IEICE TRANS. INF. & SYST., VOL.E104–D, NO.8 AUGUST 2021

PAPER Special Section on Computational Intelligence and Big Data for Scientific and Technological Resources and Services

Performance Evaluation of Online Machine Learning Models Based on Cyclic Dynamic and Feature-Adaptive Time Series

Ahmed Salih AL-KHALEEFA[†], **Rosilah HASSAN**[††], **Mohd Riduan AHMAD**[†††], **Faizan QAMAR**[††a)], ***Nonmembers***, **Zheng WEN**[††††], ***Member***, **Azana Hafizah MOHD AMAN**[††], ***and*** **Keping YU**[†††††b)], ***Nonmembers***

SUMMARY Machine learning is becoming an attractive topic for researchers and industrial firms in the area of computational intelligence because of its proven effectiveness and performance in resolving real-world problems. However, some challenges such as precise search, intelligent discovery and intelligent learning need to be addressed and solved. One most important challenge is the non-steady performance of various machine learning models during online learning and operation. Online learning is the ability of a machine-learning model to modernize information without retraining the scheme when new information is available. To address this challenge, we evaluate and analyze four widely used online machine learning models: Online Sequential Extreme Learning Machine (OSELM), Feature Adaptive OSELM (FA-OSELM), Knowledge Preserving OSELM (KP-OSELM), and Infinite Term Memory OSELM (ITM-OSELM). Specifically, we provide a testbed for the models by building a framework and configuring various evaluation scenarios given different factors in the topological and mathematical aspects of the models. Furthermore, we generate different characteristics of the time series to be learned. Results prove the real impact of the tested parameters and scenarios on the models. In terms of accuracy, KP-OSELM and ITM-OSELM are superior to OSELM and FA-OSELM. With regard to time efficiency related to the percentage of decreases in active features, ITM-OSELM is superior to KP-OSELM.

key words: *online learning, indoor positioning system, cyclic dynamic, feature-adaptive time series, machine learning*

1. Introduction

The rapid advancement in high-performance computing and the pervasive use of machine learning makes it an emerging research area [1]. Machine learning has made dramatic improvements and is a core sub-area of artificial intelligence [2], [3]. It also enables computers to discover themselves without being explicitly programmed [4], [5]. This topic has garnered the interest of academia and industry because of many reasons. First, data are generated daily from different sources and platforms and regularly stored, thereby opening the door to the building of numerous models that are trained on such data and translate knowledge to smart systems [6]. Second, the fast development of hardware power enables the execution of models within a reasonable time. Hence, these models could be commercialized for real-world applications [7]. Third, the nature of real-life models is complicated and cannot be expressed in mathematical equations [8]. However, when machine learning is coupled with data availability, it provides a remarkable way of expressing complicated models accurately [9]. A good example is a driverless car that requires a complex system to simulate driver behavior [10], [11]. When such a system is trained on data generated from many hours of driving, it becomes an autonomous system that can partially or fully replace actual drivers [12].

The typical approach to building a machine learning model is to train the model using readily available data. The training allows the optimum system configuration to be determined without changing the system after the operation. However, in most real-life applications, data are generated sequentially or incrementally. This type of system is termed as incremental learning, online learning, or concept drift [13]. At present, incremental learning applies to various scenarios and applications. Incremental learning can be applied to the field of security and intrusion detection [14]. Another field is robotics, for which the incremental learning model has been designed in the dominion of autonomous control [15], service robotics [16], computer vision [17], self-localization [18], or interactive kinesthetic teaching [19], [20]. Meanwhile, the domain of autonomous driving is gaining traction with autonomous vehicle legislation already enacted in eight states in the united states [21], [22]. Another emerging area, caused by everywhere sensors within smartphones, addresses activity identification and modeling [23]–[25]. Image processing is also another field in which image and video data are usually collected in a streaming fashion and are thus useful in incremental learning. Common problems in this context range from object recognition [26], [27], image segmentation [28], [29], and image representation [30], [31] to

Manuscript received November 9, 2020.
Manuscript revised March 6, 2021.
Manuscript publicized May 14, 2021.

[†]The author is with Department of Computer Engineering, Faculty of Information Technology, Imam Jafar Al-Sadiq University, Maysan 10011, Iraq.
[††]The authors are with Centre for Cyber Security, Faculty of Information Science and Technology (FTSM), Universiti Kebangsaan Malaysia (UKM), Bangi 43600, Selangor, Malaysia.
[†††]The author is with Broadband and Networking (BBNET) Research Group, Center for Telecommunication and Research Innovation (CeTRI), Fakulti Kejuruteraan Elektronik dan Kejuruteraan Komputer (FKEKK), Universiti Teknikal Malaysia Melaka (UTeM), Hang Tuah Jaya, Durian Tunggal 76100, Melaka, Malaysia.
[††††]The author is with Department of Communications and Computer Engineering, Faculty of School of Fundamental Science and Engineering, Waseda University, Tokyo, 169–0072 Japan.
[†††††]The author is with Global Information and Telecommunication Institute, Waseda University, Tokyo, 169–8050 Japan.
a) E-mail: faizanqamar@ukm.edu.my (Corresponding author)
b) E-mail: keping.yu@aoni.waseda.jp (Corresponding author)
DOI: 10.1587/transinf.2020BDP0002

video surveillance, person identification, and visual tracking [32], [35]. In many real-world applications of classification, instant prediction of samples is not feasible due to the embedded dynamic of the data. This is handled by considering the time dimension in the prediction. Examples are time series generated from road data, intrusion detection system (IDS) data, localization, etc. [36].

Online learning is efficient when neural networks (NNs) are expected to require knowledge updates while in operation and when data are expected to arrive sequentially while NNs are operating [37]. Various models have been developed for online learning. Most of these models consider data with fixed dimensions and, hence, the same number of active features. However, when NNs operate in the real world, the type of active features and their numbers likely show great variability. As a result, sequential data are subjected to dimension changes that require a different number of inputs for the NN for every change. The classical approach is to recreate a new NN and to repeat the training [38] or to transfer knowledge from the old NN to the new NN to avoid retraining [39].

Online sequential extreme learning machine (OSELM) as a famous NN of shallow type is prone to huge knowledge loss whenever the NN changes. For feature adaptive online sequential extreme learning (FA-OSELM), transfer learning is useful for transferring knowledge related to active features in the current and previous NNs; this process causes knowledge loss while the NNs transform. Two novel approaches, namely, knowledge preserving OSELM (KP-OSELM) [40] and infinite term memory OSELM (ITM-OSELM) [41], were proposed in previous research. In KP-OSELM, the NN is fixed with many inputs equal to the total number of features with the use of an encoding approach for non-active features. In ITM-OSELM, the NN changes according to the active features. However, this model is supported by two things: transfer learning to transfer knowledge from the old NN to the current NN and external memory to restore old knowledge related to new active features and to preserve current knowledge related to new non-active features. The work provides the following contributions:

1. It provides a quantitative evaluation and characterization of the four sequential classification models based on different types of online sequential data generated from different fields, namely, one from intrusion detection system ID and two from indoor localization.
2. It covers the response to cyclic dynamic and feature adaptive aspects based on all states of configurations of the classifier that includes all combination of number of neurons and type of activation functions.
3. It generated the evaluation based on cyclic dynamic and features adaptive nature in the sequential data. The differences in the behaviors of the models are then summarized, and recommendations for their application are presented.

The rest of the paper is organized as follows: the most recent and relevant works published within the same area are highlighted in Sect. 2; the four online learning models, i.e., OSELM, FA-OSELM, KP-OSELM, and ITM-OSELM are explained in Sect. 3; the complete research methodology is discussed in Sect. 4; the experimental findings and evaluation are provided in Sect. 5; lastly, the conclusion and future work is discussed in Sect. 6.

