

A progressive learning for structural tolerance online sequential extreme learning machine

Sarutte Atsawaraungsuk¹, Wasaya Boonphairote², Kritsanapong Somsuk³, Chanwit Suwannapong⁴,
Suchart Khummanee⁵

¹Computer Education Department, Education Faculty, Udon Thani Rajabhat University, Thailand

²Language Center, Udon Thani University, Udon Thani, Thailand

³Computer and Communication Engineering Department, Technology Faculty, Udon Thani University, Thailand

⁴Faculty of Industrial Technology, Nakhon Phanom University, Thailand

⁵Computer Science Department, Informatics Faculty, Mahasarakham University, Mahasarakham, Thailand

Article Info

Article history:

Received Sep 24, 2022

Revised Mar 09, 2023

Accepted Mar 25, 2023

Keywords:

Extreme learning machine

Progressive learning

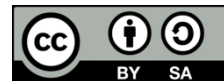
Sequential learning

Stable sequential learning

ABSTRACT

This article discusses the progressive learning for structural tolerance online sequential extreme learning machine (PSTOS-ELM). PSTOS-ELM can save robust accuracy while updating the new data and the new class data on the online training situation. The robustness accuracy arises from using the householder block exact QR decomposition recursive least squares (HBQRD-RLS) of the PSTOS-ELM. This method is suitable for applications that have data streaming and often have new class data. Our experiment compares the PSTOS-ELM accuracy and accuracy robustness while data is updating with the batch-extreme learning machine (ELM) and structural tolerance online sequential extreme learning machine (STOS-ELM) that both must retrain the data in a new class data case. The experimental results show that PSTOS-ELM has accuracy and robustness comparable to ELM and STOS-ELM while also can update new class data immediately.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Sarutte Atsawaraungsuk

Computer Education Department, Education Faculty

Udon Thani Rajabhat University, 64 Thaharn Road, Muang, Udon Thani 41000, Thailand

Email: sarutte@udru.ac.th

1. INTRODUCTION

The extreme learning machine (ELM) is a fast data training algorithm with the single-hidden layer feed-forward networks (SLFNs) structure proposed [1]-[3]. In addition, the ELM has the least-squares solution to lead to satisfactory accuracy. However, the ELM relies on a batch-mode learning mechanism. When batch-mode ELM trains new data, batch-mode ELM has to retrain all of the data in the dataset, including trained data and the new coming data. For example, ELM recognized 1,000 samples. And there is the new samples 10 samples for training. ELM must train 1,010 samples (all samples) to update their knowledge. It will always happen when new data updating. Therefore, this convention requires multiple times to retrain the data.

Online learning [4] is a method of machine learning that can update the training data in sequential order. However, the batch learning methods can create the machine learning model by learning here on the entire training data set at once. That online learning can take machine learning to learn from new data close to real-time and is used in many applications such as intrusion detection [5] and facial expression recognition [6], [7].

In the case of ELM with online learning, Liang *et al.* [8] proposed the online sequential ELM (OS-ELM) that enables online data update capability with comparable accuracy to ELM. The online update capability brings OS-ELM can update the new training data without retraining the trained data. However, OS-ELM may have a loss of information problem when updating the data continuously. When the OS-ELM has completed the new

data updating process of each training, the weight and bias matrix of OS-ELM are stored and updated in the memory. By the memory limitation, the memory has to finite the long decimal places by rounding that may affect the weight and bias loss information. This situation is called a round-off error. The round-off error will accumulate depending on the number of updating data. Therefore, The error in the data update process has to be determined to reduce the OS-ELM generalization [9].

Horata *et al.* [10] proposed structural tolerance OS-ELM (STOS-ELM), which is the applied OS-ELM to handle the OS-ELM robustness problem while updating the data. STOS-ELM works with the householder block exact QR decomposition recursive least squares (HBQRD-RLS) to enable online learning. Besides, HBQRD-RLS can reduce the effect of the rounded-off error and improve robustness in the STOS-ELM online update situation [10]-[12].

OS-ELM and STOS-ELM are matched for applications with the dataset in that they have to learn a new class of data by retraining all of the data. It might not support the applications that work with real-time arrived data where the nature of training data is unknown. Therefore, a learning technique must be adapted to this situation [9].

The progressive learning has inspiration from the human learning paradigm. Human learning can continue to learn whenever a new phenomenon has encountered. Human can resume, adapt, and grow to learn the phenomenon while still keeping existing knowledge learned thus far. Progressive learning is used in many applications such as vegetable disease recognition [13] and COVID-19 diagnosis [14].

Venkatesan and Er [15] proposed the progressive learning technique for OS ELM (POS-ELM) is OS-ELM worked with progressive learning technique. This technique can support new class data OS-ELM learning by retaining the knowledge of previous class data. It is like human learning theory. POS-ELM can break the number of class constraints of the OS-ELM in the data updating process. However, POS-ELM still has the round-off error as same as OS-ELM.

This article discusses the problems and proposes progressive learning for structural tolerance OS-ELM (PSTOS-ELM), which aims to improve the robustness of STOS-ELM in a new class data updating situation. Our experiment shows that PSTOS-ELM has higher accuracy and more robustness than POS-ELM. That effect from PSTOS-ELM has a progressive learning technique.

This article is structured as: materials and methods section describes the details of ELM, OS-ELM, STOS-ELM, and the proposed algorithm PSTOS-ELM. The experiment section explains the experimental details and results. And the last section is the conclusion.

2. METHOD

2.1. Extreme learning machine

The ELM is a fast data training algorithm [1], [2]. Given ELM have K hidden nodes with the D -dimension input. The samples could be formatted as $(x_i, t_i), i = 1, 2, \dots, N$ where $x_i \in R^D$ is the training sample members of $X_{N \times D}$ and $t_i \in R^C$ is the target sample members of $T_{N \times C}$ by C is the number of classes. ELM can be expressed in least squares form:

$$\beta_{K \times C} = H_{K \times N}^\dagger T_{N \times C} \quad (1)$$

Where $\beta_{K \times C} = [\beta_{j1}, \beta_{j2}, \dots, \beta_{jC}]^T, j = 1, 2, \dots, K$ is a matrix of output weights. H^\dagger is the pseudo-inverse of H that can calculate from the Moore-Penrose formulation. The ELM's input formulation is a linear equation that has the following description:

$$H = g(XW + B) = [h_{ij}] = [g(x_i \cdot w_j + b_j)] \quad (2)$$

Where $g(\cdot)$ is activation function and input weights $W_{D \times K} = [w_{1j}, w_{2j}, \dots, w_{Dj}]^T, j = 1, 2, \dots, K$ and biases $B_{N \times K} = [b_j, b_j, \dots, b_j]^T, j = 1, 2, \dots, K$ be randomly generated in the range $[0, 1]$ and $[-1, 1]$, respectively.

