Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Significance of activation functions in developing an online classifier for semiconductor defect detection

Md Meftahul Ferdaus [a], Bangjian Zhou [a], Ji Wei Yoon [a], Kain Lu Low [b], Jieming Pan [b], Joydeep Ghosh [b], Min Wu [a], Xiaoli Li [a], Aaron Voon-Yew Thean [b], J. Senthilnath [a],*

[a] *Institute for Infocomm Research, Agency for Science, Technology and Research (A*STAR), 138632, Singapore*
[b] *Electrical and Computer Engineering, National University of Singapore, 117583, Singapore*

ARTICLE INFO

Article history:
Received 24 September 2021
Received in revised form 9 April 2022
Accepted 12 April 2022
Available online 21 April 2022

Keywords:
Leaky ReLU
Online learning
Defect detection
Prequential
Semiconductors

ABSTRACT

In anomaly detection problems for advanced semiconductor devices, non-visual defects occur frequently. Machine learning (ML) algorithms have the advantage of identifying such defects. However, in this real-world problem, data comes sequentially in a streaming fashion, thus, we may not have sufficient data to train an ML model in batch mode. In such a scenario, online ML models are useful to detect defects immediately since they work in a single-pass mode. Besides, when data is collected from more realistic non-stationary monitoring environments, online ML models with evolving architecture are more practical. Thus, evolving and online ML models are developed in this work to detect defects in technology computer-aided design (TCAD)-based digital twin model of advanced nano-scaled semiconductor devices such as a fin field-effect transistor (FinFET) and a gate-all-around field-effect transistor (GAA-FET). Activation functions (AFs) in deep neural networks (DNNs) and membership functions (MFs) in neuro-fuzzy systems (NFSs) play an important role in the performance of those ML models. This work focuses on analyzing the effects of various AFs/MFs in our developed online ML models while detecting defects in real-world nano-scaled semiconductor devices, where significant training samples are not available. From various semiconductor datasets having fewer samples, it has been observed that the proposed evolving neuro-fuzzy system (ENFS) with Leaky-ReLU MF performs better (improvement in the range of 1.9% to 30.8% considering overall classification accuracy) than the other DNN or ENFS-based online ML models. Having an evolving architecture and online learning mechanism, besides anomaly detection, the proposed model's performance has also been evaluated for handling large data streams problems with concept drift. The performance of the proposed method has been compared with some recently developed baselines under the prequential test-then-train protocol. The classification rates of the proposed method has an improvement in the range of 1.1% to 65.9% than the existing methods. The code of this work has been made publicly available at https://github.com/MdFerdaus/LREC.

1. Introduction

Miniaturization and fast-paced recent developments in sensors and actuators have enabled us to measure the states of various tools or parts of a manufacturing process. Measuring states from various points provide us with real-time information about the health of the whole process. This information can then be utilized to train a machine learning (ML) model to identify defects in a certain component [1]. Such anomaly detection procedure is important for the manufacturer to meet the consumers' demand and supply constantly. Further, the ML-guided process monitoring system can minimize the predictive maintenance cost by lowering the labor cost and wastage of materials. However, there exist numerous challenges in using ML to identify the defects in advanced miniaturized devices [2]. Among a variety of miniaturized devices, this work deals with the challenges of analyzing defects in some advanced semiconductor devices, namely, a fin field-effect transistor (FinFET) and a gate-all-around field-effect transistor (GAA-FET). Some challenges of using ML in these devices are discussed in the following paragraph.

When the dimension of semiconductor devices scales down at a nano-meter technology node and beyond, complexity in both interconnection and transistor process escalates [3]. Thus, chances of having non-visual defects in these devices increase.

* Corresponding author.
E-mail addresses: ferdaus_meftahul@i2r.a-star.edu.sg (M.M. Ferdaus), Zhou_Bangjian@i2r.a-star.edu.sg (B. Zhou), Yoon_Ji_Wei@i2r.a-star.edu.sg (J.W. Yoon), kainlulow@sjtu.edu.cn (K.L. Low), jieming.p@u.nus.edu (J. Pan), elejg@nus.edu.sg (J. Ghosh), wumin@i2r.a-star.edu.sg (M. Wu), xlli@i2r.a-star.edu.sg (X. Li), aaron.thean@nus.edu.sg (A.V.-Y. Thean), J_Senthilnath@i2r.a-star.edu.sg (J. Senthilnath).

https://doi.org/10.1016/j.knosys.2022.108818
0950-7051/© 2022 Elsevier B.V. All rights reserved.

Indeed, several tools are used to improve general failure analysis methods, for example, nano-probing using scanning electron microscopy, conductive atomic force microscope, and transmission electron microscopy (TEM). Nonetheless, most of these solutions are time-consuming and hard to implement. In contrast to a FinFET device, a futuristic FET called gate-all-around FET (GAA-FET) device is expected to provide a better gate voltage-regulation over the channel, consequently minimizes the leakage of current that may occur due to several short channel effects [4]. In GAA-FET, super-lattices consist of Si and SiGe layers. The lattice mismatch between Si and SiGe induces strain in both layers which may cause dislocation defects. It can also be considered as a non-visual defect. It has been observed that faults occur very rarely in the semiconductor. Therefore, it becomes tedious to collect enough statistically-significant failing samples. In such circumstances, an ML-guided defect detector could be an effective replacement for TEM analysis-based solutions or an alternative to domain experts' review [5]. Since faults occur rarely, it causes an imbalance in the number of samples at different classes. Dealing with such a multi-class imbalanced classification problem is really challenging for the ML models.

While handling defect identification problems in advanced semiconductor devices, class imbalance dataset is not the only challenge for ML models. In many cases, manufacturers desire to detect the defect in an online fashion while data keeps coming sequentially in a streaming fashion. The majority of the ML models, operated in batch learning mode, are not compatible to deal with such scenarios. Besides, the distribution of data may change at a varying rate, consequently posing a challenge to fixed architecture ML models. To overcome the above-mentioned limitations, online ML models with evolving architecture are useful [6,7]. These models' ability to operate in a single-pass mode supports them to deal with sequentially incoming data streams effectively [8,9], where their evolving architecture helps to deal with varying data distribution. Thus, we are focusing on developing novel evolving/online ML models to detect defects in an online fashion, where we are emphasizing the utilization of various AFs/MFs to detect defects in the advanced semiconductor industry.

With the recent advancements in hardware and software technology, researchers have implemented various ML models [10–16] for different defect detection problems [17,18]. The majority of these models operate in batch learning mode. Unlike those examples, online ML models based on deep neural networks (DNNs) and evolving neuro-fuzzy systems (ENFSs) are proposed in this work since they can deal with streams of data coming sequentially. To handle data streams in complex environments, an ensemble fuzzy classifier called parsimonious ensemble (pENsemble) was developed in [19]. Their method was equipped with a dynamic online feature selection mechanism that can select or deselect the input features on the fly. Besides, the ensemble pruning mechanism supports pENsemble to reduce its computational complexity and to deal with the rapidly altering environment. Researchers developed a deep stacked stochastic configuration network (DSSCN) [20] for handling non-stationary data streams in the lifelong learning environment. Continuously generated data streams are handled in DSSCN by its self-constructive architecture through automatic extraction of both hidden units and layers. They have tested their algorithm in a prequential test-then-train process. Inspired by the better generalization power of the deep learning methods, a deep evolving fuzzy neural network (DEVFNN) is developed for handling non-stationary data streams [21]. Each layer of their network is derived from Generic Classifier (gClass) [22]. A new layer is added to deepen DEVFNN's architecture when a real drift is detected. Besides, it is equipped with a layer merging mechanism, which supports reducing network complexity by compromising the generalization performance negligibly. Again, the number of rules in each layer is adjusted automatically from the data streams. The performance of their algorithm was also tested by following the test-then-train protocol.

