Pattern Recognition Letters 128 (2019) 544–550

Contents lists available at ScienceDirect

Pattern Recognition Letters

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

Auto-detection of epileptic seizure events using deep neural network with different feature scaling techniques

Thara D.K.[a,*], PremaSudha B.G[b], Fan Xiong[c]

[a] Assistant Professor, Dept of ISE, CIT, Tumakuru, Karnataka, India
[b] Research Guide, Dept of CSE, SIT, Tumakuru, Karnataka, India
[c] Electrical Engineer, Life Science Group, Bio-Rad Laboratories, CA, United States

ARTICLE INFO

Article history:
Received 7 September 2019
Revised 22 October 2019
Accepted 26 October 2019
Available online 31 October 2019

Keywords:
Deep neural network
Epilepsy
Seizure
Feature scaling
Loss

ABSTRACT

Misdiagnosis of epilepsy is more seen in manual analysis of electroencephalogram (EEG) signals for epileptic seizure event detection. Therefore, automated systems for epilepsy detection are required to help neurologists in diagnosing epilepsy. These automated systems act as supporting systems for the neurologists to diagnose epilepsy with good accuracy in less time. In this paper an attempt is made to develop an automated seizure detection method using deep neural network using the dataset collected from Bonn University, Germany. The results of the experiment are compared with the existing machine learning method. Our model gives better results compared to ML methods without the need of feature extraction. It is important to perform normalization of the dataset using feature scaling techniques to obtain good accuracy in the results. In this experiment we also worked on feature scaling of the dataset. At first we tried using StandardScaler and calculated loss using mean squared error. For this we achieved an accuracy of 97.21%, Sensitivity 98.17%, Specificity 94.93%, F1_score 98.48%, MCC 91.96% and ROC 97.55%. Experiment was continued to compare the performance of four different feature scaling techniques and four different loss functions. From the experimental results it was observed that StandardScaler and RobustScaler are equally good and are the best feature scaling techniques. Loss computed using Mean squared error works better in combination with all feature scaling techniques.

1. Introduction

Epilepsy is a chronic neurological disorder caused by unprovoked recurrent seizures [1,2]. A seizure is a sudden rush of electrical signals exchanged inside the brain. This electrical activity is caused by the complex chemical changes inside the nerve cells. The nerve cells are composed of positive ions (sodium, potassium & calcium) and negative ions (chlorine). The pumping of these ions from the nerve cells generates electrical signal. Normally the amplitudes of these electrical signals lie within 10 to 100 µV [3], whereas during seizure, the amplitudes range from 0.5 to 1.5 mV and may also rise up to several millivolts during spikes. Seizures are broadly of two types, focal seizure and generalized seizure. Focal seizure affects only one location of the brain. Person affected from focal seizure can maintain consciousness, may experience muscle contractions followed by relaxations. In case of generalized seizure, entire brain gets affected and is considered to be the most dangerous type of seizure. The person may lose consciousness, lose balance and fall, frothing from the mouth, Eyes rolling upwards, Tongue biting, lip smacking etc.

Most commonly used tool for diagnosing epilepsy is electroencephalogram (EEG) [4], in short called as EEG. EEG [5] records the electrical activity happening inside the brain. EEG is recorded by placing electrodes on the scalp of the patient. Electrodes can be placed either intracranial or extra cranial. Intracranial is, placement of electrodes inside the scalp by making an incision and hence it is invasive. Whereas extra cranial is placement of electrodes outside the scalp and hence it is non-invasive. In traditional approach, neurologists analyze EEG recordings manually to diagnose epilepsy. Manual method is very laborious and highly time consuming. Percentage of misdiagnosis is more in case of traditional approach. Developing efficient automatic methods for detecting epilepsy is crucial to help the life of epileptic patients. Around 14 people per 1000 population are suffering from epilepsy in countries like India. This may be due to the increased risk of endemic conditions like malaria, increased number of road accidents; injuries at the time of

* Corresponding author.
E-mail address: tarakmurthy10@gmail.com (T. D.K.).

https://doi.org/10.1016/j.patrec.2019.10.029
0167-8655/© 2019 Elsevier B.V. All rights reserved.

birth; and not having well equipped medical infrastructure. Nearly 80% of the people are suffering from epilepsy in developing countries like INDIA. All over the world around 2.4 million people are diagnosed with epilepsy. Around 4 to 10 people are detected with epilepsy for every 1000 people.

