A Single Hidden Layer Artificial Neural Network Model that Employs Algebraic Neurons: Algebraic Learning Machine

Ömer Faruk Ertuğrul (omerfarukertugrul@gmail.com)
Batman Universitesi

Research Article

Keywords: Algebraic neurons, randomized artificial neural network, single hidden layer feed-forward artificial neural network

DOI: https://doi.org/10.21203/rs.3.rs-351062/v1

License: © This work is licensed under a Creative Commons Attribution 4.0 International License. Read Full License
A Single Hidden Layer Artificial Neural Network Model that Employs Algebraic Neurons: Algebraic Learning Machine

Ömer Faruk Ertuğrul

Department of Electrical and Electronic Engineering, Batman University, 72060 Batman, Turkey.

*Corresponding author: E-mail: omerfarukertugrul@gmail.com;
Tel.: +90-488-2173719; Fax: +90-488-2153601

Abstract

Artificial neural networks (ANN) have been employed successfully because of their high modeling capability. Many versions of the ANN have been proposed to increase the modeling ability. Since ANN based on the biological neural network system, the only mathematical operation is summation or subtraction (while the coefficients are negative). This research was done to investigate the application of other mathematical operations, which are multiplication, division, logarithm, and exponential, in nodes. Based on this fact, a novel a single hidden layer feed-forward artificial neural network (SLFN) model, which was called was algebraic learning machine (ALM), was proposed. The proposed ALM was evaluated and validated with 60 different benchmark datasets. Obtained results were compared with results obtained by each of the extreme learning machine (ELM), randomized artificial neural network, and random vector functional link, and back-propagation trained SLFN methods. Achieved results show that the proposed method is successful enough to be employed in classification and regression.

Keywords: Algebraic neurons; randomized artificial neural network; single hidden layer feed-forward artificial neural network

1. Introduction

The learning is one of the major abilities of humanity [1]. Therefore, many of the artificial learning methods have been built on the aim of modeling human learning mechanisms. One of the popular machine learning methods is an artificial neural network (ANN), which is based on the human neural network [2-4].

ANN has been employed in many applications since 1943 such as medical, economy, robotics [5]. The major reason behind the success of ANN is its high adapting capability of modeling not only linear but also nonlinear issues. The adaptation of ANN means determining the optimal values of its free parameters and this determining stage is called training. In order
to increase the adapting capability of ANN, various ANN structures (e.g., feed-forward, recurrent, single hidden layer feed-forward) and ANN training methods (gradient-based methods, randomized methods) have been proposed [6-9].

In each of them, the inputs (and in some cases, recurrent outputs) are weighted according to the learned weights by employed training method and these weighted inputs are summed (for some cases, it is subtraction because of negative weights) and biased [10, 11]. Finally, the nonlinearity of the ANN is gained by employing the transfer (activation) function to the obtained summation. As seen in this basic process, the main algebraic operation in ANN is summation and subtraction as seen in the existing literature in ANN [11-13].

Based on the summation operator, it can be easily said that the existing ANN methods can be employed successfully in modeling a system or a phenomenon that has a summation relationship between input variables (parameters). According to the best knowledge of the author, there is not any research in the literature that employed any algebraic operation other than summation or subtraction. Although, there is a large and a growing literature about the success of ANN that applied summation and/or subtraction in the nodes [8, 14-16], the result of employing different algebraic operations in nodes has not been investigated yet.

The research questions in this study focused on whether the success of the adaptation of systems that may have other algebraic relationships between input variables/parameters such as production, division, exponential, and logarithmic. The motivation behind this paper is to seek the answer to this question. To evaluate and validate the effectiveness of applying various algebraic operations in the nodes, a novel single hidden layer feed-forward (SLFN) ANN structure was proposed, which was called algebraic learning machine (ALM) and the proposed ALM was trained by the methodology of randomized ANN [17]. The proposed approach was validated by Backpropagation trained feed-forward artificial neural network (BPN), generalized regression artificial neural network (GRN), randomized artificial neural network (RNN), random vector functional link (RVFL), and extreme learning machine (ELM) methods [17-21].

The main contribution of this paper is employing various algebraic operators in the nodes and by this to increase the modeling capability of the traditional ANNs. In order to validate the proposed approach, 60 benchmark datasets (30 of them are classification and the others are regression datasets) were employed according to the given methodology in Section
2. Obtained results in terms of success and processing time were given and discussed in Section 3, and the last section was concluded in the paper.

