AN EFFICIENT FUZZY NEURAL NETWORK TRAINING MODEL FOR SUPERVISED PATTERN CLASSIFICATION SYSTEM

Dr. R.Manjula Devi, Dr. S.J. Suji Prasad, Ms.C.Sagana
Assistant Professor(Senior Grade) / CSE, Kongu Engineering College, Perundurai, Kongu Engineering College, Perundurai
rmanjuladevi.gem@gmail.com
Assistant Professor(Senior Grade) / EIE, Kongu Engineering College, Perundurai
sj sujiprasad@gmail.com
Assistant Professor / CSE, Kongu Engineering College, Perundurai
saganac@gmail.com

Abstract

Among the existing NN architectures, Multilayer Feedforward Neural Network (MFNN) with single hidden layer architecture has been scrutinized thoroughly as best for solving nonlinear classification problem. The training time is consumed more for very huge training datasets in the MFNN training phase. In order to reduce the training time, a simple and fast training algorithm called Exponential Adaptive Skipping Training (EAST) Algorithm was presented that improves the training speed by significantly reducing the total number of training input samples consumed by MFNN for training at every single epoch. Although the training performance of EAST achieves faster, it still lacks in the accuracy rate due to high skipping factor. In order to improve the accuracy rate of the training algorithm, Hybrid system has been suggested in which the neural network is trained with the fuzzified data. In this paper, a z-Score Fuzzy Exponential Adaptive Skipping Training (z-FEAST) algorithm is proposed which is based on the fuzzification of EAST. The evaluation of the proposed z-FEAST algorithm is demonstrated effectively using the benchmark datasets - Iris, Waveform, Heart Disease and Breast Cancer for different learning rate. Simulation study proved that z-FEAST training algorithm improves the accuracy rate.

Keywords: Adaptive Skipping, Neural Network, Training Algorithm, Training Speed, MFNN, Fuzzification.

1. INTRODUCTION

Due to the implicit characteristics of approximating any nonlinear classification problem, Multilayer Feedforward Neural Network (MFNN) with a single hidden layer architecture has been scrutinized thoroughly as best for solving this problem (Mehra and Wah 1992; Hornik et al 1989). For training the above network, the Back Propagation learning algorithm has been practiced (Rumelhart and McClelland 1986; Saman and Bryan 2011). In order to enhance the training performance, the training speed is the factor that is considered to be very important. The training speed is highly depends on the dimensionality of training dataset. In general, training MFNN with a larger training datasets will generalize the network well. But, lengthy training time is needed for larger training dataset [3] which influence the training speed. In order to improve the training speed, EAST algorithm was exercised [6]. It exhibits the training input samples randomly for training which diminishes the total training input samples exponentially which in reduce the overall total training time, thereby speeding up the training process. But the accuracy rate is greatly affected. Since the Fuzzy Logic (FL) enhances the NN generalization capability and also Neuro fuzzy hybrid system are universal approximators (Kosko 1994), a new Neuro fuzzy hybrid system with z-Score function has been put forward for improving the accuracy rate of EAST.

2. RELATED WORKS

Typically, by incorporating the advantage of both neural network and the fuzzy system, Neuro fuzzy hybrid system is more impressive than either the neural network or the fuzzy system. Anandakumar et al [1] proposed a classification model using Modified Levenberg-Marquardt learning algorithm that improves the accuracy and also consumes less time for convergence. Initially, the statistical ANOVA ranking technique is applied to find the higher ranked dataset. In order to analyze the public transportation system service quality, an ANN model is adapted [3]. Kulkarni and Shinde [5] proposed Neuro-fuzzy classification model for supervised data classification. By using Fuzzification method, membership value is calculated for each attribute values of the given class in the membership matrix. For ANN training, this matrix is fed as an input to the model and obtains the corresponding membership value for each pattern to the target classes. At the end of each iteration, the target class for each pattern is predicted using Defuzzification method. MFNN has been trained by Levenberg-Marquardt (LM) algorithm [9], CAST [8], EAST [6] and LAST [7] to develop a fast ANN model for nonlinear pattern classification. Patricia Melin et al [10] applied competitive neural network trained with learning vector quantization algorithm for electrocardiogram signals classification. Taskin kavzoglu et al. [13] described the way of representing the training datasets for improving the performance of classification methods. The data representation relates the training dataset size and quality. In order to identify the outlier in the training dataset, the quality analysis is used. After some refinements, representation data is formed by conducting the training data selection which is an iterative process. Quang Hung do et al [14] implemented Neuro-Fuzzy approach for solving multiclass classification problem to predict the students’ academic performance.
3. PROPOSED Z-FEAST METHODOLOGY