1.1. Related work

So far in the literature, lots of experiments are carried out to automate seizure event detection. For example, Nipun Dilesh Perera et al. [6] tried to develop a patient specific seizure detection system using discrete wavelet Packet Transform to extract wavelet packet energy using a single channel EEG to train SVM classifier in offline. They obtained sensitivity of 87.1% to 90.16% and specificity close to 100% for a total of 12 patients. Two more works on EEG signal classification using support vector machine classifier are: Chun Chen et al. [7] used Support Vector Machine for classification purpose and wavelet transform for feature extraction for EEG detection and got an accuracy of 97.62%, Abdulkadir Sengur et al. [8] have also used Support Vector Machine for classification purpose and time frequency texture descriptors as a feature extraction technique for epileptic seizure detection. They ended with the classification accuracy of 90.3%. Suvadeep Bose et al. [9] have used Discrete Wavelet technique for feature extraction and Random Forest for classifying seizure events from non-seizure events. In most of these research works, it can be observed that EEG seizure detection using machine learning classifier requires feature extraction technique. For every classifier a sophisticated feature extraction technique needs to be identified. This is the major drawback in machine learning. On the other hand deep learning techniques learn features from data as they do not require separate feature extractors [10]. Also in case of machine learning, classifiers require less training time and more testing time. On the other hand deep learning techniques take more training time. But once they are trained, they take less testing time. In this paper an attempt is made to build a method for automated detection of epileptic seizure events using deep neural network.

Arunkumar et al. [11], have made a remarkable contribution in the field of Epilepsy research. They have done a wonderful team work to automate the detection of focal EEG. Their paper systematically describes the methods employed for auto detection of focal and non-focal EEG. They used Bern Barcelona database for their work. They extracted 5 entropy features from the database and fed the features to 5 different classifiers (Naïve Bayes, Radial Basis function, Support vector machines, K nearest neighbor and Non-Nested Generalized Exemplars). Non-Nested Generalized Exemplars classifier outperformed all the remaining classifiers with a highest classification accuracy of 98%, sensitivity of 100% and specificity of 96%. This work gave us the motivation to take up Epilepsy subject for research. This paper will be the base paper for our entire research work. So far, among all the related works that are gone through, this is the work showing amazing performance in identifying focal EEG with a maximum computation time of 0.054 s. The paper gives clear explanation of the way the work is carried out. I have used their implementation in my research work. Their approach is very well documented in the article [11] and the implementation results are excellent even on my EEG dataset.

The authors have made a significant contribution to improve the life of focal epilepsy patients by developing a handy tool to analyze focal EEG. Misdiagnoses or late diagnoses create large impairment in the life of epilepsy patients. Many times neurologists fail to identify exact location of the brain affected by seizures. The method identified by Arunkumar and team acts as a supporting system for the neurologists in analyzing focal EEG. Inspired by the work done by Arunkumar and team, we have taken up Epilepsy as a subject for our research.

1.2. Contribution

In this work, we applied deep neural networks experimenting different feature scaling techniques for EEG data. Deep neural network is equipped with strong learning capabilities. Its large set of functions makes it possible to learn the differences among seizure and non-seizure events effectively and also to handle the non-stationary and non-linear property of EEG signals. The objectives of this paper are:

(1) To re execute the earlier work using PyEEG [12] and perform classification of EEG data on the same database using machine learning classification algorithms.
(2) To develop auto detection method using deep neural network and show that it performs better for classification of EEG data when compared to machine learning techniques using the same dataset used in objective 1.
(3) Extend the experiment to demonstrate the performance of four different feature scaling techniques and loss functions.

The paper is organized as follows: Section 2 presents the details of materials and methods used for this experiment. It includes details of the dataset, feature scaling method, loss functions available and proposed deep neural network method. Section 3 gives results and discussions followed by conclusion in Section 4.

2. Materials and methods

This section gives the details of the dataset used for the experiments, describes different feature scaling techniques used for standardizing the dataset and various loss functions to estimate the loss value.

2.1. Dataset

EEG recording is done by placing electrodes on the scalp of the patient. The electrodes are placed on the basis of 10–20 international system from nasion to inion as shown in the Fig. 1 below. Since it is difficult to collect EEG seizure event data from the patient directly in the neurocentre, therefore the same database used in [12] is used in our experiment.

The EEG dataset used in this experiment is collected from University Bonn, Germany. The data was sampled at the rate of 173.61 Hz. The spectral bandwidth ranges from 0.5 Hz to 85 Hz. The original dataset is a bunch of five sets, set A, set B, set C, set D and set E, each set representing the data of a single person. Each set is the recording of the electrical activity inside the brain for the duration of 23.6 s. Each set in the dataset consists of 100 channels, each channel having time series 4097 data points. Set A and B consists of EEG recorded from 5 healthy volunteers, Set C and D consists of intracranial EEG recording during inter-ictal periods. Set E consists of intracranial EEG recorded from an epileptic patient during ictal period.

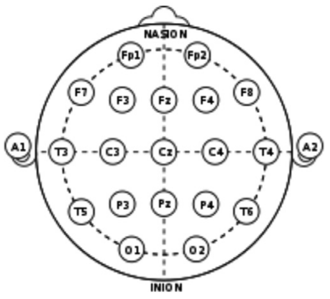

Fig. 1. Montages.