2. Related Work

Different incremental learning models have been formulated for renowned machine learning models. For the support vector machine ELM, many incremental models are available. The previous model developed for ELM was based on incremental learning by [38]. This model facilitates the transition from one time-training approach for ELM to a batch-based mode in which the model accepts sequential input data. The new approach modifies the ELM's training equations to be recursive. The incremental extreme learning machine (IELM) systematizes the batch-based ELM solution that uses the least-squares approach in a sequential method [42]. The batch version works using randomized input weights, and the complexity of model training is significantly reduced. This static network requires the number of hidden neurons to be predefined. This approach allows the processing of data one by one or in bulk, thereby considerably decreasing the general processing time. In initializing the output weights of the model, the number of examples should be equal to or more than the number of hidden neurons used in the network. The incremental-ELM (I-ELM) and convex I-ELM (CI-ELM) methods used for extreme learning machines are unable to handle faults. The research by [43] recommends two fault-tolerant I-ELM algorithms: fault-tolerant CI-ELM (FTCI-ELM) and fault-tolerant I-ELM (FTI-ELM). FTI-ELM merely tunes the output weight of the recent additive node to minimize the training set error of faulty networks. The model retains all the prior learned weights as unchanged. Moreover, this model's fault-tolerant performance is superior to those of CI-ELM and I-ELM. FTCI-ELM has been recommended for the best performance. The fault-tolerant version FTCT-ELM modifies the output weights of freshly added nodes. In addition, a simple algorithm is employed to modify the output weights and to optimize a reduction in the training set error in faulty networks. The authors in [44] proposed the I-ELM, whose basis remains the ELM, although it is used for different applications and entails different computational efficiencies and costs. Another research in [45] proposed an incremental type 2 metacognitive ELM. This machine, called evolving type-2 ELM (eT2ELM), is designed to cope with increased complexity, high dimension, concept drift, and uncertainty. The eT2ELM proposes three aspects: 1) what to learn, 2) how to learn, and 3) when to learn. The first component (i.e., what to learn) selects training samples according to their importance. The online certainty-based active learning method is used to update the model, thereby rendering eT2ELM as a semi-supervised classifier. The how-to-learn element connects extreme learning theory

to the evolving concept in which hidden nodes are automatically generated and pruned using data streams without any requirement to tune the hidden nodes. The when-to-learn component uses the standard sample reservation strategy. A generalized interval type 2 fuzzy NN is introduced as a cognitive component. Here, a hidden node is constructed on the interval type 2 multivariate Gaussian function while using a subset of the Chebyshev series in the output node. Twelve data streams with various concept drifts are used to validate the efficacy of the proposed eT2ELM numerically. The authors in [46] demonstrated the use of ELM as a base classifier to adaptively determine the number of neurons in the hidden layer. Performance improvement is achieved using a random selection of activation functions from a set of functions. In the final step, the algorithm trains a set of classifiers. The weighted voting strategy is used to calculate the decision results for unlabeled data. Each classifier is incrementally updated with the new data if the concept in the data streams remains stable. If, however, a drift is present, weak classifiers are cleared away.

The incremental models based on OSELM consider transfer learning. The authors in [39] used ELM in a transfer learning framework. The framework could undertake the addition or removal of access points from the environment. This process leads to changes in the fingerprint model. Transfer learning is used to facilitate the NN's adaption to new situations without the need for fingerprints. If the old information is required in the new system, it can be moved using two matrices: the input weight transfer matrix and the input weight supplement vector. The supplement vector enables the system to perform mandatory adjustments to adapt to the changing dimensions of feature matrices among domains alongside online sequential learning. The model is suitable for evading conventional and exhausting training procedures when an expected update happens in data distribution due to environmental or domain alterations. A drawback of FA-OSELM is that it transfers merely the last state of knowledge. This limitation was addressed by the work of [40]. This work resulted in the modification of the widely used OSELM to achieve enhanced localization results using dynamic and cyclical behavior. The model is known as the KP-OSELM. This change is brought about by the imposition of a condition that the total number of inputs should be equal to the total number of features (active and non-active included). Furthermore, non-active features are encoded as zeros for use with the tansig activation function. This new approach eliminates the need to change the NN topology in response to changes in feature count. However, the computational load increases as the number of features peaks. This drawback was addressed by [41] and by attaching an external memory (EM) to the OSELM. The EM preserves knowledge specific to the old non-active features and restores knowledge specific to new active features. This work, along with the study of [40], provides the framework for the only OSELM variants capable of processing online learning while preserving old knowledge regardless of knowledge aging.

From the incremental models discussed earlier, different incremental models have been formulated based on the original OSLEM. The models exhibit variabilities in the mode of tackling updated learning and dynamical alteration in stream data. The objective of the current study is to compare four key models: FA-OSELM, OSELM, ITM-OSELM, and KP-OSELM.

3. Online Machine Learning Models

This segment presents a detail of the four machine learning models studied in this work. The first model is the elementary incremental learning model OSELM. This model does not comprise any knowledge transfer when the quantity of features is altered. The second model, FA-OSELM, is based on the transfer of knowledge to the target when a change occurs in the count of features that possess the same capability of incremental learning as OSELM. The third model, KP-OSELM, is a knowledge preserving model that consists of an incremental learning capacity without the need for knowledge transfer. The ITM-OSELM encompasses transfer learning, incremental learning, and EM for reinstating old knowledge. These methods were selected because they are single hidden layer neural networks with an online learning algorithm which makes them suitable for handling the aspect of dynamical changes with fast response time. A qualitative comparison between them is presented in Table 1. As it is depicted in the table, there are four main aspects that are considered in selecting the models, namely, the cyclic dynamic, the feature adaptive features, and the knowledge preservation. These four models are discussed in the following subsections.

3.1 Online Sequential Extreme Learning Machine

At the outset, a host of applications lack data. However, with respect to time, a continuous generation of data takes place. The availability of a new block of data requires the model to be trained on that specific block. [42] proposed a mathematical model called the OSELM to facilitate online sequential learning from ELM. It is about the base line approach of the subsequent algorithms that were used in the comparison. It is an online variant of training single hidden layer neural network in fast way with using Moore-Penrose inverse instead of traditional back propagation. The article is cited for more details and the procedure is given as pseudocode with explaining the input and output as $\varkappa = \left\{(X_i, t_i)|X_i \in R^n, t_i \in R^m, i = 1, \ldots, \tilde{N}\right\}$ which denotes multi-dimensional time series and trained neural net-

Table 1 A qualitative comparison between the four models

Models	Supporting cyclic dynamic	Feature adaptive	Knowledge preservation
OSELM	×	×	×
FA-OSELM	×	✓	×
ITM-OSELM	✓	✓	✓
KP-OSELM	✓	✓	✓

work based on the input, respectively. Consider a set of N training samples (with a input vector and a target output vector), $(x_j, t_j) \in \mathbb{R}^n \times \mathbb{R}^m$, are used for training an OSELM with L number of hidden nodes. In a perfect case, the output of this OSELM to x_j should be

$$f(x_j) = \sum_{i=1}^{L} \beta_i G(\alpha_i, b_i, x_j) = t_j for j = 1, \ldots, N \quad (1)$$

here α_i and b_i are the input weights and bias (learning parameters) of the hidden nodes, β_i is the output weight and $G(\alpha_i, b_i, x_j)$ is the output of the ith hidden neuron to the input vector x_j. The definition of $G(\alpha_i, b_i, x_j)$ for additive hidden neuron and radial basis function are shown as follows:

$$G(\alpha_i, b_i, x_j) = \frac{1}{1 + exp(-\alpha_i x_j + b_i)}, b_i \in R \quad (2)$$

$$G(\alpha_i, b_i, x_j) = exp{-b_i}\|x_j - \alpha_i\|^2, b_i \in R^+ \quad (3)$$

The OSELM model consists of two phases. The first phase is the boosting phase, which uses a few batches of training data used in the initialization stage to train the single-layer feed-forward NNs. This phase relies on the primitive ELM method. After the boosting phase, all data used in this training phase are discarded, and then the OS-ELM relies on learning the training data individually or in parts. Training data, once consumed, are discarded. An ELM is a linear combination of L activation functions:

$$f(x) = \sum_{i=1}^{L} h_i(x)\beta_i = h^T(x)\beta \quad (4)$$

here $h(x) = [h_1(x), \ldots h_L(x)]^T$ is called the ELM *feature vector*. The procedure used in the OSELM algorithm is described in algorithm 1.

Algorithm 1 Pseudocode of OSELM

1: **Input:** $\varkappa = \left\{(X_i, t_i) | X_i \in R^n, t_i \in R^m, i = 1, \ldots, \tilde{N}\right\}$;
2: **Output:** trained SLFN
3: **procedure** ELM (OSELM) Algorithm:
4: **Boosting step:**
5: Assign random input weight W_i and bias b_i or center μ_i and impact width $\sigma_i, i = 1, \ldots, N$.
6: Calculate the initial hidden layer output matrix $H_0 = [h_i, \ldots, h_{\tilde{N}}]^2$,
7: where $h_i = [g(W_1.X_i + b_1), \ldots, g(W_{\tilde{N}}.X_i + b_{\tilde{N}})]^T, i = 1, \ldots, \tilde{N}$.
8: Estimate the initial output weight
9: $\beta^{(0)} = M_0 H_0^T T_0, WhereM_0 = (H_0^T H_0) and T_0 = [t_1, \ldots, t_{\tilde{N}}]^T$.
10: Set $k = 0$.
11: **Sequential Learning step:**
12: For each further coming observation
13: $(X_i, t_i), where X_i \in R^n, t_i \in R^m and i = \tilde{N} + 1, \tilde{N} + 2, \tilde{N} + 3, \ldots,$.
14: Calculate the hidden layer output vector
15: $h_{(k+1)} = [g(W_1.X_i + b_1), \ldots, g(W_{\tilde{N}}.X_i + b_{\tilde{N}})]^T$.
16: Calculate the latest output weight
17: β^{k+1} based on RLS algorithm:
18: $M_{k+1} = M_K - \frac{M_k h_k + h_{k+1}^T M_k}{1 + h_{k+1}^T M_k h_{k+1}} \beta^{(k+1)} = \beta^{(k)} + M_k h_{k+1}(t_i^T - h_{k+1}^T \beta^{(k)})$
19: Set k=k+1.