2.2. Online sequential extreme learning machine

An OS-ELM [8] can update its knowledge by training exclusively on new data. The OS-ELM calculation can be summarized. In case of the number of hidden nodes K is less than or equal to the number of samples $N(N_{k-1} + N_k)$ samples, the previously trained samples $(X_{k-1(N_{k-1} \times D)}, T_{k-1(N_{k-1} \times C)})N_{k-1}$ samples and newly delivered samples $(X_{k(N_k \times D)}, T_{k(N_k \times C)})N_k$ samples, the output weights β_k are formulate.

$$\beta_k = K_k^{-1} \begin{bmatrix} H_{k-1} \\ H_k \end{bmatrix}^T \begin{bmatrix} T_{k-1} \\ T_k \end{bmatrix} \quad (3)$$

Where k is an index of newly delivered data and $k - 1$ is the index of previously trained data.

$$K_k = \begin{bmatrix} H_{k-1} \\ H_k \end{bmatrix}^T \begin{bmatrix} H_{k-1} \\ H_k \end{bmatrix} \quad (4)$$

For sequential learning, in (4) can be rewritten as the conditions of K_{k-1} as:

$$K_k = K_{k-1} + (H_k^T H_k) \quad (5)$$

Thus, the result of the combination of (3) and (5) is:

$$\beta_k = \beta_{k-1} + K_k^{-1} H_k^T (T_k - H_k \beta_{k-1}) \quad (6)$$

In (5) can be rewritten using the Woodbury formula [16] to calculate K_k^{-1} :

$$P_k = P_{k-1} - P_{k-1} H_k^T (I + H_k^T P_{k-1} H_k)^{-1} H_k P_{k-1} \quad (7)$$

Where $P_k = K_k^{-1}$.

2.3. Structural tolerance sequential extreme learning machine

STOS-ELM [10] allows robust sequential learning to ELM by using the HBQRD-RLS to store and update the square root factor covariance matrix (\tilde{R}_k^{-1}) for the output weight (β_k) updating. When new data is delivered, STOS-ELM calculation steps are described. Initial phase, beginning with the calculation of \tilde{R}_{k-1}^{-1} at time $k = 1$ where $\tilde{R}_0^{-1} \approx H_0^\dagger$ by:

$$\tilde{R}_0^{-1} = R_0^{-1} Q^T \quad (8)$$

From $Q^T H \beta = Q^T T$ by $QR = H$, this process is called QR decomposition. Q is an orthogonal matrix that have property $Q^T Q = I_{N \times N}$ where $Q \in R^{N \times N}$ and R is $N \times K$ upper triangular matrix that has the same values as the square root of $H^T H$ or $R^T R = H^T H$. As a result, in (8) can be solved in the triangular system [17]. That the initial output weight (β_0) can be calculated by:

$$\beta_0 = \tilde{R}_0^{-1} T_0 \quad (9)$$

Sequential phase, STOS-ELM uses the HBQRD-RLS to produce the relational matrix:

$$G_k = -\tilde{R}_{k-1}^{-T} H_k^T \quad (10)$$

Where G_k is the relational matrix between newly delivered data and previously trained data.

The next step is to store and update the square root factor covariance matrix \tilde{R}_k^{-1} for F_k and E_k^T producing. By applying Lemma 1 in [18], HBQRD-RLS was working based on householder transformation [19], [20], to produce an orthogonal matrix $U(k)$ such that.

$$U(k) \begin{bmatrix} I_{N_k \times N_k} & 0_{N_k \times K} \\ G_k (N_0 \times N_k) & (\tilde{R}_{k-1}^{-1} (N_0 \times K))^T \end{bmatrix} = \begin{bmatrix} F_k (N_k \times N_k) & E_k^T (N_k \times K) \\ 0_{N_0 \times N_k} & (\tilde{R}_k^{-1} (N_0 \times K))^T \end{bmatrix} \quad (11)$$

Where I is the identity matrix and 0 is the zero matrix. The last step is to update the new output weight β_k :

$$\beta_k = \beta_{k-1} + E_k (F_k^{-1})^T (T_k - H_k \beta_{k-1}) \quad (12)$$

Where $(T_k - H_k \beta_{k-1})$, F_k and E_k^T are called Kalman gain.

2.4. Progressive structural tolerance online sequential extreme learning machine (the proposed method)

PSTOS-ELM is STOS-ELM that can learn a new class by maintaining the old knowledge (trained class). In the training phase, while the P new class data is coming to PSTOS-ELM, the output weight matrix at $k - 1$ time β_{k-1} is recalibrated to support the new class by using a recalibrated matrix $\Delta \beta_k$ that can be written.

$$\beta_{k-1} < -[\beta_{k-1} \quad \Delta \beta_k] \quad (13)$$

The recalibrated matrix ($\Delta \beta_k$) is the output weight matrix value of the not trained class that is calculated by multiple of the square root factor covariance matrix of the newly delivered data (\hat{R}_k^{-1}).

$$\Delta\beta_k = \hat{R}_{k(K \times N_k)}^{-1} \begin{bmatrix} -1 & \cdots & -1 \\ \vdots & \ddots & \vdots \\ -1 & \cdots & -1 \end{bmatrix}_{N_k \times P} \quad (14)$$

Where $\hat{R}_k^{-1} = R_k^{-1}Q^T$ that Q is Q of \hat{R}_k in QR decomposition. PSTOS-ELM is summarized in Algorithm 1.

Algorithm 1. PSTOS-ELM algorithm

Initial phase

1. Define parameters in K hidden nodes and set $k = 1$.
2. Calculate initial data X_{k-1} to the hidden layer output matrix H_{k-1} by using (2).
3. Initial $\hat{R}_{k-1}^{-1} \approx H_{k-1}^\dagger$ and corresponding solution $\beta_{k-1} = R_{k-1}^{-1}Q^T T_{k-1}$.

Sequential phase

4. When a new sample set X_k come to the system, X_k has calculated to H_k by (2).
5. If new classes are introduced, update β_{k-1} by using (13).
6. Calculate G_k as in (10).
7. Store and update \hat{R}_k^{-1} , F_k and E_k^T by using (11).
8. Update the output weights β_k at the k time of the new data is coming as in (12).
9. Plus k to 1 when the new sample set is coming to the training process, and then go to step (4).

End

3. RESULTS AND DISCUSSION

3.1. Experimental setup

This section is the experimental setup that will describe the details of the datasets and how to prepare them, start with the dataset. The six datasets are from the University of Irvine, California (UCI) [21] repository and were selected the same as Venkatesan and Er's experiment [15]. The dataset details are shown in Table 1.

Table 1. The dataset

Dataset	A number of classes	A number of features
Iris	3	4
Balance	3	4
Waveform	3	32
Wine	3	13
Satellite	5	36
Optdigits	10	63

The datasets are used to evaluate the the performance of the methods. All methods are run in MATLAB version R2014a on a computer with the environment Core i3 3.40 GHz RAM 8.00 GB. Training and testing data preparation in our experiments can be described.

- a) Separate the data of each class into two groups: 70 percent and 30 percent are training data and testing data, respectively [22].
- b) Sort the training data by the number of the class in ascending order.
- c) Define the two first classes of data to the initial data and the remaining class data to the sequential data.

This process is used to validate the performance in a new class data update situation that depends on the random input weights and biases with ten rounds. For each round, the input weights and biases will be generated in one set for all methods. The numbers of the hidden nodes of all methods are varied in the range of [1,200].