3.1 Overview of z-FEAST Architecture

The overall architecture of z-FEAST classifier is represented diagrammatically in Figure 1. This Neuro-Fuzzy classifier for pattern classification consists of various blocks as specified in the figure. It consists of three steps: Fuzzification, ANN training with the backpropagation algorithm and Defuzzification[5].

Assume that the network contains \( n \) input nodes in the input layer, \( p \) hidden nodes in the hidden layer and \( m \) output nodes in the output layer. Since the above network is highly interconnected, the nodes in each layer are connected with all the nodes in the next layer. Let \( P \) represent the number of input patterns in the training dataset. The input matrix, \( X \), of size \( p \times n \) is presented to the network. The number of nodes in the input layer is equivalent to the number of columns in the input matrix, \( X \). Each row in \( X \) is considered to be a real-valued vector generated from the network.

The input matrix obtained as output of this fuzzification process. The size of the matrix is \( p+1 \times (n+1) \) columns in the input matrix, \( Z(x) \), where \( 1 \leq i \leq n \).

In the Fuzzification process, the given training dataset is fed as input and the \( Z \)-score function is used as membership function. The membership matrix is obtained as output of this fuzzification process. The size of the matrix is \( S \times D \times C \), where \( S \) is the number of input samples in the training dataset, \( D \) is the number of features / attributes and \( C \) is the number of target class. Then, this membership matrix, that is fuzzified data, is fed as input to the MFNN. The summed real-valued vector generated from the hidden layer is represented as \( ze \in \mathbb{R}^{m} \), where \( 1 \leq i \leq m \). The estimated output real-valued vector generated from the network is denoted as \( ye \in \mathbb{R}^{m} \), where \( 1 \leq i \leq m \) and the corresponding target vector is represented as \( t \in \mathbb{R}^{m} \), where \( 1 \leq i \leq m \). Let \( f \) signifies the \( i \)-th iteration number.

Then, the network generated output, \( y_{i} \), is given as input to the defuzzification process. The MAX defuzzification method is applied for this process by assigning the pattern to the highest membership class. The defuzzified vector is compared with the target vector for calculating the error rate.

Let \( f_{i}(x) \) be the activation function used in the hidden layer and \( f_{i}(x) \) be the activation function used in the output layer. Let \( V_{b} \) be the \( n \times p \) weight matrix contains input-to-hidden weight coefficient for the link from the input node \( i \) to the hidden node \( j \) and \( V_{b} \) be the bias weight to the hidden node \( j \). Let \( W_{b} \) be the \( p \times m \) weight matrix contains hidden-to-output weight coefficient for the link from the hidden node \( j \) to the output node \( k \) and \( W_{b} \) be the bias weight to the output node \( k \).

