Scientific Research Publishing

World Journal of Neuroscience, 2026, 16(3), 127-155
https://www.scirp.org/journal/wjns
ISSN Online: 2162-2019
ISSN Print: 2162-2000

Analysis and Interpretation of EEG Signals for the Design of an Automated Epilepsy Detection through an Ensemble of Intelligent Systems

Romain Atangana[1,2,3*], Amstrong Emini Me Zenanga[3], Daniel Tchiotsop[1], Godpromesse Kenne[1]

[1]Research Unit in Automation and Applied Computer Science (UR-AIA), IUT of Bandjoun, University of Dschang, Bandjoun, Cameroon
[2]Research Unit in Condensed Matter, Electronics and Signal Processing (UR-MACETS), Faculty of Science, University of Dschang, Dschang, Cameroon
[3]Department of Computer Science, Higher Teacher Training College, University of Bertoua, Bertoua, Cameroun
Email: *zenanga75@gmail.com

How to cite this paper: Atangana, R., Zenanga, A.E.M., Tchiotsop, D. and Kenne, G. (2026) Analysis and Interpretation of EEG Signals for the Design of an Automated Epilepsy Detection through an Ensemble of Intelligent Systems. *World Journal of Neuroscience*, **16**, 127-155.
https://doi.org/10.4236/wjns.2026.163012

Received: March 8, 2026
Accepted: June 27, 2026
Published: June 30, 2026

Abstract

Epileptic seizures are characterized by abnormal neuronal discharges that cause significant disturbances in brain electrical activity. Electroencephalography (EEG) is widely used for epilepsy diagnosis, but manual interpretation of EEG signals is time-consuming and prone to human error. In this study, we propose an automated epilepsy detection system based on an ensemble of intelligent classifiers. The proposed framework first employs Discrete Wavelet Transform (DWT) to decompose EEG signals into five frequency sub-bands (Delta, Theta, Alpha, Beta and Gamma). Linear Discriminant Analysis (LDA) is then applied for feature dimensionality reduction. Three machine learning classifiers, namely Multilayer Perceptron (MLP), Support Vector Machine (SVM) and K-Nearest Neighbors (KNN), are combined using a majority voting strategy within a multi-agent boosting framework. Experiments were conducted on the publicly available University of Bonn EEG dataset. The proposed ensemble system achieved a classification accuracy of 100% with a significantly reduced global interpretation error compared to individual classifiers. These results demonstrate the effectiveness of the proposed intelligent ensemble approach for assisting neurologists in epilepsy diagnosis.

Keywords

Epilepsy Seizure, EEG Signals, Intelligent System, Discrete Wavelet Transform (DWT), Schapire Boosting Technique, ROC Curves, Confusion Matrix

1. Introduction

Epilepsy is a chronic neurological disorder that affects more than 50 million people worldwide, with approximately 2.4 million new cases reported annually, representing nearly 1% of the global population [1]. It is characterized by recurrent seizures caused by abnormal and excessive neuronal discharges in the brain, which may lead to involuntary movements, loss of consciousness, memory disturbances, and other neurological manifestations [2]. Because epilepsy can significantly affect quality of life and increase the risk of premature death, early and reliable diagnosis remains a major clinical challenge.

Electroencephalography (EEG) is one of the most widely used tools for the diagnosis and monitoring of epilepsy, as it records the electrical activity of the brain and helps identify abnormal patterns such as spikes, sharp waves, and rhythmic discharges associated with epileptic events [3]. However, manual visual inspection of EEG recordings by neurologists is time-consuming, subjective, and prone to human error due to the nonlinear and non-stationary nature of EEG signals. In addition, the complexity of EEG interpretation can lead to misdiagnosis or delayed detection of epileptic seizures.

To overcome these limitations, numerous studies have proposed automated systems for EEG signal analysis using signal processing and machine learning techniques. These approaches aim to extract discriminative features from EEG signals and classify them into normal or epileptic states using intelligent classifiers [4]. Such systems can significantly improve diagnostic accuracy while reducing the time required for clinical analysis.

In this work, we propose an automated epilepsy detection framework based on EEG signal analysis. The proposed method combines Discrete Wavelet Transform (DWT) for signal decomposition into frequency sub-bands and Linear Discriminant Analysis (LDA) for dimensionality reduction. These features are then classified using an ensemble of machine learning classifiers, namely Multilayer Perceptron (MLP), Support Vector Machine (SVM), and K-Nearest Neighbors (KNN), combined through a boosting-based multi-agent system and a majority voting strategy. Experiments conducted on the publicly available University of Bonn EEG dataset demonstrate that the proposed approach effectively discriminates between normal and epileptic EEG signals, providing a reliable decision-support tool for neurologists.

2. Related Work

Research on automated epilepsy detection using electroencephalogram (EEG) signals has attracted significant attention over the past two decades. The evolution of this field can be broadly divided into three main stages: traditional signal-processing approaches, hybrid machine learning frameworks, and deep learning-based systems.

2.1. Early Signal-Processing and Machine Learning Methods

Early studies on epilepsy detection mainly relied on handcrafted feature extrac-

tion combined with classical classifiers. For instance, Faust *et al* [5] investigated automatic seizure detection using power spectral density analysis combined with classifiers such as Support Vector Machines (SVM), Artificial Neural Networks (ANN), and Gaussian models. Their results demonstrated that machine learning could effectively discriminate epileptic EEG signals from normal brain activity.

Similarly, Lima *et al.* [6] explored the application of least-square SVM (LS-SVM) for epilepsy diagnosis. Their study showed that optimizing kernel parameters significantly improves classification performance and generalization capability.

In subsequent work, Acharya *et al.* [7] applied recurrence quantification analysis to extract nonlinear features from EEG signals and evaluated several classifiers, including SVM, fuzzy logic, KNN, Naïve Bayes, decision trees, and probabilistic neural networks. Their results indicated that SVM and fuzzy logic provided superior performance for discriminating between normal, pre-ictal, and ictal EEG signals.

During the same period, research also focused on artifact removal and signal denoising, which are critical preprocessing steps in EEG analysis. Guerrero Mosquera *et al.* [8] proposed an artifact removal method based on Independent Component Analysis (ICA) combined with adaptive filtering. Likewise, Guerrero-Mosquera *et al.* [8] introduced a blind source separation technique to eliminate ocular and muscular artifacts while preserving important brain activity information.

Other studies investigated wavelet-based feature extraction. For example, Acharya *et al.* [9] employed wavelet decomposition combined with principal component analysis (PCA) and ANOVA feature selection for automatic EEG classification. Their approach achieved high classification accuracy on the Bonn EEG dataset. Similarly, Tzallas *et al.* [10] highlighted the importance of automated EEG analysis in improving epilepsy diagnosis and clinical decision-making.

2.2. Hybrid Machine Learning and Ensemble Learning Approaches

Between 2013 and 2020, research progressively shifted toward hybrid frameworks combining multiple feature extraction methods and ensemble classifiers. For example, Yung-Chen Liu *et al.* [11] proposed a two-stage approach using nonlinear energy operators for peak detection followed by AdaBoost classification. Their results demonstrated the effectiveness of boosting techniques in improving seizure detection performance.

Later, Siuly *et al.* [3] integrated genetic algorithms with SVM classifiers to optimize training parameters, achieving improved classification accuracy. Similarly, Wang *et al.* [12] developed a multi-domain feature extraction approach combining time-domain, frequency-domain, and nonlinear features with dimensionality reduction techniques.

Another significant contribution was made by Guo *et al.* [13], who proposed an automatic feature reduction framework based on wavelet decomposition and correlation-based feature selection. Their model employed multiple classifiers, in-

cluding Random Forest, SVM, Logistic Decision Trees, Radial Basis Networks, and MLP neural networks, demonstrating the effectiveness of ensemble learning in multi-class EEG classification.

In recent years, hybrid machine learning techniques have been further enhanced using boosting and dimensionality reduction strategies. For instance, Al-Hadeethi *et al.* [14] introduced an adaptive boosting LS-SVM framework combined with covariance-based feature reduction for epileptic seizure detection. Their results showed that boosting techniques significantly improve classification accuracy by emphasizing difficult training samples.

2.3. Deep Learning and Modern EEG Analysis Techniques

More recently, advances in deep learning have significantly transformed EEG signal analysis. Deep architectures allow automatic feature learning directly from raw EEG signals, reducing the need for handcrafted features.

For example, Srinivasan *et al.* [15] proposed a hybrid deep learning model combining a three-dimensional deep convolutional autoencoder with a Bidirectional Long Short-Term Memory (Bi-LSTM) classifier. Their approach achieved high performance on the CHB-MIT EEG dataset, demonstrating the potential of deep neural networks for epilepsy detection.

Similarly, Yılmaz *et al.* [16] proposed a combined deep learning architecture integrating both time-series EEG signals and time-frequency representations generated using Continuous Wavelet Transform (CWT) and Short-Time Fourier Transform (STFT). Their method achieved strong performance across multiple public datasets.

Other recent works have explored alternative signal processing strategies. Wijayanto *et al.* [17] used compressive sensing to reduce EEG signal dimensionality before classification with SVM. In addition, they combined wavelet transform and Fourier transform for feature extraction before classification using neural networks.

Recent studies have also introduced explainable artificial intelligence techniques. Bhandage *et al.* [18] proposed a deep learning-based approach using spectrogram images and pretrained convolutional neural networks such as MobileNetV2 and EfficientNet. Their work also integrated explainability tools such as SmoothGradCAM++ to improve interpretability.