3.2 Feature Adaptive Online Sequential Extreme Learning Machine

In the case of FA-OSELM [39], a pre-trained NN is used to transfer old knowledge to a new network if a difference exists between the numbers of features of two networks. If the numbers of hidden nodes L are the same, FA-OSELM uses an input weight supplement vector Q_i along with an input weight transfer matrix J to transfer from the old weights a_i to the new weight a_i'. To perform this operation, FA-OSELM uses an equation that considers the feature changes from m_t to x_{t+1}. The equation is expressed as

$$\{a_i' = a_i.J + Q_i\}_{i=1}^{L}, \quad (5)$$

where

$$J = \begin{bmatrix} J_{11} & \ldots & J_{1x_{t+1}} \\ \vdots & \ddots & \vdots \\ J_{x_t 1} & \ldots & J_{x_t x_{t+1}} \end{bmatrix}_{x_t \times x(t+1)} \quad (6)$$

$$Q_i = [Q_1 \ldots Q_{x_{t+1}}]_{1 \times x(t+1)} \quad (7)$$

Matrix J should adhere to the following rules:

1. Each line must have a single "1" while the rest of them have "0".
2. Each column must not have more than a single "1", while the other lines have all "0".
3. The equation $J_{ij} = 1$ is used after a feature dimension is modified. This equation indicates that the original feature vector's i^{th} dimension becomes the new feature vector's j^{th} dimension. When the feature dimension increases, Q_i serves as the supplement. A corresponding input weight is added to account for the new feature addition. The following rules are applicable to the supplement Q_i.
4. Low feature dimensions signify that Q_i can be assumed to be an all-zero vector. Hence, additional input weights are not required for the new features added.
5. In the case of an increase in feature dimension, as the i^{th} item of a_i' represents the new feature, the a_i distribution should be used as a basis to conduct a random generation of the i^{th} item of Q_i.

3.3 Knowledge Preserving Online Sequential Extreme Learning Machine

KP-OSELM [44] is a new form of OSELM. This model features knowledge preservation power by using a fixed count of inputs that equals the system's total number of active and non-active features. KP-OSELM uses zero values to encode non-active features in case tansig is used as the activation function. Algorithm 2 contains the pseudocode for KP-OSELM. The non-active features are coded using the *Encode*() command when a new data chunk arrives.

Algorithm 2 Pseudocode of KP-OSELM

```
Inputs: D_k          // data chunks
        y_k          // chunk D_k vector of labels
        SLFN_0       // initial NN
Outputs:
        ACC          // accuracy
procedure Training and prediction using KP-OSELM
    Start
    x_0 = Encode(D_0)   // encode non-active features
    SLFN_1 = OSELMTrain(SLFN_0, x_0, y_0)
    Fork = 1untilN
    x_k = Encode(D_k)
    ŷ_k = Predict(SLFN_k, x_k)
    ACC = calculateAccuracy(ŷ_k, y_k)
    SLFN_{k+1} = OSELMTrain(SLFN_k, x_k, y_k)
    End
```

Algorithm 3 Pseudocode of ITM-OSELM

```
Inputs: D_t = {D_0, D_1, ..... }   // sequence of labeled data
        L   // number of hidden neurons
        g   // activation function
Outputs:
        yp          // predicted classes
        Ac          // accuracy
procedure Training and prediction using KP-OSELM
    Starts
    activeFeatures = checkActive(D(0))
    currentClassifier = initiateClassifier(activeFeatures,L)
    currentEM = initiate(N,L)
    yp = predict(currentClassifer,D(0).x,g)
    Ac(0) = calculateAccuracy(yp,D(0).y)
    currentClassifier = OSELM(currentClassifier, D(0).x,D(0).y,g)
    for D(i) do
        [Change,activeFeatures,newActive,oldActive] = checkActive(D(i),D(i-1))
        if Change then
            nextEM = EMUpdateEM(currentEM,oldActive)
            nextClassifier = transferLearning(currentClassifier,activeFeatures)
            nextClassifier = updateNewActive(nextEM,newActive)
            currentClassifer = nextClassifier
            currentEM = nextEM
        yp = predict(currentClassifier,D(i).x,g)
        Ac(i) = calculateAccuracy(yp,D(i).y)
        currentClassifier = OSELM(currentClassifier,D(i).x,D(i).y,g)
    End
```

3.4 Infinite Term Memory Online Sequential Extreme Learning Machine

ITM-OSELM is a new form of OSELM that can handle online learning capacity in addition to handling a variable number of features and preserving old knowledge. This model consists of two parts. The first part is transfer learning that facilitates the transfer of knowledge from the previous network to the current network. The other part, EM, facilitates the restoration of knowledge pertaining to new active features. EM is also responsible for the preservation of knowledge of old, non-active features. Algorithm 3 highlights the pseudocode of ITM-OSELM. In the case of a change in the number of active features, the **EMUpdateEM**() function is triggered to update the memory. The **updateNewActive**() function, when triggered, restores from memory the knowledge specific to the new active features [45].

Comparing with these models from the perspective of elements, we find that all of them have the same OSELM as a core. However, they are different in including some other elements, namely, transfer learning and memory. While ITM-OSELM includes transfer learning and memory, FA and KP-OSELM include only transfer learning and FA-OSELM. This is also depicted in Table 2.

Table 2 Comparing of the models from the perspective of elements

Elements	FA	ITM	KP
OSELM	×	×	×
Transfer learning	×	✓	×
Memory	✓	✓	✓

4. Methodology

This section provides the developed methodology of exploring the performance of various classifiers given time series and multiclass data. The formulation of the problem is provided in Sect. 4.1. The general methodology is given in Sect. 4.2. Section 4.3 provides the approach to generating the time series data. Section 4.4 presents the classification model. Sections 4.5 and 4.6 provide the evaluation scenarios and measures, respectively. Finally, Sect. 4.7 provides the dataset description.

4.1 Problem Formulation

Let us assume $d = \left\{(x_t^j, y_t^{j+1})\right\}, t = 1, 2, \ldots, T, j = 1, 2, \ldots, T-1$, where $x_t \in R^n$ and $y_t \in N^+$; here, R is the real number set and N^+ is the integer number set. t denotes the time stamp when the data are generated, and j denotes the timestamp when the data are known. Let us assume a neural network NN that can be trained on the data from moment 1 until $t-1$. x_t^j is used to predict y_t^{j+1}. The predicted values are denoted as $\hat{y_t}^{j+1}$. The goal is to minimize the difference between y_t^j and $\hat{y_t}^j$. This type of problem is an online learning problem. Generally, x_t^j and y_t^{j+1} can be a chunk of records instead of one record. The goal is to evaluate four variants of C_t thoroughly, namely, $OSELM_t$, $FA-OSELM_t$, $ITM-OSELM_t$, $KP-OSELM_t$ in terms of the size of non-active features in the chunck, the characteristics of the classifiers and their configuration, and the cyclicity of the sequential data.

4.2 General Framework

The framework of exploring and comparing the performance of different online classifiers is presented in Fig. 1. A separate block is established for generating the time series from an existing dataset. Thereafter, the generated time series goes to the online classifiers. Each of the online classifiers generates the predicted $(y_t)^j$ to be compared with the

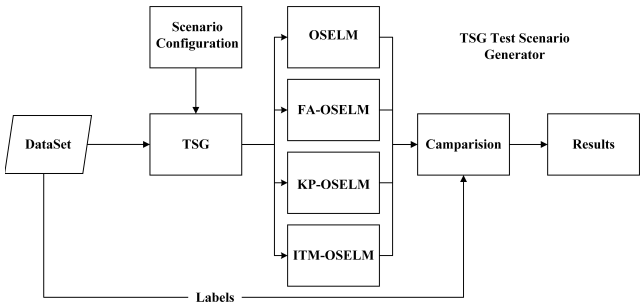

Fig. 1 Framework for evaluating various online learning models using TSG

ground truth $(y_t)^j$. The comparison is performed in a separate block that is responsible for generating the evaluation measures. Another important part of our framework is the evaluation scenario that configures time series generation to provide various scenarios. Test scenario generator (TSG) is responsible for converting an existing dataset to time series.

4.3 Time Series Data Generation

Most data in machine learning do not consider time. However, time matters in online learning. In other words, the training of a classifier is provided at a certain time, and the causality constraint prevents full training of the classifier because training cannot be done when no data are generated. Two subscripts are presented for any sample of data (x_t^j, y_t^{j+1}). The coefficient t indicates the time when the sample is generated. The coefficient j indicates the time when the sample is known to the classifier for learning. The class of the record is known at a later time. However, the classifier needs to predict the class using its previous knowledge. For generating the time series, the same cyclic dynamic generator of [47] is used. The sin function that is used to generate the time series can be replaced with any other periodic function, such as tag and cos.