3.2 Proposed z- FEAST Algorithm

The working principle of the z- FEAST algorithm that is incorporated in the BPN algorithm is summarized below:

Step 1. Weight Initialization: Initialize weights to small random values;

Step 2. Furnish the input sample: Disseminate to the input layer an input sample vector \( x_{i} \) having desired output vector \( y_{i} \);

Step 3. Fuzzification Process:
Convert Crisp to fuzzy value for the input vector \( x_{i} \). Input vectors are fuzzified using \( Z \)-score method. \( Z \)-score is modeled mathematically as

\[
Z(x; c, \sigma) = \begin{cases} 
\frac{(x-c)}{\sigma}, & x-c > 0 \\
\frac{(x-c)}{\sigma}, & x-c < 0 
\end{cases}
\]  

(1)
Where

- $x$ – Feature value
- $C$ – MF centre
- $\sigma$ is the MFs width

The Membership Function's centre, $C$, is given by,

$$C_d = \frac{1}{S} \sum_{j=1}^{S} x_{jd}$$  \hspace{1cm} (2)

Where

- $d=1, 2, \ldots, D$
- $j=1, 2, \ldots, S$ and
- $x_{jd}$ is the $d^{th}$ feature of sample $j$.

The Membership Function's width, $\sigma$, is given by,

$$\sigma_d = \frac{1}{S-1} \left( |x_{1d} - c_d| + |x_{2d} - c_d| + \ldots + |x_{sd} - c_d| \right)$$  \hspace{1cm} (3)

Where

- $x_{1d}, x_{2d}, \ldots, x_{sd}$ are the $d^{th}$ feature of the $s^{th}$ pattern and
- $C_C$ denote the mean value of $d^{th}$ feature given in Equation (2)

The membership matrix, $f_x$, that is generated using the equation (1). In this matrix, $g_{s,c}(d)$ represent the membership value of $d^{th}$ feature of $s^{th}$ pattern to the $c^{th}$ class. $f_x$ is given as

$$f_x = \begin{pmatrix}
    z_{1,1}(x_1) & z_{1,2}(x_1) & \cdots & z_{1,C}(x_1) \\
    z_{2,1}(x_1) & z_{2,2}(x_1) & \cdots & z_{2,C}(x_1) \\
    \vdots & \vdots & \ddots & \vdots \\
    z_{S,1}(x_1) & z_{S,2}(x_1) & \cdots & z_{S,C}(x_1)
\end{pmatrix}$$

Step 4. Forward Phase: Starting from the first hidden layer and propagating towards the output layer:

a. Calculate the activation values for the Hidden layer as:

i. Estimate the net output value

$$x_{inl}(it) = v_{ij}(it) + \sum_{i=1}^{n} f_{x_i}(it) \cdot v_{ij}(it)$$  \hspace{1cm} (3)

ii. Estimate the actual output

$$z_i(it) = \frac{1}{1 + e^{-x_{inl}}}$$  \hspace{1cm} (4)

b. Calculate the activation values for the Output layer as:

i. Estimate the net output value

$$y_{inl}(it) = w_{ok}(it) + \sum_{j=1}^{p} z_j(it) \cdot w_{oj}(it)$$  \hspace{1cm} (5)

ii. Estimate the actual output

$$y_k(it) = \frac{1}{1 + e^{-y_{inl}}}$$  \hspace{1cm} (6)

Step 5. Output errors: Calculate the error terms at the output layer as:

$$\delta_k(it) = [t_k - y_k(it)] \cdot f'(y_k(it))$$  \hspace{1cm} (7)

Differentiate the activation function in Equation (6),

$$f'(y_k(it)) = \frac{\partial(y_k(it))}{\partial x} = y_k(it) \times (1 - y_k(it))$$  \hspace{1cm} (8)

Substitute the resultant value of Equation (8) in (7)

$$\delta_k(it) = y_k(it) \cdot [1 - y_k(it)] \cdot [t_k - y_k(it)]$$  \hspace{1cm} (9)

Step 6. Backward Phase: Propagate error backward to the input layer through the hidden layer using the error term
\[ \delta_j(it) = \sum_{k=1}^{m} \delta_k(it).w_{jk}(it) \cdot f'(z_j(it)) \quad (10) \]

Differentiate the activation function in Equation 4,
\[ f'(z_j(it)) = \frac{\partial (z_j(it))}{\partial x} = z_j(it) \times (1 - z_j(it)) \quad (11) \]

