Online Sequential Extreme Learning Machine based Intrinsic Plasticity for Classification

Zongying Liu
Faculty of Information Technology
King Mongkut's Institute of Technology Ladkrabang
Bangkok 10520, Thailand
nxlzy17@gmail.com

Kitsuchart Pasupa
Faculty of Information Technology
King Mongkut's Institute of Technology Ladkrabang
Bangkok 10520, Thailand
kitsuchart@it.kmitl.ac.th

***Abstract*—Random determination of input weights leads to unstable performance in Online Sequential Extreme Learning Machines (OS-ELM), so obtaining reliable input weights was expected to improve the model performance. We designed a new model—the OS-ELM based Intrinsic Plasticity with a new weight selection scheme (NOS-ELM-IP) to enhance the forecast stability and accuracy for classification. In this model, the input weights were selected by a new weight selection method, which replaced the original random selection part in OS-ELM. Moreover, the Intrinsic Plasticity idea was used to find the gain and bias, used in the sequential training part of OS-ELM. It maximized the information of hidden neurons and enlarged the memory. The experimental results show that the proposed new weight selection method and Intrinsic Plasticity rule enhanced the overall performance in classification tasks for binary and multi-class data sets.**

***Index Terms*—Online Sequential Learning, Intrinsic Plasticity, Xavier, Uniformed Distribution.**

I. Introduction

Classification is a general problem in pattern recognition. It is considered as a supervised learning task that seeks the relationship between a set of given input features and its corresponding target value. Many classical algorithms, such as Logistic Regression [1], Naïve Bayes [2], Fisher's Discriminant Analysis [3], Support Vector Machine [4], k-nearest neighbor [5], were designed to solve the binary and multi-class classification problems. With a dramatic increase in data size and hardware performance, many algorithms have been introduced or re-introduced with promising results and employed in our daily life [6].

Generally, traditional machine learning algorithms are trained offline by a batch of data. However, with a broad range of applications in the real world emerging, combined with an explosion of data, streaming data frequently appears now. Thus, online learning is necessary as it can update the model when a new data point arrives [7]. Online algorithms are more effective than offline ones for a large collections of data, that is steadily growing and gradually changed overtime [8]. Many online learning algorithms have been described, *e.g.* online support vector machine [9] and online multiple kernel learning for classification [10].

Extreme Learning Machines (ELM) are thousands time faster than traditional feed-forward neural networks, such as back-propagation [11], because input weights are randomly selected and they do not iterate. However, it still has drawbacks in applications: For example, an ELM cannot quickly update the model after a new sample arrived, due to because it relies an off-line learning algorithm, which limits the development of real-world applications. In 2006, Liang *et al.* described an on-line learning algorithm, their "Online Sequential Extreme Learning Machine" (OS-ELM) [12]. This is a versatile sequential learning algorithm. Chunks of training samples, with fixed or varied length, are presented one-by-one to the training algorithm which can train it in any order. Although it solves the model updating problem, the random selection of input weights still leads to unstable prediction results with the same parameter settings. This directly affects performance of the OS-ELM model.

Random weight initialization can lead to poor performance, not only in ELM, but also in many other algorithms, for example an Echo State Network (ESN) [13] or deep learning [14]. Recently, Wu *et al.* showed that uniform random weight distribution, in ESN, led to better performance than a Gaussian distribution [13]. Glorot and Begio developed a new initialization scheme—the Xavier initialization method—that used a uniform distribution and normalized initialization method to approximately satisfy the dual objectives of maintaining activation and back-propagated gradients variances [14]. The Xavier method was used to select the weights in deep neural networks to enhance performance [15], [16]. Further, the range of random selection also affects the performance of the random projection ESN and ELM models [17], [18]. If the initial weights are too large or too small, the model may start with saturated neurons and be slow or fail to converge. To the best of our knowledge, no one has used the Xavier initialization method in OS-ELM and controlled the bounds of input weights to find the proper values of the input weights for OS-ELM.

In 2005, Intrinsic Plasticity (IP) was introduced by Triesch [19]. It assumed that the neuron firing rate distribution was approximately exponential. Schrauwen *et al.* improved ESN robustness by introducing IP into the model [20]. Using the IP idea, the probability density of a neuron output was tuned to an exponential distribution, which maximized the information of hidden neurons. Further, IP enabled the connectivity matrix of the reservoir in ESN to have a much

large memory. Thus, the model performed better than using a Laplace or Gaussian distribution to connect the nodes of reservoir. It was also successfully applied in ELM [21].