Bhandage *et al.* [18] proposed an epilepsy detection approach based on spectrogram images generated from EEG signals and deep learning models, with an emphasis on model interpretability using explainable artificial intelligence techniques.

Finally, Lan Wei and Mooney [19] proposed a LightGBM-based classification model trained on large-scale TUH EEG data, highlighting the importance of using realistic clinical datasets for model generalization.

2.4. Positioning of the Proposed Approach

From the above literature review, it can be observed that research in automated

epilepsy detection has progressively evolved from traditional signal-processing methods to hybrid machine learning approaches and, more recently, deep learning-based frameworks. However, despite the promising performance of deep learning models, hybrid intelligent systems combining signal processing techniques with ensemble learning strategies remain highly competitive due to their robustness, interpretability, and lower computational requirements.

In this context, the present study proposes a boosting-based multi-agent intelligent system that integrates Discrete Wavelet Transform (DWT) for signal decomposition, Linear Discriminant Analysis (LDA) for dimensionality reduction, and three complementary classifiers (MLP, SVM, and KNN) combined through a majority voting mechanism. This approach aims to improve classification robustness and reduce interpretation errors in EEG-based epilepsy diagnosis.

3. Material and Method

3.1. Hardware and Software Environment

The simulations and implementation of the proposed algorithms were carried out using standard computing platforms. The experiments were conducted on personal computers equipped with Intel Core processors and running Windows operating systems. The systems were configured with RAM capacities ranging from 4 GB to 16 GB and standard storage devices.

For the implementation and evaluation of the proposed epilepsy detection framework, scientific computing software tools were used. In particular, MATLAB (Matrix Laboratory) was employed for signal processing, feature extraction, dimensionality reduction, and classifier training. MATLAB provides a powerful environment for numerical computation and signal analysis, which makes it particularly suitable for EEG signal processing tasks.

The following operations were performed using MATLAB:

- Import and preprocessing of EEG signals from the clinical database;
- Discrete Wavelet Transform (DWT) decomposition of EEG signals into frequency sub-bands;
- Extraction of statistical features such as minimum, maximum, mean, standard deviation, and energy values;
- Dimensionality reduction of the extracted feature vectors using Linear Discriminant Analysis (LDA);
- Training and testing of machine learning classifiers including Support Vector Machine (SVM) and K-Nearest Neighbors (KNN);
- Evaluation of classification performance using confusion matrices, Receiver Operating Characteristic (ROC) curves, and other performance metrics.

In addition, the Java programming language was used to implement the Multilayer Perceptron (MLP) classifier and to simulate the convergence behavior of the neural network during the training phase. Java also allowed the visualization of the generalization performance of the model on the testing dataset.

3.2. EEG Database and Experimental Protocol

3.2.1. EEG Database

The experimental evaluation of the proposed method was conducted using the publicly available EEG dataset provided by the Epilepsy Center of the University of Bonn [20].

This database consists of five datasets labeled A, B, C, D, and E, each containing 100 EEG signal segments. Each segment comprises 4,097 samples recorded at a sampling frequency of 173.61 Hz, corresponding to a duration of 23.6 seconds. The EEG signals were acquired using a 12-bit resolution system with a band-pass filter ranging from 0.53 Hz to 40 Hz (12 dB/octave).

The dataset includes recordings from three different subject groups:

- Healthy subjects (normal EEG signals): datasets A and B;
- Epileptic patients during inter-ictal periods: datasets C and D;
- Epileptic patients during seizure activity (ictal state): dataset E.

The Bonn EEG dataset is widely used as a benchmark in epilepsy detection studies due to its well-structured format and clinically validated recordings. The signals provided in this dataset are already preprocessed and considered artifact-free, and therefore no additional artifact removal was applied in this study.

3.2.2. Experimental Protocol

For each classification task (A-E, B-E, CD-E, and AB-CD), a total of 200 EEG segments were used (100 segments per class).

The dataset was randomly divided into three subsets:

- 70% for training;
- 15% for validation;
- 15% for testing.

To ensure the robustness and reliability of the results, the experiments were repeated multiple times using different random splits, and the average performance metrics were reported.

The following classifier configurations were used:

- Multilayer Perceptron (MLP): 12 input neurons, 12 hidden neurons, and 2 output neurons;
- Support Vector Machine (SVM): polynomial kernel with optimized hyperparameters;
- K-Nearest Neighbors (KNN): Euclidean distance metric with K values of 5, 10, and 15.

3.3. Proposed Method

The proposed epilepsy detection framework is based on a hybrid intelligent system that combines signal processing techniques with machine learning classifiers. The overall methodology is structured into four main stages: EEG signal preprocessing, feature extraction, dimensionality reduction, and classification using an ensemble of intelligent classifiers.

In the first stage, EEG signals are preprocessed to enhance signal quality and remove noise components. This is followed by feature extraction using the Discrete Wavelet Transform (DWT), which decomposes the EEG signals into multiple frequency sub-bands corresponding to clinical rhythms.

Next, dimensionality reduction is performed using Linear Discriminant Analysis (LDA) in order to reduce redundancy in the feature space while preserving class separability. The resulting feature vectors are then used as inputs to the classification models.

Finally, classification is carried out using an ensemble of intelligent classifiers, including Multilayer Perceptron (MLP), Support Vector Machine (SVM), and K-Nearest Neighbors (KNN), organized within a boosting-based multi-agent system.

To prevent data leakage, all preprocessing steps were performed exclusively on the training data. In particular, LDA was fitted only on the training set and then applied to the validation and test sets.

Classifier training was conducted using training data only, while model evaluation was performed on unseen test data. Furthermore, the meta-classifiers in the multi-agent system were trained exclusively using outputs generated during the training phase.

3.3.1. Feature Extraction Pipeline (DWT + LDA)

In this study, the Discrete Wavelet Transform (DWT) was applied using the Daubechies wavelet of order 4 (db4), which is well suited for EEG signal analysis. Each EEG signal was decomposed into four levels, resulting in five sub-bands corresponding to clinical rhythms: Delta (D4), Theta (D3), Alpha (D2), Beta (D1), and Gamma (A4).

From each sub-band, five statistical features were extracted: maximum value, mean value, standard deviation, and spectral energy. This resulted in a total of 25 features per EEG segment.

Before feature extraction, EEG signals were filtered using a finite impulse response (FIR) filter of order 30 with a passband ranging from 0.043 Hz to 64 Hz.

The Bonn EEG dataset was used as provided, as it is already preprocessed and artifact-free. Therefore, no additional artifact removal method was applied.

3.3.2. Mono Classification

In the mono-classification stage, each classifier is trained and evaluated independently in order to assess its individual ability to discriminate between normal and epileptic EEG signals. This step is important because it provides a baseline analysis of the performance of each learning model before combining them in the proposed ensemble framework. In this work, three classifiers were investigated separately, namely the Multilayer Perceptron Neural Network (MLPNN), Support Vector Machine (SVM), and K-Nearest Neighbors (KNN). The objective of this stage is to identify the strengths and limitations of each classifier when used as a standalone decision model for epilepsy detection [21].

3.3.3. Multi-Layer Perceptron Neural Network (MLPNN) Classifier

1) Overview

The Multilayer Perceptron Neural Network (MLPNN) is a widely used supervised learning model for pattern recognition and classification tasks. It is particularly well suited for biomedical signal analysis due to its ability to model complex and nonlinear relationships between input features and output classes.

In this study, the MLPNN classifier is used to classify EEG feature vectors obtained after Discrete Wavelet Transform (DWT)-based feature extraction and Linear Discriminant Analysis (LDA)-based dimensionality reduction.

2) Network Architecture

An MLPNN is composed of three main types of layers: an input layer, one or more hidden layers, and an output layer. Neurons in each layer are interconnected through weighted connections, enabling the network to learn hierarchical feature representations.

In this work, the MLPNN architecture consists of:

- An input layer of 12 neurons (corresponding to the reduced feature space);
- A hidden layer of 12 neurons;
- An output layer of 2 neurons for binary classification.

3) Learning Process

The training of the MLPNN is performed using the backpropagation algorithm, which relies on gradient descent optimization to minimize the classification error.

The learning process is divided into two main stages:

- **Forward propagation**: The input feature vector is propagated through the network. Each neuron computes its output using an activation function, and the final layer produces the predicted class label.
- **Backward propagation**: The error between the predicted output and the true label is propagated backward through the network. This allows the computation of gradients of the loss function with respect to the network parameters, which are then updated iteratively to minimize the overall error.

This iterative optimization process continues until convergence or until a predefined stopping criterion is reached.

4) Relevance for EEG Classification

The MLPNN is particularly suitable for EEG signal classification due to its ability to learn nonlinear decision boundaries. EEG data are inherently complex and often exhibit overlapping class distributions, making linear classifiers less effective. The MLPNN can capture these nonlinear relationships, thereby improving classification performance.

5) Training and Evaluation Protocol

The MLPNN was trained using 70% of the dataset, with 15% used for validation and 15% for testing. The performance of the classifier was evaluated using standard metrics, including accuracy, sensitivity, and specificity.

The results obtained in mono-classification were compared with those of Support Vector Machine (SVM) and K-Nearest Neighbors (KNN) classifiers in order

to assess its effectiveness as a standalone model before its integration into the proposed boosting-based multi-agent system.

6) Backpropagation Algorithm

The backpropagation algorithm is a supervised learning method used to train multilayer neural networks by minimizing the error between the network output and the desired target. It is based on the principle of error correction through gradient descent.

At the output layer, the correction is guided by the difference between the predicted output and the desired output. For the hidden layers, the correction depends on the contribution of each hidden neuron to the output error. In other words, the error is propagated backward through the network so that each weight is adjusted according to its influence on the final classification result.