The pseudocode is presented in algorithm 4. The dataset is initially converted to an adaptive feature dataset through the generate active features function. The process is completed by encoding a set of non-active features for one class with a foreknown value that does not match any value in the features. The class is generated from a period function. The period function tests the performance of the learner with a cyclic dynamic nature. A cyclic dynamic nature involves the frequent repetition of the class with different values of features. Each class has B records, the features' values of which are extracted randomly from the processed dataset that provides a fixed number of active features for each class. The result is a time series dataset with any desired length. This time series maintains two aspects: the number of active features changes from one class to another, and the sequence of classes is repeated periodically. The first aspect enables the testing of adaptive features, and the second one enables the testing of cyclic dynamics. The cyclic pattern can be generated by using following equation.

$$y_t = \left\{ \left\lceil \frac{(y_{max} - 1)}{c} \sin\left(\frac{2\pi t}{T}\right) \right\rceil + 1 \right\} \tag{8}$$

where,

y_{max} denotes the maximum code of the classes.
y_t is the class that occurs at moment t.

4.4 Classification

In this phase, various classifiers are tested based on the provided time series data generated from the previous stage. The condition for any classifier is to have the capacity to handle online learning. The data are presented online to the classifier. As explained earlier, any record or sets of records are not labeled at the same time they are provided to the classifier. However, in the next moment, when a new set of data is generated, the labeling information of the previous one is provided. Thus, before the output of any dataset is predicted, the classifier needs to be trained on the previous chunk. The training is accumulative, which means that the knowledge is built up while training. The classifiers used have the same essential online learning core, which is the approach of OSELM. The differences have been previously discussed.

Algorithm 4 Pseudocode of generating active features

```
Inputs:
    A       // dataset
    B       // records per sample
    C       // time series Length
    D       // classes
    E       // time series period
Outputs:
    F       // time series data
procedure GenerateActiveFeatures
    Starts
    A = GenerateActiveFeatures(A);
    t = 1
    for i=1 to C do
        y = sin(2 * pi * t/E)
        yt = Quantize(y, D)
        for j=1 to B do
            xt = Extract(yt, A)
            F(t).x = xt
            F(t).y = yt
            t = t + 1
    End
```

4.5 Evaluation Measures

The two evaluation measures generated are execution time and accuracy. Execution time indicates the time required to train the model on a previous chunk and then predict the current chunk. Accuracy implies correct classifications measured as a fraction of the total number of classifications. Other evaluation measures include true positive rate (TPR), true negative rate (TNR), false-positive rate (FPR), and false-negative rate (FNR). The equations are specified in Table 3.

Table 3 Evolution measures of the classification system

Measure name	Description/Eq.
Positive (P)	The number of real positive cases in the data
Negative (N)	The number of real negative cases in the data
True Positive (TP)	These refer to the positive tuples that were correctly labeled
False Positive (FP)	These are the negative tuples that were incorrectly labeled
True Negative (TN)	These are the negative tuples that were correctly labeled
False Negative (FN)	These are the positive tuples that were mislabelled as negative
Accuracy (ACC)	$\frac{TP+TN}{P+N} = \frac{TP+TN}{TP+TN+FP+FN}$
True Positive Rate (TPR)	$\frac{TP}{P} = \frac{TP}{TN+FN}$
True Negative Rate (TNR)	$\frac{TN}{N} = \frac{TN}{TN+FP}$
False Positive Rate (FPR)	$\frac{FP}{N} = \frac{FP}{FP+TN} = 1 - TNR$
False Negative Rate (FNR)	$\frac{FN}{P} = \frac{FN}{FN+TP} = 1 - TPR$

4.6 Evaluation Scenarios

The evaluation was done using MATLAB 2019a, we run our experiment on a computer with Windows 10, Processor of Intel (R) Core (TM) i7-6500U and RAM of 8.00 GB. The evaluation scenarios are generated based on the configuration of the time series generator. We change various parameters and study the influence of each of them on the evaluation measures provided previously. The parameters to be changed are the number of neurons, the activation function types, the percentage of active features, and the period of the tested time series.

The first variable to change is the number of neurons in the hidden layer. The variable starts with an initial value equal to the number of inputs that changes until the maximum value is reached. The goal is to study the effect of this parameter on accuracy and computational time. This scenario is tested on the four classifiers. The second scenario to apply is the percentage of active features. This percentage is changed within five ranges: 10%-20%, 20%-40%, 40%-60%, 60%-80%, and 80%-100%. The goal is to explore the effect of this percentage on performance from two aspects: accuracy and computational cost. Another measure to test the period's influence on system performance. The input T is changed in the pseudocode to obtain different time series with different values of T. The final factor to be investigated is the activation function. We have three types of activation functions that are used in each of the models: tansig, sin, and sigmoid.

The evaluation algorithm follows the pseudocode provided in algorithm 5. As it is seen in the pseudocode, the evaluation starts with changing the number of neurons according to the range, the type of activation function, the range of periods, and the percentage of active features. Next, it evaluates each of the four models accordingly and it adds its evaluation results to the output.

Algorithm 5 Pseudocode of evaluation feature

```
Inputs:
    TimeSeries
    RangeOfNumberOfNeurons
    TypesOfActivationFunction
    RangeOfPeriods
    PercentageOfActiveFeature
Outputs:
    Evaluation Results
procedure EvaluationFeatures
    Starts
    for Each numberofNeurons do
        for Each typeofActivationFunc do
            for Each raneofPeriod do
                for Each percentageofActivationFunc do
                    w = Predict(OSELM)
                    x = predict(FA-OSELM)
                    y = Predict(ITM-OSELM)
                    z = Predict(KP-OSELM)
                    Calculate Accuracy and Time
                    Add to the Evaluation Results
    End
```

4.7 Dataset Description

The Knowledge Discovery and Data Mining (KDD) competition, held in 1999, provided the KDD99 dataset [48]. These data were provided by Lee and Stolfo [49]. Pfahringer [50] used bagging and boosting to differentiate such data from other datasets. This work served as a benchmark for researchers after having won the first place in the competition. The data focus on the security domain, particularly intrusion detection.

These data are essential for machine learning. The output classes consist of five main categories: Denial of Service, Root 2 Local, probe, User 2 Root, and the normal category. This dataset consists of a set of 38 attacks; the training phase has 24 attack types, whereas the testing phase has 14 attack types. The 14 new attacks act as a theoretical test on the IDS capability to generalize unknown attacks. The detection of the new set of 14 attacks is difficult for machine learning-based IDS [51]. KDD99 is an old dataset, however, it is still a benchmarking data for evolving behavior of attacks in intrusion detection systems. In the work of [52] which is 2020 work, it is indicated that IDS is "Many studies have used these datasets in their work" and it has focused on the analysis it is behavior which provides its relevance in IDS. Also, it has various challenges from the perspective of classification, we present them as follows:

1. It has an evolving aspect due to the evolving of attacks with respect to time.
2. It has a concept drift issue.
3. It has a class imbalance.
4. It is type of big data due to the large number of records.

Two supplementary datasets from Wi-Fi-based localization are utilized: TempereU and UJIIndoorLoc. The UJIIndoorLoc database contains data pertaining to three build-

ings of the Jaume I University. These buildings have at least four levels and an area of 110,000 m^2 [53]. The UJIIndoorLoc database may be utilized in classification. Regression, identification of floors and buildings, and an estimate of coordinates (longitude and latitude) are some examples. This database was formulated in 2013 with more than 20 distinctive users and 25 Android units. The database comprises 1,111 validation and test records and 19,937 training/reference records. The database has 529 attributes with Wi-Fi fingerprints, which include the coordinates of the information sources.