To avoid the limitations mentioned above, we describe a new weight selection approach for OS-ELM, so that its neurons will not start training in saturation and the algorithm will be robust. Further, we added IP to improve OS-ELM robustness, because it had similar characteristic to ELM and ESN, leading to an improved method, NOS-ELM-IP.

The paper is organized as follows. Section II briefly sets out the OS-ELM algorithm. The following section introduces the new weight selection technique, IP rule, and our algorithm. Section IV describes our experiments. Section V compares performance, with the conventional OS-ELM, on a well-studied set of benchmarks and a real-world data set.

II. Review of OS-ELM

ELM has attracted many researchers, because of its speed and performance and many ELM variants have been applied to real-world applications. Because many applications have real-time streaming data, an online sequential model is necessary. This model updates itself based on incoming samples. This section explains an ELM variant, "OS-ELM", introduced by Liang and her colleagues [12].

OS-ELM is a single hidden layer feedforward network with radial basis function (RBF) hidden nodes in a unified framework [12]. There are two main parts in its training phase: initialization and online sequential learning parts. Assuming that X is an input matrix, with dimensions of $N \times D$, where N is the number of samples and D is the number of features. The input matrix comes with its corresponding output column vector, $\boldsymbol{y}$, with length D. There are u neurons in the hidden layer of OS-ELM with RBF $g(\cdot)$ as an activation function. An input weight matrix $\mathrm{W} \in \mathcal{R}^{u \times D}$ and a bias vector $\boldsymbol{b} \in \mathcal{R}^{u \times 1}$ are randomly selected.

In initialization, the initial training size is defined as c where $c < N$. The initial hidden matrix, H_0, can be computed by $g(\mathrm{W}_0\mathrm{X}_0^\top + \boldsymbol{b}_0)$, where X_0 is the initial training samples matrix, W_0 are the initial weights and $\boldsymbol{b}_0$ is an initial bias. Thus, the initial output weights, $\boldsymbol{\beta}_0$, can be calculated:

$$\boldsymbol{\beta}_0 = \mathrm{Z}_0\mathrm{H}_0^\top \boldsymbol{y}_0, \tag{1}$$

where Z_0 is an interval matrix, that can be calculated from $(\mathrm{H}_0^\top \mathrm{H}_0)^{-1}$, and $\boldsymbol{y}_0$ represents the initial target data.

Then the sequential learning part starts from the samples, $\boldsymbol{x}_{c+1}$ to $\boldsymbol{x}_N$. The partial hidden layer output matrix, at l-th sequential training data,

$$\mathrm{H}_l = g(\mathrm{W} \cdot \boldsymbol{x}_l + \boldsymbol{b}), \tag{2}$$

where W are the input weights, $\boldsymbol{b}$ is a bias and $\boldsymbol{x}_l$ is the sequential training data at sample l, $l = [c+1, \ldots, N]$. Thus, the output weights, $\boldsymbol{\beta}_l$, can be updated by

$$\boldsymbol{\beta}_l = \boldsymbol{\beta}_{l-1} + \mathrm{Z}_l\mathrm{H}_l^\top (y_l - \mathrm{H}_l\boldsymbol{\beta}_{l-1}), \tag{3}$$

where $\boldsymbol{\beta}_{l-1}$ are the previous output weights. Z_l is an interval matrix:

$$\mathrm{Z}_l = \mathrm{Z}_{l-1} - \mathrm{Z}_{l-1}\mathrm{H}_l^\top (\mathrm{I} + \mathrm{H}_l\mathrm{Z}_{l-1}\mathrm{H}_l^\top)^{-1}\mathrm{H}_l\mathrm{Z}_{l-1}, \tag{4}$$

where I is a sparse identity matrix.

III. Methodology

In this paper, we describe a new method to overcome the drawbacks of OS-ELM and enhance classification performance. Firstly, we replaced the random weight selection in OS-ELM, by introducing a new weight selection method, that reduces forecasting instability and improves prediction. Then, the IP rule was applied in the online sequential part to produce the desired output distributions of the hidden neurons.

A. New Weight Selection

Due to the random selection for ELM input weights, output forecasts can be unstable, even using the same parameters. Thus, a new weight selection inspired by Xavier initialization was used, instead of random selection to generate more stable results.