Substitute the resultant value of Equation (11) in (10)
\[ \delta_j(it) = \sum_{k=1}^{m} \delta_k(it).w_{jk}(it) \cdot z_j(it) \times [1 - z_j(it)] \quad (12) \]

**Step 7. Weight Amendment:** Update weights using the Delta-Learning Rule
a. **Weight amendment** for Output Unit
   \[ W_{jk}(it + 1) = W_{jk}(it) + \alpha(it).\delta_k(it).z_j(it) \quad (13) \]

b. **Weight amendment** for Hidden Unit
   \[ V_{ij}(it + 1) = V_{ij}(it) + \alpha(it).\delta_j(it).x_i(it) \quad (14) \]

**Step 8. EAST Algorithm:** Incorporating the EAST algorithm
a. **Compare** the error value, \(|t_k - y_k|\) with threshold value, \(d_{\text{max}}\)
   \[ |t_k - y_k(it)| < d_{\text{max}} \quad (15) \]
   If equation 15 generates 0, then the \(x_i\) is correct
b. **Compute** the probability value for all input samples
   \[ \text{prob}(x^i) = \begin{cases} 0, & \text{if } x_i \text{ is correct and epoch number } < n \\ 1, & \text{otherwise} \end{cases} \quad (16) \]
c. **Calculate** the skipping factor, \(sf\) for all input samples
   i. Initialize the value of \(sf\) to zero (for first epoch)
   ii. Increment the value of \(sf\) exponentially for correctly classified samples alone.
d. **Skip** the training samples with \(\text{prob}(=0)\) for the next \(sf\) epoch

**Step 9. Defuzzification Process:**
The Fuzzy to Crisp conversion for the output variable is done using centroid method
\[ x^* = \frac{\int \mu_C(x).xdx}{\int \mu_C(x)dx} \]
To assign the class label,
\[
A = \begin{bmatrix}
O_{11} & O_{12} & \cdots & O_{1c} \\
O_{21} & O_{22} & \cdots & O_{2c} \\
\vdots & \vdots & \ddots & \vdots \\
O_{s1} & O_{s2} & \cdots & O_{sc}
\end{bmatrix}
\]
Where \(O_{sc}\) is the output of \(s^{th}\) pattern to the \(c^{th}\) class

**Step 10. Repeat** steps 1-7 until the halting criterion is satisfied, which may be chosen as the Root Mean Square Error (RMSE), elapsed epochs and desired accuracy

**Working Flow of z-FEAST Algorithm**
The block diagram of the proposed strategy is diagrammatically represented in the following figure
4. EXPERIMENTAL SETUP AND RESULT

4.1 Experimental Layout

A 3-layer feedforward neural network is adopted for the simulation of all the training algorithms with the selected training architecture and training parameters mentioned in Table 1. The simulation of all the training algorithms is repeated for two different learning rates such as $10^{-4}$ (0.0001) and $10^{-3}$ (0.001).

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Learning Rate</th>
<th>MFNN Architecture</th>
<th>Momentum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heart</td>
<td>$10^{-4}$</td>
<td>13x5x1</td>
<td>0.9</td>
</tr>
<tr>
<td></td>
<td>$10^{-3}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Breast Cancer</td>
<td>$10^{-4}$</td>
<td>31x15x1</td>
<td>0.9</td>
</tr>
<tr>
<td></td>
<td>$10^{-3}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Iris</td>
<td>$10^{-4}$</td>
<td>4 x 5 x 1</td>
<td>0.8</td>
</tr>
<tr>
<td></td>
<td>$10^{-3}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Waveform</td>
<td>$10^{-4}$</td>
<td>21x10x1</td>
<td>0.7</td>
</tr>
<tr>
<td></td>
<td>$10^{-3}$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For training Heart dataset, 13, 5 and 1 neurons in the input, hidden and output layers respectively are used. And, for training Breast Cancer dataset, the NN architecture with 31, 15 and 1 neurons in the input, hidden and output layers respectively, is used. The NN architecture with 4, 5 and 1 neurons in the input, hidden and output layers respectively, is used for training Iris database. For training waveform dataset, 21, 10 and 1 neurons in the input, hidden and output layers respectively, is used.