2. Materials and Method

2.1. Material

To validate the proposed approach 30 benchmark classification datasets, whose details were given in Table 1, have been employed in tests. The first 5 datasets, which are the Lithuanian, Highleyman, banana-shaped, spherical, and multi-class, are synthetic datasets [22], while the other employed datasets that were given in Table 1 are real-world datasets [23].

| Dataset                        | Number of attributes | Number of observations | Number of classes | Source |
|--------------------------------|----------------------|------------------------|-------------------|--------|
| Lithuanian                     | 2                    | 1000                   | 2                 | [22]   |
| Highleyman                     | 2                    | 1000                   | 2                 | [22]   |
| Banana shaped                  | 2                    | 1000                   | 2                 | [22]   |
| Spherical                      | 2                    | 1000                   | 2                 | [22]   |
| Multi-class                    | 2                    | 1000                   | 8                 | [22]   |
| Liver                          | 7                    | 345                    | 2                 | [23]   |
| Pima Indian diabetes           | 8                    | 762                    | 2                 | [23]   |
| Hepatitis                      | 19                   | 150                    | 2                 | [23]   |
| Banana                         | 2                    | 5300                   | 2                 | [23]   |
| Image segmentation             | 19                   | 2315                   | 7                 | [23]   |
| Satellite image                | 36                   | 6435                   | 7                 | [23]   |
| Statlog (shuttle)              | 9                    | 58000                  | 7                 | [23]   |
| Abalone                        | 8                    | 4177                   | 29                | [23]   |
| Wine                           | 13                   | 178                    | 3                 | [23]   |
| Breast tissue                  | 10                   | 106                    | 4                 | [23]   |
| Cardiotocography               | 22                   | 2126                   | 3                 | [23]   |
| Skin segmentation              | 3                    | 245057                 | 2                 | [23]   |
| Seeds                          | 7                    | 210                    | 3                 | [23]   |
| EEG eye state                  | 15                   | 14980                  | 2                 | [23]   |
| Seismic bumps                  | 18                   | 2584                   | 2                 | [23]   |
| Banknote authentication        | 4                    | 1372                   | 2                 | [23]   |
| Balance scale                  | 4                    | 625                    | 3                 | [23]   |
| Acute inflammations            | 7                    | 120                    | 2                 | [23]   |
| Dermatology                    | 35                   | 366                    | 6                 | [23]   |
| Diabetic retinopathy Debrecen  | 19                   | 1151                   | 2                 | [23]   |
| Fertility                      | 9                    | 100                    | 2                 | [23]   |
| Glass                          | 10                   | 214                    | 2                 | [23]   |
| Haberman                       | 3                    | 306                    | 2                 | [23]   |
| Hayes-Roth                     | 5                    | 132                    | 3                 | [23]   |
| QSAR biodegradation            | 41                   | 1055                   | 2                 | [23]   |
The employed 30 benchmark regression datasets are summarized in Table 2. The approximate sinc dataset is a synthetic dataset [24], while the other employed datasets are real-world datasets [23, 25].

| Dataset                   | Number of attributes | Number of observations | Source |
|---------------------------|----------------------|------------------------|--------|
| Approximate sinc          | 1                    | 5000                   | [24]   |
| Forest fire               | 13                   | 517                    | [23]   |
| CASP 5-9                  | 9                    | 45730                  | [23]   |
| Istanbul stock exchange    | 7                    | 537                    | [23]   |
| Ailerons                  | 40                   | 13750                  | [25]   |
| Delta ailerons            | 5                    | 7129                   | [25]   |
| Auto-price                | 15                   | 159                    | [23]   |
| Bank-8FM                  | 8                    | 6481                   | [25]   |
| Boston housing            | 14                   | 506                    | [23]   |
| Breast cancer             | 32                   | 194                    | [23]   |
| California housing        | 9                    | 26940                  | [25]   |
| Census-8L                 | 8                    | 22784                  | [25]   |
| Census-8H                 | 8                    | 22784                  | [25]   |
| Census-16L                | 16                   | 22784                  | [25]   |
| Census-16H                | 16                   | 22784                  | [25]   |
| CPU-small                 | 12                   | 8192                   | [23]   |
| CPU                       | 21                   | 8192                   | [23]   |
| Diabetes child            | 2                    | 43                     | [25]   |
| Delta elevators           | 6                    | 9517                   | [25]   |
| Elevators                 | 18                   | 16599                  | [25]   |
| Kinematics                | 8                    | 8192                   | [25]   |
| Machine-CPU               | 6                    | 209                    | [25]   |
| Puma-8NH                  | 8                    | 6677                   | [25]   |
| Puma-32H                  | 32                   | 4938                   | [25]   |
| Pyrimidines               | 28                   | 74                     | [25]   |
| Servo                     | 4                    | 167                    | [23]   |
| Stocks                    | 9                    | 950                    | [25]   |
| Triazines                 | 60                   | 186                    | [25]   |
| Energy efficiency: cooling| 8                    | 768                    | [23]   |
| Energy efficiency: heating| 8                    | 768                    | [23]   |