In testing the IPSs that rely on Wi-Fi/wireless LAN fingerprints, the TempereU database is used. The datasets of TempereU are meant for indoor localization. Made by Lohan and Talvitie, this database is used to validate techniques specific to indoor localization [54]. This database consists

Table 4 Evaluation measures for the four online models with respect to the number of neurons and activation function type for the KDD99 dataset

Dataset - KDD99														
Classifier		ITM-OSELM			KP-OSELM			FA-OSELM			OSELM			
	NoN/AF	sin	tansig	sigmoid	sin	tansig	sigmoid	sin	tansig	sigmoid	sin	tansig	sigmoid	Max
TPR	500	65.17%	61.05%	40.76%	64.01%	59.82%	36.67%	43.75%	45.92%	17.84%	12.27%	30.97%	27.39%	sin-ITM
	2000	75.98%	74.19%	50.81%	75.15%	74.05%	50.72%	49.89%	51.89%	33.62%	21.84%	21.12%	34.87%	sin-ITM
	5000	77.81%	76.21%	60.12%	78.24%	76.42%	61.72%	52.13%	51.79%	40.69%	18.78%	24.21%	26.94%	sin-KP
FPR	500	8.70%	9.73%	14.81%	8.99%	10.05%	15.83%	14.06%	13.52%	20.54%	21.93%	17.26%	18.15%	sin-ITM
	2000	6%	6.45%	12.30%	6.21%	6.48%	12.32%	12.53%	12.03%	16.60%	19.54%	19.72%	16.28%	sin-ITM
	5000	5.54%	5.94%	9.96%	5.44%	5.89%	9.57%	11.97%	12.05%	14.83%	20.31%	18.95%	18.26%	sin-KP
TNR	500	91.29%	90.26%	85.19%	91.00%	89.95%	84.17%	85.94%	86.48%	79.46%	78.07%	82.74%	81.85%	sin-ITM
	2000	93.99%	93.55%	87.70%	93.79%	93.51%	87.68%	87.47%	87.97%	83.40%	80.46%	80.28%	83.72%	sin-ITM
	5000	94.45%	94.05%	90.03%	94.56%	94.10%	90.43%	88.03%	87.95%	85.17%	79.69%	81.05%	81.74%	sin-KP
FNR	500	34.83%	38.95%	59.24	35.99%	40.18%	63.33%	56.25%	54.08%	82.16%	87.73%	69.03%	72.61%	sin-ITM
	2000	24.02%	25.81%	49.19%	24.85%	25.95%	49.28%	50.11%	48.11%	66.38%	78.16%	78.88%	65.13%	sin-ITM
	5000	22.19	23.79%	39.88%	21.76%	23.58%	38.28%	47.87%	48.21%	59.31%	81.22%	75.79%	73.06%	sin-KP

Table 5 Evaluation measures for the four online models with respect to the number of neurons and activation function type for the TampereU dataset

Dataset - TampereU														
Classifier		ITM-OSELM			KP-OSELM			FA-OSELM			OSELM			
	NoN/AF	sin	tansig	sigmoid	sin	tansig	sigmoid	sin	tansig	sigmoid	sin	tansig	sigmoid	Max
TPR	500	64.90%	63.56%	63.70%	66.06%	64.54%	65.65%	29.74%	32.26%	35.65%	27.02%	37.00%	34.79%	sin-KP
	2000	67.97%	66.42%	64.08%	68.20%	66.02%	64.26%	30.49%	30.29%	31.24%	24.56%	20.60%	29.79%	sin-KP
	5000	67.37%	67.42%	63.55%	67.13%	67.24%	63.47%	31.63%	29.41%	30.99%	32.17%	17.45%	31.23%	tansig-ITM
FPR	500	11.70%	12.15%	12.10%	11.31%	11.82%	11.45%	23.42%	22.58%	21.45%	24.33%	21.00%	21.74%	sin-KP
	2000	10.68%	11.19%	11.97%	10.60%	11.33%	11.91%	23.17%	23.24%	22.92%	25.15%	26.47%	23.40%	sin-KP
	5000	10.88%	10.86%	12.15%	10.96%	10.92%	12.18%	22.79%	23.53%	23.00%	22.61%	27.52%	22.92%	tansig-ITM
TNR	500	88.30%	87.85%	87.90%	88.69%	88.18%	88.55%	76.58%	77.42%	78.55%	75.67%	79.00%	78.26%	sin-KP
	2000	89.32%	88.81%	88.03%	89.40%	88.87%	88.09%	76.83%	76.76%	77.08%	74.85%	73.53%	76.60%	sin-KP
	5000	89.12%	89.14%	87.85%	89.04%	89.08%	87.82%	77.21%	76.47%	77.00%	77.39%	72.48%	77.08%	tansig-ITM
FNR	500	35.10%	36.44%	36.30%	33.94%	35.46%	34.35%	70.26%	67.74%	64.35%	72.98%	63.00%	65.21%	sin-KP
	2000	32.30%	33.58%	35.92%	31.80%	33.98%	35.74%	69.51%	69.71%	68.76%	75.44%	79.40%	70.21%	sin-KP
	5000	32.63	32.58%	36.45%	32.87	32.76%	36.53%	68.37%	70.59%	69.01%	67.83%	82.55%	68.77%	tansig-ITM

Table 6 Evaluation measures for the four online models with respect to the number of neurons and activation function type for the UJIIndoorLoc dataset

Dataset - UJIIndoorLoc														
Classifier		ITM-OSELM			KP-OSELM			FA-OSELM			OSELM			
	NoN/AF	sin	tansig	sigmoid	sin	tansig	sigmoid	sin	tansig	sigmoid	sin	tansig	sigmoid	Max
TPR	500	62.60%	70.98%	70.23%	61.87%	71.13%	70.15%	22.22%	33.30%	32.63%	21.33%	17.46%	24.20%	tansig-KP
	2000	73.01%	71.76%	71.15%	74.78%	71.76%	71.21%	19.90%	38.93%	38.93%	20.00%	18.47%	18.79%	sin-KP
	5000	73.83%	71.56%	71.76%	74.88%	71.76%	71.76%	22.32%	38.93%	38.93%	20.10%	18.16%	25.93%	sin-KP
FPR	500	9.35%	7.25%	7.44%	9.53%	7.21%	7.46%	19.45%	16.67%	16.84%	19.67%	20.64%	18.95%	tansig-KP
	2000	6.74%	7.06%	7.21%	6.30%	7.06%	7.19%	20.03%	15.27%	15.27%	20.00%	20.38%	20.30%	sin-KP
	5000	6.54%	7.10%	7.06%	6.28%	7.06%	7.06%	19.42%	15.27%	15.27%	19.97%	20.46%	18.52%	sin-KP
TNR	500	90.65%	92.75%	92.56%	90.47%	92.78%	92.54%	80.55%	83.33%	83.16%	80.33%	79.36%	81.05%	tansig-KP
	2000	93.25%	92.94%	92.79%	93.69%	92.94%	92.80%	79,97%	84.73%	84.73%	80.00%	79.62%	79.70%	sin-KP
	5000	93.46%	92.89%	92.94%	93.72%	92.94%	92.94%	80.58%	84.73%	84.73%	80.03%	79.54%	81.48%	sin-KP
FNR	500	37.40%	29.02%	29.77%	38.13%	28.87%	29.85%	77.78%	66.70%	67.37%	78.67%	82.54%	75.80%	tansig-KP
	2000	26.99%	28.24%	28.85%	25.22%	28.24%	28.79%	80.10%	61.07%	61.07%	80.00%	81.53%	81.21%	sin-KP
	5000	26.17%	28.44%	28.24%	25.12%	28.24%	28.24%	77.68%	61.07%	61.07%	79.90%	81.84%	74.07%	sin-KP

of data pertaining to two buildings of the Tampere University of Technology. The buildings have three and four levels. The TempereU database contains 1,478 training and reference records and 489 test attributes specific to the first building [55]. The test attribute count for the second building is 312. This database is also a storehouse of coordinates, namely, latitude, longitude, and height, in addition to the Wi-Fi fingerprints of 209 wireless access points.

5. Experimental Results and Evaluation

This section presents the experimental work of the study. The results of the number of features and a thorough analysis of the types of activation functions are presented. Next, the results of the effects of the percentage of active features are discussed and analyzed. Finally, the results of the effects of the period are discussed.

5.1 Analysis of the Number of Features and Types of Activation Functions

Tables 4, 5, and 6 show the relation between three factors: the number of neurons in the hidden layer, the type of activation function, and the model tested. The following conclusions can be drawn:

1. The number of neurons increases in the TPR and TNR when KDD99 increases. This finding is interpreted by the added capacity to preserve additional knowledge. However, many neurons higher than a certain threshold will result in a decrease in accuracy because of overfitting. This condition appears after increasing the number of neurons from 2000 to 5000 in the TampereU and UJIIndoorLoc datasets, respectively, because of overfitting.

2. The type of activation function shows interesting behavior. For example, sin achieves the highest number of TPR and TNR and the lowest number of FPR and FNR for the KDD99 dataset. Moreover, sin achieves the best measures for the number of neurons (2000) and (5000), and tansig achieves the best measures for the number of neurons (500) in UJIIndoorLoc. The activation functions sin and tansig achieve the best measures for TampereU but with a different number of neurons. Thus, the type of activation function plays a crucial role in model performance. However, changing the number of neurons might require changing the type of activation function to sustain the level of performance.

3. Among the scenarios, the best models are ITM-OSELM and KP-OSELM. Furthermore, these models have similar prediction performances because of their common knowledge preservation. Also, the random factor that results from the random weights in the input hidden layer can cause a slight difference between the models.

4. Among the scenarios and datasets, the weakest model is OSELM. This model lacks knowledge of transfer and preservation when the number of features changes.

5. The level of TNR, relative to the level of TPR, reveals that TNR has a higher range, as reflected by the low number of positive records relative to the number of negative records during testing.