3.2. Results

3.2.1. Accuracy of PSTOS-ELM

This section reports the performance result of ELM, OS-ELM, POS-ELM [15], STOS-ELM, and PSTOS-ELM. The five methods are evaluated by their performance by using the following: the average accuracy of the 10-round test, max accuracy of the 10-round test, min accuracy of the 10-round test, standard deviation (SD) of the 10-round test, and the number of the hidden node that take the best accuracy to the methods as shown in Table 2. The bold letters show the best value of each dataset (the meta-metrics evaluation [23]-[24]).

Table 2 shows the performance of the methods over the six datasets. The results show that PSTOS-ELM has average accuracy, max accuracy, and min accuracy slightly lower than STOS-ELM, which has the highest accuracy on average accuracy. The difference between the average, max, and min accuracy of STOS-ELM and PSTOS-ELM are 0.0017, 0.0019, and 0.0033, respectively.

Table 2. The performance of methods

Dataset	Method	Accuracy	Max	Min	SD	A number of node
Balance	ELM	0.9143	0.9206	0.9101	0.0033	15
	OS-ELM	0.9143	0.9206	0.9101	0.0033	15
	STOS-ELM	0.9143	0.9206	0.9101	0.0033	15
	POS-ELM	0.8360	0.8889	0.7831	0.0402	58
	PSTOS-ELM	0.9122	0.9206	0.8995	0.0062	17
Iris	ELM	0.9667	0.9778	0.9333	0.0157	21
	OS-ELM	0.9622	1.0000	0.9111	0.0235	105
	STOS-ELM	0.9689	0.9778	0.9333	0.0155	21
	POS-ELM	0.9622	0.9778	0.9111	0.0211	104
	PSTOS-ELM	0.9644	0.9778	0.9333	0.0187	21
Optdigits	ELM	0.8200	0.8215	0.8186	0.0009	195
	OS-ELM	0.5785	0.6525	0.5201	0.0390	28
	STOS-ELM	0.8199	0.8209	0.8186	0.0008	194
	POS-ELM	0.3471	0.5437	0.2417	0.0881	118
	PSTOS-ELM	0.8198	0.8209	0.8186	0.0007	194
Satellite	ELM	0.7342	0.7378	0.7311	0.0020	200
	OS-ELM	0.5251	0.5959	0.4514	0.0569	89
	STOS-ELM	0.7342	0.7378	0.7311	0.0020	200
	POS-ELM	0.4771	0.6054	0.2912	0.1181	86
	PSTOS-ELM	0.7342	0.7372	0.7311	0.0020	200
Waveform	ELM	0.8580	0.8594	0.8568	0.0009	190
	OS-ELM	0.8211	0.8354	0.7808	0.0203	184
	STOS-ELM	0.8578	0.8594	0.8568	0.0008	190
	POS-ELM	0.7783	0.8315	0.6749	0.0524	147
	PSTOS-ELM	0.8577	0.8588	0.8561	0.0008	190
Wine	ELM	0.9836	1.0000	0.9818	0.0057	42
	OS-ELM	0.9855	1.0000	0.9818	0.0077	42
	STOS-ELM	0.9855	1.0000	0.9818	0.0077	42
	POS-ELM	0.9818	0.9818	0.9818	0.0000	28
	PSTOS-ELM	0.9818	0.9818	0.9818	0.0000	12
Average accuracy	ELM	0.8795	0.8862	0.8719	0.0040	–
	OS-ELM	0.7978	0.8341	0.7592	0.0258	–
	STOS-ELM	0.8801	0.8861	0.8719	0.0040	–
	POS-ELM	0.7304	0.8032	0.6473	0.0505	–
	PSTOS-ELM	0.8784	0.8828	0.8701	0.0050	–

3.2.2. Robustness with the best-hidden node when data updating

This experiment simulates randomization of weight and bias situations that aims to analyze the robustness and accuracy of the methods on a 10-rounds test with different weights and biases in data updating situations. The methods use the best-hidden node that is shown in Table 2. Figure 1(a) to Figure 1(e), Figure 2(a) to Figure 2(e), and Figure 3(a) to Figure 3(e) show the average accuracy (red line), max accuracy (green line), and min accuracy (blue line) of ELM, OS-ELM, STOS-ELM, POS-ELM, and PSTOS-ELM with the appropriate number of the hidden node. Each figure demonstrates the accuracy of three datasets: optdigits, satellite, and balance. The x -axis in the figure shows the percent of data updating that starts with data in the third class of the dataset. If the dataset has more than three classes, the red line will have appeared in the percent that the new class data has trained.

From Figure 1 to Figure 3, the results can be divided into 2 groups as follows. In the first group (Figure 1 to Figure 2), ELM (Figure 1 to Figure 2(a)), STOS-ELM (Figure 1 to Figure 2(c)), and PSTOS-ELM (Figure 1 to Figure 2(e)) have accuracy that tends to grow with robustly. POS-ELM (Figure 1 to Figure 2(d)) has an accuracy trend to grow, but POS-ELM has less accuracy and robustness than ELM. For OS-ELM (Figure 1 to Figure 2(b)), the accuracy trend is like POS-ELM, but the min accuracy trend is lower than. Optdigits, satellite, and waveform datasets are in the first group. In the second group (Figure 3), all of the methods have the same growth trends. Balance, iris, and wine are in the second group.

The extra case of the satellite dataset has one different point from the first group (Figure 2). After OS-ELM updates the data on the fifth class to OS-ELM (Figure 2(b)), OS-ELM has a down accuracy trend. On the other hand, POS-ELM (Figure 2(d)) has an up-accuracy trend. However, OS-ELM has accuracy comparable to POS-ELM.

3.2.3. Robustness over a different number of nodes when data updating

This experiment simulates the accuracy of each hidden node. The experiment aims to analyze the robust and accuracy of the methods in the different numbers of hidden nodes. Figure 4 to Figure 6 show the accuracy of ELM, OS-ELM, STOS-ELM, POS-ELM, and PSTOS-ELM in each percent of updated data (x -axis) with the different numbers of hidden nodes. Each figure demonstrates the accuracy in three datasets: satellite, iris, and wine.

From Figure 4(a) to Figure 6(e), the results can be separated into three groups as follows. In the first group (Figure 4), ELM (Figure 4(a)), STOS-ELM (Figure 4(c)), and PSTOS-ELM (Figure 4(e)) have accuracy that tends to grow in all hidden nodes. On the other hand, OS-ELM (Figure 4(b)) and POS-ELM (Figure 4(d)) accuracy trend up slightly and then drop in some hidden nodes. Satellite, balance, optdigits, and waveform datasets are in the first group.

In the second group (Figure 5), STOS-ELM (Figure 5(c)) and PSTOS-ELM (Figure 5(e)) have an accuracy trend to grow in all hidden nodes. On the other hand, ELM (Figure 5(a)), OS-ELM (Figure 5(b)), and POS-ELM (Figure 5(d)) accuracy trend up slightly and then drop in some hidden nodes. Iris dataset is in the second group.

In the third group (Figure 6), OS-ELM (Figure 6(b)), POS-ELM (Figure 6(d)), STOS-ELM (Figure 6(c)), and PSTOS-ELM (Figure 6(e)) have an accuracy trend to grow in all hidden nodes. Only ELM (Figure 6(a)) has an accuracy trend that grows slightly, drops sharply, and then rises at 100 hidden nodes. And at 150 and 200 hidden nodes, ELM trend accuracy is less grower than the other method. Iris dataset is in the third group.