2.2. Algebraic learning machine (ALM)

The proposed ALM is built on a single hidden layer feed-forward artificial neural network (SLFN) as given in Figure 1 [21]. As seen in Figure 1, there are different types of neurons, which are summation, production, exponential, logistic, and trigonometric neurons, in
the hidden layer. Each of them performed different algebraic operators at its core. The previous neurons are only done summation or subtraction operations. The algebraic operator, which is employed in the core of the neurons in the hidden layer, performed in ALM is not limited to summation, production, exponential, logistic, and trigonometric operators. The researchers may use any algebraic operator that may represent the modeled system or phenomenon. The output of each type of the neurons in the hidden layer, which can be seen in Figure 1, can be calculated as follows.

\[ O_{1 \cdots a}^s = g\left( \sum_{i=1}^{n} w_{i,j}^s x_i + b_j^s \right) \]  \hspace{1cm} (1) \\

\[ O_{1 \cdots b}^p = g\left( \prod_{i=1}^{n} w_{i,j}^p x_i + b_j^p \right) \]  \hspace{1cm} (2) \\

\[ O_{1 \cdots c}^e = g\left( \sum_{i=1}^{n} e^{w_{i,j}^e x_i} + b_j^e \right) \]  \hspace{1cm} (3) \\

\[ O_{1 \cdots d}^l = g\left( \sum_{i=1}^{n} \log (w_{i,j}^l x_i) + b_j^l \right) \]  \hspace{1cm} (4) \\

\[ O_{1 \cdots f}^t = g\left( \sum_{i=1}^{n} \sin(w_{i,j}^t x_i) + b_j^t \right) \]  \hspace{1cm} (5) \\

where \( x, O, n, g(.), w_{i,j}, b_j \), are the input, output of the neuron in the hidden layer, the number of features in the input, the activation (transfer) function, weights in the input layer, and biases, respectively. Furthermore, \( s, p, e, l, \) and \( t \) are the summation, production, exponential, logarithmic, and trigonometric types of neurons in the hidden layer, which has \( a, b, c, d, \) and \( f \) number of neurons in the hidden layer, respectively. The output of ALM is given below

\[ y = \sum_{j=1}^{m} \beta_j O_j + B \]  \hspace{1cm} (6) \\

where \( m, \) and \( B, \) are the number of neurons in the hidden layer and the bias in the output layer, respectively. In ALM, the training methodology is based on a randomized artificial neural network as given below [17].
Figure 1. The structure of the single hidden layer feed-forward artificial neural network.
The number of each of the neuron type in the hidden layer are assigned by the user/researcher,

1. The activation functions in the hidden layer are assigned by the user/researcher (please note that each of the arbitrarily bounded and non-constant activation functions can be employed as an activation function [26]),
2. Weights in the input layer are assigned arbitrarily,
3. Biases in the hidden layer are assigned arbitrarily,

After assigning these parameters it can be seen that the only unknowns are the weights and bias in the output layer. These free parameters must be determined to minimize the norm error, which can be shown below [21, 24].

\[
\min \left\| \sum_{k=1}^{N} (O_k - y_k)^2 \right\|
\]

(6)

To achieve the unknown/un-assigned weights and biases, a training dataset is required and \( N \) shows the number of observations in this dataset. After applying the training dataset, the following matrix can be obtained.

\[
\begin{bmatrix}
  y_1 \\
  y_2 \\
  \vdots \\
  y_{N-1} \\
  y_N
\end{bmatrix}_{1 \times N} =
\begin{bmatrix}
  O_1^1 & O_2^1 & \cdots & O_m^1 \\
  O_1^2 & O_2^2 & \cdots & O_m^2 \\
  \vdots & \vdots & \ddots & \vdots \\
  O_{N-1}^1 & O_{N-1}^2 & \cdots & O_{N-1}^m \\
  O_N^1 & O_N^2 & \cdots & O_N^m
\end{bmatrix}_{N \times m} \ast 
\begin{bmatrix}
  \beta_1 \\
  \beta_2 \\
  \vdots \\
  \beta_{m-1} \\
  \beta_m
\end{bmatrix}_{m \times 1}
\]

(7)

As seen the matrix that consists of the output of the neurons in the hidden layer is not a square matrix. Therefore, the matrix that consists of the weights in the output layer can be easily calculated analytically by the generalized Moore–Penrose inverse method or by Fisher method [17, 21, 24].

\[
\hat{\beta} = O^\dagger y
\]

(7)

where \( O^\dagger \) shows the generalized Moore–Penrose inverse of the outputs of the neurons in the hidden layer.