This iterative optimization process continues until the network reaches an acceptable level of error or until a predefined stopping criterion is met. Once trained, the MLPNN can be used in the testing phase to classify unseen EEG signals and evaluate its generalization capability.

Learning by gradient descent:

- Output layer: Correction guided by the error between the desired and the actual output;
- Hidden layers: Correction about the sensibility (influence of the neuron on the error in the output layer).

Outputs values of the neurons (Figure 1):

Value y_j^t of the neuron j for the input data X^t

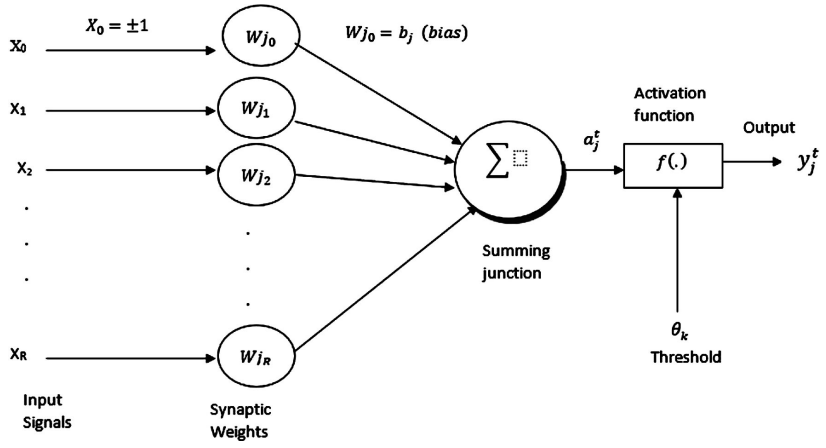


Figure 1. Mathematical model of ANN.

$$y_j^t = f\left(a_j^t\right) = f\left(\sum_{i=1}^{R} \omega_{j,i} \cdot y_i^t + \omega_{j,0}\right) \tag{1}$$

***f*: activation function of the neuron**

$a_j^t = \sum_{i=1}^{R} \omega_{j,i} \cdot y_i^t + \omega_{j,0}$ is the cumulative sum of the input neurons;

$\omega_{j,i}$: weights of the link connecting the neuron *j* to the neuron *i* of the previous layer;

$\omega_{j,0}$: bias of the neuron j;

y_i^t : output of the neuron i of the previous layer for the input data X^t ;

R : number of the neurons on the previous layer.

Output error to the output layer:

A set of data $\chi = \left\{x^t, r^t\right\}_{t=1}^N$ with $r^t = \left[r_1^t \;\; r_2^t \;\; \cdots \;\; r_K^t\right]^{\mathrm{T}} = \begin{bmatrix} r_1^t \\ r_2^t \\ \vdots \\ r_K^t \end{bmatrix}$

where $r_j^t = 1$ if $x^t \in C_j$, otherwise $r_j^t = 0$.

Error observed for the data x^t on the neuron j on the output layer

$$e_j^t = r_j^t - y_j^t \tag{2}$$

Quadratic error observed for the data x^t on all the K neurons of the output layer (one neuron per class).

$$E^t = \frac{1}{2}\sum_{j=1}^K \left(e_j^t\right)^2 \tag{3}$$

Quadratic mean error observed for the data set χ .

$$E = \frac{1}{N}\sum_{t=1}^N E^t \tag{4}$$

Correction of the error for the output layer:

Correction of the weights through descent gradient of the quadratic mean error

$$\Delta\omega_{j,i} = -\eta\frac{\partial E}{\partial \omega_{j,i}} = -\eta\frac{\partial\left(\frac{1}{N}\sum_{t=1}^N E^t\right)}{\partial \omega_{j,i}} \tag{5}$$

$$\Delta\omega_{j,i} = -\frac{\eta}{N}\sum_{t=1}^N \frac{\partial E^t}{\partial \omega_{j,i}}$$

The error of the neuron j depends on the neurons of the previous layer:

Development while using the chaining derivative rule $\left(\frac{\partial f}{\partial x} = \frac{\partial f}{\partial y}\frac{\partial y}{\partial x}\right)$

$$\frac{\partial E^t}{\partial \omega_{i,j}} = \frac{\partial E^t}{\partial e_j^t}\frac{\partial e^t}{\partial y_j^t}\frac{\partial y_j^t}{\partial a_j}\frac{\partial a_j^t}{\partial \omega_{j,i}} \tag{6}$$

$$\frac{\partial E^t}{\partial \omega_{j,0}} = \frac{\partial E^t}{\partial e_j^t}\frac{\partial e^t}{\partial y_j^t}\frac{\partial y_j^t}{\partial a_j}\frac{\partial a_j^t}{\partial \omega_{j,0}} \tag{7}$$

Calculation of the partial derivatives:

Development with the sigmoid activation function

$$y_j^t = \frac{1}{1+\exp\left(-a_j^t\right)} \tag{8}$$

$$\frac{\partial E^t}{\partial e_j^t} = \frac{\partial}{\partial e_j^t}\left[\frac{1}{2}\sum_{j=1}^K \left(e_j^t\right)^2\right] = e_j^t \tag{9}$$

$$\frac{\partial e_j^t}{\partial y_j^t} = \frac{\partial\left[r_j^t - y_j^t\right]}{\partial y_j^t} = -1 \tag{10}$$

$$\frac{\partial y_j^t}{\partial a_j^t} = \frac{\partial\left[\frac{1}{1+\exp\left(-a_j^t\right)}\right]}{\partial a_j^t} = \frac{\exp\left(-a_j^t\right)}{\left[1+\exp\left(-a_j^t\right)\right]^2} \tag{11}$$

$$= \left[\frac{1}{1+\exp\left(-a_j^t\right)}\right]\left[1-\frac{1}{1+\exp\left(-a_j^t\right)}\right]$$

$$\frac{\partial y_j^t}{\partial a_j^t} = \frac{1}{1+\exp\left(-a_j^t\right)} * \frac{\mathrm{e}^{-a_j^t}+1-1}{1+\exp\left(-a_j^t\right)} = y_j^t\left(1-y_j^t\right) \tag{12}$$

$$\frac{\partial y_j^t}{\partial a_j^t} = y_j^t\left(1-y_j^t\right) \tag{13}$$

$$\frac{\partial a_j^t}{\partial \omega_{j,i}} = \frac{\partial}{\partial \omega_{j,i}}\sum_{i=1}^{R}\omega_{j,i}y_i^t + \omega_{j.0} = y_i^t \tag{14}$$

$$\frac{\partial a_j^t}{\partial \omega_{j,0}} = 1 \tag{15}$$

Learning for the output layer:

Adjusting of the weights of the output layer:

$$\Delta\omega_{j,i} = -\frac{\eta}{N}\sum_{t=1}^{N}\frac{\partial E^t}{\partial \omega_{j,i}} = -\frac{\eta}{N}\sum_{t=1}^{N}\frac{\partial E^t}{\partial e_j^t}\frac{\partial e_j^t}{\partial y_j^t}\frac{\partial y_j^t}{\partial a_j^t}\frac{\partial a_j^t}{\partial \omega_{j,i}} \tag{16}$$

$$\Delta\omega_{j,i} = +\frac{\eta}{N}\sum_{t=1}^{N}e_j^t y_j^t\left(1-y_j^t\right)y_i^t \tag{17}$$

$$\Delta\omega_{j,0} = -\frac{\eta}{N}\sum_{t=1}^{N}\frac{\partial E^t}{\partial \omega_{j,0}} = -\frac{\eta}{N}\sum_{t=1}^{N}\frac{\partial E^t}{\partial e_j^t}\frac{\partial e_j^t}{\partial y_j^t}\frac{\partial y_j^t}{\partial a_j^t}\frac{\partial a_j^t}{\partial \omega_{j,0}} \tag{18}$$

$$\Delta\omega_{j,0} = \frac{\eta}{N}\sum_{t=1}^{N}e_j^t y_j^t\left(1-y_j^t\right) \tag{19}$$

Delta Rule:

- Let a delta δ_j^t, which corresponds to the local gradient of the neuron *j* for the input data X^t.

$$\delta_j^t = e_j^t y_j^t\left(1-y_j^t\right) \tag{20}$$

$$\text{Thus}\quad \Delta\omega_{j,i} = \frac{\eta}{N}\sum_{t=1}^{N}\delta_j^t y_i^t \tag{21}$$

$$\text{And}\quad \Delta\omega_{j,0} = \frac{\eta}{N}\sum_{t=1}^{N}\delta_j^t \tag{22}$$

Correction of the error for the hidden layers:

- Gradient of the error for the hidden layers.

$$\frac{\partial E^t}{\partial \omega_{j,i}} = \frac{\partial E^t}{\partial y_j^t}\frac{\partial y_j^t}{\partial a_j^t}\frac{\partial a_j^t}{\partial \omega_{j,i}} \tag{23}$$

Only $\frac{\partial E^t}{\partial y_j^t}$ changes $\frac{\partial y_j^t}{\partial a_j^t}$ and $\frac{\partial a_j^t}{\partial \omega_{j,i}}$ are the same as on the output layer.