Table 7 Accuracies of four online learning models with respect to the percentages of active features for three datasets

Models	Per./CL	ITM-OSELM	KP-OSELM	FA-OSELM	OSELM
KDD99	10-20%	42.17%	41.42%	16.67%	16.67%
	20-40%	54.58%	54.75%	22.25%	23.08%
	40-60%	70.58%	70.56%	35.42%	31.83%
	60-80%	68.75%	67.58%	30.25%	26.89%
	80-100%	71.61%	69.75%	26.58%	07.63%
TampereU	10-20%	95.42%	95.42%	37.56%	27.64%
	20-40%	99.67%	99.67%	35.83%	27.69%
	40-60%	99.67%	99.67%	28.31%	27.11%
	60-80%	99.33%	100%	46.03%	24.69%
	80-100%	90.17%	89.61%	40.14%	23.14%
UJIIndoorLoc	10-20%	100%	100%	25.47%	15.42%
	20-40%	98.72%	98.17%	24.81%	17.47%
	40-60%	90.92%	91.28%	23.28%	21.83%
	60-80%	86.69%	85.50%	20.94%	19.03%
	80-100%	97.11%	96.00%	24.94%	26.92%

Table 8 Execution times of four online learning models with respect to the percentages of active features for three datasets.

Model	Per./CL	ITM-OSELM	KP-OSELM
KDD99	10-20%	23.7474	26.5752
	20-40%	25.5523	26.8614
	40-60%	25.1697	25.9639
	60-80%	26.9561	25.1547
	80-100%	25.8241	27.0670
TampereU	10-20%	36.9818	45.3966
	20-40%	39.6630	45.9803
	40-60%	42.7834	46.4927
	60-80%	47.3053	45.9603
	80-100%	52.1914	46.1674
UJIIndoorLoc	10-20%	46.1292	62.0675
	20-40%	55.1569	62.8888
	40-60%	59.3844	61.6400
	60-80%	66.1538	61.3029
	80-100%	67.0744	65.8962

5.2 Analysis of Accuracy and Execution Time with Respect to the Percentage of Active Features

The accuracy of each of the four models with respect to changes in active features is determined, and the results are presented in Table 7. The percentage of active features is not related to the generated accuracy of the model. KP-OSELM and ITM-OSELM are superior to FA-OSELM and OSELM with respect to accuracy; hence, the difference between them should be analyzed from the aspect of computational cost. The percentage of active features plays a role in the computational time of the model. We generate the execution time for five levels of the percentage of active features and compare the execution times of ITM-OSELM and KP-OSELM. Table 8 shows that ITM-OSELM's execution time is affected by percentage. The trend in computational time increases with the percentage of active features. For the KDD99 dataset, the computational time increases from 23.7474 for a percentage of 10%-20% to 25.8241 for a percentage of 80%-100%. For the TampereU dataset, the com-

Table 9 Accuracy of four online learning models with respect to different periods of time series for three datasets

Model	No./CL	ITM-OSELM	KP-OSELM	FA-OSELM	OSELM
KDD99	20	88.64%	86.60%	29.00%	17.13%
	30	84.39%	83.96%	32.07%	23.77%
	40	82.28%	82.17%	29.33%	32.07%
	50	81.50%	80.79%	32.32%	30.79%
TampereU	20	98.73%	99.00%	44.78%	34.82%
	30	98.22%	97.49%	37.26%	33.68%
	40	96.90%	96.46%	36.21%	31.70%
	50	95.64%	95.81%	39.30%	34.25%
UJIIndoorLoc	20	83.82%	84.78%	28.69%	24.76%
	30	71.97%	69.12%	28.42%	26.58%
	40	62.83%	62.94%	27.61%	24.61%
	50	56.99%	58.82%	28.88%	25.55%

Table 10 Student t-test for statistical differences between ITM-OSELM, KP-OSELM, FA-OSELM, and OSELM

Algorithm	ITM-OSELM	KP-OSELM	FA-OSELM	OSELM
ITM-OSELM	N	0.373409874	$2.83616E-09$	$6.49452E-09$
KP-OSELM	0.3734099	N	$2.11299E-09$	$5.09957E-09$

putational time increases from 36.9818 for a percentage of 10%-20% to 52.1914 for a percentage of 80%-100%. For the UJIIndoorLoc dataset, the computation time increases from 46.1292 for a percentage of 10%-20% to 67.0744 for a percentage of 80%-100%. This resulted is interpreted by the adaptive inputs of the topology that changes according to the number of active features.

For KP-OSELM, the computational complexity changes within the range of [25.1547-27.067] for the KDD99 dataset, within the range of [45.3966-46.4927] for the TampereU dataset, and within the range of [61.3029-65.8962] for the UJIIndoorLoc dataset. The small range indicates a non-change in computational time depending on the percentage of active features. This result is interpreted by the fixed structure of the network in which the active and non-active features are fed into the network with an encoding of the non-active features with zeros.

5.3 Period Analysis

The effects of the periods of cyclic dynamic signals on the accuracies of online learning models are analyzed in Table 9. A range of period values starting from 20 and ending at 50 with a step of 10 is taken. The increase in period causes a decline in the accuracies of KP-OSELM and ITM-OSELM. The longest period equates to less frequent training, whereas the shortest period equates to more frequent training. The shortest period consequently results in high accuracy. FA-OSELM shows a random behavior with frequency because knowledge is only transferred from the previous state to the current state. OSELM behaves the same way as FA-OSELM because of the lack of knowledge transfer.

To verify the statistical difference between the four models, we apply a two-tailed distribution t-test and determine the probabilities associated with this test. The results are presented in Table 10. Apparently, KP-OSELM and ITM-OSELM do not show any statistical difference in accuracy because the p-value is higher than 0.05. However, both models show statistical differences with FA-OSELM and OSELM with a p-value lower than 0.01. This finding reveals the significant improvement of the models of KP-OSELM and ITM-OSELM over FA-OSELM and OSELM.

6. Conclusion and Future Work

Cyclic dynamic and feature adaptive aspects of time series are faced in many time series classification problems in real world. The former aspect indicates to the repeated patterns throughout the time series in the time domain and the latter indicates to the disabled or enable of subset of features which makes the feature space of variable length.

In this article, four online learning models, namely, OSELM, FA-OSELM, KP-OSELM, and ITM-OSELM, were evaluated in terms of the capability of handling time series these two aspects based on various scenarios. The tested scenarios included the number of neurons in the hidden layer, types of activation function, percentage of active features, and length of the period of tested time series. The evaluation measures were accuracy and execution time. The results showed that KP-OSELM and ITM-OSELM are superior to FA-OSELM and that FA-OSELM is superior to OSELM. This superiority is attributed to the accuracy of the models when the time series is combined with many cycles. This attribute is caused by the aspect of knowledge preservation. FA-OSELM is superior to OSELM because of its learning transfer. The execution times of ITM-OSELM and KP-OSELM when the percentages of active features change were also studied. A low percentage of active features resulted in a time-efficient ITM-OSELM relative to KP-OSELM. Furthermore, the accuracies of ITM-OSELM and KP-OSELM decreased when the period length increased.

Future studies should focus on the improvement of ITM-OSELM and KP-OSELM using other supporting models to make them stable with respect to changes in the periods of time series learned by the models. Furthermore, an approach to activation function selection should be proposed to study the best characteristics of activation functions in online learning. In addition, there is a need to build types of ITM-OSELM and KP-OSELM that support kernel and reduced kernels variants of ELM.

Acknowledgments

This research was funded by the Universiti Kebangsaan Malaysia, under code grant DIP-2018-040, and Malaysian Ministry of Education, Universiti Teknikal Malaysia Melaka, under grant PJP/2018/FKEKK(3B)/S01615.

References

[1] X. Wang, F. Jiang, L. Zhong, Y. Ji, S. Yamada, K. Takano, and G. Xue, "Intelligent Post-Disaster Networking by Exploiting Crowd