Forrest Sheng Bao et al. [12] used PyEEG framework, an open source python module to extract EEG features. They extracted power spectral density, fractal dimensions and entropies from EEG dataset. In this work the same work is reproduced by extracting the mentioned EEG features. Further we used support vector machine and k nearest neighbor algorithms for classifying seizure events from non-seizure events. This work gave the classification accuracy of 94% and 79.4% respectively for SVM and KNN respectively.

For the proposed experiment, the EEG recording of 1 s duration is collected from each set. The collected dataset is restructured such that the restructured dataset is a vector consisting of 179 attributes. The last column (179th column) represents the class label. Thus the resulting vector is a multivariate time series dataset with 179 columns and 11500 instances with no missing values. Deep neural network is used to classify seizure events from non-seizure events. The proposed approach resulted with the classification accuracy of 97.21%.

To obtain better results performed feature scaling of the dataset is performed. The next section describes different feature scaling techniques used to standardize the data and different loss functions used to calculate error in each epoch.

2.2. Feature scaling

Feature scaling also called standardization is a step during data pre-processing. It is performed to normalize the data within a particular range. Feature scaling helps in accelerating the calculations in the algorithm. Feature scaling of the data is the common requirement for the experiments carried out using Keras, Scikit learn and deep learning. The dataset used in this experiment contains variables that are different in scale. Therefore, feature scaling of the dataset is performed to change the feature vector into the format that is more suitable for deep learning techniques. There are many different scalers available for feature scaling of the dataset. Most commonly used ones are StandardScaler(), MinMaxScaler(), Normalizer() and RobustScaler().

StandardScaler() [13] transforms the dataset such that the resulting distribution's mean value is zero and the standard deviation is one. Transformed value is obtained by subtracting mean value from the original value and dividing by the standard deviation. The formula given below is used for the transformation.

$$z = \frac{x - \mu}{\sigma} \tag{1}$$

Where, z is the transformed value of the feature

x is the original value
μ is the mean
σ is the standard deviation

x is the value of the features in the dataset (11500 rows and 178 columns) used in our experiment. The value ranges from −1500 to +1500 which is the voltage level in micro volts.

MinMaxScaler () [14] scales the data such that all the values in the dataset fall between 0 and 1.

$$t = \frac{X - X\min}{X\max - X\min} \tag{2}$$

Where, t is the transformed value of the feature

X is the original value
Xmin, Xmax are the minimum and maximum values of the feature x.

RobustScaler () [13] removes the median and scales the data according to the quartile range which ranges from 25th quartile to 75th quartile. The transformed values of the dataset are comparatively larger than the previous scalers. The values in the dataset are transformed to fall between the range [−2, 3]. This is similar to minmaxscaler but uses inter quartile range instead of min max

$$t = \frac{x_i - Q_1(x)}{Q_3(x) - Q_1(x)} \tag{3}$$

Where, t is the transformed value of the feature

x is the original value
$Q_1(x)$ and $Q_3(x)$ are the inter quartile range

Normalizer() [14] scales data for each sample to a unit norm. This scaling technique scales both NumPy arrays and SciPy sparse matrix. For example, if the feature values were x, y and z then the transformed normalized value for x would be as shown below:

$$z = \frac{x_i}{\sqrt{x_i^2 + y_i^2 + z_i^2}} \tag{4}$$

2.3. Loss functions

Our model is trained using stochastic gradient descent algorithm. Each time it is evaluated, it exhibits some amount of error. Error is nothing but the difference between the actual target value and the predicted value. It is also called loss. Loss has to be estimated at each current state of the model. For this an appropriate loss function should be identified to estimate the loss value. Weights and biases must be updated at each evaluation to reduce the loss value in the next evaluation.

Four main loss functions that are widely used are:

Mean squared error [12]: It is sum of the squared difference between the actual target value and the predicted value. It can be used as 'mse' or 'mean squared error' in Keras at the time of compiling the model.

$$MSE = \frac{\sum_{i=1}^{n} \left(y_i - y_i^p\right)^2}{n} \tag{5}$$

where, y_i and y_i^p are the actual and predicted target values respectively.

Mean absolute error [12]: It is the sum of absolute value of the difference between the actual target value and the predicted value. It can be used as 'mae' or 'mean absolute error' in Keras at the time of compiling the model.

$$MAE = \frac{\sum_{i=1}^{n} \left|y_i - y_i^p\right|}{n} \tag{6}$$

where, y_i and y_i^p are the actual and predicted target values respectively.

Logcosh [14] It is the loss function based on the logarithm of the hyperbolic cosine function of the difference between actual target value and the predicted value. It can be used as 'logcosh' in Keras at the time of compiling the model.

$$L(y, y^p) = \sum_{i=1}^{n} \log\left(\cosh\left(y_i^p - y_i\right)\right) \tag{7}$$

where, y_i and y_i^p are the actual and predicted target values respectively.