2.3. Validation Process

Each of the processes was applied according to 5-folds cross-validation. BPN, GRN, RNN, RVFL, and ELM methods were employed in the validation process [17-21]. In order to achieve a fair comparison, the same partitions and optimization rules were employed for each of the employed methods. To evaluate and validate the proposed approach, the following methodology was performed as summarized in three steps.
1. **step:** Normalizing the employed dataset into the range of -1 to 1.

2. **step:** Optimizing the activation function and the number of neurons in the hidden layer in each of the employed methods, which are BPN, GRN, RNN, RVFL, and ELM. The optimization stage was performed by trials. The employed activation functions in trials were sigmoid, triangular basis, hard limit, sign, tangent sigmoid, and saturation linear, while the employed numbers of neurons in the hidden layer were 10, 20, 50, 100, and 150. Please note that, in RNN, only the sigmoid activation function was used in optimizing, training, and test stages due to the methodology of RNN [17]. The optimizing was performed based on obtained accuracy (%) in classification datasets and obtained root mean square error (RMSE) in regression datasets. These metrics are calculated as follows [27].

\[
\text{Accuracy} \%(\%) = 100 \times \frac{\# \text{True classified samples}}{\# \text{All samples}}\% \quad (4)
\]

\[
\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (f_i - y_i)^2} \quad (5)
\]

where \( f, y, \) and \( N \) are desired value, obtained values, and the number of observations (samples) in the dataset, respectively.

3. **step:** Employing BPN, GRN, RNN, ELM, RVFL, and the proposed ALM in each dataset based on 5-folds cross-validation.

3. **Results and Discussion**

3.1. **Analyzing the outcomes obtained based on the employed number of neuron types in the hidden layer**

To investigate the outcomes of the neuron types on the success of the ALM, 2 classification datasets, which are the Lithuanian (a synthetic dataset) and the liver (a real-world dataset) dataset, and 2 regression datasets, which are the approximate sinc (a synthetic dataset) and the forest fire (a real-world dataset), were employed. The following neuron types were employed in tests.

- a. All types of neurons, which are summation, production, exponential, logistic, and trigonometric neurons,
- b. Only summation neurons,
- c. Only production neurons,
- d. Only exponential neurons,
e. Only logistic neurons,
f. Only trigonometric neurons,

Obtained testing accuracies (%) in classification datasets and achieved testing RMSEs in regression datasets based on the employed activation functions, which are sigmoid, triangular basis, hard limit, sign, tangent sigmoid, and saturation linear, and the employed numbers of neurons in the hidden layer, which are 10, 20, 50, 100, and 150, given in Figures 2-5 (please note that each of the reported results is obtained based on 5-folds cross-validation).

As seen in these figures, obtained accuracies are highly dependent on the employed artificial neural network parameters. This is an expected result because successive modeling yields higher success rates in testing. Furthermore, the training and testing success rates (accuracy (%) in classification and RMSE in regression datasets) and processing duration (s), which shows computational complexity, in ALM with optimized parameters are summarized in Tables 3 and 4.

Table 3. Obtained mean accuracies and process times in the Lithuanian and the liver datasets

| Neuron Types       | Lithuanian | Training Accuracy (%) | Training Duration (s) | Testing Accuracy (%) | Testing Duration (s) |
|--------------------|------------|-----------------------|-----------------------|----------------------|----------------------|
| All equal          |            | 97.80                 | 0.0325                | 95.70                | 0.0325               |
| Only summation     |            | 97.40                 | **0.0119**             | 95.30                | **0.0119**           |
| Only production    |            | 97.73                 | 0.0367                | 94.80                | 0.0367               |
| Only exponential   |            | 97.65                 | 0.0194                | 96.00                | 0.0194               |
| Only logistic      |            | **97.98**             | 0.0292                | 95.30                | 0.0292               |
| Only trigonometric |            | 97.75                 | 0.0152                | **96.20**            | 0.0152               |

| Neuron Types       | Liver      | Training Accuracy (%) | Training Duration (s) | Testing Accuracy (%) | Testing Duration (s) |
|--------------------|------------|-----------------------|-----------------------|----------------------|----------------------|
| All equal          |            | 82.40                 | 0.0191                | 71.45                | 0.0191               |
| Only summation     |            | 82.19                 | 0.0197                | 71.99                | 0.0197               |
| Only production    |            | 81.67                 | **0.0170**             | **72.31**            | **0.0170**           |
| Only exponential   |            | 82.45                 | 0.0218                | 71.62                | 0.0218               |
| Only logistic      |            | 81.63                 | 0.0181                | 72.14                | 0.0181               |
| Only trigonometric |            | **81.97**             | **0.0148**             | 71.97                | **0.0148**           |
Figure 2. Obtained accuracies based on number of neurons and activation function in the Lithuanian: a) all neuron types, b) summation neurons, c) production neurons, d) exponential, e) logistic, f) trigonometric
Figure 3. Obtained accuracies based on number of neurons and activation function in the liver: a) all neuron types, b) summation neurons, c) production neurons, d) exponential, e) logistic, f) trigonometric
Figure 4. Obtained accuracies based on number of neurons and activation function in the approximate sinc: a) all neuron types, b) summation neurons, c) production neurons, d) exponential, e) logistic, f) trigonometric.
Figure 5. Obtained accuracies based on number of neurons and activation function in the forest fire: a) all neuron types, b) summation neurons, c) production neurons, d) exponential, e) logistic, f) trigonometric
Table 4. Obtained mean accuracies and process times in the approximate sinc and the forest fire datasets