Majority of above mentioned neuro-fuzzy systems are rooted with either univariate Gaussian membership functions (MFs) [23,24], or multivariate MFs [25–27]. In this work, besides using these MFs, a Leaky ReLU MF-based novel ENFS is proposed. The influence of these MFs on ENFS-based autonomous and online classifiers has also been observed in this work while detecting defects in advanced semiconductor devices. In addition, to understand the impact of various AFs like ReLU, leaky ReLU, sigmoid, and tanh, they have been used in DNN to develop new online learning methods to detect defects in advanced semiconductor devices.

2. Related work

Researchers have tried various ML techniques to detect defects in semiconductor manufacturing. Some of them developed reference cycles for data comparison [28]. A standard cycle is created from data to be considered clean, which has been used as a reference to compare with the defective cycles. Such a technique supports them in determining effectively the process parameters that are causing the problems. Some researchers have tried to use such a reference cycle to conduct cluster analysis [29]. Applying cluster analysis, data are processed to detect and visualize defects. Various clustering algorithms such as k-means, k-medoids, CLARA, agglomerative, and divisive approaches are employed to both univariate and multivariate data to cluster the clean data with the reference cycle, whereas the defective cycles are clustered separately. Multi-Layer Perceptron (MLP) and Long Short-Term Memory (LSTM) models are also used to identify defects in semiconductor manufacturing as a time-series forecasting [30]. They have used the clustering technique to create labels for supervised forecasters like MLP and LSTM. They trained their models with clean data so that they can forecast clean data with lower residual values. In such a setting, a higher residual value indicates defective data. A conditional generative adversarial network-based model is developed in [31] to detect and classify defects in the semiconductor manufacturing process. In another work, a single convolutional neural network (CNN) is utilized to identify and classify defects in the wafer surface. However, these approaches have utilized a lot of scanning electron microscope (SEM) images to train their model, which may not be available in many real-world problems. Our work is focused on supervised learning-based ML models to detect defects in semiconductor devices. However, rather than focusing on differentiating the clean from the defective data, we focus on developing a classifier to classify various faults in semiconductor devices using tabular data. In real-world problems, data comes sequentially, and there may exist various drifts in the dataset. To deal with these issues, we emphasize online learning-based evolving structured classifiers over batch learning-based ML models as used in the literature to detect defects.

In online learning algorithms, the models are updated from a continuous streams of data, where multiple passes over data are not performed. In data stream analysis, concept drift is commonly observed phenomenon, which indicates the arbitrary change of statistical behaviors of a target space over time [6]. The generalized framework for handling concept drifts in machine learning has also been illustrated in [6]. Among various ML models, evolving architecture-based online learning algorithms have been successfully employed by many researchers [6,19,32–34], which motivates us to further develop new evolving ML model. In our proposed evolving neuro-fuzzy system, we have focused on reducing the number of learning parameters through the utilization of a Leaky-ReLU-based membership function. To sum up, we have firstly used our proposed method to detect defect in semiconductor, which has been later reformed into prequential test-then-train protocol to deal with large data stream problems with various concept drifts.

The main contributions in this work can be summarized as follows:

- Unlike the commonly used batch-processed DNN, an online learning-based DNN classifier is proposed in this paper. The influence of various AFs such as ReLU, Leaky ReLU, Sigmoid, and tanh in the online DNN model have been investigated while detecting defects in advanced semiconductor devices and/or semiconductor manufacturing process.
- Impact of various MFs such as univariate Gaussian function, multivariate Gaussian function, and Leaky ReLU in ENFSs has also been observed. The incorporation of the proposed Leaky ReLU MF in neuro-fuzzy architecture has reduced the number of learning parameters significantly. If n is the number of input features and R denotes the number of rules, then the total number of learning parameters in a univariate Gaussian function-based ENFS is $(R \times n) + R + (n + 1) \times R$. In the case of multivariate Gaussian function-based ENFS, it is $(R + n) + (n \times n) \times R + (n + 1) \times R$. In our proposed Leaky ReLU-based ENFS, it has been reduced to only $(n + 1) \times R$. The reduction of learning parameters supports the proposed ENFS to minimize the computational burden.
- Real-world datasets from our TCAD-based digital twin model of nano-scaled FinFET and GAA-FET are considered. Some challenging features associated with these datasets are as follows: (1) less number of samples; (2) imbalance and multiclass (9 and 10 classes). Thus, these datasets are utilized to evaluate the performance of our proposed online classifiers along with some benchmark models.
- A popular evaluation mechanism for online and evolving learning models in handling data streams namely prequential test-then-train protocol is used to evaluate the performance of the proposed LREC. Under such settings, the performance of LREC has been compared with recently developed baselines.

The remaining paper can be organized as follows: Section 3.1 refers to the problem state of the work. The proposed Leaky ReLU MF-based ENFS named as Leaky ReLU-based Evolving Classifier (LREC) is explained along with its architecture in Section 3. Section 4 explains the online learning policy of the proposed LREC. All the semiconductor datasets, used to evaluate the online classifiers, are detailed in Section 5. The performance of various online classifiers in detecting defects of semiconductor devices is reported and described in Section 6. Finally, the paper ends with the concluding remarks in Section 8.

3. Leaky ReLU-based evolving classifier

The design of ENFS-based classifier involves a high number of learning parameters due to the utilization of various Gaussian MFs. Such feature may hamper their performance in detecting defects online. With the motivation of using less learning parameters, the popular leaky ReLU is incorporated as an MF in our proposed LREC. LREC is used to detect the location of defects in advanced FET devices namely a FinFET and a GAA-FET device with only a few samples extracted from the experimental analysis. The detailed architecture of the LREC is shown in Fig. 3, where the process of collecting the input data and passing it to LREC is discussed in the Experiment Section.

3.1. Problem statement

In this work, we proposed an evolving classifier called LREC in solving two different problems, such as (i) detection of anomaly through a single-pass online learning mechanism; and (ii) handling concept drift through the prequential test-then-train protocol. In single-pass settings, data comes in a sequential online manner to train LREC in a single-pass fashion. It is tested after training is finished and it can alter the architecture during the training phase. To get a clearer overview, the single-pass-based working mechanism of LREC is portrayed in Fig. 1. Unlike the state of the arts evolving models, it requires less parameters, which supports its real-world deployment in applications like semiconductor defect detection. On the other hand, in a prequential test-then-train procedure, a model is tested first to evaluate its generalization performance before it is being trained. Such a mechanism is helpful in data streams by considering the fact of the arrival of data streams without labels since the ground truth may not be captured instantaneously. A clearer view of the workflow in LREC, while operating in a prequential test-then-train setting is shown in Fig. 2. Here, $C_t = [X_1, X_2, \ldots, X_p] \in \Re^{N \times n}$ is representing unlabeled data streams with a size of N and input dimension of P, $Y \in \Re^N$ is the true class labels for N unlabeled data points.