As seen in Figure 4 to Figure 6, accuracy in some hidden nodes has a downtrend. It noticed how final accuracy has robustness in a wide range of the number of hidden nodes. The notice will be issued to find in the next section.

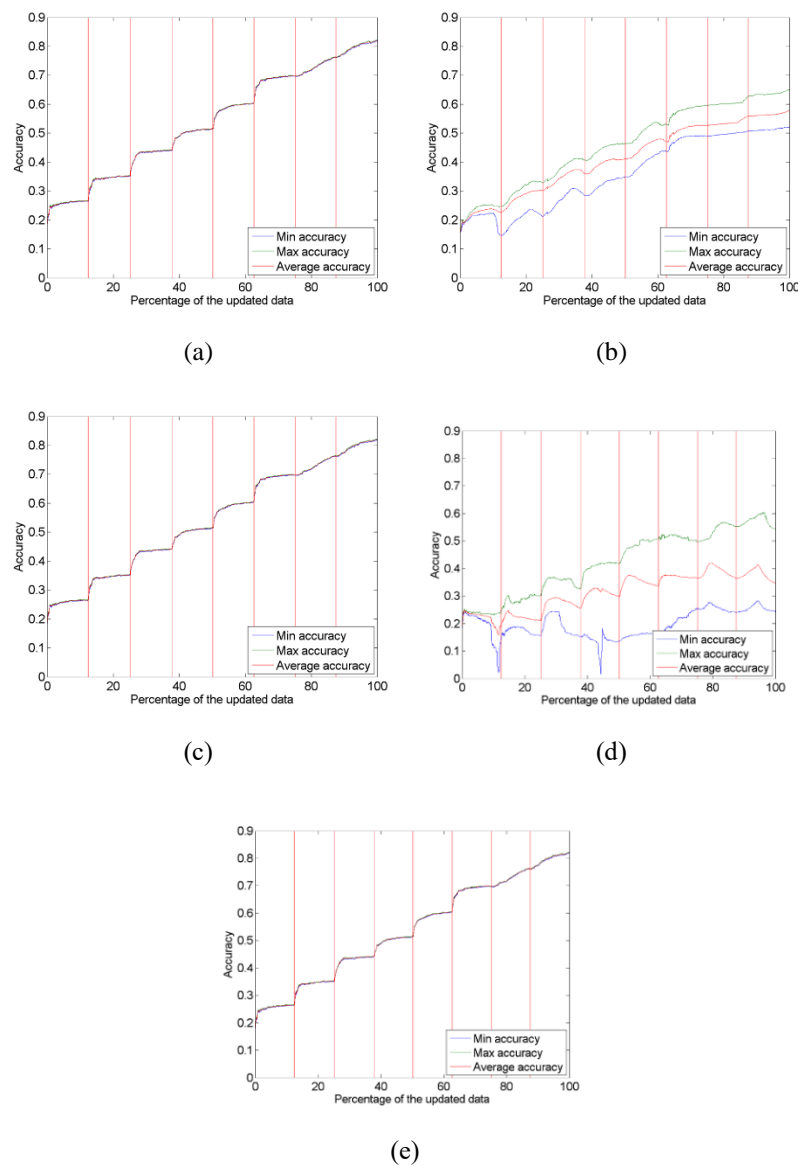


Figure 1. The accuracy of the methods while data updating with their best hidden node in optdigits dataset: (a) ELM, (b) OS-ELM, (c) STOS-ELM, (d) POS-ELM, and (e) PSTOS-ELM

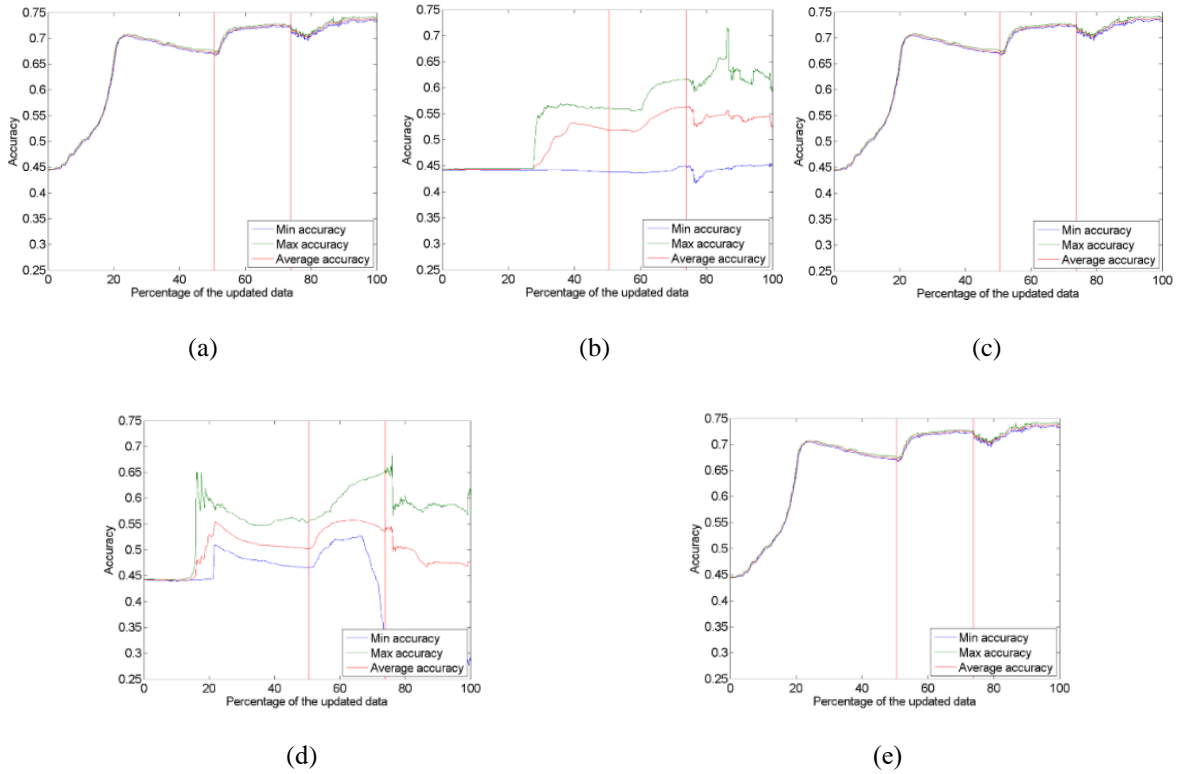


Figure 2. The accuracy of the methods while data updating with their best hidden node in satellite dataset: (a) ELM, (b) OS-ELM, (c) STOS-ELM, (d) POS-ELM, and (e) PSTOS-ELM