| Neuron Types      | Training RMSE | Training Duration (s) | Testing RMSE | Testing Duration (s) |
|-------------------|---------------|-----------------------|--------------|----------------------|
| **Approximate Sinc** |               |                       |              |                      |
| All equal         | 0.5693        | 0.1280                | 0.5743       | 0.1280               |
| Only summation    | **0.5711**    | **0.0624**            | 0.5742       | **0.0624**           |
| Only production   | 0.5677        | 0.1260                | **0.5741**   | 0.1260               |
| Only exponential  | 0.5702        | 0.0679                | 0.5742       | 0.0679               |
| Only logistic     | **0.5649**    | 0.1192                | 0.5742       | 0.1192               |
| Only trigonometric| 0.5709        | 0.0753                | 0.5742       | 0.0753               |
| **Forest Fire**   |               |                       |              |                      |
| All equal         | 0.0504        | 0.0204                | 0.0457       | 0.0204               |
| Only summation    | 0.0501        | 0.0223                | 0.0459       | 0.0223               |
| Only production   | 0.0520        | 0.0192                | 0.0461       | 0.0192               |
| Only exponential  | 0.0504        | **0.0150**            | **0.0451**   | **0.0150**           |
| Only logistic     | 0.0515        | 0.0225                | 0.0462       | 0.0225               |
| Only trigonometric| **0.0499**    | 0.0174                | 0.0460       | 0.0174               |

As seen in Tables 3 and 4, there is not any relationship between obtained success and the neuron types. For instance, in the Lithuanian dataset, the most suitable activation function was found as trigonometric neurons, which can be because of the data distribution of the Lithuanian dataset that can be seen in Figure 6.

As a summary, the obtained major result well suits the literature findings that the optimal network parameters depend on the data distribution of the dataset. The optimal neuron types, the number of neurons in the hidden layer, and the activation function must be determined to achieve better modeling; in other words, to achieve higher success.
3.2. Obtained success rates in ALM

The proposed ALM, BP, GRN, RNN, RVFL, and ELM were employed in the datasets given in Section 2. Each of these methods was optimized and applied according to 5-folds cross-validation. Obtained accuracies (%) and RMSEs were given in Tables 5 and 6, respectively, while the best ones were given in bold. Please note that, in tests in this section, the neuron types were not optimized; all types of neurons, which are summation, production, exponential, logistic, and trigonometric neurons, were employed together.

Table 5. Obtained mean accuracies (%) in classification datasets by BP, RNN, RVFL, ELM, and ALM

| Datasets     | BP   | RNN  | RFVL | ELM  | ALM  | BP   | RNN  | RFVL | ELM  | ALM  |
|--------------|------|------|------|------|------|------|------|------|------|------|
| Lithuanian   | 95.521 | 96.750 | 96.650 | 97.275 | 97.725 | 93.988 | 95.000 | 94.900 | 95.200 | 95.400 |
| Highleyman   | 91.538 | 92.950 | 90.125 | 94.350 | 95.500 | 88.536 | 88.200 | 87.800 | 91.700 | 90.300 |
| Banana Shaped | 98.746 | 98.600 | 98.100 | 98.950 | 99.250 | 96.646 | 96.800 | 96.500 | 97.400 | 97.300 |
| Spherical    | 83.823 | 85.250 | 84.775 | 87.000 | 87.975 | 78.908 | 78.600 | 79.300 | 79.200 | 79.400 |

Figure 6. The Lithuanian dataset
Table 6. Obtained mean accuracies in regression datasets by GRNN, RNN, RVFL, ELM, and ALM.