According to the idea of Nguyen-Widrow algorithm (Nguyen and Widrow 1990), the NN weight coefficients are initialized with the random values within the specified range -0.5 to +0.5 for faster learning.

4.2 Evaluation Method

The Fivefold cross validation method (Witten and Frank 2000) is performed on the above datasets to evaluate and compare the performance of the proposed training algorithms empirically on unseen data. The input sample in each
Dataset is randomly split into five equal sized disjoint folds. Among these five folds, a single fold is retained as the validation data for testing the network, and the remaining four folds are used as training data for training the network. The validation process is then repeated for five times, with each of the five folds used exactly once as the validation data. The results taken from the five training folds can then be averaged to produce a final result. The advantage of using this validation method is that all observations are used for both training and validation, and each observation is used for validation exactly once and also to avoid over-fitting (Peterson et al 1995).

The performance measures that are considered to evaluate the training algorithm are training time and classification accuracy. A good training algorithm will cut down the training time, while accomplishing better accuracy which is proved in our proposed work. The classification accuracy is calculated using the following formula:

\[
\text{Classification accuracy} = \frac{\text{Number of samples correctly classified}}{\text{Total number of samples}}
\]

The simulations of all the proposed training algorithms are done using MATLAB R2010b on a machine with the configuration of Intel® Core i5-3210M processor, 4 GB of RAM and CPU speed of 2.50GHz.

### 4.3 Dataset Description

The performance of all proposed AST algorithm is assessed for the classification problem on the benchmark two-class classification and multi-class classification datasets. The real-world benchmark datasets utilized for two-class classification problem are Heart and Breast Cancer Dataset, and multiclass classification problem are Iris and Waveform Dataset. The fore-mentioned datasets were fetched from the UCIMLR (University of California at Irvine Machine Learning Repository) (Asuncion and Newman 2007).

The specification of the benchmark datasets utilized for training in the research is summarized in Table 2.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>No. of Attributes</th>
<th>No. of Classes</th>
<th>No. of Instances</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iris</td>
<td>4</td>
<td>3</td>
<td>150</td>
</tr>
<tr>
<td>Waveform</td>
<td>21</td>
<td>3</td>
<td>5000</td>
</tr>
<tr>
<td>Heart</td>
<td>13</td>
<td>2</td>
<td>270</td>
</tr>
<tr>
<td>Breast Cancer</td>
<td>31</td>
<td>2</td>
<td>569</td>
</tr>
</tbody>
</table>

#### 4.3.1 Multiclass Problems

**Iris Dataset**

In the IRIS dataset, the number of iris flower samples is 150 which is gathered from three different flower varieties equally. The varieties are listed as Iris Setosa, Iris Versicolour and Iris Virginica which is identified using width and length of Iris sepal, and width and length of Iris petal. Among these varieties, Iris Setosa is easier to be separated from the other two varieties, while the other two varieties, Iris Virginica and Iris Versicolour, are partially obscured and harder to be distinguished.

**Waveform Dataset**

In the Waveform database generator dataset, the total number of wave’s samples is 5000 with 21 attributes which are equally divided into three wave classes (Asuncion and Newman 2007). These samples are collected from the generation of 2 of 3 “base” waves.

#### 4.3.2 Two-Class Problems

**Heart Dataset**

In the Statlog Heart disease database, the samples with 13 attributes are collected from 270 patients. Among these samples, the number of samples with heart disease ‘absent’ is 150 and with heart disease ‘present’ is 120.

**Breast Cancer Dataset**

In the Wisconsin Breast Cancer Diagnosis Dataset, the samples are collected from the patient’s characteristics of 569 with 32 features. Among these samples, 357 samples are diagnosed as benign and 212 samples are diagnosed as malignant class.