3.2. LREC architecture

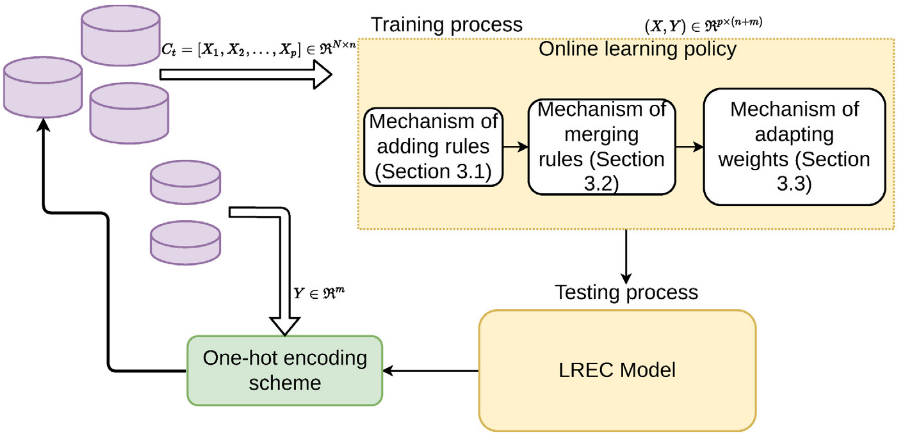
The proposed LREC is bottomed with Takagi–Sugeno (TS)-fuzzy architecture. Therefore, the LREC structure consists of three parts, namely, (1) fuzzification; (2) rule base; (3) defuzzification. The incoming streams of crisp data are fuzzified in layer 1 (fuzzification layer). From previous research [32,35], it is observed that the generation of linear equations in the consequent part of the TS-fuzzy system supports hyper-plane-shaped clusters (HPSCs) to develop an efficient TS-fuzzy model than the hyper-spherical-shaped cluster-based models. The MF in PALM [32] can accommodate the hyper-planes directly, which can be expressed as $\exp\left(-\Gamma \frac{d(j)}{\max(d(j))}\right)$, where j is the index of the current rules, Γ is a tuning parameter with a range of $[1 - 100]$ [32,35] to control the fuzziness in membership grades and $d(j)$ is a distance of the input at a certain time step to the jth hyper-plane. The Γ is required to be varied with respect to the various dataset. To mitigate such shortcoming, the concept of leaky ReLU is incorporated in LREC to accommodate the hyper-planes directly as follows:

$$f_{L1_j} = \mu_j = f(x) = \begin{cases} A\omega_j x & : \omega_j x < 0 \\ \omega_j x & : \omega_j x > 0 \end{cases} \tag{1}$$

where A is a constant with small value ($A < 0.1$), $x = x_e \in \Re^{1\times(n+1)}$ is the extended input vector. At the kth time step, it can be expressed as $x_e^k = [1, x_1^k, x_2^k, \ldots, x_n^k]$, here n is the number of input features and $\omega_j \in \Re^{(n+1)\times j}$ stands for the weight vector of the jth rule. This study focuses on type-1 TS-fuzzy architecture to develop the LREC algorithm. The next layer of LREC is rule base. The IF-THEN rule of the proposed learning algorithm is illustrated as follows:

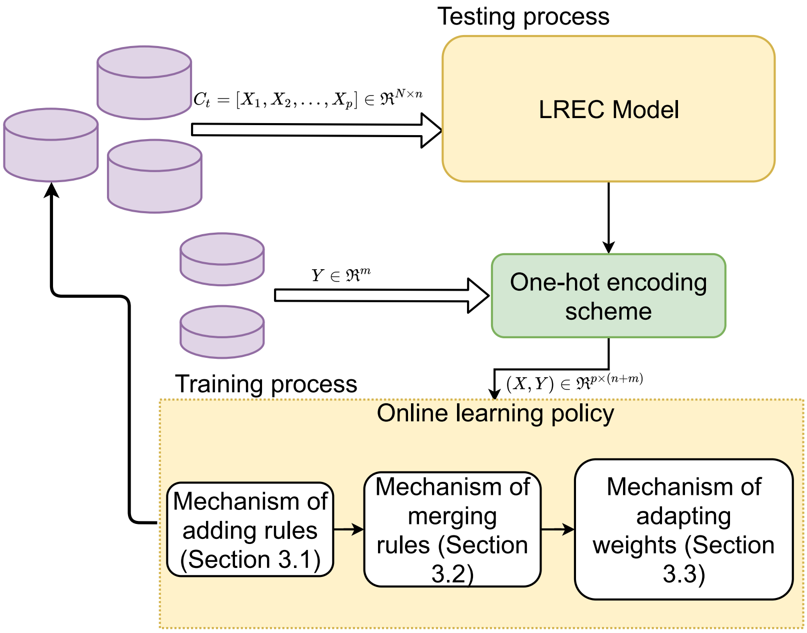
$$R^j : \quad \text{IF } X_n \text{ is close to } f_{L2_j} \text{ THEN } y_j = \lambda_j x_e^T \omega_j \tag{2}$$

where $\lambda_j = (\tau_j / \sum_{i=1}^{R} \tau_i)$ is the normalized firing strength for the jth rule, τ_j is the firing strength of the jth rule, which is the Cartesian product of respective fuzzy sets of that rule. τ_j can be expressed as $\tau_j = \prod_{i=1}^{n} \mu_{ji}$, ω_j is the vector of consequent parameters called weight for the jth rule, y_j is the consequent part of the jth rule. Since the rule base is the layer after fuzzification,



Training process
$(X,Y) \in \Re^{p\times(n+m)}$
$C_t = [X_1, X_2, \dots, X_p] \in \Re^{N\times n}$
Online learning policy
Mechanism of adding rules (Section 3.1)
Mechanism of merging rules (Section 3.2)
Mechanism of adapting weights (Section 3.3)
$Y \in \Re^m$
Testing process
One-hot encoding scheme
LREC Model

Fig. 1. Work flow in single-pass online LREC.



Testing process
$C_t = [X_1, X_2, \dots, X_p] \in \Re^{N\times n}$
LREC Model
$Y \in \Re^m$
One-hot encoding scheme
Training process
$(X,Y) \in \Re^{p\times(n+m)}$
Online learning policy
Mechanism of adding rules (Section 3.1)
Mechanism of merging rules (Section 3.2)
Mechanism of adapting weights (Section 3.3)

Fig. 2. Work flow in LREC under prequential test-then-train procedure.