- The error for a neuron of the hidden layer depends on the error of the K neurons of the next layer. (Back-propagation of the error)

$$E^t = \frac{1}{2}\sum_{k=1}^{N}\left(e_k^t\right)^2 \tag{24}$$

$$\frac{\partial E^t}{\partial y_j^t} = \frac{\partial}{\partial y_j^t}\left[\frac{1}{2}\sum_k\left(e_k^t\right)^2\right] = \sum_k e_k^t \frac{\partial e_k^t}{\partial y_j^t} \tag{25}$$

$$\frac{\partial E^t}{\partial y_j^t} = \sum_k e_k^t \frac{\partial e_k^t}{\partial a_k^t}\frac{\partial a_k^t}{\partial y_j^t} = \sum_k e_k^t \frac{\partial\left[r_k^t - y_k^t\right]}{\partial a_k^t}\frac{\partial\left[\sum_l \omega_{k,l} y_k^t + \omega_{k,0}\right]}{\partial y_j^t} \tag{26}$$

$$\frac{\partial E^t}{\partial y_j^t} = \sum_k e_k^t\left[-y_k^t\left(1-y_k^t\right)\right]\omega_{k,j} \tag{27}$$

$$\delta_k^t = e_k^t\left[y_k^t\left(1-y_k^t\right)\right] \tag{28}$$

$$\frac{\partial E^t}{\partial y_j^t} = -\sum_k \delta_k^t \omega_{k,j} \tag{29}$$

Correction of the corresponding error.

$$\frac{\partial E^t}{\partial \omega_{j,i}} = \frac{\partial E^t}{\partial y_j^t}\frac{\partial y_j^t}{\partial a_j^t}\frac{\partial a_j^t}{\partial \omega_{j,i}} = -\left[\sum_k \delta_k^t \omega_{k,j}\right] y_j^t\left(1-y_j^t\right) y_i^t \tag{30}$$

$$\delta_j^t = y_j^t\left(1-y_j^t\right)\sum_k \delta_k^t \omega_{k,j} \tag{31}$$

$$\Delta\omega_{j,i} = -\eta\frac{\partial E}{\partial \omega_{j,i}} = -\frac{\eta}{N}\sum_{t=1}^{N}\frac{\partial E^t}{\partial \omega_{j,i}} = \frac{\eta}{N}\sum_{t=1}^{N}\delta_j^t y_i^t \tag{32}$$

$$\Delta\omega_{j,0} = -\eta\frac{\partial E}{\partial \omega_{j,0}} = -\frac{\eta}{N}\sum_{t=1}^{N}\frac{\partial E^t}{\partial \omega_{j,0}} = \frac{\eta}{N}\sum_{t=1}^{N}\delta_j^t \tag{33}$$

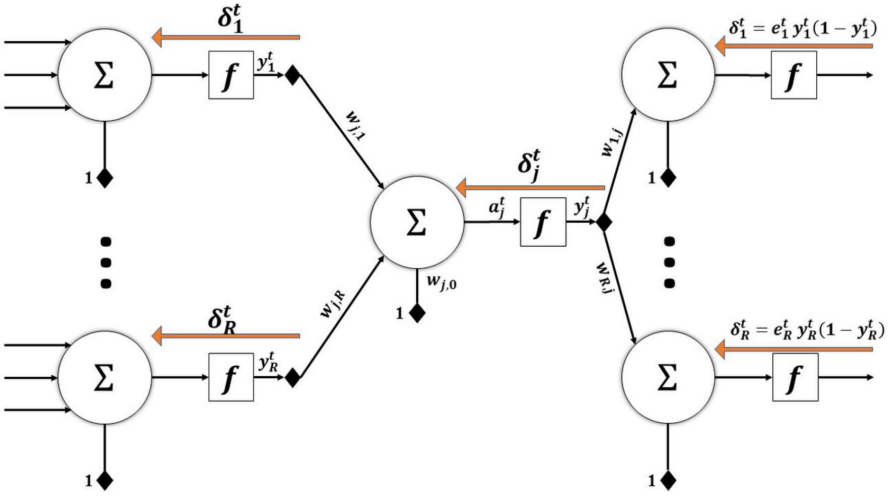


Figure 2. Back-propagation of error.

The back-propagation algorithm can be summarized by the following steps (Figure 2):

Step1: Normalize the training data $X_i^t \in [-1,1]$ and the desired outputs $\widetilde{r_i^t} \in \{0,05;0,95\}$.

Step 2: Initialize the weights and bias to small random values, $\omega_{j,i} \in \mathcal{U}(-0,5,0,5)$.

Step3: Select the structure of the network (such as the number of hidden layers and number of neurons for each layer).

Step4: Choose activation functions for the neurons. These activation functions can be uniform or they can be different for different layers.

Step5: Select the training pair from the training set. Apply the input vector to the network input and propagate the signal forward through the network.

Step6: Calculate the output of the network based on the initial weights and input set.

Step7: Calculate the error between network output and desired output.

$$e_j^t = \tilde{r}_j^t - y_j^t; j = 1,\cdots,K; t = 1,\cdots,N \tag{34}$$

Step8: Propagate error backward and adjust the weights in such a way that minimizes the error. Start from the output layer and go backward to the input layer

$$\omega_{j,i} = \omega_{j,i} + \Delta\omega_{j,i} = \omega_{j,i} + \frac{\eta}{N}\sum_t \delta_j^t y_i^t \tag{35}$$

$$\omega_{j,0} = \omega_{j,0} + \Delta\omega_{j,0} = \omega_{j,0} + \frac{\eta}{N}\sum_t \delta_j^t \tag{36}$$

where the local gradient or Delta is defined as:

$$\delta_j^t = \begin{cases} e_j^t y_j^t\left(1-y_j^t\right), & \text{if } j \text{ belongs to the output layer} \\ y_j^t\left(1-y_j^t\right)\sum_k \delta_k^t \omega_{k,j}, & \text{if } j \text{ belongs to the hidden layer} \end{cases} \tag{37}$$

Step9: Repeat step 5 - 8 for each vector in the training set until the error for the set is lower than the required minimum error.

After enough repetitions of these steps the error between the actual output and the target outputs should be reduced to an acceptable value, and the network is said to be trained. At this point the network can be used in the recall or generalization phases where the weights are not changed.

7) SVM classifier

Support vector machines or wide margin separators (SVMs) are a set of supervised learning techniques designed to solve discrimination and regression problems. and regression problems. SVMs are a generalization of linear classifiers. SVMs were developed in the 1990s from Vladimir Vapnik's theoretical considerations on the development of a Vapnik-Chervonenkis theory [22]. SVMs were quickly adopted for their ability to work with high-dimensional data, their low number of hyperparameters their theoretical guarantees and their good results in practice.

Principle:

Linearly separate positive and negative attributes in the set of attributes. Each attribute must be represented by an n-dimensional vector. the hyperplane that separates the attributes into two classes. The class of positive attributes and the class of negative attributes [23]. This separation must be made by ensuring that the margin between the nearest positive and negative attributes is maximal. As we aim to maximize this margin, we'll refer to this as the wide-margin separator method. Consider the hyperplane defined by (**Figure 3**):

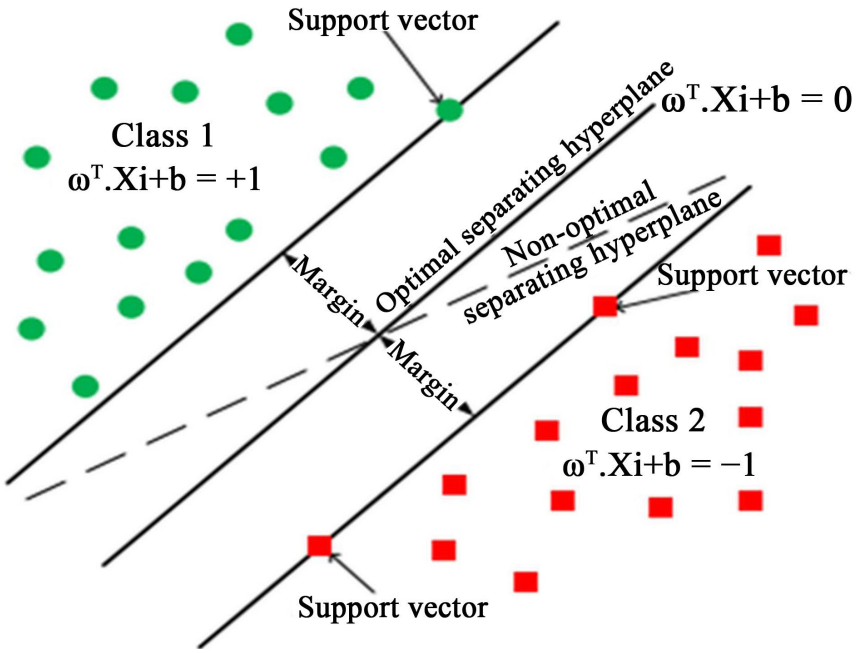


Figure 3. Support machine classifier.

The optimal hyperplane separating the points of two classes is the one that passes "to the middle" of these classes, *i.e.* whose distance to the nearest points of the two classes is maximum. These nearest attributes, which are sufficient to determine this hyperplane, are called support vectors or critical attributes. The optimal hyperplane is perpendicular to the shortest line segment joining a learning attribute to the hyperplane.

On the other hand, only points on boundary hyperplanes play a role. These points are called support vectors by Vapnik. The optimal hyperplane separating the points of two classes is the one that passes "in the middle" these classes, *i.e.* whose distance to the closest points of the two classes is maximum. These nearest attributes, which are sufficient to determine this hyperplane, are called support vectors or critical attributes. The optimal hyperplane is perpendicular to the shortest line segment joining a learning attribute to the hyperplane.