### 4.4 EXPERIMENTAL RESULT

Table 3 to 10 shows the experimental results of EAST, FEAST and z-FEAST algorithms observed at each step across five repeats of fivefold cross validation using two different learning rates such as $1e^{-4}$ and $1e^{-3}$. From these table 3 to 10, the EAST algorithm yields improved computational training speed in terms of the total number of trained input samples as well as total training time over FEAST and z-FEAST. But, when the skipping factor goes higher, the accuracy of EAST system is affected highly. But, z-FEAST improves the accuracy rate of the system.
Table 3. Comparison Results Trained by the Iris Dataset with 1e-4 Learning Rate

<table>
<thead>
<tr>
<th>Testing Fold</th>
<th>Number of Epochs</th>
<th>EAST</th>
<th>FEAST</th>
<th>z-FEAST</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total Number of Input Samples</td>
<td>Training Time (in Sec)</td>
<td>Accuracy (%)</td>
<td>Total Number of Input Samples</td>
</tr>
<tr>
<td>1</td>
<td>5442</td>
<td>208755</td>
<td>8.2995</td>
<td>73.33</td>
</tr>
<tr>
<td>2</td>
<td>5902</td>
<td>240293</td>
<td>8.5218</td>
<td>76.67</td>
</tr>
<tr>
<td>3</td>
<td>5332</td>
<td>206029</td>
<td>8.296</td>
<td>80</td>
</tr>
<tr>
<td>4</td>
<td>5439</td>
<td>223245</td>
<td>8.2565</td>
<td>80</td>
</tr>
<tr>
<td>5</td>
<td>5161</td>
<td>203116</td>
<td>7.8261</td>
<td>76.67</td>
</tr>
<tr>
<td>AVG</td>
<td>5455</td>
<td>216288</td>
<td>8.24</td>
<td>77.33</td>
</tr>
</tbody>
</table>

Table 4. Comparison Results Trained by the IRIS Dataset with 1e-3 Learning Rate

<table>
<thead>
<tr>
<th>Testing Fold</th>
<th>Number of Epochs</th>
<th>EAST</th>
<th>FEAST</th>
<th>z-FEAST</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total Number of Input Samples</td>
<td>Training Time (in Sec)</td>
<td>Accuracy (%)</td>
<td>Total Number of Input Samples</td>
</tr>
<tr>
<td>1</td>
<td>547</td>
<td>22339</td>
<td>0.7867</td>
<td>76.67</td>
</tr>
<tr>
<td>2</td>
<td>526</td>
<td>21369</td>
<td>0.7537</td>
<td>80</td>
</tr>
<tr>
<td>3</td>
<td>535</td>
<td>21735</td>
<td>0.7667</td>
<td>76.67</td>
</tr>
<tr>
<td>4</td>
<td>545</td>
<td>22120</td>
<td>0.7756</td>
<td>80</td>
</tr>
<tr>
<td>5</td>
<td>510</td>
<td>20735</td>
<td>0.7306</td>
<td>76.67</td>
</tr>
<tr>
<td>AVG</td>
<td>533</td>
<td>21660</td>
<td>0.76</td>
<td>78</td>
</tr>
</tbody>
</table>

Table 5. Comparison Results Trained by the Waveform Dataset with 1e-4 Learning Rate