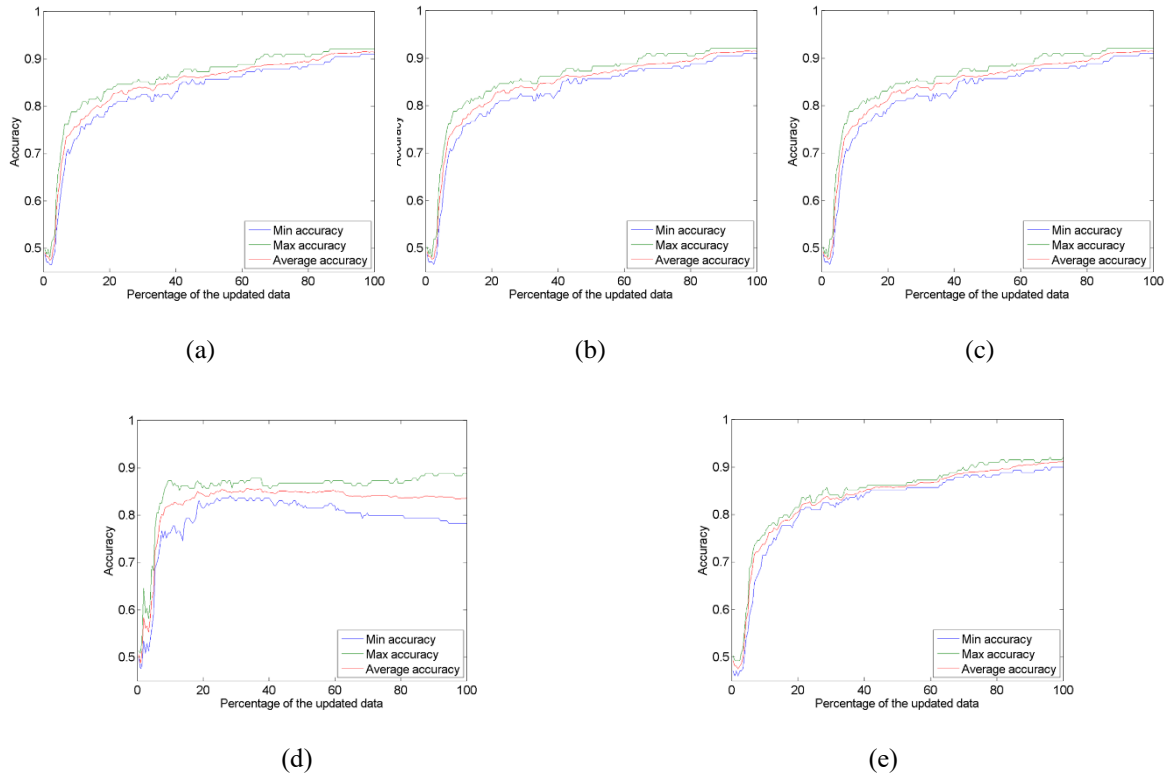


Figure 3. The accuracy of the methods while data updating with their best hidden node in balance dataset: (a) ELM, (b) OS-ELM, (c) STOS-ELM, (d) POS-ELM, and (e) PSTOS-ELM

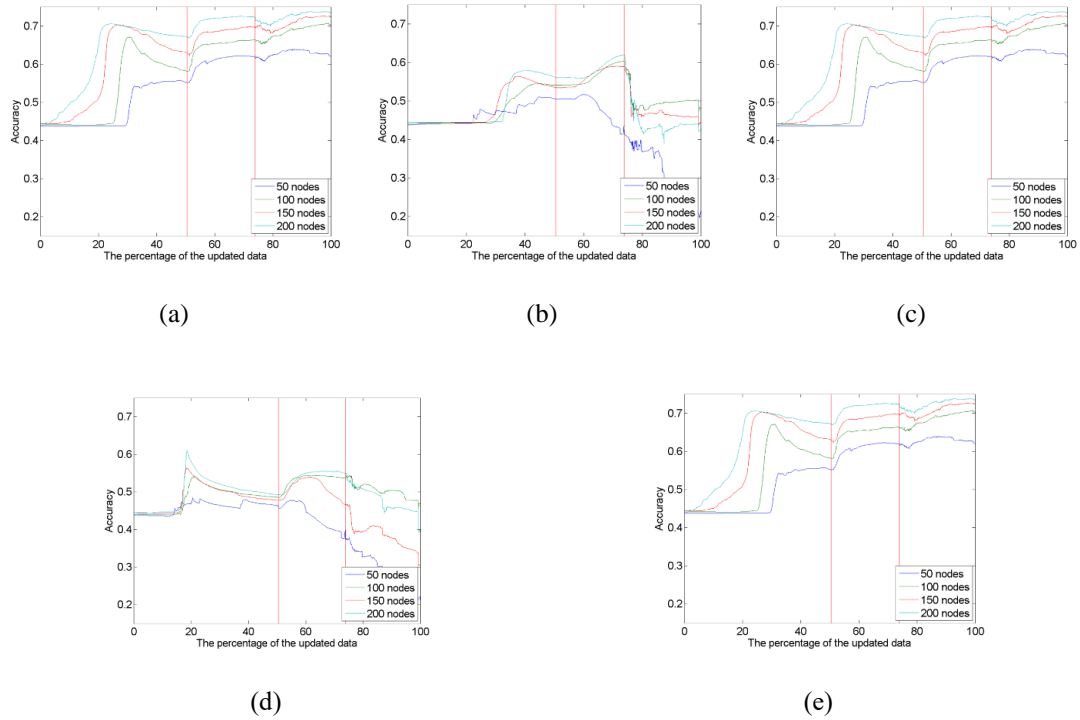


Figure 4. Accuracy of the methods with different number of hidden nodes in satellite dataset: (a) ELM, (b) OS-ELM, (c) STOS-ELM, (d) POS-ELM, and (e) PSTOS-ELM