Binary cross entropy [15]: It is the loss function which identifies the loss based on the probability value. The probability value lies between 0 and 1. A perfect model with zero loss will have probability value 0. Below given is the formula used for binary classification.

$$BCE = -(y\log(p) + (1 - y)\log(1 - p)) \tag{8}$$

The next section describes the details of the proposed deep neural network model.

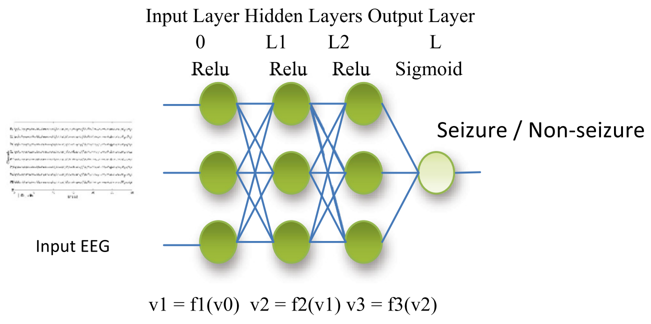
Input Layer Hidden Layers Output Layer
0 L1 L2 L
Relu Relu Relu Sigmoid
Seizure / Non-seizure
Input EEG
v1 = f1(v0) v2 = f2(v1) v3 = f3(v2)

Fig. 2. Architecture of Deep neural network.

2.4. Proposed deep neural network model

An important component of machine learning algorithms is feature engineering. They require carefully designed features by the domain experts. One of the main reasons for the success of deep learning technique is its ability to learn high level representations directly from the data without much involvement of the domain experts. Unlike neural network, deep neural network comprises of multiple non linear hidden layers. Each layer receives the output of the previous layer and learns the data progressively from bottom to top.

In case of proposed DNN model, the network performs its own data preprocessing and feature extraction is not required. The relevant information passes through multiple layers in the network. The network is self directed for the relevant data analysis. The proposed network consists of four layers including input and output layer. The activation function Relu is used on the first three layers and since this is a binary classification problem, sigmoid function is used on the output layer and the same is depicted in Fig. 2.

The step by step representation of the reproduced machine learning model is shown in Fig. 3 and the proposed DNN model in Fig. 4. EEG data from the University of Bonn is used as input for both the models. In case of machine learning model, PyEEG is used for extracting features from the input dataset. Five features namely, Detrended Fluctuation Analysis, Higuchi Fractal Dimension, SVD_Entropy, Fisher Information and Petrosian Fractal Dimension values are extracted using the functions, pyeeg.dfa(), pyeeg.hfd(), pyeeg.svd_entropy(), pyeeg.fisher_info() and pyeeg.pfd() from pyeeg module. The extracted feature values are stored in the form of a matrix. 67% of the rows are used for train set and 33% of rows are used for test set. Feature matrix is fed as input to classification algorithms SVM and KNN and ended with the classification accuracy of 94% and 79.7% respectively.

If a DNN consists of L layers, where, the input and output layers are 0 and L respectively. The output vector v_l of the lth layer is computed using the following equation.

$$v_l = f\left(z^l\right) = f\left(W^l v_{l-1} + b^l\right) \tag{9}$$

Where, v_{l-1} is the output vector of the previous layer.

v_l is the output vector of the present layer
W^l is the weight matrix
b^l is the bias vector.

The proposed method is implemented using deep neural network model in Jupyter notebook using python 3.5, Tensorflow, Keras and Scikit learn. This is a binary classification problem in which all seizure events in the dataset belong to class 0 and remaining all events belongs to class 1. The dataset is divided into test set consisting of 2300 instances and train set consisting of 9200 instances. Feature scaling of the dataset is performed with StandardScaler. In Keras the first layer is the first hidden layer. Batch size was set to 10. The layer 1 of the model takes input array of shape (*, 178) and outputs array of shape (10, 80). The structure of the designed model is given below:

Layer info	Output size	Param#
Dense1	(10, 80)	14320
Dense2	(10, 80)	6480
Dense3	(10, 80)	6480
Desne4	(10, 1)	81

Total Parameters: 27361
Trainable parameters: 27361
Non-trainable parameters: None

The model runs with the activation function "Relu" on hidden layers as shown in Fig. 3. The equation used for this is:

$$R(x) = \max(0, x) \tag{10}$$

And on the output layer the activation function "sigmoid" is used as shown below:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{11}$$

The learning process of the model should be configured before training the model. This is done using compile method. The model is compiled using optimizer function "adam" and loss using "mean squared error". Adam is the most efficient optimizer algorithm with adaptive learning rate (computing individual learning rate for each parameter) which is specially designed for training deep neural networks. After compilation, the model is trained. The training instances are passed as input to the model with batch size 10 and epochs 150. The test set is then passed to the model for detecting seizure events. The complete workflow of the experiment is depicted in Fig. 5.

3. Results and discussions

The results of the model are evaluated and printed using the following performance parameters shown in Table 1. Feature scaling of the dataset is performed using StandardScaler and loss is calculated using Mean squared error.