<table>
<thead>
<tr>
<th>Testing Fold</th>
<th>Number of Epochs</th>
<th>EAST</th>
<th>FEAST</th>
<th>z-FEAST</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total Number of Input Samples</td>
<td>Training Time (in Sec)</td>
<td>Accuracy (%)</td>
<td>Total Number of Input Samples</td>
</tr>
<tr>
<td>1</td>
<td>8187</td>
<td>16974989</td>
<td>17.28264</td>
<td>79.8</td>
</tr>
<tr>
<td>2</td>
<td>8973</td>
<td>17897431</td>
<td>30.3537</td>
<td>80.2</td>
</tr>
<tr>
<td>3</td>
<td>8929</td>
<td>17812293</td>
<td>30.22541</td>
<td>81.1</td>
</tr>
<tr>
<td>4</td>
<td>8903</td>
<td>17806977</td>
<td>29.0942</td>
<td>80.9</td>
</tr>
<tr>
<td>5</td>
<td>8887</td>
<td>17144339</td>
<td>28.6921</td>
<td>79.9</td>
</tr>
<tr>
<td>AVG</td>
<td>8776</td>
<td>17527206</td>
<td>27.13</td>
<td>80.38</td>
</tr>
</tbody>
</table>

Table 6. Comparison Results Trained by the Waveform Dataset with 1e-3 Learning Rate
<table>
<thead>
<tr>
<th>Testing Fold</th>
<th>Number of Epochs</th>
<th>EAST</th>
<th>FEAST</th>
<th>z-FEAST</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total Number of Input Samples</td>
<td>Training Time (in Sec)</td>
<td>Accuracy (%)</td>
<td>Total Number of Input Samples</td>
</tr>
<tr>
<td>1</td>
<td>823</td>
<td>161159</td>
<td>2.6747</td>
<td>81.1</td>
</tr>
<tr>
<td>2</td>
<td>894</td>
<td>178533</td>
<td>2.9381</td>
<td>80.6</td>
</tr>
<tr>
<td>3</td>
<td>891</td>
<td>176121</td>
<td>2.8975</td>
<td>79.9</td>
</tr>
<tr>
<td>4</td>
<td>890</td>
<td>178488</td>
<td>2.8904</td>
<td>80.5</td>
</tr>
<tr>
<td>5</td>
<td>890</td>
<td>165932</td>
<td>2.8696</td>
<td>80.1</td>
</tr>
<tr>
<td>AVG</td>
<td>878</td>
<td>172047</td>
<td>2.85</td>
<td>80.44</td>
</tr>
</tbody>
</table>

Table 7. Comparison Results Trained by the Heart Dataset with 1e-4 Learning Rate

<table>
<thead>
<tr>
<th>Testing Fold</th>
<th>Number of Epochs</th>
<th>EAST</th>
<th>FEAST</th>
<th>z-FEAST</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total Number of Input Samples</td>
<td>Training Time (in Sec)</td>
<td>Accuracy (%)</td>
<td>Total Number of Input Samples</td>
</tr>
<tr>
<td>1</td>
<td>7485</td>
<td>713559</td>
<td>23.2651</td>
<td>75.93</td>
</tr>
<tr>
<td>2</td>
<td>7529</td>
<td>809372</td>
<td>25.3458</td>
<td>74.07</td>
</tr>
<tr>
<td>3</td>
<td>7569</td>
<td>820114</td>
<td>27.84309</td>
<td>75.93</td>
</tr>
<tr>
<td>4</td>
<td>7567</td>
<td>813699</td>
<td>26.6308</td>
<td>79.63</td>
</tr>
<tr>
<td>5</td>
<td>7567</td>
<td>811180</td>
<td>25.9578</td>
<td>77.78</td>
</tr>
<tr>
<td>AVG</td>
<td>7543</td>
<td>793585</td>
<td>25.81</td>
<td>76.67</td>
</tr>
</tbody>
</table>

Table 8. Comparison Results Trained by the Heart Dataset with 1e-3 Learning Rate
5  829  99092  3.993784  77.78  116799  4.08  92.59  102895  3.49  94.40  
AVG  829  95137  3.56  75.56  116826  4.46  92.54  99590  3.18  94.40  

Table 9. Comparison Results Trained by the Breast Cancer Dataset with 1e-4 Learning Rate