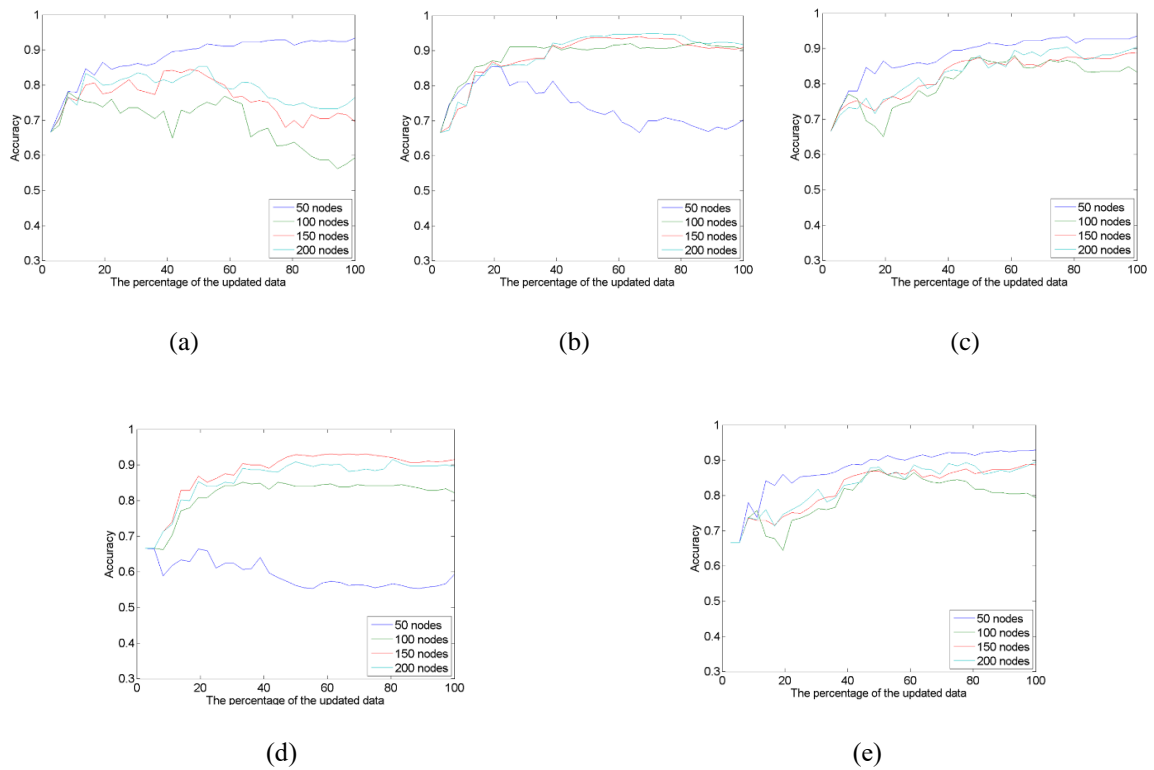


Figure 5. Accuracy of the methods with different number of hidden nodes in iris dataset: (a) ELM, (b) OS-ELM, (c) STOS-ELM, (d) POS-ELM, and (e) PSTOS-ELM

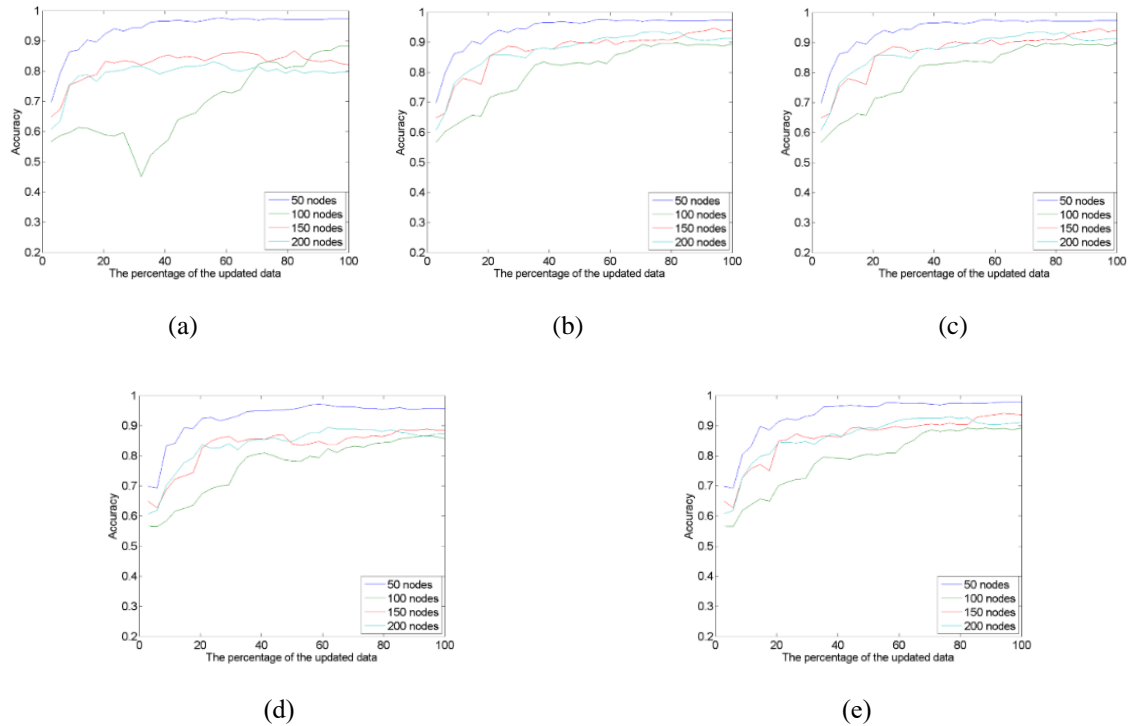


Figure 6. Accuracy of the methods with different number of hidden nodes in wine dataset: (a) ELM, (b) OS-ELM, (c) STOS-ELM, (d) POS-ELM, and (e) PSTOS-ELM

3.2.4. Robustness when the number of hidden nodes varied in a wide range

This experiment aims to extend more information from the previous section by analysing the accuracies of the methods of hidden nodes in the range of 1 to 200 hidden nodes. The result is shown in Figure 7. Our dataset selection according to 3 groups in section 3.2.3 that selected dataset for each group is the waveform, iris, and wine dataset, respectively.

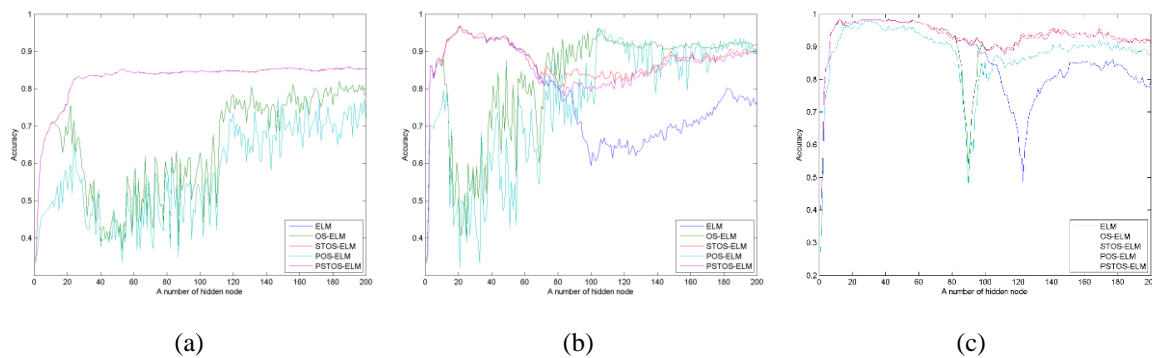


Figure 7. The accuracies of the methods while the number of hidden nodes varied in the range of 1 to 200 hidden nodes: (a) waveform dataset, (b) iris dataset, and (c) wine dataset

For group 1 in Figure 7(a), ELM, STOS-ELM, and PSTOS-ELM have accuracy that tends to grow and be stable. OS-ELM and POS-ELM have accuracies that trend lower than ELM, STOS-ELM, and PSTOS-ELM. In addition, OS-ELM and POS-ELM have fluctuating accuracy trends.

For group 2 in Figure 7(b), ELM, STOS-ELM, and PSTOS-ELM have a similar accuracy trend. But ELM has low accuracy trend than STOS-ELM and PSTOS-ELM in between 80 to 200 hidden nodes. For OS-ELM and POS-ELM, their accuracy trends to drop sharply between 10 to 20 hidden nodes. After that the accuracy trends to go up to their accuracy is higher than ELM, OS-ELM, and STOS-ELM since above 80 hidden nodes.

For group 3 in Figure 7(c), all methods have similar accuracy trend. But OS-ELM and POS-ELM have accuracy that tends to drop and up between 83-100 hidden nodes. And ELM has lower accuracy trend than other methods since above 100 hidden nodes. From the result, if the number of hidden nodes has too much, overfitting will happen that the decision boundary is too fit with the data. On the opposite, the model will not match the complex data.