The variance of initial weights of the i-layer in the deep learning model was

$$\mathrm{Var}(\mathrm{W}_{ini}^i) = \frac{2}{n_i + n_{i+1}}, \tag{5}$$

where n_j is the number of neurons in the j-th hidden layer. To maintain the normal distribution in each layer, the weights of each layer were computed by the random selection with normal distribution divided by the square root of the number of samples [16], [22]. Therefore, the initial weights, W_{ini}, can be calculated based on Xavier:

$$\mathrm{W}_{ini} = \frac{\mathrm{R}}{\sqrt{N}}, \tag{6}$$

where R is a random matrix from a normal distribution, $\mathrm{R} \in \mathcal{R}^{N \times u}$ and N is the number of samples in the training data. Similarly, the bias $\boldsymbol{b}$ can be generated randomly with the dimensions of $u \times 1$.

It was reported that the range of random selection in input weights has an impact on the performance in ELM and ESN [17], [18]—the smaller the range, the better the performance. The original OS-ELM used weights $\in -1 \ldots 1$ in the input weight matrix. Here, we use a smaller range by scaling them with the following conditions:

$$\mathrm{W} = \begin{cases} \mathrm{W}_{ini} \cdot w_{\max}, & \text{if } w_{\max} < 1 \\ \mathrm{W}_{ini} \cdot \frac{1}{w_{\max}}, & \text{if } w_{\max} > 1 \end{cases} \tag{7}$$

where $w_{\max}$ is the maximum value of the elements in $|\mathrm{W}_{ini}|$, $\max(|\mathrm{W}_{ini}|)$. Therefore, the random input weight matrix in OS-ELM was replaced by new weights W.

B. Intrinsic Plasticity

IP is an unsupervised learning rule. It mainly focuses on seeking the best gain ($\boldsymbol{u}$) and bias (V) in each IP epoch based on the sequential training data. Schrauwen *et al.* successfully applied IP in ESN [20]. Here, we modified the IP rule and used it in the OS-ELM sequential learning part to improve classification.

If the initial gain is $\boldsymbol{u}$ ($[(N-c)\times 1]$) and bias is V ($[(N-c)\times D]$), these two parameters are iteratively updated by parameters $\boldsymbol{p}$ and Q, computed in equations (8) and (9), respectively.

$$\boldsymbol{p} = \eta/\boldsymbol{u} + \mathrm{Q}\boldsymbol{x}_i, \tag{8}$$

$$\mathrm{Q} = -\eta((-\mu/\sigma^2) + \boldsymbol{a}/\sigma^2(2\sigma^2 + 1 - \boldsymbol{a}^\top\boldsymbol{a} + \mu\boldsymbol{a})), \tag{9}$$

where η is a learning rate, μ represents the mean of sequential data, σ is the standard deviation of sequential data and $\boldsymbol{a}$ is the vector that can be computed based on $u_{l-c}^{(i)}$ and $\boldsymbol{v}_{l-c}^{(i)}$, that is from the element of $\boldsymbol{u}^{(i)} = [u_1^{(i)}, u_2^{(i)}, \ldots, u_{N-c}^{(i)}]^\top$ and $\mathrm{V}^{(i)} = [\boldsymbol{v}_1^{(i)}, \boldsymbol{v}_2^{(i)}, \ldots, \boldsymbol{v}_{N-c}^{(i)}]^\top$ in the i-th epoch.

$$\boldsymbol{a} = g(u_{l-c}^{(i)}\boldsymbol{x}_l + \boldsymbol{v}_{l-c}^{(i)}), \tag{10}$$

where $\boldsymbol{x}_l$ is the l-th sample in the sequential training data, $l = [l-c, \ldots, N]$.

Therefore, $\boldsymbol{u}^{(i)}$ and $\mathrm{V}^{(i)}$ can be updated by equations (11) and (12), respectively.

$$\boldsymbol{u}^{(i)} = \boldsymbol{u}^{(i-1)} + \boldsymbol{p} \tag{11}$$

$$\mathrm{V}^{(i)} = \mathrm{V}^{(i-1)} + \mathrm{Q} \tag{12}$$

The algorithm will terminate when (i) the norms of the differences between the gains and biases in current and previous epochs are smaller than a threshold (tol) or (ii) the maximum epoch $nEpoch$ is reached. In this study, the threshold tol was defined as 0.1×10^{-6} and $nEpoch$ was set to 100.