| Datasets                    | GRNN | RNN | RFVL | ELM | ALM |
|-----------------------------|------|-----|------|-----|-----|
| Approximate Sinc            | 0.574| 0.574| 0.574| 0.570| 0.565|
| Forest Fire                 | 0.057| 0.050| 0.045| 0.050| 0.040|
| CASP 5-9                    | 0.100| 0.037| 0.039| 0.037| 0.034|
| Istanbul Stock Exchange     | 0.187| 0.101| 0.138| 0.100| 0.073|
| Delta Ailerons              | 0.122| 0.071| 0.075| 0.071| 0.068|
| Ailerons                    | 0.000| 0.000| 0.000| 0.000| 0.000|
| Auto-Price                  | 0.152| 0.004| 0.084| 0.004| 0.004|
| Bank-SFM                    | 0.186| 0.043| 0.048| 0.042| 0.039|
| Boston Housing              | 0.162| 0.037| 0.099| 0.036| 0.023|
| Breast Cancer               | 0.264| 0.046| 0.258| 0.043| 0.000|
The results in Tables 5 and 6 show that the proposed approach is good enough compared to other employed methods, which are ALM, BP/GRN, RNN, RVFL, and ELM. It can be seen from these results that the training success of the ALM is higher than the other employed methods in each of the datasets. Similarly, in most of the datasets, the success of the ALM is higher than the others. Another metric in the applicability of a machine learning method is computational complexity. In order to compare the proposed ALM with other employed methods in terms of the computational complexity, the process duration, which is in second (s) were used. Obtained time durations in regression and classification datasets are summarized in Tables 7 and 8, respectively.

Table 7. Processing duration in classification datasets by BP, RNN, RVFL, ELM, and ALM

| Datasets       | Training Time (s) | Testing Time (s) |
|----------------|-------------------|------------------|
|                | BP    | RNN   | RFVL  | ELM   | ALM  | BP     | RNN   | RFVL  | ELM   | ALM  |
| Lithuanian     | 0.794 | 0.018 | 0.021 | 0.030 | 0.031| 0.058  | 0.011 | 0.003 | 0.030 | 0.033|
| Highleyman     | 0.350 | 0.019 | 0.014 | 0.023 | 0.026| 0.056  | 0.011 | 0.002 | 0.023 | 0.026|
| Banana Shaped  | 0.372 | 0.015 | 0.022 | 0.031 | 0.043| 0.044  | 0.014 | 0.002 | 0.031 | 0.038|
| Spherical      | 0.248 | 0.017 | 0.020 | 0.022 | 0.022| 0.025  | 0.012 | 0.003 | 0.022 | 0.025|
| Multi-Class    | 1.703 | 0.017 | 0.022 | 0.028 | 0.041| 0.027  | 0.017 | 0.005 | 0.028 | 0.040|
| Datasets                       | GRNN | RNN | RFVL | ELM | ALM   | GRNN | RNN | RFVL | ELM | ALM   |
|-------------------------------|------|-----|------|-----|-------|------|-----|------|-----|-------|
| Abalone                       | 5.163| 0.036| 0.029| 0.041| 0.056 | 0.019| 0.012| 0.003| 0.041| 0.045 |
| Pima Indian Diabetes          | 3.333| 0.041| 0.022| 0.047| 0.058 | 0.020| 0.014| 0.006| 0.047| 0.052 |
| Breast tissue                 | 1.338| 0.019| 0.010| 0.019| 0.024 | 0.003| 0.010| 0.003| 0.019| 0.028 |
| Statlog (Shuttle)             | 28.273| 5   | 1.126| 0.668| 1.091 | 1.389| 0.075| 0.100| 1.166| 1.091 |
| Banknote authentication       | 1.344| 0.054| 0.043| 0.055| 0.056 | 0.031| 0.015| 0.007| 0.055| 0.071 |
| Balance Scale                 | 1.045| 0.037| 0.024| 0.036| 0.052 | 0.028| 0.011| 0.006| 0.036| 0.041 |
| Skin segmentation             | 164.533| 4  | 4.689| 2.286| 4.501 | 6.029| 0.216| 0.354| 0.359| 4.501 |
| EEG eye state                 | 1138.43| 4  | 0.317| 0.207| 0.332 | 0.393| 0.059| 0.028| 0.052| 0.332 |
| Seismic Bumps                 | 18.723| 0.088| 0.078| 0.082| 0.123 | 0.022| 0.016| 0.016| 0.082| 0.099 |
| Fertility                     | 1.363| 0.018| 0.010| 0.013| 0.015 | 0.028| 0.007| 0.001| 0.013| 0.014 |
| Glass                         | 4.694| 0.026| 0.012| 0.026| 0.030 | 0.014| 0.010| 0.005| 0.026| 0.030 |
| Haberman                      | 1.281| 0.035| 0.014| 0.032| 0.043 | 0.016| 0.009| 0.003| 0.032| 0.037 |
| Hayes-Roth                    | 0.883| 0.021| 0.009| 0.017| 0.018 | 0.017| 0.007| 0.001| 0.017| 0.021 |
| QSAR biodegradation           | 100.597| 0.045| 0.054| 0.050| 0.068 | 0.005| 0.016| 0.008| 0.050| 0.063 |

Table 8. Processing duration in regression datasets by BP, RNN, RVFL, ELM, and ALM
Although ALM is not faster than the other methods as seen in Tables 7 and 8, the obtained process durations are still in an acceptable range.

