k + 1$
End For
Map X_{m+1}, the output of the m_{th} layer, to the output layer.
Compute the classification results by using the above trained OSML-GRELMA model.

The main objective and training success of the proposed OSML-GRELMA approach is based on evolutionary identification of the underlying structure of the input data flows to produce the final model. It basically uses the knowledge of labelled data to investigate the distribution of the input data, aiming at enhancing the outcome of the learning process using an adaptive scheme. In this sense, it includes procedures that approach unsupervised learning, where inputs come from the same marginal distribution or follow a common cluster structure.

4.2 Adaptive Random Forests for Streaming Data

As it can be seen, data flow management and especially knowledge extraction with ML algorithms from these flows are unlikely to be performed by applying iterations over input data. Accordingly, adapting the Random Forest algorithm [29] to streaming data, depends on a suitable accumulation process that is partially achieved by a bootstrap method and at the same time by limiting each decision to divide the sheets into a subset of attributes. This is achieved with the modification of the base tree algorithm, by effectively reducing the set of features examined for further separation into random subsets of magnitude m, where $m < M$ (M corresponds to the total number of attributes that are examined in each case. In the non-streaming bagging, each of the n-base models is trained in a Z-sized bootstrap sample created by random samples with replacement from the original training kit. Every bootstrapped sample contains an original training snapshot K, where $P\ (K = k)$ follows a binomial distribution. For large values of Z this binomial distribution is attached to a Poisson one, with $\lambda = 1$. On the other hand, according to the ARF approach for streaming data, a Poisson distribution is used with $\lambda = 6$. This "feedback" has the practical effect of increasing the probability of assigning higher weights to instances during the training of the basic models [29].

ARF is an adaptation of the original Random Forest algorithm, which has been successfully applied to a multitude of machine learning tasks. In layman's terms the original Random Forest algorithm is an ensemble of decision trees, which are trained using bagging and where the node splits are limited to a random subset of the original set of features. The "Adaptive" part of ARF comes from its mechanisms to adapt to different kinds of concept drifts, given the same hyper-parameters. Specifically, the 3 most important aspects of the ARF algorithm are it adds diversity through resampling ("bagging"); it adds diversity through randomly selecting subsets of features for node splits and it has one drift and warning detector per base tree, which cause selective resets in response to drifts. It also allows training background trees, which start training if a warning is detected and replace the active tree if the warning escalates to a drift. ARF was designed to be "embarrassingly" parallel, in other words, there are no dependencies between trees. The overall ARF pseudo-code is presented below in Algorithm 4 [29].

Algorithm 4. Adaptive Random Forests

```
function ARF (m, n, δw, δd)
 T ← CreateTrees(n)
 W ← InitWeits(n)
 B ← Ø
 while HasNext(S) do
   (x, y) ← next(S)
   for all t ∈ T do
        ỹ ← predict(t, x)
        W(t) ← P(W(t), ỹ, y)
        RFTreeTrain (m, t, x, y)
        if C (δw, t, x, y) then
             b ← CreateTrees()
             B(t) ← b
        end if
   end for
   for all b ∈ B do
        RFTreeTrain (m, b, x, y)
   end for
 end while
end function
```

Where m: maximum features evaluated per split; n: total number of trees ($n = |T|$); δ_w: warning threshold; δ_d: drift threshold; c(·): change detection method; S: Data stream; B: Set of background trees; W(t): Tree t weight; P(·): Learning performance estimation function [29].

5 Datasets

The following four main categories of sounds have been determined in order to create highly complex scenarios that can potentially include the most likely cases that can be detected in an underwater space:

- Fishes: Several species of fish produce sounds with various mechanisms such as teeth, pharynx, fins, and shuttle bladder. 1076 sounds belonging to 10 fish species have been included in this category (e.g. Bidyanus Bidyanus, Epinephelus Adscensionis, Cynoscion Regalis, Carassius Auratus, Cyprinus Carpio, Rutilus Rutilus, Salmo Trutta, Oreochromis Mossambicus, Micropterus Salmoides, Oncorhynchus Mykiss).
- Mammals: Marine mammals produce and use sounds to orientate and communicate with each other. Totally 836 sounds belonging to 8 species of mammals are included in this category. (e.g. Delphinus Delphis, Erignathus Barbatus, Balaena Mysticetus, Phocoena Phocoena, Neophocaena Phocaenoides, Trichechus, Tursiops Truncates, Phoca Hispida).
- Anthropogenic Sounds: It comprises of 684 sounds belonging to 9 classes (Ship, Sonar, Zodiac, Torpedo, Wind Turbine, Scuba Noise, Bubble Curtain, Personal Water Craft, Airgun).
- Natural Sounds: Totally 477 sounds belong here classified in six clusters (Earthquake, Hydrothermal Vents, Ice Cracking, Rainfall, Lightning, Waves).

The Feature Extraction process [30] enables capturing of characteristics that precisely determine the uniqueness of each sound and helps distinguish between acoustic categories. The categories distinction is based on 34 characteristics related to statistical measurements obtained from the signal frequency information. In this research effort we have extracted the short-term feature sequences for an audio signal, using a frame size of 50 ms and a frame step of 25 ms (50% overlap). All sounds had a sampling rate of 44.1 kHz, 16-bit stereo resolution while their average duration was 10.3 s.

6 Results

In data batch cases using multiple classifiers, for estimating the real error during training, the full probability density of both categories should be known [31, 32]. The classification performance is estimated by the *Total Accuracy* (TA), *Root Mean Squared Error* (RMSE), *Precision* (PRE), *Recall* (REC), *F-Score* and *ROC Area* indices [33, 34]. The 10-fold cross validation is employed in this stage in order to obtain performance indices. Analytical values of the predictive capacity of the algorithm are presented in the following Tables 1, 2, 3, 4, 5 and 6.

In the case of stream data classification, we need to compare classification performance in terms of *Accuracy Kappa* statistic and *Kappa-Temporal* statistic. This is done by using the traditional immediate setting. The true label is presented right after the instance used for testing or the delayed setting (where there is a real delay between the moment an instance is presented and the moment its true label becomes available) [33].

Table 1. Performance metrics of Categories_Dataset

Classifier	Classification accuracy & performance metrics					
	TA	RMSE	PRE	REC	F-Score	ROC Area
OSML-GRELMA	96.08%	0.1376	0.960	0.960	0.960	0.970

Table 2. Confusion matrix of Categories_Dataset

Fishes	Mammals	Anthr_Sounds	Natural_Sounds	
1042	13	12	9	Fishes
14	**797**	17	8	Mammals
5	7	**659**	13	Anthr_Sounds
4	6	10	**457**	Natural_Sounds

Table 3. Performance metrics of Mammals_Dataset

Classifier	Classification accuracy & performance metrics					
	TA	RMSE	PRE	REC	F-Score	ROC Area
OSML-GRELMA	92.18%	0.1571	0.922	0.922	0.922	0.955

Table 4. Confusion matrix of Mammals_Dataset

a	b	c	d	e	f	g	h	
142	2	0	1	1	0	1	0	a = Delphinus Delphis
1	**101**	3	0	0	5	0	4	b = Erignathus Barbatus
1	2	**122**	1	0	0	0	2	c = Balaena Mysticetus
1	1	2	**61**	1	1	0	3	d = Phocoena Phocoena
2	2	3	1	**51**	0	2	2	e = Neophocaena Phocaenoides
2	0	2	0	1	**82**	0	2	f = Trichechus
2	0	0	0	0	1	**51**	1	g = Tursiops Truncates
2	2	1	1	1	0	1	**162**	h = Phoca Hispida

Table 5. Performance metrics of Fishes_Dataset

Classifier	Classification accuracy & performance metrics					
	TA	RMSE	PRE	REC	F-Score	ROC Area
OSML-GRELMA	87.91%	0.1512	0.879	0.879	0.879	0.920