The gain and bias of IP will be updated based on the sequential training data. In the sequential part, the updated $\boldsymbol{u}^{(i)}$ and $\mathrm{V}^{(i)}$ will be used in the activation function .

C. OS-ELM based Intrinsic Plasticity with New Weight Selection

In this subsection, we used the new weight selection method and IP rule, in our NOS-ELM-IP method. In the implementation of IP rule into the sequential training part, the input data of activation function requires to be updated based on the $(l-c)$-th element of $\boldsymbol{u}^{(i)}$ and $\mathrm{V}^{(i)}$ in the i-th epoch. The updated input data can be computed by,

$$\boldsymbol{x}_{new} = u_{l-c}^{(i)}\boldsymbol{x}_l + \boldsymbol{v}_{l-c}^{(i)}, \tag{13}$$

where $l = [(l-c), \ldots, N]$, $u_{l-c}^{(i)}$ is the $(l-c)$-th observation in $\boldsymbol{u}^{(i)}$, and $\boldsymbol{v}_{l-c}^{(i)}$ is the $(l-c)$-th row of matrix, $\mathrm{V}^{(i)}$. Then, the hidden layer output matrix based on the new input data can be calculated using (2). Finally, the output weights can be computed by (3). Pseudo-code for NOS-ELM-IP is shown in Algorithm 1.

IV. Experiment Framework

The main aim of this section is to demonstrate the effectiveness of the two new methods, used in OS-ELM, *i.e.* the new input weights selection and the intrinsic plasticity rule.

A. Data sets

Tests were run on a set of commonly used benchmarks [23]–[26], available at UCI Machine Learning Repository [27], and a real-world data set. The data sets included binary class and multi-class classification tasks. With the rapid development of software and hardware, wearable devices become popularization. The increasing number of researchers pay more attention to human activity recognition. Therefore, we chose the Human Activities and Postural Transitions' Recognition using Smartphone Data (HAPT) [28] as the real-world data set to be evaluated. It consists of 3-axial linear acceleration (from the embedded accelerometer) and 3-axial angular velocity (from the gyroscope) of a smartphone. Details of each data set are in Table I.

TABLE I
Details of Data sets

Data Set	Features	Classes	Samples
Flare	9	2	144
Breast	9	2	263
Diabetes	8	2	768
Heart	13	2	270
Thyroid	5	2	215
Banana	2	2	5300
Titanic	6	2	714
German	20	2	1000
Iris	4	3	150
Twonorm	20	2	7400
HAPT	561	6	10929

B. Experimental Setting

We compared our models: (i) OS-ELM with Xavier initialization method (XOS-ELM), (ii) OS-ELM with the new weight selection approach (NOS-ELM), and (iii) OS-ELM based IP with the new weight selection approach (NOS-ELM-IP), with the conventional OS-ELM on the data sets in Table I. We evaluated performance with five-fold cross-validation on each of the ten benchmark data sets. The average classification error across five-folds was the performance metric. Since HAPT data set is partitioned into two sets—70% for the training set and 30% for test sets, we simply reported performance from the test set.

It was seen that each algorithm had parameters, that needed to be tuned. We used a simple grid search for each algorithm. In OS-ELM, the number of hidden neurons u was varied in the set [10, 20, ..., 100, 200, ..., 1000, 2000]. In XOS-ELM and NOS-ELM, the only parameter is the number of hidden neurons. Since the input weight selection technique changed, the suitable number of hidden neurons in this model also changed. Therefore, we searched parameters in the same range as those for OS-ELM. In NOS-ELM-IP, we simply set the number of hidden neurons to be the same as for NOS-ELM

Algorithm 1 Learning Phase of NOS-ELM-IP

Require: Input data matrix, X, the target value, $\boldsymbol{y}$, initial training size, c, number of training data N, number of hidden neurons, u, initial $\boldsymbol{u}^{(0)}$ and $\mathrm{V}^{(0)}$ set to one, tol = 0.1×10^{-6}, Gaussian function, $g(\cdot)$, and $\boldsymbol{a}$ = g($u^{(0)}\boldsymbol{x}_c + \boldsymbol{v}^{(0)}$).

Ensure: Output weights ($\boldsymbol{\beta}$).