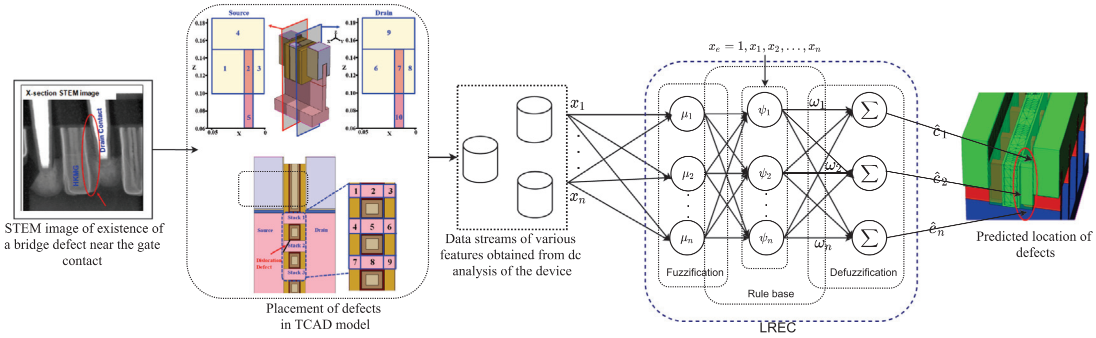


Fig. 3. Overall flow of LREC-based defect detection.

the output of this layer is indicating the consequent part. For a certain rule, it can be expressed as follows:

$$f_{L2_j} = x_e^T \omega_j \tag{3}$$

The last layer is known as the defuzzification layer, where the fuzzified output of the previous layer is defuzzified to the desired crisp output. This crisp output is the weighted average of the individual rules' contribution, which is presented as follows:

$$f_{L3} = \hat{y} = \sum_{j=1}^{R} \lambda_j x_e^T \omega_j \tag{4}$$

The term λ in Eq. (4) is guaranteeing the partition of unity where the sum of normalized membership degree is unity. Usually, in a TS-fuzzy-based learning algorithm, both antecedent and consequent parameters are to be learned. However, there are no antecedent parameters like centers, widths in LREC. Thus, in this work, we need to learn only the consequent parameters.

In this work, for all the evolving framework-based classifiers, the true class labels are transformed to either 0 or 1. For instance, a class label of 2 is converted as [0, 1]. Theoretically, the concept of rule-based one versus rest fuzzy classifier [36] is inserted here which provides more convenience in multi-class classification problems. The final decision from the proposed classifier can be expressed as follows:

$$O_{LR} = \arg\max_{c=1,\ldots,M}(O^{c}_{LR}) \tag{5}$$

where O_{LR} is denoting the predicted class label, $c = 1, 2, \ldots, M$ is the number of classes. The desired class is predicted from the maximal rule firing strength over all the rules. The online learning mechanism of the proposed LREC's consequent parameters along with the evolving structure learning policy of LREC is discussed in the next section.

4. Online Learning Mechanism in LREC

In this section, the online learning strategy to construct the LREC autonomously is discussed. Here, data to train the model is collected in a sequential manner, rather than a batch of data. In some cases, new data may or may not contain novel information than the previously existed ones. Hence, to accommodate the variety of continuously non-stationary data during the training phase, LREC can alter its structure by adding or merging the rules.

4.1. Mechanism of adding rules

In LREC, autonomous construction of fuzzy rules is realized using self constructive clustering (SCC) [32,37–39]. In this mechanism, both the input and output coherence are calculated to measure the significance of a rule. To calculate the coherence, the correlation between the existing samples and the target concept is analyzed. Let us assume that $\mathcal{H}_j \in \Re^{R\times(1+n)}$ is a hyper-plane for the jth rule, $X_t \in \Re^{n}$ is the input vector, $T_t \in \Re^{m}$ is the target vector, where n and m are the input and output dimensions respectively, and $t = 1, 2, \ldots, n/m$. Now the input and output coherence of the LREC can be expressed as follows:

$$I_c(\mathcal{H}_j,\, X_t) = \xi(\mathcal{H}_j,\, X_t) \tag{6}$$

$$O_c(\mathcal{H}_j,\, X_t) = \xi(X_t, T_t) - \xi(\mathcal{H}_j, T_t) \tag{7}$$

where $\xi(\,)$ expresses the correlation function. In LREC, Maximal information Compression Index (MCI) [40] method-based measurement of correlation is utilized in the SSC method as follows:

$$\xi(X_t, T_t) = \frac{1}{2}(\mathrm{var}(X_t) + \mathrm{var}(T_t)) \\ -\sqrt{(\mathrm{var}(X_t) + \mathrm{var}(T_t))^2 - 4\mathrm{var}(X_t)(T_t)(1 - \rho(X_t, T_t)^2))} \tag{8}$$

$$\rho(X_t, T_t) = \frac{\mathrm{cov}(X_t, T_t)}{\sqrt{\mathrm{var}(X_t)\mathrm{var}(T_t)}} \tag{9}$$

where $\mathrm{var}(X_t)$, $\mathrm{var}(T_t)$ express the variance of X_t and T_t respectively, $\mathrm{cov}(X_t, T_t)$ presents the covariance between two variables X_t and T_t, $\rho(X_t, T_t)$ stands for Pearson correlation index of X_t and T_t. In a similar way, the correlation $\xi(\mathcal{H}_j,\, X_t)$ and $\xi(\mathcal{H}_j,\, T_t)$ can be measured using Eqs. (8) and (9). In addition, the MCI method measures the compressed information when a newly observed sample is ignored. The similarity between $\mathcal{H}_j$ and X_t is explored directly while calculating $I_c(\mathcal{H}_j,\, X_t)$. On the other hand, $O_c(\mathcal{H}_j,\, X_t)$ examines the dissimilarity between $\mathcal{H}_j$ and X_t in an indirect way by considering the target vector as a reference.

The value of I_c and O_c should fulfill the following conditions to append rules:

$$I_c(\mathcal{H}_j,\, X_t) > b_1 \quad \text{and} \quad O_c(\mathcal{H}_j,\, X_t) < b_2 \tag{10}$$

where the ranges for the predetermined thresholds b_1 and b_2 for all three datasets used in this study are as follows: $b_1 \in [0.03,\ 10.0]$, and $b_2 \in [0.01,\ 1.0]$.

4.2. Mechanism of merging rules

For merging HPSCs, the angle between the hyper-planes is measured in [41]. It is not sufficient to understand their relationship in the target-space since it cannot entail the spatial proximity between HPSCs. From this gap, not only the angles but also the minimum distance between clusters are measured in LREC. By following the same approach in [25], the angle between HPSCs are measured as follows:

$$\theta_{j,j+1} = \arccos\left(\left|\frac{\omega_j^T \omega_{j+1}}{|\omega_j \parallel \omega_{j+1}|}\right|\right) \tag{11}$$

where the obtained angle $\theta_{j,j+1}$ between jth and $(j+1)$th cluster is ranged between 0 and π radian, $\omega_j = \left[b_{1,j},\ b_{2,j},\ \ldots,\ b_{k,i}\right],\ \omega_{j+1} = \left[b_{1,j+1},\ b_{2,j+1},\ \ldots,\ b_{k,j+1}\right]$. Now, the minimum distance between the clusters is measured as follows:

$$d_{j,j+1} = \left|(a_1 - a_2).\frac{(b_1 \times b_2)}{|b_1 \times b_2|}\right| \tag{12}$$

where a_1, a_2, b_1, b_2 are the parameters of the two considered hyper-planes expressed as $l_{R1} = a_1 + xb_1$, and $l_{R2} = a_2 + xb_2$. Now, the conditions to merge the hyper-planes can be expressed as:

$$\theta_{j,j+1} \le c_1 \text{ and } d_{j,j+1} \le c_2 \tag{13}$$

where $c_1 \in [0.01, 0.1]$ and $c_2 \in [0.001, 0.1]$ are user-defined parameters for all three datasets used in this work.