Table 6. Confusion matrix of Fishes_Dataset

a	b	c	d	e	f	g	h	i	j
139	5	1	2	1	0	1	0	1	3
4	**91**	4	1	3	0	0	2	4	0
0	3	**116**	1	2	0	2	4	2	2
1	0	1	**88**	1	0	0	1	0	1
0	3	1	1	**100**	0	0	2	1	2
1	0	0	0	1	**58**	0	2	0	3
1	0	1	0	0	0	**94**	2	0	0
3	5	6	1	3	1	1	**103**	1	8
0	3	2	0	3	0	0	1	**80**	3
1	0	3	1	3	1	1	3	1	**78**

a = Bid/nus Bidyanus, b = Epin/lus Adscen/nis, c = Cyn/cion Regalis, d = Cara/us Auratus
e = Cyp/nus Carpio, f = Rutilus Rutilus,
g = SalmoTru, h = Oreo/mis Mossambicus
i = Micr/rus Salmoides, j = Oncor/hus Mykiss

Table 7 below, presents the results of the scenarios applied on streaming data in this research. Validation of the results was done by employing the Prequential Evaluation method [34]. The training window used 1000 instances. It should be clarified that the following Table 7 uses average values for every evaluation measure.

Table 7. Validation metrics when streaming data are used

Classifier	Classification accuracy & performance metrics		
	Accuracy	Kappa statistic	Kappa temporal statistic
Categories_Dataset			
ARF	94.48%	73.91%	74.53%
Mammals_Dataset			
ARF	92.16%	71.37%	73.14%
Fish_Dataset			
ARF	88.11%	68.59%	70.36%

7 Discussion and Conclusions

This paper presents an innovative, reliable, low-demand and highly effective system of MAHE and sound analysis, based on sophisticated computational intelligence. The development of FLAME_H is based on the optimal combination of two highly efficient and fast learning algorithms that create a comprehensive intelligent system of active environmental security using a Lambda Architecture approach. The sophisticated application described herein, combined with the promising results that have emerged, constitutes a credible innovative proposal for the standardization and design of biosecurity and biodiversity protection. This implementation follows a Reactive strategy for dealing with invasive species as it combines training of two counter diametrically opposite classifiers to detect incoming contrasts and to discard them. Training is done by using datasets that respond to specialized, realistic scenarios. In addition, this framework implements a Big data analysis approach that attempts to balance latency, throughput, and fault tolerance using integrated and accurate views of historical data, while at the same time it is making optimum use of new entrant data flows. The operating scenarios proposed with the combination of batch and streaming data, create capabilities for a fully-defined configuration of model's parameters and for high-precision classification or correlation.

The basic innovation of the proposed FLAME_H is the implementation of an intelligent ML system, based solely on fully automated methods of detecting sound events using COIN. This innovation provides important solutions and improves the way environmental problems and, in particular, biodiversity and bio-security mechanisms work and deal. Also, a significant innovation is the architecture of the proposed computational intelligence system, which combines and exploits Lambda architecture, that is, the combination of both batch and streaming data analysis, using fast and extremely accurate ML algorithms to solve a multidimensional and complex real-life problem. ML delivers intelligence and significantly boosts the environmental protection mechanisms as it is an important defense tool against asymmetric environmental threats. The FLAME_H simplifies and automates the sound recognition and the invasive species detection procedures, while minimizing human intervention by combining the EL_GROSEMMARI and ARF algorithms for the first time in the literature. Finally, one more innovation is found in the way of collecting and selecting the data, (which emerged after extensive research) as well as the development of the final data set used, which is complex and has a high dimension, but it can be used effectively in training.

Future extensions-improvements should focus on further optimizing the parameters of the algorithm used by the Lambda architecture, so that an even more efficient, accurate, and faster classification process is achieved. Also, it would be important to study the expansion of this particular system by implementing Lambda architecture in a parallel and distributed data analysis system (Hadoop). Finally, an additional element that could be studied in the direction of future expansion concerns the operation of FLAME_H with methods of self-improvement and meta-learning in order to fully automate the process of locating the species.