Big Data," IEEE Network, vol.34, no.4, pp.49–55, 2020.
[2] Z. Guo, K. Yu, Y. Li, G. Srivastava, and J.C. -W. Lin, "Deep Learning-Embedded Social Internet of Things for Ambiguity-Aware Social Recommendations," IEEE Transactions on Network Science and Engineering, 2021. doi: 10.1109/TNSE.2021.3049262
[3] Z. Guo, Y. Shen, A.K. Bashir, M. Imran, N. Kumar, D. Zhang, and K. Yu, "Robust Spammer Detection Using Collaborative Neural Network in Internet-of-Things Applications," IEEE Internet of Things Journal, vol.8, no.12, pp.9549–9558, 2020. doi: 10.1109/JIOT.2020.3003802
[4] J. Zhang, K. Yu, Z. Wen, X. Qi, and A.K. Paul, "3D Reconstruction for Motion Blurred Images Using Deep Learning-based Intelligent Systems," CMC-Computers, Materials & Continua, vol.66, no.2, pp.2087–2104, 2021.
[5] Q. Li, H. Qu, Z. Liu, N. Zhou, W. Sun, S. Sigg, and J. Li, "AF-DCGAN: Amplitude-Feature Deep Convolutional GAN for Fingerprint Construction in Indoor Localization System," IEEE Transactions on Emerging Topics in Computational Intelligence, vol.5, no.3, pp.468–480, 2021. doi: 10.1109/TETCI.2019.2948058
[6] L. Zhou, S. Pan, J. Wang, and A.V. Vasilakos, "Machine learning on big data: Opportunities and challenges," Neurocomputing, vol.237, pp.350–361, 2017.
[7] P. Yao, H. Wu, B. Gao, J. Tang, Q. Zhang, W. Zhang, J.J. Yang, and H. Qian, "Fully hardware-implemented memristor convolutional neural network," Nature, vol.577, no.7792, pp.641–646, 2020.
[8] K.-P. Yu, L. Tan, M. Aloqaily, H. Yang, and Y. Jararweh, "Blockchain-Enhanced Data Sharing with Traceable and Direct Revocation in IIoT," IEEE Transactions on Industrial Informatics, 2021. doi: 10.1109/TII.2021.3049141
[9] H. Hu, Z. Liu, and J. An, "Mining Mobile Intelligence for Wireless Systems: A Deep Neural Network Approach," IEEE Computational Intelligence Magazine, vol.15, no.1, pp.24–31, Feb. 2020.
[10] K. Yu, L. Lin, M. Alazab, L. Tan, and B. Gu, "Deep Learning-Based Traffic Safety Solution for a Mixture of Autonomous and Manual Vehicles in a 5G-Enabled Intelligent Transportation System," IEEE Transactions on Intelligent Transportation Systems, pp.1–11, 2020. doi: 10.1109/TITS.2020.3042504
[11] Z. Zhou, H. Liao, X. Wang, S. Mumtaz, J. Rodriguez, "When Vehicular Fog Computing Meets Autonomous Driving: Computational Resource Management and Task Offloading," IEEE Network, vol.34, no.6, pp.70–76, 2020.
[12] J.-A. Bolte, A. Bar, D. Lipinski, and T. Fingscheidt, "Towards corner case detection for autonomous driving," 2019 IEEE Intelligent Vehicles Symposium (IV), IEEE, pp.438–445, 2019.
[13] V. Mangat, V. Gupta, and R. Vig, "Methods to Investigate Concept Drift in Big Data Streams," in Knowledge Computing and Its Applications: Springer, pp.51–74, 2018.
[14] A. Patel, M. Taghavi, K. Bakhtiyari, and J.C. JúNior, "An intrusion detection and prevention system in cloud computing: A systematic review," Journal of network and computer applications, vol.36, no.1, pp.25–41, 2013.
[15] M. Wang and C. Wang, "Learning from adaptive neural dynamic surface control of strict-feedback systems," IEEE transactions on neural networks and learning systems, vol.26, no.6, pp.1247–1259, 2014.
[16] F.I. Doğan, I. Bozcan, M. Celik, and S. Kalkan, "Cinet: A learning based approach to incremental context modeling in robots," 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, pp.4641–4646, 2018.
[17] H. Zhang, P. Wu, A. Beck, Z. Zhang, and X. Gao, "Adaptive incremental learning of image semantics with application to social robot," Neurocomputing, vol.173, pp.93–101, 2016.
[18] A.M. Ali and M.J. Nordin, "SIFT based monocular SLAM with multi-clouds features for indoor navigation," TENCON 2010-2010 IEEE Region 10 Conference, IEEE, pp.2326–2331, 2010.
[19] M. Saveriano, S.-I. An, and D. Lee, "Incremental kinesthetic teaching of end-effector and null-space motion primitives," 2015 IEEE International Conference on Robotics and Automation (ICRA), IEEE, pp.3570–3575, 2015.
[20] A.M. Ghalamzan E, C. Paxton, G.D. Hager, and L. Bascetta, "An incremental approach to learning generalizable robot tasks from human demonstration," 2015 IEEE international conference on robotics and automation (ICRA), IEEE, pp.5616–5621, 2015.
[21] A.R. Fayjie, S. Hossain, D. Oualid, and D.-J. Lee, "Driverless car: Autonomous driving using deep reinforcement learning in urban environment," 2018 15th International Conference on Ubiquitous Robots (UR), IEEE, pp.896–901, 2018.
[22] R. Allamaraju, H. Kingravi, A. Axelrod, G. Chowdhary, R. Grande, J.P. How, C. Crick, and W. Sheng, "Human aware UAS path planning in urban environments using nonstationary MDPs," 2014 IEEE International Conference on Robotics and Automation (ICRA), IEEE, pp.1161–1167, 2014.
[23] M. Hasan and A.K. Roy-Chowdhury, "Incremental learning of human activity models from videos," Computer Vision and Image Understanding, vol.144, pp.24–35, 2016.
[24] Z.S. Abdallah, M.M. Gaber, B. Srinivasan, and S. Krishnaswamy, "Adaptive mobile activity recognition system with evolving data streams," Neurocomputing, vol.150, pp.304–317, 2015.
[25] C.C. Loy, T. Xiang, and S. Gong, "Incremental activity modeling in multiple disjoint cameras," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.34, no.9, pp.1799–1813, 2011.
[26] M.B. Naceur, R. Saouli, M. Akil, and R. Kachouri, "Fully automatic brain tumor segmentation using end-to-end incremental deep neural networks in MRI images," Computer methods and programs in biomedicine, vol.166, pp.39–49, 2018.
[27] X. Bai, P. Ren, H. Zhang, and J. Zhou, "An incremental structured part model for object recognition," Neurocomputing, vol.154, pp.189–199, 2015.
[28] J. Dou, J. Li, Q. Qin, and Z. Tu, "Moving object detection based on incremental learning low rank representation and spatial constraint," Neurocomputing, vol.168, pp.382–400, 2015.
[29] M.K.S. Alsmadi, K.B. Omar, S.A. Noah, and I. Almarashdah, "Fish recognition based on the combination between robust feature selection, image segmentation and geometrical parameter techniques using Artificial Neural Network and Decision Tree," arXiv preprint arXiv:0912.0986, 2009.
[30] K. Diaz-Chito, F.J. Ferri, and W. Diaz-Villanueva, "Incremental generalized discriminative common vectors for image classification," IEEE transactions on neural networks and learning systems, vol.26, no.8, pp.1761–1775, 2014.
[31] N.A.M. Zin, S.N.H.S. Abdullah, and A. Abdullah, "Improved CAMshift based on supervised learning," Robot Intelligence Technology and Applications 2012, Springer, vol.208, pp.611–621, 2013.
[32] G. Yang, Z. Hu, and J. Tang, "Robust visual tracking via incremental subspace learning and local sparse representation," Arabian Journal for Science and Engineering, vol.43, no.2, pp.627–636, 2018.
[33] J. Dou, J. Li, Q. Qin, and Z. Tu, "Robust visual tracking based on incremental discriminative projective non-negative matrix factorization," Neurocomputing, vol.166, pp.210–228, 2015.
[34] M.A. Zulkifley, N.S. Samanu, N.A.A.N. Zulkepeli, Z. Kadim, and H.H. Woon, "Kalman filter-based aggressive behaviour detection for indoor environment," Information Science and Applications (ICISA) 2016, Springer, vol.376, pp.829–837, 2016.
[35] Z. Ismail and R. Hassan, "A performance study of various mobility speed on AODV routing protocol in homogeneous and heterogeneous MANET," The 17th Asia Pacific Conference on Communications, IEEE, pp.637–642, 2011.
[36] A. Nielsen, Practical Time Series Analysis: Prediction with Statistics and Machine Learning, O'Reilly Media, 2019.
[37] X. Chen, C. Wu, Z. Liu, N. Zhang, and Y. Ji, "Computation Offloading in Beyond 5G Networks: A Distributed Learning Framework and Applications," IEEE Wireless Communications, vol.28, no.2, pp.56–62, 2021. (accept for publication)
[38] G.-B. Huang, N.-Y. Liang, H.-J. Rong, P. Saratchandran, and N.