When merging two rules or clusters, the less dominant one between them is deleted to maintain the structural simplicity in LREC. On the other hand, the dominant rule with a higher number of samples has a higher influence during the merging scenario and presents the underlying data distribution. While two HPSCs are merged, the weighted average strategy is employed for simplicity as follows:

$$\omega_j^{new} = \frac{\omega_j^{old}N_j^{old} + \omega_{j+1}^{old}N_{j+1}^{old}}{N_j^{old} + N_{j+1}^{old}} \tag{14}$$

$$N_j^{new} = N_j^{old} + N_{j+1}^{old} \tag{15}$$

where ω_j^{old} and ω_{j+1}^{old} are output weight vectors of the jth and $(j+1)$th rule respectively, and ω_j^{new} is expressing the output weight vector of the merged rule, N is the number of samples, also called as population in a HPSC or rule. When $N_j > N_{j+1}$, then the jth rule is more influential than the $(j+1)$th rule.

4.3. Mechanism of adapting weights

To adapt the weights of LREC, in this work, Fuzzily Weighted Generalized Recursive Least Square (FWGRLS) method [25,32] has been employed. The FWGRLS method-based loss function for the jth rule of the LREC is expressed as follows:

$$\mathcal{L}_j = (y - x_e\pi_j)\Lambda_j(y - x_e\pi_j) + \\ 2\beta\varphi(\pi_j) + (\pi - \pi_j)(C_j x_e)^{-1}(\pi - \pi_j) \tag{16}$$

where Λ_j is a diagonal matrix with the diagonal element of the jth rule, β is a regularization parameter, φ is a decaying factor, C_j is the covariance matrix and π_j is the local subsystem of the jth rule. The overall cost function can be illustrated as follows:

$$\mathcal{L} = \sum_{j=1}^{R} \mathcal{L}_j \tag{17}$$

The recursive calculation for a different element of the loss function in Eq. (16) can be presented as follows:

$$\pi_j^k = \pi_j^{k-1} - \beta C_j^k \nabla\varphi\pi_j^{k-1} + \Upsilon(k)(y^k - x_e\pi_j^k) \tag{18}$$

$$C_j^k = C_j^{k-1} - \Upsilon^k x_e C_j^{k-1} \tag{19}$$

$$\Upsilon^k = C_j^{k-1} x_e \left(\frac{1}{\Lambda_j} + x_e C_j^{k-1} x_e^T\right)^{-1} \tag{20}$$

with the initial conditions

$$\pi_1(1) = 0 \quad \text{and} \quad C_1(1) = \Omega I. \tag{21}$$

where Υ^k is representing the Kalman gain, $\Omega = 10^5$ is a large positive constant, β is a regularization parameter. The quadratic weight decay function is utilized in LREC, which can be expressed as follows:

$$\varphi(\pi_j^{k-1}) = \frac{1}{2}(\pi_j^{k-1})^2 \tag{22}$$

Its gradient can be presented as follows:

$$\nabla\varphi(\pi_j^{k-1}) = \pi_j^{k-1} \tag{23}$$

Utilization of this function strengthens the generalization capability of LREC by maintaining the dynamic of the weights into small values [42]. To get a clearer view about the work flow in LREC, the pseudocode of LREC is attached here as Algorithms 1, 2 and 3.

5. Experiments in detecting defects

In this work, to analyze the impact of various AFs/MFs in our developed online DNN and ENFS, two real-world multi-class datasets of advanced FET devices and a benchmark binary-class semiconductor dataset from UCI machine learning repository [43] are considered. Specification of these datasets such as number of training and testing samples, number of features, and number of classes is shown in Table 1. Major challenges associated with two real-world datasets are (i) very less number of samples; and (ii) higher number of defect classes. The process of extracting the data and complexities associated with them to classify are discussed in the subsections below. Also, classification of the benchmark dataset is very challenging since it is a high dimensional (590 input features) and highly imbalanced dataset. This dataset contains 1567 observations taken from a wafer fabrication production line, where each observation is a vector of 590 sensor measurements along with one-hot encoded labels of pass and fail. Between these two classes, 1463 examples pass the test with only 104 failed cases.

5.1. Datasets from advanced transistors

Bridging defects in a FinFET and dislocation defects in a GAA-FET device are detected using online DNN and ENFS. These FETs are modeled using TCAD simulation, from where different defect data are collected. These TCAD models are compared with actual device defects and corresponding current–voltage (I–V) curves, thus, they are digital twin of actual FinFET and GAA-FET device. In the TCAD simulation, patterning in the gate and chemical–mechanical polishing are introduced as a source for the bridge defect in FinFET based on the previous research in [44]. The planar scanning transmission electron microscopy (STEM) image of an actual bridge defect, its TCAD model, various location of defects in FinFET is explained in details in [5]. Based on the location of the defects, the whole region of the FinFET model is divided into 10 sub-regions. These sub-regions are considered as different classes in the extracted data for the ML algorithms. Twenty different electrical characteristics of the FinFET are considered as feature set for the learning algorithms. A total of 270 samples with various electrical features of FinFET are extracted from the I–V curves of the TCAD simulation.

Besides FinFET, TCAD model of a three-stack GAA-FET structure with a fixed dimension and a single dislocation defect is also considered. Within the distribution of defects in the channel of the nanosheet stacks, 9 sub-regions of defects are considered. Each sub-region is considered as a class for the learning algorithms. Forty electrical features from the I–V curves of GAA-FET with introduced defects are recorded where the impact of different positions of dislocation defects in GAA-FET are analyzed. The extracted GAA-FET dataset consists of 90 samples, each contains 40 features extracted from I–V curves to identify defects from all 9 sub-regions. These FinFET and GAA-FET datasets are passed to LREC in a streaming fashion as shown in Fig. 3.

Table 1
Specification of real-world and benchmark classification datasets.

Datasets	#training samples	#testing samples	#features	#classes
FinFET[a]	216	54	20	10
GAA-FET[b]	72	18	40	9
SECOM	1096	471	590	2

[a] Bridge defect.
[b] Dislocation defect.

5.2. Class distribution in semiconductor defect datasets

To get a clearer idea about the challenges associated with the data distributions of the three datasets utilized in this paper, the t-SNE plots of their class distribution are pictured in Fig. 4. From the class distribution of the FinFET dataset (with all 270 samples) as shown in Fig. 4(a), some observations are as follows: (1) there exists 10 fault classes, and the number of samples per class is not equal. Out of 270 samples that are distributed among 10 classes, both class 4 and class 9 contains 60 samples, whereas class 5 has only 13 samples; (2) most of the classes are scattered in multiple clusters; (3) samples from different classes are overlapping with each other. For instance, most of the samples from class 6 are overlapping with class 9. A similar phenomenon is also witnessed among some samples of class 1 and class 4. On the other hand, the GAA-FET dataset is comparatively smaller as it consists of only 90 samples with 9 dislocation defect classes. From Fig. 4(b), it is clearly observed that the classes are not overlapping with each other. However, samples from the same class are not clusters in the same location. Other than the samples from class 6, samples of other classes are scattered. In the SECOM dataset (https://archive.ics.uci.edu/ml/datasets/SECOM), though it contains only 2 classes, the minority class inhabits only 14% of the total samples. Besides, these minority samples are not clustered in a certain place. Rather, they are scattered randomly and overlapping with the majority class samples as shown in Fig. 4(c). Such behaviors of the above-mentioned datasets pose challenges to the proposed and benchmark online ML models.