Confusion matrix is computed using Scikit learn and plotted using Matplotlib (Fig. 6). Confusion matrix consists of the following values:

True positive	*False negative*
False positive	*True negative*

The values obtained from the model are shown below:

```
[[1828 26]
[38 408]]
```

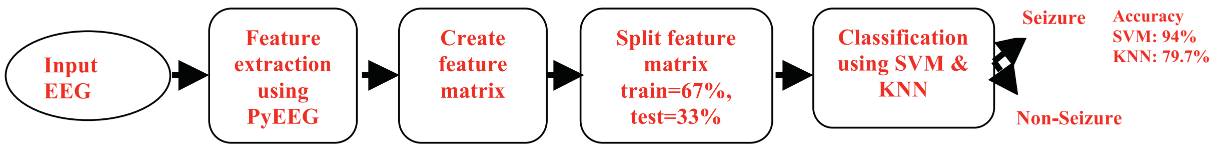
Input EEG
Feature extraction using PyEEG
Create feature matrix
Split feature matrix train=67%, test=33%
Classification using SVM & KNN
Seizure
Non-Seizure
Accuracy
SVM: 94%
KNN: 79.7%

Fig. 3. Based on earlier work: Automatic seizure detection using PyEEG and Machine learning algorithms.

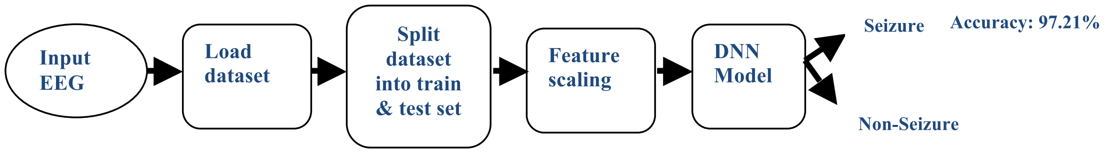

Fig. 4. Proposed work: Automatic seizure detection using deep neural network.

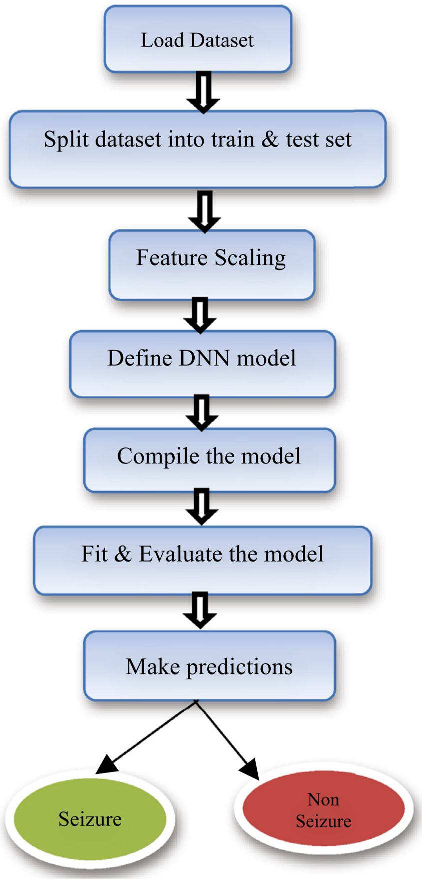

Fig. 5. Work flow of the proposed experiment using DNN.

Table 1
Values of the performance parameters of the DNN model.

Parameter	Values obtained from model (Training & Testing phase)
No. of test instances	2300
Real seizure events (P)	1866
Real non-seizure events (*N*)	434
Correctly detected	2243
Sensitivity	98.59%
Specificity	91.47%
Accuracy	97.21%
F1_score	0.927273
Precision	0.940092
Recall	0.914798
MCC	0.910201
ROC AUC	0.981682
True Positive rate(TPR)	0.9796
False Positive rate (FPR)	0.0599

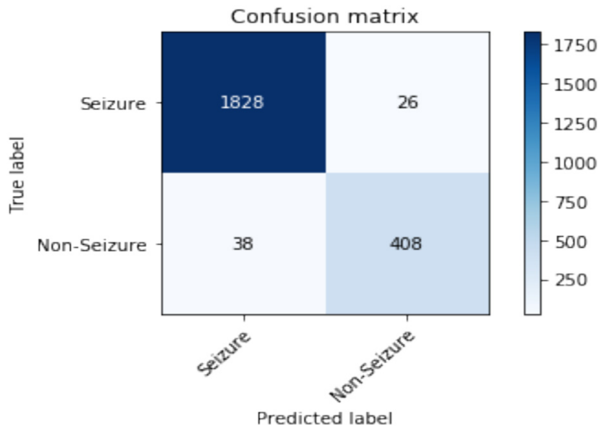

Fig. 6. Confusion matrix of the DNN model.