References

1. Rahel, F., Olden, J.D.: Assessing the effects of climate change on aquatic invasive species. Conserv. Biol. **22**(3), 521–533 (2008). https://doi.org/10.1111/j.1523-1739.2008.00950.x
2. Abdulla, A., Linden, O.: Maritime Traffic Effects on Biodiversity in the Mediterranean Sea: Review of Impacts, Priority Areas and Mitigation Measures. IUCN, Centre for Mediterranean Cooperation, vol. 184, pp. 08 (2008)

3. Miller, W.: The structure of species, outcomes of speciation and the species problem: ideas for paleobiology. Paleogeograohy Palaeoclimatol. Palaeoecol. **176**(1–4), 1–10 (2001). https://doi.org/10.1016/s0031-0182(01)00346-7
4. Lyon, R.: Human and Machine Hearing: Extracting Meaning from Sound. Cambridge University Press, Cambridge (2017). https://doi.org/10.1017/9781139051699
5. Deng, L., Yu, D.: Deep learning: methods and applications. Found. Trends Signal Process. **7** (3–4), 197–387 (2014). https://doi.org/10.1561/2000000039
6. Zhang, J., Yin, J., Zhang, Q., Shi, J., Li, Y.: Robust sound event classification with bilinear multi-column ELM-AE and two-stage ensemble learning. Eurasip J. Audio Speech Music Process. **2017**(1), 11 (2017). https://doi.org/10.1186/s13636-017-0109-1
7. Dedić, N., Stanier, C.: Towards differentiating business intelligence, big data, data analytics and knowledge discovery. In: Piazolo, F., Geist, V., Brehm, L., Schmidt, R. (eds.) ERP Future 2016. LNBIP, vol. 285, pp. 114–122. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-58801-8_10
8. Kiran, M., Murphy, P., Monga, I., Dugan, J., Baveja, S.S.: Lambda architecture for cost-effective batch and speed big data processing, pp. 2785–2792 (2015). https://doi.org/10.1109/bigdata.2015.7364082
9. http://www.cabi.org/isc
10. Yamato, Y., Kumazaki, H., Fukumoto, Y.: Proposal of lambda architecture adoption for real time predictive maintenance. In: 2016 CANDAR, pp. 713–715. IEEE (2016). https://doi.org/10.1109/candar.2016.0130
11. Demertzis, K., Iliadis, L.: Detecting invasive species with a bio-inspired semisupervised neurocomputing approach: the case of Lagocephalus sceleratus. Neural Comput. Appl. **28** (6), 1225–1234 (2017). https://doi.org/10.1007/s00521-016-2591-2
12. Demertzis, K., Iliadis, L.: Intelligent bio-inspired detection of food borne pathogen by DNA barcodes: the case of invasive fish species lagocephalus sceleratus. In: Iliadis, L., Jayne, C. (eds.) Engineering Applications of Neural Networks EANN 2015 Communications in Computer and Information Science, vol. 517, pp. 89–99. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-23983-5_9
13. Demertzis, K., Iliadis, L., Anezakis, V.D.: A deep spiking machine-hearing system for the case of invasive fish species. In: Proceedings of 2017 IEEE International Conference on Innovations in Intelligent Systems and Applications, Gdynia, Poland, pp. 23–28 (2017). https://doi.org/10.1109/inista.2017.8001126
14. Demertzis, K., Iliadis, L.: Adaptive elitist differential evolution extreme learning machines on big data: intelligent recognition of invasive species. In: Angelov, P., Manolopoulos, Y., Iliadis, L., Roy, A., Vellasco, M. (eds.) INNS 2016. AISC, vol. 529, pp. 333–345. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-47898-2_34
15. Demertzis, K., Iliadis, L.: Evolving smart URL filter in a zone-based policy firewall for detecting algorithmically generated malicious domains. In: Gammerman, A., Vovk, V., Papadopoulos, H. (eds.) SLDS 2015. LNCS (LNAI), vol. 9047, pp. 223–233. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-17091-6_17
16. Demertzis, K., Iliadis, L.: SAME: an intelligent anti-malware extension for android art virtual machine. In: Núñez, M., Nguyen, N.T., Camacho, D., Trawiński, B. (eds.) ICCCI 2015. LNCS (LNAI), vol. 9330, pp. 235–245. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24306-1_23
17. Hinton, G., Deng, L., Yu, D., et al.: Deep neural networks for acoustic modeling in speech recognition. IEEE Signal Process. Mag. **29**(6), 82–97 (2014). https://doi.org/10.1109/MSP.2012.2205597

18. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q. (eds.) Proceedings of the 25th International Conference on Neural Information Processing Systems, NIPS 2012, USA, pp. 1097–1105 (2012)
19. Collobert, R., Weston, J.: A unified architecture for natural language processing: deep neural networks with multitask learning. In: William, W., Mccallum, C., Mccallum, A., Sam, T.R. (eds.) Proceedings of the 25th International Conference on Machine Learning, ICML 2008, pp. 160–167. ACM, New York (2008). https://doi.org/10.1145/1390156.1390177, ISBN 978-1-60558-205-4
20. Deselaers, T., Hasan, S., Bender, O., Ney, H.: A deep learning approach to machine transliteration. In: Proceedings of the Fourth Workshop on Statistical Machine Translation StatMT 2009, pp. 233–241. Association for Computational Linguistics, Stroudsburg (2009)
21. Lanez, N.D., Georgievy, P., Qendro, L.: DeepEar: robust smartphone audio sensing in unconstrained acoustic environments using deep learning. In: Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing, pp. 283–294 (2015). http://dx.doi.org/10.1145/2750858.2804262
22. Alom, M.Z., Alam, M., Taha, T.M., Iftekharuddin, K.M.: Object recognition using cellular simultaneous recurrent networks and convolutional neural network. In: Proceedings of the International Joint Conference on Neural Networks (IJCNN 2017), pp. 2873–2880, Anchorage (2017). https://doi.org/10.1109/ijcnn.2017.7966211
23. Zhang, D., Lee, D.J., Zhang, M., Tippetts, B.J., Lillywhite, K.D.: Object recognition algorithm for the automatic identification and removal of invasive fish. Biosyst. Eng. **145**, 65–75 (2016). https://doi.org/10.1016/j.biosystemseng
24. Han, K., Yu, D., Tashev, I.: Speech emotion recognition using deep neural network and extreme learning machine. In: Chng, E.S., Li, H., Meng, H., Ma, B., Xie, L. (eds.) INTERSPEECH 2014, Proceedings of the Annual Conference of the International Speech Communication Association, pp. 223–227 (2014)
25. Zhao, Z., Zhang, S.H., Xu, Z.Y., et al.: Automated bird acoustic event detection and robust species classification. Ecol. Inf. **39**, 99–108 (2017). https://doi.org/10.1016/j.ecoinf.2017.04.003
26. Cambria, E., Guang-Bin, H.: Extreme learning machines. In: IEEE InTeLLIGenT SYSTemS, 541-1672/13 (2013)
27. Huang, G.B.: An insight into extreme learning machines: random neurons, random features and kernels. Cogn. Comput. **6**(3), 376–390 (2014). https://doi.org/10.1007/s12559-014-9255-2
28. Sun, K., Zhang, J., Zhang, C., Hu, J.: Generalized extreme learning machine autoencoder and a new deep neural network. Neurocomputing **230**, 374–381 (2017)
29. Gomes, H.M., Bifet, A., Read, J., et al.: Adaptive random forests for evolving data stream classification. Mach. Learn. **106**(9–10), 1469–1495 (2017). https://doi.org/10.1007/s10994-017-5642-8
30. Giannakopoulos, T.: pyAudioAnalysis: an open-source Python library for audio signal analysis. PLoS ONE **10**(12) (2015). https://doi.org/10.1371/journal.pone.0144610
31. Žliobaitė, I., Bifet, A., Read, J., Pfahringer, B., Holmes, G.: Evaluation methods and decision theory for classification of streaming data with temporal dependence. Mach. Learn. **98**(3), 455–482 (2015). https://doi.org/10.1007/s10994-014-5441-4

32. Vinagre, J., Jorge, A.M., Gama, J.: Evaluation of recommender systems in streaming environments. In: Workshop on Recommender Systems Evaluation: Dimensions and Design (REDD 2014), Held in Conjunction with RecSys (2014). https://doi.org/10.13140/2.1.4381.5367
33. Mao, J., Jain, A.K., Duin, P.W.: Statistical pattern recognition: a review. IEEE Trans. Pattern Anal. Mach. Intell. **22**(1), 4–37 (2000). https://doi.org/10.1109/34.824819
34. Fawcett, T.: An introduction to ROC analysis. Pattern Recogn. Lett. Sci. **27**(8), 861–874 (2006). https://doi.org/10.1016/j.patrec.2005.10.010