### 3.3. Discussion

Given results in Tables 5-8 are summarized in Table 9 based on employed methods, which are ALM, BP, GRN, RNN, RVFL, and ELM.

**Table 9.** Summarized results achieved by BP, GRNN, RNN, RVFL, ELM, and ALM

| Phase       | Method  | Accuracy (%) | Time (s) | RMSE  | Time (s) |
|-------------|---------|--------------|----------|-------|----------|
| **Training**|         |              |          |       |          |
|             | BP/GRNN | 91.697       | 180.642  | 0.163 | 0.049    |
|             | RNN     | 92.551       | 0.247    | 0.077 | 0.187    |
|             | RFVL    | 91.258       | 0.139    | 0.117 | 0.093    |
|             | ELM     | 92.873       | 0.241    | 0.075 | 0.179    |
|             | ALM     | **94.237**   | 0.316    | **0.070** | 0.229   |
| **Testing** |         |              |          |       |          |
|             | BP/GRNN | 77.660       | 0.035    | 0.171 | 1.025    |
|             | RNN     | 78.199       | 0.028    | 0.130 | 0.021    |
|             | RFVL    | 77.882       | **0.025** | 0.106 | **0.018** |
|             | ELM     | 78.962       | 0.241    | 0.112 | 0.179    |
|             | ALM     | **79.841**   | 0.301    | **0.105** | 0.223   |
As seen in Table 9, although the best mean process durations were obtained by RVFL, the best mean training and testing success rates were obtained by the proposed ALM. The success rates achieved by ALM can also be increased by optimizing the neuron types that were employed in the hidden layer. As it was described in the previous section, the neuron types were not optimized to achieve faster results and all algebraic neurons were employed together. Based on the results reported in Tables 5, 6 and 9, the reason behind these successes achieved by ALM maybe its higher ability in modeling by providing higher nonlinearity [27], because of employing algebraic operators such as summation, production, exponential, logistic, and trigonometric neurons.

The main contributions of this paper to the literature are (1) employing algebraic operators, which are summation, production, exponential, logistic, and trigonometric operators, alone or together instead of using only summation or subtraction operators (2) providing higher nonlinearity by using different algebraic operators instead of summation, and (3) higher success rates were achieved because of having higher modeling ability.

Although the ALM requires more time in training and testing compared with RNN, ELM, and RVFL, it is faster than BP. Therefore, the computational cost of ALM found in an acceptable range. On the other hand, determining the optimal neuron types in the hidden layer is a novel issue, which may be solved by employing trials or by an expert (based on the modeled system or phenomenon). This may be noted as a disadvantage while optimizing is a common problem in all types of ANNs.

4. Conclusion

In this paper, a novel artificial neural network method, which was called ALM, was proposed. In this approach, various types of algebraic operators (functions), which are summation, production, exponential, logistic, and trigonometric neurons, can be employed together or alone in the core of the neurons in the hidden layer instead of using the only summation as in traditional ANN methods. To the authors’ best knowledge this is the first attempt to evaluate using different algebraic operators in neurons. Furthermore, using different operators yield providing higher nonlinearity and therefore, a higher ability to model a system or a phenomenon. Obtained results by the proposed approach were compared with some other popular ANNs such as BP, RNN, RVFL, and ELM. The proposed approach found successful enough in terms of both achieved success rates and process speeds.
Compliance with Ethical Standards

Conflict of Interest: Author Ömer Faruk Ertuğrul declares that he has no conflict of interest.

Ethical Approval: This article does not contain any studies with human participants or animals performed by any of the authors.

References

[1] Ertuğrul, Ö. F., & Tağluk, M. E. (2017). A novel machine learning method based on generalized behavioral learning theory. Neural Computing and Applications, 28(12), 3921-3939.

[2] Chen, M., Challita, U., Saad, W., Yin, C., & Debbah, M. (2019). Artificial neural networks-based machine learning for wireless networks: A tutorial. IEEE Communications Surveys & Tutorials, 21(4), 3039-3071.

[3] Eskov, V. M., Pyatin, V. F., Eskov, V. V., & Ilyashenko, L. K. (2019). The heuristic work of the brain and artificial neural networks. Biophysics, 64(2), 293-299.

[4] Amer, M., & Maul, T. (2019). A review of modularization techniques in artificial neural networks. Artificial Intelligence Review, 52(1), 527-561.

[5] McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics, 5(4), 115-133.

[6] Reyes-Galaviz, O. F., Pedrycz, W., He, Z., & Pizzi, N. J. (2017). A supervised gradient-based learning algorithm for optimized entity resolution. Data & Knowledge Engineering, 112, 106-129.