On the other hand, only points on boundary hyperplanes play a role. These points are called support vectors by Vapnik. They are also critical points, as they determine the optimal hyperplane.

$$W^{\mathrm{T}}x + W_0 = 0$$

Margin is the Euclidian distance $\frac{1}{\|W\|}$ between the two parallel hyperplanes described by:

$$W^{\mathrm{T}}x+W_0=1 \text{ et } W^{\mathrm{T}}x+W_0=-1$$

Let x_i the training data, with classes $y_i\in\{-1,1\}, i=1,2,\cdots,N$ respectively for a binary classification problem. The aim is to optimize the operation so that the training error is minimal and to minimize the wide margin separators between the hyperplanes in the previous equation.

The SVM classifier solves this problem by seeking to solve the optimization problem described by the following equation

$$\text{Minimise } L(W,W_0,\xi)=\frac{1}{2}\|W\|^2+C\sum_{i=1}^{N}\xi_i$$

Ce qui conduit à

$$\begin{cases} W^{\mathrm{T}}x+W_0\geq 1-\xi_i & \text{si } y_i\in 1 \\ W^{\mathrm{T}}x+W_0\leq -1+\xi_i & \text{si } y_i\in -1 \\ \text{et} \\ \xi_i\geq 0 \end{cases}$$

Definitions:

Define hyperplanes H_1 and H_2 such as:

$$W\cdot x_i+b\geq +1 \text{ if } y_i=+1$$

$$W\cdot x_i+b\leq -1 \text{ if } y_i=-1$$

H_1 and H_2 are planes such that:

$$H_1:\ W\cdot x_i+b=1$$

$$H_2:\ W\cdot x_i+b=-1$$

The points on the planes H_1 and H_2 are support vectors.

d^+ is the shortest distance from H to the nearest positive point of H_1.

d^- is the shortest distance from H to the nearest negative point of H_2.

The margin is defined by d^++d^-.

Maximizing the margin:

The aim is to design a classifier with as large a margin as possible. Recall that the distance from a point with coordinates (x_0,y_0) to a straight line with equation $Ax+By+c=0$ is given by

$$d(M_0,D)=\frac{|Ax_0+By_0+c|}{\sqrt{A^2+B^2}}$$

The distance between H and H_1 is given by:

$$d=\frac{|W\cdot x+b|}{\|W\|}=\frac{1}{\|W\|}$$

The distance between H_1 and H_2 is given by:

$$d(H_1, H_2) = \frac{2}{\|W\|}$$

In order to maximize the margin, we need to minimize $\|W\|$ with the condition that there are no data points between H_1 and H_2

$$i.e.: \ W \cdot x_i + b \geq +1, \text{ if } y_i = +1$$

$$W \cdot x_i + b \leq -1, \text{ if } y_i = -1$$

Maximizing the margin therefore amounts to minimizing $\|W\|$ under constraints

$$\begin{cases} \min \frac{1}{2}\|W\|^2 \\ \forall i \ \ y_i\left(W \cdot x_i + b\right) \geq 1 \end{cases}$$ Tapez une équation ici.

We must therefore determine W and b minimizing $\frac{1}{2}\|W\|^2$ (in order to maximize the power of generalization), under the constraints (separating hyperplane)

$$y_i\left[\left(W \cdot x_i\right) + b\right] \geq 1, i = 1, \cdots, n$$

Transformation of the Optimization problem

Lagrange multipliers method

$$\begin{cases} L(W, b, \alpha) = \frac{1}{2}\|W\|^2 - \sum_{i=1}^{n} \alpha_i \left\{\left(W \cdot x_i + b\right) y_i - 1\right\} \\ \qquad\qquad = \frac{1}{2}\|W\|^2 - \sum_{i=1}^{n} \alpha_i \left\{\left(W \cdot x_i + b\right) y_i\right\} + \sum_{i=1}^{n} \alpha_i \\ \forall i, \ \ x_i \geq 0 \end{cases}$$

- Dual problem.

$$\begin{cases} \underbrace{\max}_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \left(x_i x_j\right) \\ \forall i, \alpha_i \geq 0 \\ \sum_{i=1}^{n} \alpha_i y_i = 0 \end{cases}$$

Solution to the optimization problem:

$$\begin{cases} D(x) = \left(W^* x + b^*\right) \text{ with } *: \text{estimate} \\ W^* = \sum_{i=1}^{m} \alpha_i^* y_i x_i \\ W_0^* = y_s - \sum_{i=1}^{m} \alpha_i^* y_i \left(x_i \cdot x_s\right) \ \left(x_s y_s\right) \text{ being any point of the support} \end{cases}$$

3.3.4. KNN Classifier

It's a very simple and straightforward approach, requiring no training but simply the storage of training data. The principle is as follows: an unknown class of data is compared with all stored data. For the new data item, the majority class is chosen from among its K nearest neighbors (this can be a cumbersome process for large databases), in the sense of a chosen distance. In order to find the K nearest to the data to be classified, we can choose the Euclidean distance. Given two pieces

of data represented by two vectors X_i and X_j, the distance between these two pieces of data is given by:

$$d\left(X_i, X_j\right) = \sqrt{\sum_{k=1}^{d}\left(x_{ik} - x_{jk}\right)^2} \tag{38}$$

This involves implementing the *K*-nearest neighbor algorithm to predict the classes of new data from labeled data (training data).

Next, let's extend the code for the K-NN case for a value of $K \geq 1$.

3.3.5. Multiclassification

1) SCHAPIRE boosting algorithm

The algorithm starts with a training set T and the choice of using the induction technique. The Schapire Boosting algorithm is thus defined as follows: - A subset T1 selected at random from a sample of the training data is extracted from T. T1 is used to induce the first classifier C1. A further training subset T2 is created so that C1 obtains a 50% classification rate on it. From the subset thus created, the second classifier C2 is induced. As the two classifiers C1 and C2 have each been induced from different training data, they will inevitably differ in the way they label new data. A tie-breaker is therefore necessary.

To this end, a third subset of training data T3 is created, consisting solely of data on which C1 and C2 disagree. From this third subset T3, the third classifier C3 is induced; From this point on, apply majority voting for classification. A recursive implementation of the above algorithm can theoretically reduce the error rate significantly, since each additional level of recursivity can reduce the error. Assume that each induced classifier has an error rate below a certain ε. It has been proved that the error rate of the set resulting from the majority vote of the triplet of classifiers is less than 3ε 2-2ε 2, which is always less than ε [24]. The main difficulty is to find good training data for each of the training subsets. Another difficulty is that even if we succeed in creating all three learning subsets, it is impossible to apply the same principle recursively. Kubat has suggested that to be able to do this, we would need a rarely available and almost inexhaustible data source.

2) Implementing a multi-agent system using Schapire's Boosting approach

A multi-agent system can be defined as a group of agents working together synergistically to achieve a specific task. This is a typical illustration of a distributed system capable of partitioning a difficult task into a set of small tasks easily managed by a single agent, which can in fact be considered as an aggregate. This idea is along the same lines as Surowiecki's "Wisdom of Crowds". Surowiecki argues that, under certain controlled conditions, the aggregation of information from several sources often results in better decisions than could have been made by a single person, even an expert. Of course, not all crowds are wise. According to Surowiecki, to become wise, a crowd must meet the following criteria:

- Diversity of opinion: every member must have private information, even if it's just an eccentric interpretation of known facts;
- Independence: members' opinions are not determined by the opinions of

those around them; Decentralization: members can specialize and draw conclusions on the basis of local knowledge;

- Aggregation: there is a mechanism for transforming private judgments into a collective decision. The system we build in this article is a set of agents set up to perform EEG signal classification. For our multi-agent system to become wise, all four of Surowiecki's above-mentioned criteria must be satisfied.

3) Architecture of the Schapire Boosting multi-agent system

To provide a clear and reproducible description of the proposed multi-agent boosting framework, the overall procedure is summarized in Algorithm 1.

4) Proposed Multi-Agent Boosting Algorithm

Algorithm 1: Proposed Multi-Agent Boosting Framework

Input: EEG dataset T

(a) Split dataset into training, validation, and testing sets;

(b) Apply DWT to decompose EEG signals into sub-bands;

(c) Extract statistical features from each sub-band;

(d) Apply LDA for dimensionality reduction using training data only;

(e) Train base classifiers:

- Train CA1 - CA3 using MLP;
- Train CA4 - CA6 using SVM;
- Train CA7 - CA9 using KNN.

(f) Generate meta-training dataset from outputs of base classifiers;

(g) Train meta-classifiers:

- MA1 (MLP);
- MA2 (SVM);
- MA3 (KNN).

(h) Apply majority voting using meta-classifiers.

Output: Final classification result

The proposed algorithm provides a structured representation of the training and decision-making process of the multi-agent system, thereby improving clarity and reproducibility.

The proposed Schapire Boosting multi-agent system, illustrated in Figure X, consists of thirteen agents organized in a three-layer hierarchical architecture:

- A base layer composed of nine classifier agents (CA1 - CA9);
- An intermediate layer composed of three meta-classifier agents (MA1 - MA3);
- A final output layer consisting of a controller agent responsible for decision making.

At the base layer, the Schapire boosting strategy is implemented as follows:

- **First classifier agent (CA1)**: This agent implements a Multilayer Perceptron (MLP) model trained on a subset T1 randomly extracted from the training set T. The subset T1 is constructed by selecting 50% of the samples randomly, while the remaining 50% are generated using a data augmentation technique

applied to the selected samples. This approach increases data diversity and improves generalization.

- **Second classifier agent (CA2)**: This agent uses the same MLP-based induction technique with different parameter configurations. The training subset T2 is composed of 50% of samples correctly classified by CA1 and 50% randomly selected samples from T.
- **Third classifier agent (CA3)**: This agent is trained on a subset T3 composed of samples on which CA1 and CA2 disagree. Together, CA1, CA2, and CA3 form the first group of MLP-based classifiers.