True positive = 1828
False positive = 26
False negative = 38
True negative = 408

Sensitivity, Specificity and F1_score are the most reliable metrics for assessing the accuracy of the classification. Sensitivity (or Recall) gives the number of true positive cases that are classified as positive. Specificity gives the proportion of truly negative cases that are classified as negative. F1_score gives the measure of classification accuracy. It is calculated using precision and recall. Precision is the proportion of number of relevant instances among retrieved instances. The formulas used for calculating sensitivity, specificity, precision, recall and f1_score are shown below:

$$\text{Sensitivity (or Recall)} = \frac{TP}{TP + FN} \tag{12}$$

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{13}$$

$$\text{Pr ecision} = \frac{TP}{TP + FP} \tag{14}$$

$$\text{F1\_Score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{15}$$

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{16}$$

$$\text{True Positive rate} = \frac{TP}{P} \tag{17}$$

Where P is the number of real seizure events

$$\text{False Positive rate(FPR)} = \frac{FP}{N} \tag{18}$$

Where N is the number of real non-seizure events.

Precision-recall curves and ROC curves are the two most diagnostic tools that are useful for interpreting the probabilistic forecast of the two-class classification predictive modeling problems.

The precision-recall curve of the proposed experiment is shown in Fig. 7. Precision and recall are calculated using Eqs. (14) & (12) respectively. The curve is plotted using precision_recall_curve() function in scikit learn with precision on the y-axis and recall on the x-axis. The function takes positive outcomes

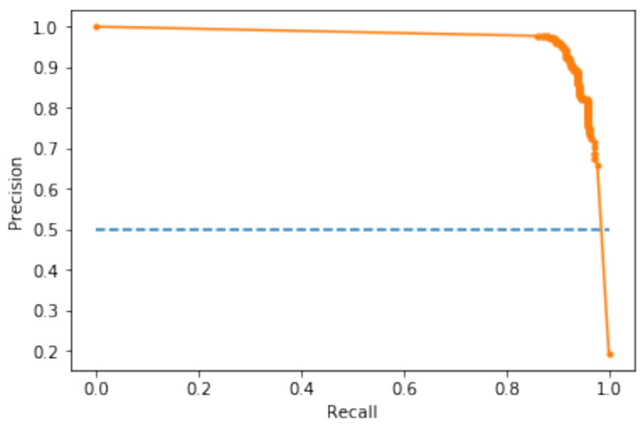

Fig. 7. precision-recall curve of the DNN model.

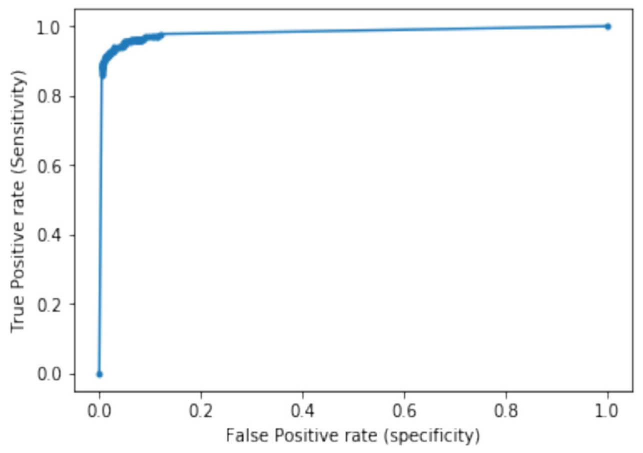

Fig. 8. ROC (Receiver Operating Characteristics) of the DNN model.

of the result and probabilities of the positive class and returns precision, recall and threshold values. Thus it gives the tradeoff between true positive points and probabilities of the positive labels at different thresholds. The curve bowing towards (1.0, 1.0) indicates that the model is a skillful model. A model with high precision and high recall gives results with correct labeling.

ROC curve shown in Fig. 8 is plotted using roc_curve() function from scikit learn with Sensitivity on the y-axis and specificity on the x-axis. ROC curve gives the tradeoff between True Positive rate(TPR) and False Positive rate (FPR), which are calculated using Eqs. (17) & (18) respectively. Fig. 8 below shows that, the curve runs from bottom left to top right with different thresholds. A model with perfect skill will have the curve from 0.0 to 0.1 as shown below.

Another performance evaluating parameter, Matthews correlation coefficient (MCC) [16] is used to measure the quality of binary and multiclass classification modeling problems. MCC is computed using Matthews_corrcoef () function from scikit learn. The correlation coefficient ranges from −1 to +1. +1 indicates perfect prediction, −1 indicates inverse prediction and 0 indicates an average prediction. Our model returned the **MCC value of 0.9196327497734866** which indicates the quality of the classification model is almost close to perfect prediction.

Accuracy and loss curves

The accuracy and loss of the model with Standard Scaler and mse as the Standardization and loss function respectively, are shown in Figs. 9 and 10.