Algorithm 1 LREC

```
Define: Training data $(X_n, T_m)$=$(x_1, ..., x_n, t_1, ..., t_m)$
Predefined thresholds $b_1, b_2$ and $c_1, c_2$
Step 1: Basic structure of LREC
procedure FUZZIFICATION
    for $j = 1$ to $R$ do
        Calculation of the leaky ReLU-based MF using Eq. (1)
    end for
end procedure
procedure RULE BASE
    for $j = 1$ to $R$ do
        Calculate consequent part for a rule using Eq. (3)
    end for
    Express the IF-THEN fuzzy rule utilizing Eq. (2)
end procedure
procedure DEFUZZIFICATION
    Calculate defuzzified crisp output using Eq. (4)
    Compute the final decision of LREC using Eq. (5)
end procedure
Step 2
procedure MECHANISM OF ADDING RULES
    See algorithm 2
end procedure
procedure MECHANISM OF MERGING RULES
    See algorithm 3
end procedure
procedure ADAPTATION OF OUTPUT WEIGHTS
    for $j = 1$ to $R$ do
        Update output weights using FWGRLS as described from Eq. (18)-(20)
    end for
end procedure
```

Algorithm 2 Mechanism of Adding Rules

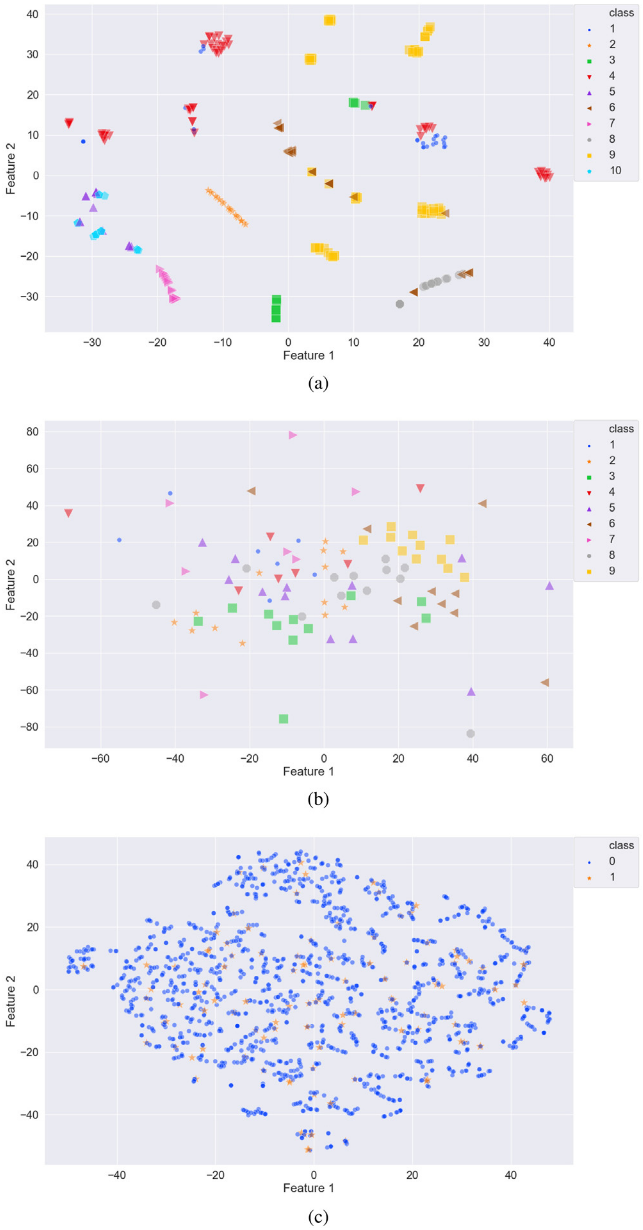
```
procedure MECHANISM OF ADDING RULES
    for $j = 1$ to $R$ do
        Calculate input coherence using Eq. (6)
    end for
    for $i = 1$ to $n$ do
        Compute $\xi(X_i, T_t)$ using Eq. (8)
        for $o = 1$ to $k$ do
            Compute $\xi(\mathcal{H}_j, T_o)$ using Eq. (8)
        end for
        Calculate output coherence using Eq. (7)
    end for
    if Eq. (10) then
        Create a new rule
    else
        Accommodated data points of a rule are updated as $N_{j*} = N_{j*} + 1$
        Take the next sample and Go to Step 1
    end if
end procedure
```

6. Demonstration of online single-pass LREC

An online DNN is implemented using Pytorch, where effect of Sigmoid, tanh, ReLU and leaky ReLU AFs in the DNN to detect defects in semiconductors are analyzed. Here, the DNN is a three hidden layered feed-forward fully connected NN, where each layer consists of 100 nodes. The DNN code has been written using python 3.6 and PyTorch 1.10.0. Among different hyper-parameters of DNN, learning rate and number of epochs have been tuned to obtain better classification accuracy. For instance, the learning rate for FinFET dataset is 1×10^{-4} for all four activation functions. In this dataset, the number of epochs for Sigmoid activation function is 3000, however, in remaining three cases, it is 1500. While detecting defects in GAA-FET, the learning rate of DNN is fixed at 1×10^{-5} and number of epochs is 5000, except for tanh activation function with 1500 epochs. In SECOM dataset, the learning rate and number of epochs are 1×10^{-3} and 10 respectively. Since it is an online DNN, the batch size is 1, the utilized optimizer is Adam. Besides, evolving and online neuro-fuzzy systems with univariate [24] and multivariate Gaussian MF [25] are considered for those defect detection problems. Also, a popular AF in the deep learning domain called leaky ReLU is proposed here as a MF for an ENFS called LREC. We repeated the experiments 5 times for each methods with same

Algorithm 3 Mechanism of Merging Rules

```
procedure MECHANISM OF MERGING RULES
    for $i = 1$ to $R$ do
        for $o = 1$ to $k$ do
            Calculate angle between rules using Eq. (11)
            Calculate minimum distance between rules using Eq. (12)
        end for
        if Eq. (13) then
            Rules are merged
        end if
    end for
end procedure
```



(a)

(b)

(c)

Fig. 4. t-SNE plots of the class distribution in (a) FinFET, (b) GAA-FET, (c) SECOM.

Table 2
Overall classification accuracy in detecting defects in semiconductors (UV: Uni-variate; MV: Multi-variate).

Online ML models	AF/MF	Overall accuracy		
		FinFET[a]	GAA-FET[b]	SECOM
DNN	ReLU	89.26±3.55	76.67±7.24	93.31±0
	Leaky-ReLU	89.26±4.01	74.44±6.33	93.31±0
	Sigmoid	86.67±0.83	84.44±4.64	93.31±0
	tanh	87.41±4.01	77.78±3.93	93.31±0
ENFS	UV Gaussian	88.89±0	72.22±0	90.23±0
	MV Gaussian	90.74±0	83.33±0	**93.42±0**
	Leaky-ReLU	**94.44±0**	**94.44±0**	91.93±0

[a] Bridge defect.
[b] Dislocation defect.

Table 3
Average classification accuracy in detecting defects in semiconductors (UV: Uni-variate; MV: Multi-variate).