Initial Training Part:

Calculate initial weights (W_{ini}) by (6); ▷ Xavier initialization method
Compute input weights by (7);
Calculate initial output weights ($\boldsymbol{\beta}_0$) by (1);
for $i \in \{1, \ldots, nEpoch\}$ **do** ▷ Intrinsic Plasticity Rule
  for $l \in \{c+1, \ldots, N\}$ **do**
    Calculate the interval parameters $\boldsymbol{p}$ and Q by (8) and (9), respectively;
    Compute $\boldsymbol{a}$ based on $u_{l-c}^{(i)}$ and $\boldsymbol{v}_{l-c}^{(i)}$ in (10).
    Update $\boldsymbol{u}^{(i)}$ and $\mathrm{V}^{(i)}$ by (11) and (12), respectively;
  end for
  if norm($\boldsymbol{u}^{(i-1)} - \boldsymbol{u}^{(i)}$) $< tol$ and norm($\mathrm{V}^{(i-1)} - \mathrm{V}^{(i)}$) $< tol$ **then**
    Break;
  end if
end for
return $\boldsymbol{u}^{(i)}$, $\mathrm{V}^{(i)}$

Sequential Training Part:

for $l \in \{c+1, \ldots, N\}$ **do**
  Calculate new input of active function by (13);
  Calculate hidden matrix by (2), based on new input, $\boldsymbol{x}_{new}$;
  Calculate output weights by (3);
end for

and searched for the IP parameter, η, using these values—$[1.0E-17, 1.5E-17, \ldots, 1.0E-12]$. The initial number of training data (L_0) was set to $\frac{N}{2}$. The optimal parameters found for each model, based on these settings, are listed in Table II.

TABLE II
PARAMETER SETTINGS FOR ALL MODELS

Data set	OS-ELM	XOS-ELM	NOS-ELM	NOS-ELM-IP	
	u	u	u	u	η
Flare	30	10	10	10	1.0×10^{-13}
Breast	40	30	30	30	1.0×10^{-14}
Diabetes	30	10	10	10	5.0×10^{-13}
Heart	20	20	20	20	2.0×10^{-12}
Thyroid	20	20	20	20	1.0×10^{-15}
Banana	50	20	20	20	1.0×10^{-17}
Titanic	30	20	20	20	1.0×10^{-14}
German	30	30	30	30	5.0×10^{-12}
Iris	10	10	10	10	1.5×10^{-13}
Twonorm	70	80	30	30	1.0×10^{-11}
HAPT	1000	500	500	500	1.5×10^{-17}

V. RESULTS AND DISCUSSION

We ran the experiment 10 times with different random splits, with the parameters in Table II. We report the mean misclassification rate for the weights selection and IP algorithm as shown in Table III and IV, respectively.

We first compared the performance of the new weight selection model, NOS-ELM, with OS-ELM and XOS-ELM. The new weight selection method clearly improved classification performance on all data sets. For all data sets, the misclassification rate in NOS-ELM reduced by 1.98% compared to OS-ELM and 0.13% compared to XOS-ELM. For NOS-ELM compared with OS-ELM, the maximum benefit appeared in the iris data set at 4.8%, while the minimum was in the Twonorm set at 0.06%. On the other hand, comparing the performance of the proposed model with XOS-ELM, the maximum improvement of NOS-ELM appeared in the Banana data set (0.4%), but there was no improvement in the Twonorm data set. Besides, the average value of the standard deviation of misclassification rate for all data sets was 1.3% in NOS-ELM, or less than that for OS-ELM (0.6%) and XOS-ELM (0.3%). This confirmed that our new weight selection method enhanced the model robustness.

Furthermore, the IP rule enhanced performance—see Table IV, which shows the difference in the misclassification rate of NOS-ELM and NOS-ELM-IP. There was no change in performance for NOS-ELM *vs* NOS-ELM-IP in Flare, Breast, Thyroid and Twonorm data sets, but NOS-ELM-IP showed improvements in the others. Overall, NOS-ELM-IP was comparable or yielded better performance than NOS-ELM.

For example, the highest improvement appeared in HAPT as the misclassification rate reduced by 1.6%, while the lowest appeared in the German data that reduced by 0.01%. Using the IP rules in NOS-ELM improved overall performance by $\sim 0.17\%$. There was no difference in the average standard deviation of misclassification rate between NOS-ELM and NOS-ELM-IP. Thus NOS-ELM-IP performed better than NOS-ELM in both binary and multi-class classification.