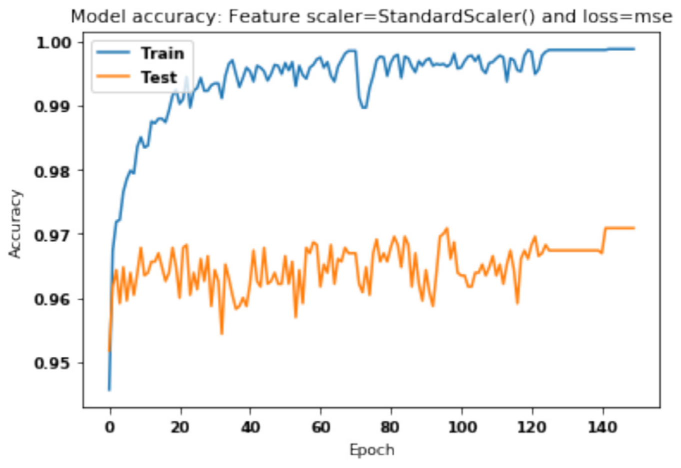

Fig. 9. Accuracy of the DNN model for train & test set.

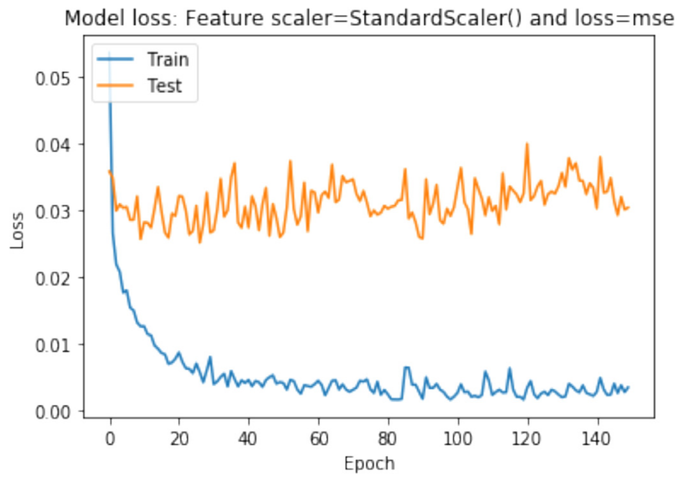

Fig. 10. Loss of the DNN model for train & test set.

The accuracy curve for train set (blue) in Fig. 9 shows that, the accuracy is around 97% in the beginning. But as the number of epochs is reaching 150, the accuracy has become almost 99.9%.Similarly, the accuracy curve for test set (green) shows that, the accuracy is around 95% in the beginning. When the number of epochs has reached 150, the accuracy has reached slightly more than 97%. Accuracy can be calculated using Eq. (16).

The loss curve for the train set (blue) in Fig. 10 shows that the amount of loss by the time number of epochs reached 150 is almost zero. In case of test set loss is less than 3% when the epoch has reached 150.

Experimental results show that deep neural network gives better performance when compared to machine learning methods. DNN has the capability of self learning from the data. Whereas in case of machine learning it needs to identify the features and extract the features using some sophisticated feature extracting technique. As shown in Table 2, our experiment out performed KNN and SVM.

Feature scaling of the dataset is a common requirement in most of the machine learning and deep learning experiments implemented in Scikit learn and Keras. Therefore in this experiment an attempt is made to check the performance of various loss functions and scaling techniques with respect to classification accuracy. The model is executed using four different Standardization functions like Standard Scaler, MinMaxScaler, Normalizer and RobustScaler. And each scaling technique, with four different loss functions like mean squared error, mean absolute error, Logcosh and binary cross

Table 2
Comparison of accuracy in% with other published works.

	Method	Feature extraction	Accuracy
Forrest Sheng Bao et al. [12]	KNN SVM	PyEEG	79.7% 94%
Abdulkadir Sengur et al. [8]	SVM	time frequency texture descriptors	90.3%
Our experiment	DNN	––––	97.21%

Table 3
Accuracy score of the DNN model with 4 different feature scaling techniques, each using 4 different loss functions.

Feature Scaling	Loss Function	Accuracy %
StandardScaler	Mean squared error	97.21
	Binary entropy	97.39
	Mean absolute error	97.08
	Logcosh	97.39
MinMaxScaler	Mean squared error	91.86
	Binary entropy	94.60
	Mean absolute error	80.60
	Logcosh	86.91
Normalizer	Mean squared error	88.17
	Binary entropy	86.78
	Mean absolute error	88.95
	Logcosh	88.69
RobustScaler	Mean squared error	97.60
	Binary entropy	97.65
	Mean absolute error	96.95
	Logcosh	97.21

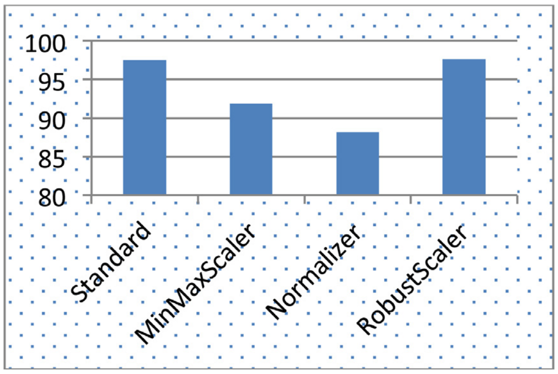

Fig. 11. Comparison of Accuracy of the model for four different feature scaling techniques using loss = 'mse'.