Online ML models	AF/MF	Average accuracy		
		FinFET[a]	GAA-FET[b]	SECOM
DNN	ReLU	85.07±4.11	77.78±6.41	50.00±0
	Leaky-ReLU	85.03±4.85	76.30±6.34	50.00±0
	Sigmoid	80.83±0.37	86.30±4.65	50.00±0
	tanh	83.93±4.71	79.26±3.31	50.00±0
ENFS	UV Gaussian	75.00±0	79.63±0	51.29±0
	MV Gaussian	81.67±0	83.33±0	50.00±0
	Leaky-ReLU	**88.33±0**	**94.44±0**	**55.20±0**

[a] Bridge defect.
[b] Dislocation defect.

hyper-parameters setting, and calculate the mean and standard-deviation for the performances. All these ENFSs are developed in MATLAB R2016b environment. The computational platform for all the numerical experiments was an Intel(R) Core(TM) i7-8700 CPU a 3.2 GHz dual processor and 32.0 GB installed memory.

The overall accuracy's (testing) for all these algorithms with three different datasets are shown in Table 2. The overall testing accuracy indicates the ratio between the predicted and true class labels during the testing phase. After obtaining such ratio for individual each class, their average is calculated and tabulated in Table 3. It has been observed from both Table 2 and Table 3 that online DNNs/ENFS with different AFs/MFs are performing well and close to each other in detecting defects in both GAA-FET and FinFET devices. Among various MF in ENFS, it is observed that multi-variate Gaussian MF is performing better than the univariate Gaussian MF since multi-variate Gaussian MF can deal with no-axis parallel data distribution. However, the best results are witnessed consistently from the proposed Leaky-ReLU MF-based ENFS called LREC in most cases of all three semiconductor defect datasets. One clear advantage of the LREC besides having an autonomous architecture is the requirement of less adapting parameters. In addition, LREC architecture is not multi-layered. Such architectural simplicity supports them to perform better with less data. Both in GAA-FET and FinFET dataset, though the highest overall and average accuracy is witnessed from our proposed LREC, still some samples are misclassified. The reason

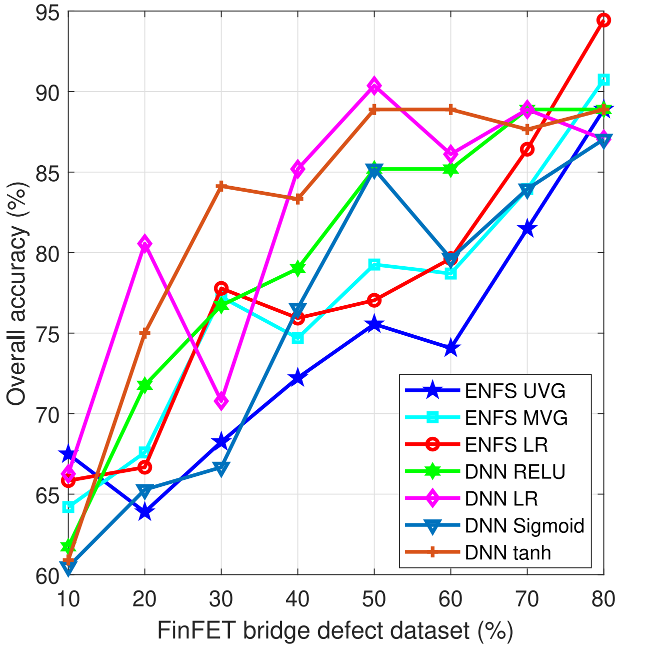


Fig. 5. Learning with less data by showing average overall accuracy for FINFET dataset.

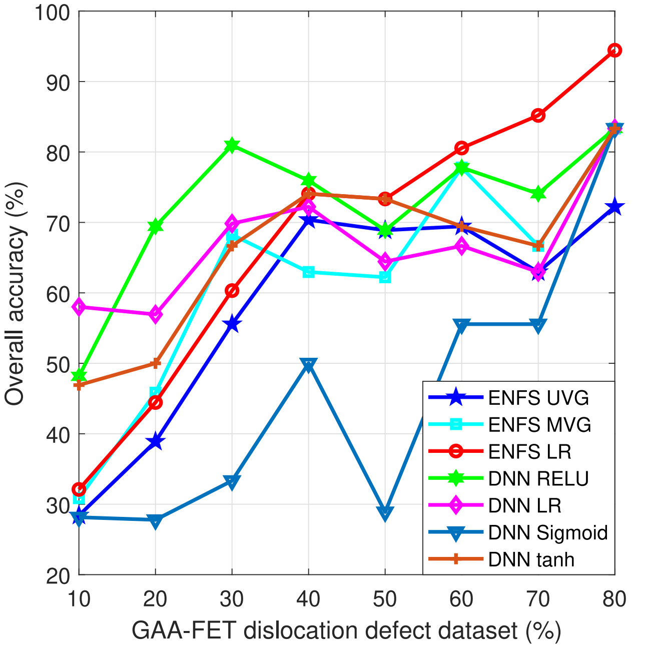


Fig. 6. Learning with less data by showing average overall accuracy for GAA-FET dataset.

behind that can be extracted from Fig. 4(a), and (b). In FinFET dataset, overlaps among samples from class 6 and class 9, and also in class 5 and class 10 cause some misclassification. Since the GAAFET dataset has a very few samples to test the LREC, and samples the scatter randomly as shown in Fig. 4(b), it is also causing some misclassification.

6.1. Learning with less data

In this subsection, the significance of evolving shallow architecture-based neuro-fyzzy systems, especially the leaky-ReLU MF-based classifying defects with less data is analyzed. Here, the number of training samples is varied from 10% to 80% of the whole dataset with an increment of 10% for defects in both the FinFET and GAA-FET devices. Besides, the number of samples per class are not the same, thus, the imbalance factor is calculated in both the training and testing phase using,

$$IBF = 1 - \frac{n}{N} \times \min_{c=1,\dots,n} N_c \qquad (24)$$

where n is the number of classes, N is the total number of samples in training/testing dataset. The value of IBF closer to 1 indicates a higher imbalance in the dataset. In the training and testing phase of FinFET, the range of IBF are [0.51–0.69] and [0.5–0.73] respectively. With the variations in the number of samples in the training and testing phase of GAA-FET, the range of IBF are [0.44–0.71] and [0.35–0.58] respectively. A very lower number of samples per class with varying IBF pose challenges to online ML models in detecting defects in both the FinFET and GAA-FET device datasets.

As pointed out earlier, defects occur very rarely in a FinFET device, which makes the collection of such defect data for ML models challenging. To increase the number of samples of FinFET's bridge defect, more data are collected from FinFET's TCAD model. From 270 samples of the FinFET's bridge defect, a randomly selected varying percentage of total samples are used to train various learning methods as shown in Fig. 5. In this figure, the curves are representing the mean value of the overall accuracy for 10 different runs. Since the FinFET dataset has a comparatively higher number of samples (270), models trained with only 27 randomly selected samples are showing accuracies more than 70% in identifying the location of the bridge defect while ENFS are used. To be specific, while the training samples are varied from 10% to 80%, the ranges of mean overall accuracy for different ENFS varies from 64.2% to 94.44%. Performance among various ENFS and DNNs are very close to each other. Some insights from these results are as follows: (1) multi-variate Gaussian MF-based ENFSs have some advantages over the univariate Gaussian MF-based ENFSs in capturing various data distribution since multi-variate Gaussian MF can accommodate non-axis parallel data distribution. It has been witnessed during the experimentation with FinFET dataset in the learning with less data scenario where better defect detection accuracy is observed in all cases; (2) utilization of Leaky-ReLU MF in LREC provide comparatively better accuracy than both univariate and multi-variate Gaussian MF-based evolving learning methods; (3) From Fig. 5, the lower rate of reduction in overall accuracy with the continuous reduction in the number of training samples indicates the ability of the evolving or online learning algorithms to predict the defects with less data.