We used the "Wilcoxon Signed Rank Test" [29] to confirm that NOS-ELM-IP was better than NOS-ELM. Table IV shows the differences between the misclassification rates of NOS-ELM and NOS-ELM-IP. Then, the absolute differences were ranked in ascending order—see the last column. Candidates with no difference were given rank 0. Next, we calculated the sum of ranks for the positive difference R^+ and negative difference R^-, leading to $R^+ = 28$ and $R^- = 0$. Hence, $W_{stat} = 0$—$W_{stat} \leq 10$ implied that the difference was statistically significant, confirming that NOS-ELM-IP performed better than NOS-ELM.

TABLE III
MEAN MISCLASSIFICATION RATE (%) FOR THE NEW ALGORITHMS, NOS-ELM AND XOS-ELM *vs* OS-ELM. BOLDED FIGURES MARK THE BEST PERFORMANCE.

Data set	Misclassification Rate (%)		
	OS-ELM	XOS-ELM	NOS-ELM
Flare	40.51 ± 2.6	37.74 ± 1.2	**37.67 ± 1.1**
Breast	28.78 ± 1.9	27.41 ± 1.2	**27.25 ± 0.9**
Diabetes	24.75 ± 1.0	23.05 ± 0.5	**23.01 ± 0.4**
Heart	19.22 ± 1.6	17.62 ± 1.2	**17.37 ± 1.1**
Thyroid	11.25 ± 3.0	9.53 ± 1.6	**9.41 ± 1.6**
Banana	42.96 ± 7.2	39.73 ± 8.1	**39.34 ± 6.3**
Titanic	19.67 ± 0.8	19.57 ± 0.8	**19.49 ± 0.8**
German	25.28 ± 0.5	25.04 ± 0.5	**25.03 ± 0.6**
Iris	12.40 ± 1.6	7.66 ± 1.8	**7.60 ± 1.7**
Twonorm	2.31 ± 0.1	2.25 ± 0.0	**2.25 ± 0.0**
HAPT	8.54 ± 0.3	5.67 ± 0.4	**5.44 ± 0.2**
Average	21.4 ± 1.9	19.57 ± 1.6	**19.44 ± 1.3**

TABLE IV
COMPARISON OF MEAN MISCLASSIFICATION RATE (%) FOR THE PROPOSED ALGORITHM, NOS-ELM AND NOS-ELM-IP.

Data set	Misclassification Rate		Difference	Rank
	NOS-ELM	NOS-ELM-IP		
Flare	37.67 ± 1.1	37.67 ± 1.1	0.00	0
Breast	27.25 ± 0.9	27.25 ± 0.9	0.00	0
Diabetes	23.01 ± 0.4	22.99 ± 0.4	0.02	2
Heart	17.37 ± 1.1	17.25 ± 1.1	0.12	6
Thyroid	9.44 ± 1.6	9.44 ± 1.6	0.00	0
Banana	39.34 ± 6.3	39.26 ± 6.4	0.08	5
Titanic	19.49 ± 0.8	19.46 ± 0.8	0.03	3
German	25.03 ± 0.6	25.02 ± 0.5	0.01	1
Iris	7.60 ± 1.7	7.53 ± 1.7	0.07	4
Twonorm	2.25 ± 0.1	2.25 ± 0.1	0.00	0
HAPT	5.44 ± 0.2	3.83 ± 0.2	1.61	7
Average	19.44 ± 1.3	19.27 ± 1.3	0.17	

VI. CONCLUSION

We described a new model, NOS-ELM-IP, that used an improved method to select the input weights, instead of the random selection in the original OS-ELM. Adding the IP rule assisted in updating the best gain and bias, that are based on sequential data and used them in the activation function. Experiments showed that OS-ELM in conjunction with the new weight selection and IP rules enhanced overall performance on benchmark data sets. Moreover, the proposed model is more robust than the others. The new model also achieved better performance in the human activity data set than the baselines. Overall, our new NOS-ELM-IP model was the best performer in binary and multi-class classification tasks over a wide range of tested data sets.

REFERENCES

[1] C. R. Boyd, M. A. Tolson, and W. S. Copes, "Evaluating trauma care: the triss method. trauma score and the injury severity score." *The Journal of trauma*, vol. 27, no. 4, pp. 370–378, 1987.

[2] M. E. Maron, "Automatic indexing: an experimental inquiry," *Journal of the ACM (JACM)*, vol. 8, no. 3, pp. 404–417, 1961.

[3] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of eugenics*, vol. 7, no. 2, pp. 179–188, 1936.

[4] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[5] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.