Therefore, selecting the appropriate number of hidden nodes is required for the best performance of ELM. However, the number of hidden node selections does not have the principle. So, the model validation has been used. The method can help ELM find the appropriate node by testing the ELM with the validation data (the data separate from the training data) and selecting the best number of hidden nodes that takes the best accuracy.

3.2.5. Performance comparison between PSTOS-ELM and the other ELMs with progressive learning

In this section, our experiment aims to compare the performance of PSTOS-ELM and the other ELM with progressive learning. The other ELMs are PL-MCCIP, PL-MCCRP, PL-MCCMP [25], POS-ELM [15], and S-ELM [26]. The performance comparison consists of the accuracy and robustness of all ELM methods are shown in Table 3.

The first line in the table shows the best accuracy of the ELM methods with the best-hidden node in the range [1,200]. The second line is the area under the curve of each accuracy from the data updating that the area under the curve can calculate from the trapz function in MATLAB. The bold letters show the best value of each dataset.

Table 3. The performance of PSTOS-ELM and the other ELMs with progressive learning

Dataset	PL-MCC _{IP}	PL-MCC _{RP}	PL-MCC _{MP}	POS-ELM	S-ELM	PSTOS-ELM
Balance	0.8571	0.4762	0.8571	0.8571	0.9206	0.9206
	169.4339	99.8386	169.4339	169.5556	170.8360	169.0212
Iris	0.9778	0.9333	0.9778	0.9556	1.0000	0.9778
	31.1111	21.6000	31.1111	31.5000	33.4556	32.0000
Optdigits	0.3966	0.1537	0.5875	0.3972	0.6200	0.8191
	1190.2444	438.2402	1117.4557	1191.2878	1434.3735	1701.3286
Satellite	0.4838	0.4196	0.5250	0.4838	0.5257	0.7351
	968.5703	422.8419	973.9280	968.5439	837.5622	1246.4811
Waveform	0.8314	0.7442	0.8314	0.8314	0.8388	0.8588
	928.6935	735.6985	928.6935	928.5456	891.3468	927.2332
Wine	1.0000	0.3091	1.0000	0.9818	1.0000	1.0000
	29.8909	10.8818	29.8909	32.2727	32.5727	32.5727
Average	0.7578	0.5060	0.7965	0.7512	0.8175	0.8852
	552.9907	288.1835	541.7522	553.6176	566.6911	684.7728

Table 3 shows that PSTOS-ELM has the highest average accuracy and the area under the curve. The result has some points of interest. As clearly seen, the other ELMs with progressive learning have low accuracy and area under the curve on optdigits and satellite datasets. Both datasets have number of class higher than three classes which may cause low accuracy. However, that problem does not affect STOS-ELM.

3.3. Discussion

This article presents PSTOS-ELM that can improve robust accuracy while updating the new data and the new class data on the online training situation. The robustness accuracy arises from using the HBQRD-RLS. HBQRD-RLS is supported by PSTOS-ELM performance as shown in the experimental results in 3 aspects.

- Accuracy and robustness: PSTOS-ELM has accuracy comparable to the batch ELM and STOS-ELM. Furthermore, PSTOS-ELM also keeps the robust property in hidden node changing and data updating situations. That is creditable to the key of PSTOS-ELM in using the HBQRD-RLS.
- Effect of progressive learning: the other ELMs with progressive learning cannot achieve robustness, especially in the dataset that has several classes. On the other hand, PSTOS-ELM almost has similar accuracy to STOS-ELM. That means PSTOS-ELM does not affect progressive learning.
- Computation: while new class samples come, STOS-ELM must recalculate the initial model by setting $H_{0:k}$ to the recent samples include the new class samples and using $\tilde{R}_{0:k}^{-1}T_{0:k}$ (9) to calculate $\beta_{0:k-1}$. The samples updating complexity of STOS-ELM is $K \times N_{0:k} \times N_{0:k} \times C$ And PSTOS-ELM uses

$$\Delta\beta_k = \tilde{R}_k^{-1} \begin{bmatrix} -1 & \cdots & -1 \\ \vdots & \ddots & \vdots \\ -1 & \cdots & -1 \end{bmatrix}_{N_k \times P} \quad (14)$$

to update the new class samples and uses (12) to concatenate this with the old beta β_{k-1} . The data updating complexity of PSTOS-ELM is $K \times N_k \times N_k \times P$ + little calculation for β_{k-1} concatenating (12). The massive difference in updating complexity between STOS-ELM

in (9) and PSTOS-ELM (14) is the size of the class (C and P) and samples ($N_{0:k}$ and N_k) that PSTOS-ELM uses only the new coming samples for training. Therefore, PSTOS-ELM uses computation less than STOS-ELM in new class data updating.




4. CONCLUSION

This article discussed the PSTOS-ELM based on the HBQRD-RLS algorithm. PSTOS-ELM can retain robust accuracy while updating the new data and the new class data on the online training situation. The results showed that PSTOS-ELM accuracy and robustness are comparable to the batch learning ELM and STOS-ELM. Furthermore, PSTOS-ELM can reduce the complexity of STOS-ELM when updating the new class data.