Sundararajan, "On-line sequential extreme learning machine," Computational Intelligence, vol.2005, pp.232–237, 2005.
[39] X. Jiang, J. Liu, Y. Chen, D. Liu, Y. Gu, and Z. Chen, "Feature adaptive online sequential extreme learning machine for lifelong indoor localization," Neural Computing and Applications, vol.27, no.1, pp.215–225, 2016.
[40] A.S. Al-Khaleefa, M.R. Ahmad, A.A.M. Isa, M.R.M. Esa, Y. Aljeroudi, M.A. Jubair, and R.F. Malik, "Knowledge Preserving OSELM Model for Wi-Fi-Based Indoor Localization," Sensors, vol.19, no.10, p.2397, 2019.
[41] A.S. Al-Khaleefa, M.R. Ahmad, A.A.M. Isa, M.R.M. Esa, A. Al-Saffar, and Y. Aljeroudi, "Infinite-Term Memory Classifier for Wi-Fi Localization Based on Dynamic Wi-Fi Simulator," IEEE Access, vol.6, pp.54769–54785, 2018.
[42] 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.
[43] H. Tang, W. Liu, W.-L. Zheng, and B.-L. Lu, "Multimodal emotion recognition using deep neural networks," International Conference on Neural Information Processing, Springer, pp.811–819, 2017.
[44] L. Guo, J.-H. Hao, and M. Liu, "An incremental extreme learning machine for online sequential learning problems," Neurocomputing, vol.128, pp.50–58, 2014.
[45] M. Pratama, G. Zhang, M.J. Er, and S. Anavatti, "An incremental type-2 meta-cognitive extreme learning machine," IEEE transactions on cybernetics, vol.47, no.2, pp.339–353, 2016.
[46] S. Xu and J. Wang, "A fast incremental extreme learning machine algorithm for data streams classification," Expert systems with applications, vol.65, pp.332–344, 2016.
[47] A.S. AL-Khaleefa, M.R. Ahmad, A.A.M. Isa, M.R.M. Esa, A. Al-Saffar, and M.H. Hassan, "Feature adaptive and cyclic dynamic learning based on infinite term memory extreme learning machine," Applied Sciences, vol.9, no.5, p.895, 2019.
[48] A. Özgür and H. Erdem, "A review of KDD99 dataset usage in intrusion detection and machine learning between 2010 and 2015," PeerJ Preprints, vol.4, e1954v1, 2016.
[49] W. Lee and S.J. Stolfo, "A framework for constructing features and models for intrusion detection systems," ACM transactions on Information and system security (TiSSEC), vol.3, no.4, pp.227–261, 2000.
[50] B. Pfahringer, "Winning the KDD99 classification cup: bagged boosting," ACM SIGKDD Explorations Newsletter, vol.1, no.2, pp.65–66, 2000.
[51] M. Sabhnani and G. Serpen, "Why machine learning algorithms fail in misuse detection on KDD intrusion detection data set," Intelligent data analysis, vol.8, no.4, pp.403–415, 2004.
[52] M.S. Al-Daweri, K.A.Z. Ariffin, S. Abdullah, and M.F.E.M. Senan, "An Analysis of the KDD99 and UNSW-NB15 Datasets for the Intrusion Detection System," Symmetry, vol.12, no.10, p.1666, 2020.
[53] J. Torres-Sospedra, R. Montoliu, A. Martinez-Usó, J.P. Avariento, T.J. Arnau, M. Benedito-Bordonau, and J. Huerta, "UJIIndoorLoc: A new multi-building and multi-floor database for WLAN fingerprint-based indoor localization problems," 2014 international conference on indoor positioning and indoor navigation (IPIN), IEEE, pp.261–270, 2014.
[54] Y. Jiang, X. Pan, K. Li, Q. Lv, R.P. Dick, M. Hannigan, and L. Shang, "Ariel: Automatic wi-fi based room fingerprinting for indoor localization," Proceedings of the 2012 ACM conference on ubiquitous computing, pp.441–450, 2012.
[55] F. Pirahansiah, S.N.H.S. Abdullah, and S. Sahran, "Simultaneous Localization And Mapping Trends And Humanoid Robot Linkages," Asia-Pacific Journal of Information Technology and Multimedia, vol.2, p.12, pp.27–38, 2013.

Ahmed Salih Al-Khaleefa received the B.Sc. degree in software engineering from Imam Ja'afar Al-Sadiq University, Iraq, in 2013, and the M.Sc. degree in computer science networks department from Universiti Kebangsaan Malaysia, Malaysia, in 2017. He received Ph.D. degree in telecommunication engineering from the Faculty of Electronic Engineering and Computer Engineering (FKEKK), Universiti Teknikal Malaysia Melaka, Malaysia. He is currently serving as Senior Lecturer in the Department of Computer Engineering, Faculty of Information Technology University of Imam Jafar Al-Sadiq, Maysan, Iraq His research interests include communication, security, routing protocols, artificial intelligence, IoT, Optimization, and Localization.

Rosilah Hassan is an Associate Professor at Universiti Kebangsaan Malaysia (UKM) in the Faculty of Information Science and Technology. She received her Ph.D. in Mobile Communication from the University of Strathclyde, United Kingdom, in May 2008. She obtained her Master of Electrical (M.E.E) Engineering in Computer and Communication from the UKM, Malaysia, in 1999. Her first Degree was BSc. in Electronic Engineering from Hanyang University, South Korea. Rosilah Hassan worked as an Engineer with Samsung Electronic Malaysia in Seremban, Malaysia before joining UKM in 1997. Besides being a senior lecturer, she got experience in management post for the university as the Deputy Director of Academic Entrepreneurship for over seven years. She also is the Head of the Network Communication Technology (NCT) lab in her Faculty. Her research interests are in Mobile Communications, Networking, IoT, and Big Data. She has had experience as an external examiner for Ph.D. and Master for both national and international level. She is also an active member of IEEE, WIE, Malaysia Society for Engineering (MySET), Malaysian Board of Technologists (MBoT), and IET.

Mohd Riduan Ahmad received the Degree (Hons.) in computer system and communication engineering from Universiti Putra Malaysia in 2003, the M.Eng. degree with a specialization in cross-layer design of MAC protocols for multi-in multi-out-based wireless sensor network from the University of Wollongong, Australia, in 2008, and the Ph.D. degree with a specialization in atmospheric discharges from Uppsala University, Sweden, in 2014. From 2015 to 2016, he was with MIT, USA, where he focused on the understanding and characterization of microwave radiation emitted by lightning flashes. He is currently a Senior Lecturer with the Faculty of Electronics and Computer Engineering, Universiti Teknikal Malaysia Melaka.

Faizan Qamar has a Ph.D. degree in Wireless Networks from the Faculty of Engineering, University of Malaya, Kuala Lumpur, Malaysia in October 2019. He had completed M.E. degree in Telecommunication from NED University, Karachi, Pakistan in 2013, and B.E. degree in Electronics from Hamdard University, Karachi, Pakistan, in 2010. He is currently serving as Senior Lecturer in the Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia (UKM), Selangor, Malaysia. He has more than eight years of research and teaching experience. He has authored and co-authored numerous ISI & Scopus journals and IEEE conference papers. He is also a reviewer of several national & international journals and IEEE conferences proceedings. His research interests include interference management, millimeter-wave communication, IoT networks, D2D communication and Quality of Service enhancement for the future wireless networks.

Zheng Wen received his B.E. degrees in Computer Science and Technology at Wuhan University, Wuhan, China, and obtained his M.Sc. and Ph.D. degrees in communications engineering from Waseda University, Tokyo, Japan. He is an Assistant Professor in Department of Communications and Computer Engineering at Waseda University. His research interests lie in the area of ICT system, including the information-centric networking, emergency management system, IoT, edge computing, big data and block-chain technologies. He is a member of IEEE, IEICE.

Azana Hafizah Mohd Aman received the B.Eng., M.Sc., and Ph.D. degrees in computer and information engineering from International Islamic University Malaysia, Malaysia. She is currently working as Senior Lecturer at the Research Center for Cyber Security, Faculty of Information Science and Technology (FTSM), The National University of Malaysia, Malaysia. Her research interests include computer system and networking, computer information and network security, the Internet of Things (IoT), cloud computing, and big data.

Keping Yu received the M.E. and Ph.D. degrees from the Graduate School of Global Information and Telecommunication Studies, Waseda University, Tokyo, Japan, in 2012 and 2016, respectively. He was a Research Associate and a Junior Researcher with the Global Information and Telecommunication Institute, Waseda University, from 2015 to 2019 and 2019 to 2020, respectively, where he is currently a Researcher (Assistant Professor). Dr. Yu has hosted and participated in more than ten projects, is involved in many standardization activities organized by ITU-T and ICNRG of IRTF, and has contributed to ITU-T Standards Y.3071 and Supplement 35. He received the Best Paper Award from ITU Kaleidoscope 2020, the Student Presentation Award from JSST 2014. He has authored 100+ publications including papers in prestigious journal/conferences such as the IEEE WirelComMag, NetMag, TFS, IoTJ, TII, T-ITS, TVT, TNSE, TGCN, CEMag, IoTMag, ICC, GLOBECOM etc. He is an Editor of IEEE Open Journal of Vehicular Technology (OJVT). He has been a Lead Guest Editor for Sensors, Peer-to-Peer Networking and Applications, Energies and Guest Editor for IEICE Transactions on Information and Systems, Computer Communications, IET Intelligent Transport Systems, Wireless Communications and Mobile Computing. He served as general co-chair and publicity co-chair of the IEEE VTC2020-Spring 1st EBTSRA workshop, general co-chair of IEEE ICCC2020 2nd EBTSRA workshop, session chair of IEEE ICCC2020, TPC co-chair of SCML2020, local chair of MONAMI 2020, Session Co-chair of CcS2020, and session chair of ITU Kaleidoscope 2016. His research interests include smart grids, information-centric networking, the Internet of Things, artificial intelligence, blockchain, and information security.