Similarly, the second group of classifiers (CA4 - CA6) implements the SVM induction technique, while the third group (CA7 - CA9) implements the KNN induction technique.

The outputs generated by the nine base classifiers are then combined to form three new training subsets (T', T'', and T'''), which are used to train the meta-classifier agents MA1, MA2, and MA3. These subsets are constructed by applying a one-hot encoding strategy to the outputs of the base classifiers.

At the final stage, the controller agent aggregates the outputs of the meta-classifiers using a majority voting mechanism to produce the final classification decision.

The meta-classifiers MA1, MA2, and MA3 respectively implement the MLP, SVM, and KNN induction techniques.

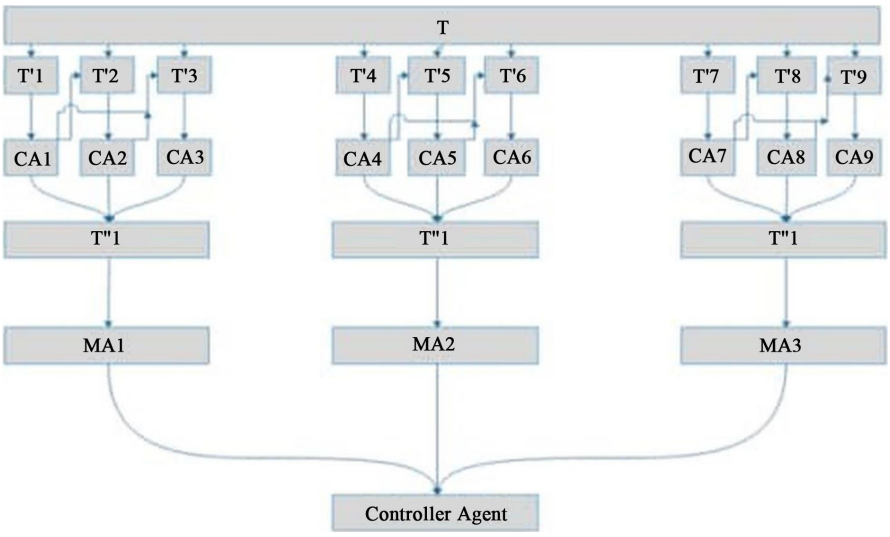


Figure 4. Schapire adaboost.

The AdaBoost (Adaptive Boosting) algorithm, introduced by Freund and Schapire (1997), is an ensemble learning method used to improve classification performance by combining several weak classifiers into a strong one. As illustrated in **Figure 4**, the algorithm assigns a weight w_i to each training sample.

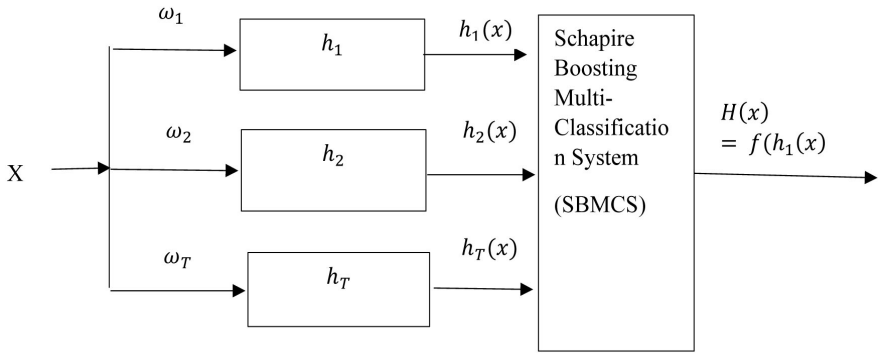
At each iteration, a weak classifier is trained on the weighted dataset. Misclassified samples receive higher weights in the next iteration, forcing subsequent clas-

sifiers to focus more on difficult cases. In contrast, correctly classified samples receive lower weights.

Each classifier is associated with a coefficient that reflects its accuracy. The final decision is obtained by combining all classifiers using a weighted voting mechanism, where more accurate classifiers have a greater influence on the final result.

This adaptive process allows AdaBoost to progressively reduce classification errors and improve the overall robustness of the model.

$$H(x) = sign\left(\sum_{i=1}^{T} \alpha_i H_i(x)\right)$$



Step 1: Creation of training subsets

- The training data will be available in packets X_k (k is the number of packets). At the start of training, only the X_1 data set is available, then only package X_2, up to package X_k.
- The system will train T_k classifiers, each of which is trained on a subset $X_k\left(t = \{1, \cdots, T_k\}\right)$ of X_k. To create these learning subsets, we assign a weight ω to each data, which will be used to define whether this data belongs to a subset.

$$\omega(k) = \frac{1}{m_k}$$

- A training subset X_{kt} is created by a weighted random selection of the data in set X_k. The higher the weight of a data item, the more likely it is chances of belonging to this training subset.

Step 2: Combining classifiers

- Each classifier is trained on the X_{kt} training subsets. Once a subset has been trained, we obtain an output h_{kt} (using a weak learning algorithm). which is the weak classifier. We then calculate the error ε_{kt} of this classifier by the formula

$$h_{kt} : \varepsilon_{kt} = \sum_{i=h_{kt}}^{T} x_i \neq y_i \omega_t(i)$$

The error is then tested: If the error $\varepsilon_{kt} > \frac{1}{2}$ we repeat the operation of randomly drawing from the learning basis and we remove the hypothesis, otherwise we continue the learning and calculate the normalized error as:

$$\alpha_{kt} = \frac{1}{2}\log\left(\frac{1-\varepsilon_{kt}}{\varepsilon_{kt}}\right)$$

Then we calculate the normalization factoras:

$$Z_t = 2\sqrt{\varepsilon_t\left(1-\varepsilon_t\right)}$$

Update weights of m individuals

$$\begin{cases} \text{if } h_i = h_i\left(x\right) \text{ then} \\ \omega_{i+1}\left(i\right) = \dfrac{\omega_t}{Z_t} * \begin{cases} \mathrm{e}^{-\alpha t} \text{ if } h_t\left(x_i\right) = y_i \\ \mathrm{e}^{\alpha t} \text{ if } h_t\left(x_i\right) \neq y_i \end{cases} \end{cases}$$

Return all h_i and $\alpha_t\left(t = \{1, \cdots, T\}\right)$.

The classification of a new individual X is based on the following equation.

$$H\left(x\right) = sign\left(\sum_{i=1}^{T} \alpha_i H_i\left(x\right)\right)$$

This being the decision of the weighted vote of the global classifier in each iteration.

4. Results

4.1. Results of the Diagnosis of Epilepsy in Mono Classification

4.1.1. Results Using the MLP

The reported performance metrics are computed based on confusion matrix values (TP, FP, FN, TN) for each classification task.

It should be noted that the sample sizes may vary depending on the classification scenario.

The term "global interpretation error" refers to the difference between the error rate obtained using individual classifiers and that obtained using the proposed ensemble system.

Calculating the performance metrics for each of the classification problems gives:

Lda_A - E

$$\begin{cases} \text{Sensibility } S_e\left(\%\right) = \dfrac{9}{9+0} * 100 = 100\% \\ \text{Specificity } S_p\left(\%\right) = \dfrac{9}{9+0} * 100 = 100\% \\ \text{Accuracy } Acc\left(\%\right) = \dfrac{9+9}{9+9+0+0} * 100 = 100\% \end{cases}$$

Lda_B - E

$$\begin{cases} \text{Sensibility } S_e\left(\%\right) = \dfrac{9}{9+1} * 100 = 90\% \\ \text{Specificity } S_p\left(\%\right) = \dfrac{8}{8+0} * 100 = 100\% \\ \text{Accuracy } Acc\left(\%\right) = \dfrac{9+8}{9+8+1+0} * 100 = 94.44\% \end{cases}$$

Lda_AB - CD

$$\begin{cases} \text{Sensibility } S_e(\%) = \dfrac{9}{9+0} * 100 = 100\% \\ \text{Specificity } S_p(\%) = \dfrac{9}{9+0} * 100 = 100\% \\ \text{Accuracy } Acc(\%) = \dfrac{9+9}{9+9+0+0} * 100 = 100\% \end{cases}$$

Lda_CD - E

$$\begin{cases} \text{Sensibility } S_e(\%) = \dfrac{9}{9+0} * 100 = 100\% \\ \text{Specificity } S_p(\%) = \dfrac{9}{9+0} * 100 = 100\% \\ \text{Accuracy } Acc(\%) = \dfrac{9+9}{9+9+0+0} * 100 = 100\% \end{cases}$$

With the Multilayer Perceptron Neural Network classifier, classification problems A - E, AB - CD, CD - E of the proposed model achieved a maximum classification rate of 100%. However, the B - E classification problem achieved a classification rate of 94.44%. These performances are summarized in Tables 1-10, Figure 5.

Table 1. Performance metrics of the MLP classifier.