<table>
<thead>
<tr>
<th>Testing Fold</th>
<th>Number of Epochs</th>
<th>EAST</th>
<th>FEAST</th>
<th>z-FEAST</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total Number of Input Samples</td>
<td>Training Time (in Sec)</td>
<td>Accuracy (%)</td>
<td>Total Number of Input Samples</td>
</tr>
<tr>
<td>1</td>
<td>6279</td>
<td>1055844</td>
<td>34.0877</td>
<td>83.33</td>
</tr>
<tr>
<td>2</td>
<td>6460</td>
<td>966328</td>
<td>30.7942</td>
<td>79.82</td>
</tr>
<tr>
<td>3</td>
<td>7976</td>
<td>1286262</td>
<td>46.8745</td>
<td>84.21</td>
</tr>
<tr>
<td>4</td>
<td>7691</td>
<td>1138979</td>
<td>43.9747</td>
<td>80.07</td>
</tr>
<tr>
<td>5</td>
<td>7439</td>
<td>1097278</td>
<td>31.3622</td>
<td>84.21</td>
</tr>
<tr>
<td>AVG</td>
<td>7169</td>
<td>1108938</td>
<td>37.42</td>
<td>82.3</td>
</tr>
</tbody>
</table>

Table 10. Comparison Results Trained by the Breast Cancer Dataset with 1e-3 Learning Rate

RESULT ANALYSIS

4.5.1 Training Samples Comparison

The comparison results of the total number of input samples consumed for training by EAST, FEAST and z-FEAST with the learning rate of 1e-4 and 1e-3 are shown in Fig.3-6.

Herewith, it is assured from the Figure 3 that the total number of training samples consumed by EAST algorithm for training under the learning rate of 1e-4 is reduced by an average of nearly 17% and 6% of FEAST and z-FEAST algorithm for Iris Dataset, 69% and 8% for Waveform Dataset, 78% and 9% for Heart Dataset and 87% and 8% for Breast Cancer Dataset respectively.
Herewith, it is assured from the Figure 4 that the total number of training samples consumed by EAST algorithm for training under the learning rate of 1e-3 is reduced by an average of nearly 15% and 3% of FEAST and z-FEAST algorithm for Iris Dataset, 79% and 11% for Waveform Dataset, 23% and 5% for Heart Dataset and 73% and 9% for Breast Cancer Dataset respectively.

### 4.5.2 Training Time Comparison

Herewith, it is concluded from the Figure 5, for training IRIS dataset, the total training time consumed by EAST algorithm with the learning rate of 1e-4 is reduced to an average of 37% of FEAST algorithm and 11% of z-FEAST algorithm, for Waveform Dataset by 35% of FEAST algorithm and 10% of z-FEAST algorithm, for Heart Dataset by 41% of FEAST algorithm and 9% of z-FEAST algorithm and for Breast Cancer Dataset by 16% of FEAST algorithm and 6% of z-FEAST algorithm respectively.
Figure 5: Comparison Result of total training time consumed with 1e-4 learning rate

Herewith, it is concluded from the Figure 6, for training IRIS dataset, the total training time consumed by EAST algorithm with the learning rate of 1e-3 is reduced to an average of 29% of FEAST algorithm and 8% of z-FEAST algorithm, for Waveform Dataset by 45% of FEAST algorithm and 10% of z-FEAST algorithm, for Heart Dataset by 25% of FEAST algorithm and 10% of z-FEAST algorithm and for Breast Cancer Dataset by 67% of FEAST algorithm and 13% of z-FEAST algorithm respectively.

Figure 6: Comparison Result of total training time consumed with 1e-3 learning rate

5. CONCLUSION

Thus, the z-Score Fuzzy Exponential Adaptive Skipping Training (Z-Feast) Algorithm is systematically investigated in order to improve the accuracy rate of EAST algorithm. And also, It is further concluded that the proposed z-FEAST algorithm is much faster than the standard BPN, LAST, CAST, HOT and EAST algorithm and also the accuracy rate is highly improved compared to EAST algorithm. The proposed z-FEAST Algorithm can be incorporated in any supervised algorithm used for training real-world supervised pattern classification.

REFERENCES