REFERENCES

- [1] G. -B. Huang, Q. -Y. Zhu, and C. -K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1–3, pp. 489–501, 2006, doi: 10.1016/j.neucom.2005.12.126.
- [2] G. -B. Huang, D. H. Wang, and Y. Lan, "Extreme learning machines: a survey," *International journal of machine learning and cybernetics*, vol. 2, pp. 107–122, 2011, doi: 10.1007/s13042-011-0019-y.
- [3] G. -B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme Learning Machine for Regression and Multiclass Classification," in *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 2, pp. 513–529, 2012, doi: 10.1109/TSMCB.2011.2168604.
- [4] Ó. F. -Romero, B. G. -Berdíñas, D. M. -Rego, B. P. -Sánchez, and D. P. -Barral, "Online machine learning," in *Efficiency and Scalability Methods for Computational Intellect*, IGI Global, 2013, pp. 27–54, doi: 10.4018/978-1-4666-3942-3.ch002.
- [5] W. -P. Cao *et al.*, "An ensemble fuzziness-based online sequential learning approach and its application," *International Conference on Knowledge Science, Engineering and Management*, 2021, pp. 255–267, doi: 10.1007/978-3-030-82136-4_21.
- [6] A. Uçar, Y. Demir, and C. Güzelış, "A new facial expression recognition based on curvelet transform and online sequential extreme learning machine initialized with spherical clustering," *Neural Computing and Applications*, vol. 27, pp. 131–142, 2016, doi: 10.1007/s00521-014-1569-1.
- [7] S. Atsawaraungsuk, T. Katanyukul, P. Polpinit, and N. E. -Anant, "Fast and robust online-learning facial expression recognition and innate novelty detection capability of extreme learning algorithms," *Progress in Artificial Intelligence*, vol. 11, pp. 151–168, 2022, doi: 10.1007/s13748-021-00266-y.
- [8] N. -Y. Liang, G. -B. Huang, P. Saratchandran, and N. Sundararajan, "A Fast and Accurate Online Sequential Learning Algorithm for Feedforward Networks," in *IEEE Transactions on Neural Networks*, vol. 17, no. 6, pp. 1411–1423, 2006, doi: 10.1109/TNN.2006.880583.
- [9] M. Moonen and J. Vandewalle, "Recursive least squares with stabilized inverse factorization," *Signal Processing*, vol. 21, no. 1, pp. 1–15, 1990, doi: 10.1016/0165-1684(90)90022-Q.
- [10] P. Horata, S. Chiewchanwattana, and K. Sunat, "Enhancement of online sequential extreme learning machine based on the householder block exact inverse QRD recursive least squares," *Neurocomputing*, vol. 149, pp. 239–252, 2015, doi: 10.1016/j.neucom.2013.10.047.
- [11] S. Atsawaraungsuk and T. Katanyukul, "Sin activation structural tolerance of online sequential circular extreme learning machine," *International Journal of Technology*, vol. 8, no. 4, 2017, doi: 10.14716/ijtech.v8i4.9476.
- [12] S. Atsawaraungsuk, T. Katanyukul, and P. Polpinit, "Identity activation structural tolerance online sequential circular extreme learning machine for highly dimensional data," *Engineering and Applied Science Research*, vol. 46, no. 2, pp. 120–129, 2019, doi: 10.14456/easr.2019.15.
- [13] J. Zhou, J. Li, C. Wang, H. Wu, C. Zhao, and Q. Wang, "A vegetable disease recognition model for complex background based on region proposal and progressive learning," *Computers and Electronics in Agriculture*, vol. 184, 2021, doi: 10.1016/j.compag.2021.106101.
- [14] Z. Yang, Y. Hou, Z. Chen, L. Zhang, and J. Chen, "A Multi-Stage Progressive Learning Strategy for Covid-19 Diagnosis Using Chest Computed Tomography with Imbalanced Data," *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 8578–8582, doi: 10.1109/ICASSP39728.2021.9414745.
- [15] R. Venkatesan and M. J. Er, "A novel progressive learning technique for multi-class classification," *Neurocomputing*, vol. 207, pp. 310–321, 2016, doi: 10.48550/arXiv.1609.00085.
- [16] M. A. Akgün, J. H. Garcelon, and R. T. Haftka, "Fast exact linear and non-linear structural reanalysis and the Sherman–Morrison–Woodbury formulas," *International Journal for Numerical Methods in Engineering*, vol. 50, no. 7, pp. 1587–1606, 2001, doi: 10.1002/nme.87.
- [17] M. Moonen and J. Vandewalle, "A square root covariance algorithm for constrained recursive least squares estimation," *Journal of VLSI signal processing systems for signal, image and video technology*, vol. 3, pp. 163–172, 1991, doi: 10.1007/BF00925827.
- [18] J. A. Apolinário, "QRD-RLS adaptive filtering," New York: Springer, 2009, doi: 10.1007/978-0-387-09734-3.
- [19] G. H. Golub and C. F. V. Loan, "Orthogonalization and Least Squares," in *Matrix computations*, 3rd ed, Baltimore, MD, USA: The Johns Hopkins University Press, 2013, vol. 3, [Online]. Available: https://twiki.cern.ch/twiki/pub/Main/AVFedotovHowToRootTDecompQRH/Golub_VanLoan.Matr_comp_3ed.pdf
- [20] C. -T. Pan and R. Plemmons, "Least squares modifications with inverse factorizations: parallel implications," in *Journal of Computational and Applied Mathematics*, vol. 27, no. 1–2, pp. 109–127, 1989, doi: 10.1016/0377-0427(89)90363-4.
- [21] A. Asuncion and D. Newman, "UCI machine learning repository," 2007. [Online]. Available: <http://archive.ics.uci.edu/>
- [22] A. Gholamy, V. Kreinovich, and O. Kosheleva, "Why 70/30 or 80/20 relation between training and testing sets: a pedagogical explanation," *International Journal of Intelligent Technologies and Applied Statistics*, 2018, doi: 10.6148/IJITAS.201806_11(2).0003.
- [23] A. Stefani and M. Xenos, "Meta-metric evaluation of e-commerce-related metrics," *Electronic Notes in Theoretical Computer Science*, vol. 233, no. 27, pp. 59–72, 2009, doi: 10.1016/j.entcs.2009.02.061.
- [24] P. Horata, S. Chiewchanwattana, and K. Sunat, "Robust extreme learning machine," *Neurocomputing*, vol. 102, pp. 31–44, 2013, doi: 10.1016/j.neucom.2011.12.045.
- [25] M. J. Er, R. Venkatesan, N. Wang, and C. -J. Chien, "Progressive learning strategies for multi-class classification," *2017 International Automatic Control Conference (CACS)*, 2017, pp. 1–6, doi: 10.1109/CACS.2017.8284266.
- [26] C. -L. Lee, Y. -T. Chen, and A. -Y. Wu, "A Scalable Extreme Learning Machine (S-ELM) for Class-Incremental ECG-Based User Identification," *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2021, pp. 1–5, doi: 10.1109/ISCAS51556.2021.9401716.




BIOGRAPHIES OF AUTHORS

Sarutte Atsawaraungsuk    received the B.Sc. and M.Sc degree in Computer Science, and the Ph.D. degree in Computer Engineering from the Khon Kaen University, Thailand. Currently, he is an assistant professor at the Department of Computer Education, Faculty of Education, Udon Thani Rajabhat University, Udon Thani, Thailand. His research interests in the Machine learning, image processing and its application. He can be contacted at email: sarutte@udru.ac.th.






Wasaya Boonphairote    obtained a bachelor of arts in English from National and Kapodistrian University of Athens, Greece and received a master of arts in Career English For International Communication from Thammasat University, Thailand. Currently, She is a lecturer at Language Center, Udon Thani Rajabhat University. The research interests focus on modern greek language, natural language processing and spoken language systems. She can be contacted at email: wasaya.bo@udru.ac.th.






Kritsanapong Somsuk    is an associate professor at the Department of Computer and Communication Engineering, Faculty of Technology, Udon Thani Rajabhat University, Udon Thani, Thailand. He obtained his M.Eng. (Computer Engineering) from Department of Computer Engineering, Faculty of Engineering, Khon Kaen University, M.Sc. (Computer Science) from Department of Computer Science, Faculty of Science, Khon Kaen University and his Ph.D. (Computer Engineering) from Department of Computer Engineering, Faculty of Engineering, Khon Kaen University. The area of research interests includes computer security, cryptography and integer factorization algorithms. He can be contacted at email: kritsanapong@udru.ac.th.



Chanwit Suwannapong    received a B.Eng., M.Eng. and Ph.D. degree in Computer Engineering from Khon Kaen University, Thailand. Currently, he is an assistant professor at the Department of Computer Engineering, Faculty of Engineering, Nakhon Phanom University, Nakhon Phanom, Thailand. He has published many publications in the area of Wireless Sensor Network, Ad Hoc Networks and Smart Agriculture. He can be contacted at email: schanwit@npu.ac.th.



Suchart Khummanee    received the B.Eng. degree in Computer Engineering from the King Mongkut's Institute of Technology Ladkrabang, the M.Sc. degree in Computer Science from the Khon Kaen University, and the Ph.D. degree in Computer Engineering from the Khon Kaen University, Thailand. He is currently a full lecturer of Computer Science at the Mahasarakham University, Thailand. He can be contacted at email: suchart.k@msu.ac.th.