Different Classifications	Sensitivity (%)	Specificity (%)	Accuracy (%)
A - E	100	100	100
B - E	90	100	94,44
CD - E	100	100	100
AB - CD	100	100	100

4.1.2. Results Using SVM

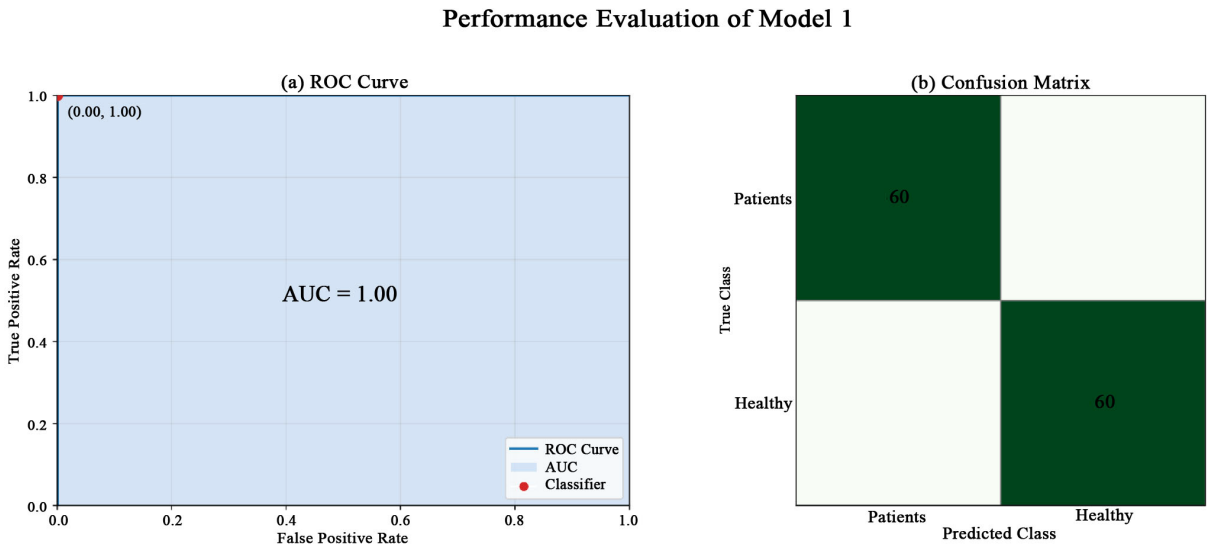


Figure 5. (a) ROC curve of Model 1 showing perfect classification (AUC = 1.00). (b) Confusion matrix indicating zero misclassification.

Lda A - E

$$\begin{cases} \text{Sensibility } S_e(\%) = \dfrac{60}{60+0} * 100 = 100\% \\ \text{Specificity } S_p(\%) = \dfrac{60}{60+0} * 100 = 100\% \\ \text{Accuracy } Acc(\%) = \dfrac{60+60}{60+60+0+0} * 100 = 100\% \end{cases}$$

Lda B - E

$$\begin{cases} \text{Sensibility } S_e(\%) = \dfrac{59}{59+1} * 100 = 98,33\% \\ \text{Specificity } S_p(\%) = \dfrac{60}{60+0} * 100 = 100\% \\ \text{Accuracy } Acc(\%) = \dfrac{59+60}{59+60+1+0} * 100 = 99,16\% \end{cases}$$

Lda CD - E

$$\begin{cases} \text{Sensibility } S_e(\%) = \dfrac{58}{58+2} * 100 = 96,66\% \\ \text{Specificity } S_p(\%) = \dfrac{59}{59+1} * 100 = 98,33\% \\ \text{Accuracy } Acc(\%) = \dfrac{58+59}{58+59+2+1} * 100 = 97,50\% \end{cases}$$

Lda AB - CD

$$\begin{cases} \text{Sensibility } S_e(\%) = \dfrac{59}{59+1} * 100 = 98.33\% \\ \text{Specificity } S_p(\%) = \dfrac{60}{60+0} * 100 = 100\% \\ \text{Accuracy } Acc(\%) = \dfrac{59+60}{59+60+1+0} * 100 = 99.16\% \end{cases}$$

Table 2. Performance metrics of the SVM classifier.

Different Classifications	Sensitivity (%)	Specificity (%)	Accuracy (%)	ROC Curve
A - E	100	100	100	1.00
B - E	98.33	100	99.16	1.00
CD - E	96.66	98.33	97.50	0.99
AB - CD	98.33	100	99.16	0.98

4.1.3. Results Using KNN

Table 3. Performance metrics of the KNN classifier.

Different Classifications	Sensitivity (%)	Specificity (%)	Accuracy (%)
A - E	100	100	100
B -E	100	100	100
CD - E	96.06	95	95.83
AB - CD	98.33	98.33	98.33

4.2. Results of the Diagnosis of Epilepsy in Multiclassification

4.2.1. Results Using MLP for Training as Classifier Agent

Table 4. Architecture parameters using MLP.

Classifier Agent	Number of Hidden layers	Activationn function	Solver
CA1	100	Hyberbolic tangent	Sgd
CA2	100	Logistic	Sgd
CA3	200	Hyperbolic tangent	Sgd

Table 5. Metric performances of Epilepsy classification with MLP agent classifier.

Classifier agent	TP	FP	FN	TN	S_e (%)	S_p (%)	Acc_Voting (%)
CA1	62	22	38	78	61.10	77.80	69.45
CA2	100	44	0	56	100	55.60	77.80
CA2	67	0	33	100	66.70	100	83.30

4.2.2. Results Using SVM for Training of Epilepsy Classification with SVM Agent Classifier

Table 6. Architecture parameters using SVM.

Classifier agent	Nucleus	C	Degre
CA4	Polynomial	1	3
CA5	Polynomial	1	4
CA6	Polynomial	0.5	3

Table 7. Metric performances of Epilepsy classification with SVM agent classifier.

Classifier agent	TP	FP	FN	TN	S_e (%)	S_p (%)	Acc_Voting (%)
CA4	100	0	0	100	100	100	100
CA5	100	0	0	100	100	100	100
CA6	100	0	0	100	100	100	100

4.2.3. Results Using KNN for Training

Table 8. Architecture parameters using KNN.

Classifier agent	Number of Neighbors K
CA7	5
CA8	10
CA9	15

Table 9. Metric performances of Epilepsy classification with KNN agent classifier.

Classifier agent	TP	FP	FN	TN	S_e (%)	S_p (%)	Acc_Voting (%)
CA7	77	0	33	100	77.80	100	88.50
CA8	89	0	22	100	88.90	100	94.00
CA9	100	0	0	100	100	100	100

Table 10. Synthesis of metric performances of our Intelligent system using majority voting.

SMA_BS	TP	FP	FN	TN	S_e (%)	S_p (%)	Acc_Voting (%)
MA1	100	0	0	100	100	100	100
MA2	100	0	0	100	100	100	100
MA3	100	0	0	100	100	100	100

In view of the performance achieved by the MLP, SVM and KNN agent-classifiers in multi-classification, it turns out that these performances can again be improved. In that view, we have implemented meta-classifiers with MLP, SVM and KNN. Thus, for training meta-classification systems we selected datasets A and E from the online database of the University of Bonn. The meta-classifiers MA1, MA2 and MA3 are therefore proposed. These meta-classifiers are trained using the MLP, SVM and KNN induction techniques proposed in mono-classification, respectively. For each meta-classifier, the three agents proposed in multi-classification are simultaneously implemented, and the majority voting algorithm is implemented by a fourth agent called the controller agent. The performance of the proposed meta-classification systems, called Schapire Boosting Multi-Agent Systems (SMA-BS) are summarized in table 10.

5. Discussion

The results obtained in this study demonstrate a significant improvement in the classification of epileptic and non-epileptic EEG signals. One of the main advantages of the proposed model lies in the use of a majority voting mechanism, which aggregates the predictions of multiple classifiers to produce a final decision. This approach enhances robustness, as no single classifier can consistently achieve optimal performance across all classification tasks due to the inherent complexity of EEG signal patterns.

The proposed Schapire Boosting Multi-Agent System (SMA-BS) presents several key advantages. First, it enables progressive learning by emphasizing difficult-to-classify samples during successive training iterations. Second, the combination of meta-classifiers significantly improves the overall performance compared to individual classifiers. In most cases, the ensemble achieves a lower error rate than the best standalone classifier; in other cases, it performs at least comparably.

From a practical perspective, this implies that in challenging classification sce-

narios, the use of an ensemble-based approach is preferable to relying on a single classifier. The resulting system can therefore be considered as an effective decision-support tool capable of providing reliable diagnostic suggestions to neurologists.

Furthermore, the proposed architecture can be implemented in a parallel or multi-processor environment, which may reduce training time and improve computational efficiency. The boosting strategy builds a strong classifier by combining multiple weak learners, allowing improved generalization and feature discrimination.

The effectiveness of boosting relies on the adaptive weighting of training samples, where misclassified instances are given higher importance in subsequent iterations. This mechanism enables the model to focus on difficult patterns and progressively improve its performance.

In terms of computational complexity, the algorithm exhibits linear complexity with respect to the number of samples during inference, while the training phase may reach quadratic complexity depending on the number of iterations. Despite this, the performance gains obtained justify the additional computational cost in many practical applications (Table 11).

Table 11. Comparative study.

Authors/ Year	Feature Extraction	Classification	Dataset	Accuracy (%)	Acc_Voting (%)
Subasi & Gursoy [25]	DWT + PCA + ICA + LDA	SVM	A - E	100	NI
Ubeyli [26]	Least Squares Features	LS-SVM	A - E	100	NI
Acharya *et al.* [27]	DWT + PCA	ANN	A - E	98.1	NI
Tzallas *et al.* [10]	FFT	ANN	A - E	97 - 100	NI
Wang *et al.* [12]	Time + Frequency + Nonlinear	SVM	A - E	99+	NI
Guo *et al.* [13]	DWT + Feature Selection	RF, SVM, MLP (Ensemble)	A - E	97.6	YES
Al-Hadeethi *et al.* [14]	DWT + Covariance	LS-SVM + Boosting	A - E	99+	YES
Bhandage *et al.* [18]	Spectrogram (CWT/STFT)	CNN (Deep Learning)	A - E	99.24	NI
	DWT + LDA	SVM	A - E	100	NI
	DWT + LDA	MLP	A - E	100	NI

Continued

DWT + LDA	KNN	A - E	100	NI
DWT + LDA	SVM + Boosting	A - E	100	100
DWT + LDA	MLP + Boosting	A - E	83.3	100
DWT + LDA	KNN + Boosting	A - E	100	100

6. Conclusion

In this study, a robust automated system for epilepsy detection based on EEG signal analysis has been proposed. The approach combines Discrete Wavelet Transform (DWT) for signal decomposition and Linear Discriminant Analysis (LDA) for dimensionality reduction with an intelligent multi-agent classification framework. The proposed architecture integrates three complementary machine learning classifiers, namely Multilayer Perceptron (MLP), Support Vector Machine (SVM), and K-Nearest Neighbors (KNN), organized within a Schapire Boosting–based multi-agent system (SMA-BS).