entropy. The performance of these standardization and loss functions are shown in Table 3 below. A graph (Fig. 11) is plotted showing the accuracy of the model with the above mentioned four standardization techniques using loss = mse. From the graph, it can be observed that both StandardScaler and RobustScaler methods are giving almost equal accuracy greater than 97%. Thus it can be concluded that, StandardScaler and RobustScaler feature scaling techniques give better accuracy compared to MinMaxScaler and Normalizer.

3.1. Advantages of the proposed system over existing method

a. In the proposed model there is no need to perform feature extraction separately.
b. Network learns the data by itself and passes relevant information through multiple layers.
c. As the amount of data increases; deep neural networks give better and better results.

4. Conclusion

The paper presents an automated epileptic seizure detection method using deep neural network. Most commonly used dataset from Bonn University Germany is used for the experiments. First, the experiment was conducted using machine learning SVM and KNN classifiers. In this case we had to extract features from the dataset and feed features to the classifiers. The obtained results gave classification accuracy of 94% and 79.7%. Second, deep neural network was used on the same dataset and ended with classification accuracy of 97.21%. From the results it can be shown that deep neural network gives better results compared to machine learning algorithms and also it is not required to use feature extraction technique in case of DNN as it learns data on its own. Feature scaling on the dataset was performed using StandardScaler method. The experiment was continued by performing feature scaling using other methods like, MinMaxScaler, Normalizer and RobustScaler. Among other methods, RobustScaler and StandardScaler showed equally good performance. Four different loss functions (Mean squared error, Binary entropy, Mean absolute error and Logcosh) were executed with each feature scaling technique. The accuracy values obtained are tabulated; mean squared error gives good performance with all four feature scaling techniques.

Conflict of Competing Interest

None.

References

[1] Y. Yuan, G. Xun, K. Jia, A. Zhang, A Multi-view Deep Learning Method For Epileptic Seizure Detection Using Short-time Fourier Transform, ACM, 2017.
[2] J.T. Oliva, J.L.G. Rosa, The Use of One-Class Classifiers For Differentiating Healthy from Epileptic EEG Segments, IEEE, 2017.
[3] A. Theodorakopoulou, Machine Learning Data Preparation For Epileptic Seizures Prediction Technological Educational Institute of Crete, Department of Informatics Engineering, June 2017.
[4] J.M. Williams, Deep Learning and Transfer Learning in the Classification of EEG Signals Computer Science and Engineering: Theses, Dissertations, and Student Research, University of Nebraska-Lincoln, 8–2017
[5] F. Artoni, A. Delorme and S. Makeig, Applying dimension reduction to EEG data by principal component analysis reduces the quality of its subsequent independent component decomposition neuroimage, March 2018.
[6] N. Dilesh Perera, C. Madarasingha, A.C. De Silva, Spatial Feature Reduction in Long-term EEG For Patient-Specific Epileptic Seizure Event Detection, ACM, 2017.
[7] C. Chen, Z. Liu, H. Li, R. Zhou, Y. Zhang, EEG Detection Based On Wavelet Transform and SVM Method, IEEE International Conference on Smart Cloud, 2016.
[8] A. Sengur, Y. Guo, Y. Akbulut, Time-frequency Texture Descriptors of EEG Signals For Efficient Detection of Epileptic Seizure, Springer, 2016.
[9] S. Bose, V. Rama, C.B. Rama Rao, EEG Signal Analysis For Seizure Detection Using Discrete Wavelet Transform and Random Forest, IEEE, 2017.
[10] L. Buitinck, G. Louppe, "API design for machine learning software: experiences from the scikit-learn project", arxiv, Sep 2013.
[11] N. Arunkumar, K. Ramkumar, V. Vekatraman, E. Abdulhay, S.L. Fernandes, S. Kadry, S. Segal, Classification of Focal and Non-Focal EEG Using entropies, Pattern Recognition Letters, Elsevier, 2017.
[12] F.S. Bao, PyEEG: AnOpen Source PythonModule for EEG/MEG Feature Extraction, Hindawi Publishing Corporation, 2011 Volume406391, doi:10.1155/2011/406391.
[13] https://scikit-learn.org/.
[14] D.M.W. Powers, Evaluation: from precision, recall and F-measure to ROC, informedness, markedness & correlation, ResearchGate (January 2011).
[15] D.K. Thara, B.G. Premasudha, A review on computer aided diagnosis of epilepsy using machine learning and deep learning, IJRAR 5 (3) (2018).
[16] J. Teo, Data Science Documentation, Oct 19, 2019 Release 0.1.