[6] M. Paliwal and U. A. Kumar, "Neural networks and statistical techniques: A review of applications," *Expert systems with applications*, vol. 36, no. 1, pp. 2–17, 2009.

[7] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[8] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. USA: Prentice Hall Press, 2009.

[9] D. M. Tax and P. Laskov, "Online svm learning: from classification to data description and back," in *2003 IEEE XIII Workshop on Neural Networks for Signal Processing (IEEE Cat. No. 03TH8718)*. IEEE, 2003, pp. 499–508.

[10] S. C. Hoi, R. Jin, P. Zhao, and T. Yang, "Online multiple kernel classification," *Machine Learning*, vol. 90, no. 2, pp. 289–316, 2013.

[11] G.-B. Huang and C.-K. Siew, "Extreme learning machine: Rbf network case," in *ICARCV 2004 8th Control, Automation, Robotics and Vision Conference, 2004.*, vol. 2. IEEE, 2004, pp. 1029–1036.

[12] N.-Y. Liang, G.-B. Huang, P. Saratchandran, and N. Sundararajan, "A fast and accurate online sequential learning algorithm for feedforward networks," *IEEE Transactions on neural networks*, vol. 17, no. 6, pp. 1411–1423, 2006.

[13] Q. Wu, E. Fokoue, and D. Kudithipudi, "On the statistical challenges of echo state networks and some potential remedies," *arXiv preprint arXiv:1802.07369*, 2018.

[14] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[15] T. A. Sai and H.-h. Lee, "Weight initialization on neural network for neuro pid controller-case study," in *2018 International Conference on Information and Communication Technology Robotics (ICT-ROBOT)*. IEEE, 2018, pp. 1–4.

[16] S. K. Kumar, "On weight initialization in deep neural networks," *arXiv preprint arXiv:1704.08863*, 2017.

[17] G. Dudek, "Extreme learning machine as a function approximator: Initialization of input weights and biases," in *Proceedings of the 9th International Conference on Computer Recognition Systems CORES 2015*. Springer, 2016, pp. 59–69.

[18] D. Ye, H. Lv, Y. Jiang, Z. Wu, Q. Bao, Y. Gao, and R. Huang, "Improved echo state network (esn) for the prediction of network traffic," in *11th EAI International Conference on Mobile Multimedia Communications*. European Alliance for Innovation (EAI), 2018.

[19] J. Triesch, "A gradient rule for the plasticity of a neuron's intrinsic excitability," in *Artificial Neural Networks: Biological Inspirations – ICANN 2005*, W. Duch, J. Kacprzyk, E. Oja, and S. Zadrożny, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 65–70.

[20] B. Schrauwen, M. Wardermann, D. Verstraeten, J. J. Steil, and D. Stroobandt, "Improving reservoirs using intrinsic plasticity," *Neurocomputing*, vol. 71, no. 7-9, pp. 1159–1171, 2008.

[21] K. Neumann and J. J. Steil, "Batch intrinsic plasticity for extreme learning machines," in *International Conference on Artificial Neural Networks*. Springer, 2011, pp. 339–346.

[22] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*, 2014, pp. 675–678.

[23] A. Chaudhary, S. Kolhe, and R. Kamal, "An improved random forest classifier for multi-class classification," *Information Processing in Agriculture*, vol. 3, no. 4, pp. 215–222, 2016.

[24] D. M. Farid, L. Zhang, C. M. Rahman, M. A. Hossain, and R. Strachan, "Hybrid decision tree and naïve bayes classifiers for multi-class classification tasks," *Expert systems with applications*, vol. 41, no. 4, pp. 1937–1946, 2014.

[25] R. F. Harrison and K. Pasupa, "A simple iterative algorithm for parsimonious binary kernel fisher discrimination," *Pattern Analysis and Applications*, vol. 13, no. 1, pp. 15–22, 2010.

[26] ——, "Sparse multinomial kernel discriminant analysis (sMKDA)," *Pattern Recognition*, vol. 42, no. 9, pp. 1795–1802, 2009.

[27] D. Dua and C. Graff, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[28] J.-L. Reyes-Ortiz, L. Oneto, A. Samà, X. Parra, and D. Anguita, "Transition-aware human activity recognition using smartphones," *Neurocomputing*, vol. 171, pp. 754–767, 2016.

[29] F. Wilcoxon, "Individual comparisons by ranking methods," in *Breakthroughs in statistics*. Springer, 1992, pp. 196–202.