[7] Migdal-Najman, K., Najman, K., & Antonowicz, P. (2019). Early warning against insolvency of enterprises based on a self-learning artificial neural network of the SOM type. In Effective Investments on Capital Markets (pp. 165-176). Springer, Cham.

[8] Zhang, L., & Suganthan, P. N. (2016). A survey of randomized algorithms for training neural networks. Information Sciences, 364, 146-155.

[9] Ertam, F., & Aydın, G. (2017, October). Data classification with deep learning using Tensorflow. In 2017 International Conference on Computer Science and Engineering (UBMK) (pp. 755-758). IEEE.

[10] Cao, W., Wang, X., Ming, Z., & Gao, J. (2018). A review on neural networks with random weights. Neurocomputing, 273, 278-287.
[11] Hunt, K. J., Sbarbaro, D., Żbikowski, R., & Gawthrop, P. J. (1992). Neural networks for control systems—a survey. *Automatica, 28*(6), 1083-1112.

[12] Leung, F. H. F., Lam, H. K., Ling, S. H., & Tam, P. K. S. (2003). Tuning of the structure and parameters of a neural network using an improved genetic algorithm. *IEEE Transactions on Neural networks, 14*(1), 79-88.

[13] Hill, T., Marquez, L., O’Connor, M., & Remus, W. (1994). Artificial neural network models for forecasting and decision making. *International journal of forecasting, 10*(1), 5-15.

[14] Li, Y., & Ma, W. (2010, October). Applications of artificial neural networks in financial economics: a survey. In *2010 International symposium on computational intelligence and design* (Vol. 1, pp. 211-214). IEEE.

[15] Li, H., Wang, X., & Ding, S. (2018). Research and development of neural network ensembles: a survey. *Artificial Intelligence Review, 49*(4), 455-479.

[16] Liu, W., Wang, Z., Liu, X., Zeng, N., Liu, Y., & Alsaadi, F. E. (2017). A survey of deep neural network architectures and their applications. *Neurocomputing, 234*, 11-26.

[17] Schmidt, W. F., Kraaijveld, M. A., & Duin, R. P. (1992, August). Feed forward neural networks with random weights. In *IEEE, 11th International Conference on Pattern Recognition* (pp. 1-4).

[18] Erb, R. J. (1993). Introduction to backpropagation neural network computation. *Pharmaceutical research, 10*(2), 165-170.

[19] Pao, Y. H., Park, G. H., & Sobajic, D. J. (1994). Learning and generalization characteristics of the random vector functional-link net. *Neurocomputing, 6*(2), 163-180.

[20] Cigizoglu, H. K. (2005). Generalized regression neural network in monthly flow forecasting. *Civil Engineering and Environmental Systems, 22*(2), 71-81.

[21] Huang, G. B., Zhu, Q. Y., & Siew, C. K. (2006). Extreme learning machine: theory and applications. *Neurocomputing, 70*(1-3), 489-501.

[22] Duin, R. P., Juszczak, P., Paclik, P., Pekalska, E., De Ridder, D., Tax, D. M., & Verzakov, S. (2007). Prtools4. 1, a matlab toolbox for pattern recognition. Delft University of technology, 60.

[23] Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA Univ. California, Sch. Inf. Comput. Sci., 2013.

[24] Huang, G. B., Zhu, Q. Y., & Siew, C. K. (2004, July). Extreme learning machine: a new learning scheme of feedforward neural networks. In *2004 IEEE international joint conference on neural networks (IEEE Cat. No. 04CH37541)* (Vol. 2, pp. 985-990). IEEE.
[25] LIACC, (2020) http://www.dcc.fc.up.pt/~ltorgo/Regression/DataSets.html.”

[26] Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural networks, 4*(2), 251-257.

[27] Ertuğrul, Ö. F. (2018). Two novel versions of randomized feed forward artificial neural networks: stochastic and pruned stochastic. *Neural Processing Letters, 48*(1), 481-516.
Figures

Figure 1

The structure of the single hidden layer feed-forward artificial neural network
Figure 2

Obtained accuracies based on number of neurons and activation function in the Lithuanian: a) all neuron types, b) summation neurons, c) production neurons, d) exponential, e) logistic, f) trigonometric

Figure 3
Obtained accuracies based on number of neurons and activation function in the liver: a) all neuron types, b) summation neurons, c) production neurons, d) exponential, e) logistic, f) trigonometric

Figure 4

Obtained accuracies based on number of neurons and activation function in the approximate sinc: a) all neuron types, b) summation neurons, c) production neurons, d) exponential, e) logistic, f) trigonometric
Figure 5

Obtained accuracies based on number of neurons and activation function in the forest fire: a) all neuron types, b) summation neurons, c) production neurons, d) exponential, e) logistic, f) trigonometric

Figure 6

The Lithuanian dataset