The experimental results obtained on the Bonn EEG dataset demonstrate that the proposed ensemble framework significantly improves classification robustness compared with individual classifiers operating in mono-classification mode. The SMA-BS system achieved a classification accuracy of 100% with a very low global error rate of 0.130, indicating a strong capability to discriminate between epileptic and non-epileptic EEG signals.

Furthermore, a comparative analysis with several state-of-the-art methods reported in the literature shows that the proposed approach performs competitively while maintaining a relatively simple and interpretable architecture. By combining signal processing techniques with ensemble machine learning strategies, the proposed system provides an effective decision-support tool for EEG interpretation.

Overall, the results highlight the potential of intelligent ensemble systems for improving automated epilepsy detection and assisting neurologists in the clinical analysis of EEG signals. Future work will focus on extending the proposed framework to larger and more heterogeneous EEG datasets, as well as integrating deep learning and real-time seizure detection mechanisms to further enhance the clinical applicability of the system.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] World Health Organization (2023) Epilepsy. https://www.who.int/health-topics/epilepsy

[2] Fisher, R.S., Acevedo, C., Arzimanoglou, A., Bogacz, A., Cross, J.H., Elger, C.E., *et al*. (2014) ILAE Official Report: A Practical Clinical Definition of Epilepsy. *Epilepsia*, **55**, 475-482. https://doi.org/10.1111/epi.12550

[3] Siuly, S., Li, Y. and Zhang, Y.C. (2016) EEG Signal Analysis and Classification. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, **11**, 141-144.

[4] Ahsan, A. and Siddiqi, S. (2023) Diagnosis and Prognosis: Literature Review on Prediction of Epilepsy Using Machine Learning Techniques. *International Journal of Computer Applications*, **185**, 10-29.

[5] Faust, O., Acharya, U.R., Min, L.C. and Sputh, B.H.C. (2010) Automatic Identification of Epileptic and Background EEG Signals Using Frequency Domain Parameters. *International Journal of Neural Systems*, **20**, 159-176. https://doi.org/10.1142/s0129065710002334

[6] Lima, C.A.M., Coelho, A.L.V. and Eisencraft, M. (2010) Tackling EEG Signal Classification with Least Squares Support Vector Machines: A Sensitivity Analysis Study. *Computers in Biology and Medicine*, **40**, 705-714. https://doi.org/10.1016/j.compbiomed.2010.06.005

[7] Acharya, U.R., Sree, S.V., Chattopadhyay, S., Yu, W. and Ang, P.C.A. (2011) Application of Recurrence Quantification Analysis for the Automated Identification of Epileptic EEG Signals. *International Journal of Neural Systems*, **21**, 199-211. https://doi.org/10.1142/s0129065711002808

[8] Guerrero-Mosquera, C. and Navia-Vázquez, A. (2012) Automatic Removal of Ocular Artefacts Using Adaptive Filtering and Independent Component Analysis for Electroencephalogram Data. *IET Signal Processing*, **6**, 99-106. https://doi.org/10.1049/iet-spr.2010.0135

[9] Rajendra Acharya, U., Vinitha Sree, S., Alvin, A.P.C. and Suri, J.S. (2012) Use of Principal Component Analysis for Automatic Classification of Epileptic EEG Activities in Wavelet Framework. *Expert Systems with Applications*, **39**, 9072-9078. https://doi.org/10.1016/j.eswa.2012.02.040

[10] Tzallas, A.T., Tsipouras, M.G. and Fotiadis, D.I. (2009) Epileptic Seizure Detection in EEGs Using Time-Frequency Analysis. *IEEE Transactions on Information Technology in Biomedicine*, **13**, 703-710. https://doi.org/10.1109/titb.2009.2017939

[11] Liu, Y., Lin, C., Tsai, J. and Sun, Y. (2013) Model-Based Spike Detection of Epileptic EEG Data. *Sensors*, **13**, 12536-12547. https://doi.org/10.3390/s130912536

[12] Wang, L., Xue, W., Li, Y., Luo, M., Huang, J., Cui, W., *et al*. (2017) Automatic Epileptic Seizure Detection in EEG Signals Using Multi-Domain Feature Extraction and Nonlinear Analysis. *Entropy*, **19**, Article 222. https://doi.org/10.3390/e19060222

[13] Guo, Y., Zhang, Y., Mursalin, M., Xu, W. and Lo, B. (2018) Automated Epileptic Seizure Detection by Analyzing Wearable EEG Signals Using Extended Correlation-Based Feature Selection. 2018 *IEEE* 15*th International Conference on Wearable and Implantable Body Sensor Networks* (*BSN*), Las Vegas, 4-7 March 2018, 66-69. https://doi.org/10.1109/bsn.2018.8329660

[14] Al-Hadeethi, H., Abdulla, S., Diykh, M., Deo, R.C. and Green, J.H. (2020) Adaptive Boost LS-SVM Classification Approach for Time-Series Signal Classification in Epileptic Seizure Diagnosis Applications. *Expert Systems with Applications*, **161**, Article ID: 113676. https://doi.org/10.1016/j.eswa.2020.113676

[15] Srinivasan, S., Dayalane, S., Mathivanan, S.K., Rajadurai, H., Jayagopal, P. and Dalu, G.T. (2023) Detection and Classification of Adult Epilepsy Using Hybrid Deep Learning Approach. *Scientific Reports*, **13**, Article No. 17574.

https://doi.org/10.1038/s41598-023-44763-7

[16] Varlı, M. and Yılmaz, H. (2023) Multiple Classification of EEG Signals and Epileptic Seizure Diagnosis with Combined Deep Learning. *Journal of Computational Science*, **67**, Article ID: 101943. https://doi.org/10.1016/j.jocs.2023.101943

[17] Wijayanto, I., Humairani, A., Hadiyoso, S., Rizal, A., Prasanna, D.L. and Tripathi, S.L. (2023) Epileptic Seizure Detection on a Compressed EEG Signal Using Energy Measurement. *Biomedical Signal Processing and Control*, **85**, Article ID: 104872. https://doi.org/10.1016/j.bspc.2023.104872

[18] Bhandage, V., Pokuri, T., Desai, D. and Jeyabose, A. (2024) Detection of Epilepsy Disorder Using Spectrogram Images Generated from Brain EEG Signals. *IEEE Access*, **12**, 195054-195064. https://doi.org/10.1109/access.2024.3520861

[19] Wei, L. and Mooney, C. (2024) An EEG-Based Automatic Classification Model for Epilepsy with Explainable Artificial Intelligence. 2024 14*th International Conference on Biomedical Engineering and Technology*, Seoul, 14-17 June 2024, 44-50. https://doi.org/10.1145/3678935.3678943

[20] Andrzejak, R.G., Lehnertz, K., Mormann, F., Rieke, C., David, P. and Elger, C.E. (2001) Indications of Nonlinear Deterministic and Finite-Dimensional Structures in Time Series of Brain Electrical Activity: Dependence on Recording Region and Brain State. *Physical Review E*, **64**, Article ID: 061907. https://doi.org/10.1103/physreve.64.061907

[21] Chandani, M. (2017) Statistical Wavelet Features, PCA, MLPNN, SVM and K-NN Based Approach for the Classification of EEG Physiological Signal. *International Journal of Industrial and Manufacturing Systems Engineering*, **2**, 57-65. https://doi.org/10.11648/j.ijimse.20170205.12

[22] Vapnik, V.N. (1999) An Overview of Statistical Learning Theory. *IEEE Transactions on Neural Networks*, **10**, 988-999. https://doi.org/10.1109/72.788640

[23] Boser, B.E., Guyon, I.M. and Vapnik, V.N. (1992) A Training Algorithm for Optimal Margin Classifiers. *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, Pittsburgh, 27-29 July 1992, 144-152. https://doi.org/10.1145/130385.130401

[24] Kubat, M. (2017) An Introduction to Machine Learning. 2nd Edition, Springer.

[25] Subasi, A. and Ismail Gursoy, M. (2010) EEG Signal Classification Using PCA, ICA, LDA and Support Vector Machines. *Expert Systems with Applications*, **37**, 8659-8666. https://doi.org/10.1016/j.eswa.2010.06.065

[26] Übeyli, E.D. (2010) Least Squares Support Vector Machine Employing Model-Based Methods Coefficients for Analysis of EEG Signals. *Expert Systems with Applications*, **37**, 233-239. https://doi.org/10.1016/j.eswa.2009.05.012

[27] Acharya, U.R., Molinari, F., Sree, S.V., Chattopadhyay, S., Ng, K. and Suri, J.S. (2012) Automated Diagnosis of Epileptic EEG Using Entropies. *Biomedical Signal Processing and Control*, **7**, 401-408. https://doi.org/10.1016/j.bspc.2011.07.007