A similar analysis for the GAA-FET's dislocation defect dataset is pictured in Fig. 6. In GAA-FET's dislocation defect dataset, we have only 90 samples with 40 different features and 9 classes. Identifying the various location of defects from such a small dataset is not easy. In contrast with online DNN, the proposed method operates from scratch with one rule at the beginning. Afterward, LREC alters its structure in terms of the number of rules based on the incoming streams of data in a sequential manner. In Tables 2 and 3, randomly selected 80% of the 90 samples were used to conduct the classification analysis. To evaluate the effectiveness of shallow evolving learning algorithms in learning from less data, the number of samples in the training stage has been reduced from 80% to 10% of total samples as shown in Fig. 6. When the training samples are decreased from 80% to 60%, the rate of reduction in accuracy is lower. Within the range between 10% and 40%, the number of samples per class has reduced to a very lower value, which causes a reduction in classification accuracy at the testing phase. Among all the ML models, Leaky-RELU-based ENFS called LREC is performing best at most of the testing ratios due to its simple and evolving structure with less parameters.

While detecting a failure in highly imbalanced semiconductor manufacturing SECOM dataset, the overall accuracy in online DNN is better or comparable with their ENFS-based variants. However, from the average accuracy, it has been witnessed that all the DNNs have failed completely to classify the minority class. Among the ENFS, the Leaky-ReLU MF-based one can classify few samples for the minority class in the testing phase and performs better than the remaining DNN and ENFS-based online ML models. Such ability of LREC is proving it as a robust classifier for handling imbalance and multi-class real-world problems with fewer samples.

The reported results in Table 2 are using testing datasets. We repeated experiments 5 times for each DNN and ENFS method, using different weight initialization. We observed that due to randomness in initialization, the DNN models' performance would vary a bit, while the ENFS models would not. This shows the ENFS models ability to quickly converge to the same state. On the other hand, to verify the proposed classifier's robustness in training and testing sets, some results are reported here. In case of GAA-FET dataset, the training and testing accuracy of LREC are 97.22% and 94.44% respectively. In DNN with ReLU, these values are 90.28% and 83.33% respectively, considering one of the best results. The lower difference between the training and testing accuracy is discarding the possibility of over-fitting or under-fitting from these algorithms.

To get an idea about the structural and computational complexity of different online ML models, the number of utilized rules (in ENFS) or nodes (in DNN) for three different datasets are summarized in Table 4. Since the DNN has a fixed architecture with 3 hidden layers and 100 nodes in each layer, the total number of nodes is always 300. Among various ENFSs, we repeated experiments for 5 times with different weight initialization for each method. Since the initial rule is set to 1 in ENFSs, few initial parameters would not affect the parameters of the final rules. Uni-variate Gaussian MF-based one generates more rules than its multi-variate Gaussian and Leaky ReLU MF-based ones. Both the multi-variate Gaussian and Leaky ReLU MF-based ENFSs generate only one rule for all the datasets. However, with the same number of rules, LREC requires less number of learning parameters to be tuned. Thus, it takes less training time than the multi-variate Gaussian MF-based variant. For instance, while detecting bridge defects in FinFET, multi-variate Gaussian MF-based ENFS requires 0.731 s to train, which is only 0.319 s in LREC.

Table 4
Number of rules/nodes in detecting defects in semiconductors (UV: Uni-variate; MV: Multi-variate).

Online ML models	AF/MF	#Rules/nodes		
		FinFET[a]	GAA-FET[b]	SECOM
DNN	ReLU	100	100	100
	Leaky-ReLU	100	100	100
	Sigmoid	100	100	100
	tanh	100	100	100
ENFS	UV Gaussian	12±0	11±0	1±0
	MV Gaussian	1±0	1±0	1±0
	Leaky-ReLU	1±0	1±0	1±0

[a] Bridge defect.
[b] Dislocation defect.

6.2. Ablation study

In this work, we have incorporated the idea of leaky ReLU as an MF in an ENFS. A clear advantage of leaky ReLU MF-based ENFS over the univariate or multivariate Gaussian MF is the lower number of learning parameters. Another recently developed ENFS with less learning parameter is PALM [32], where hyper-plane MF has been utilized. However, a tuning parameter called fuzziness in membership grade in the hyper-plane MF needs to be altered for different datasets. Besides, the MF has a dependency on true class labels in calculating the distance. The limitation of dependency on true labels can be mitigated by introducing a recurrent connection from the network output to input to replace the true class label in the distance calculation of PALM. However, these solutions will also have the same MF.

To overcome these limitations, a leaky ReLU MF is introduced in this work. Unlike the hyperplane MF-based ENFS, in LREC, we do not calculate the distance. Rather than hyper-plane, the distance can be fed to a ReLU-based MF. However, the ReLU MF-based ENFS is not implemented in this work since ReLU MF yields zero values of μ, consequently the defuzzified crisp output will be zero. In this work, we perform an ablation study among three ENFS-based evolving classifiers namely PALM, rPALM, and LREC. In this ablation study, the identification of bridge defect in FinFET is analyzed, where the evolving classifiers are trained with a various number of samples starting from 27 (10%) to 208 (80%) among 270 samples. By considering the mean of overall mean accuracy at different training/testing ratios, more than 2.2% better classification rate is witnessed in LREC than the PALM. The performance of rPALM is very close to that of LREC, where a fixed value of Γ is considered in rPALM at different training scenarios. Some insights from this ablation study are as follows: (1) in learning with less data, the performance of various ENFS-based classifiers with less learning parameters are very close to each other; (2) when the number of training samples increased from 10% to 80% of the complete set, the classification rate of LREC is more than 0.6% and 2% higher than rPALM and PALM respectively.

6.3. Explainability in LREC

The majority of the ML models are considered as a black box and difficult to interpret. However, an ENFS can be expressed as a linguistic IF-THEN rule. For instance, first rule in detecting defect of FinFET can be expressed as follows:

$$R^1 : \text{IF } X \text{ is close to } \Big([1, x_1, x_2, \ldots, x_{20}] \times [4.1827, -0.1291, 2.2296, \ldots, 0.0355]^T \Big), \tag{25}$$

$$\text{THEN } y_1 = 4.1827 - 0.0.1291x_1 + 2.2296x_2 + \cdots + 0.0355x_{20}$$

In Eq. (25), the antecedent part is same as in Eq. (1). The consequent part is simply $y_1 = x_e^1\omega$, where $\omega \in \Re^{(n+1)\times 1}$, n is the number of input dimension. Besides rule-based representation, the importance of various features has been ranked in LREC based on their corresponding weights by following the approach explained in [45]. For instance, in GAA-FET dataset, it has been witnessed from the ranking of features that feature 25 and 38 are impacting negatively to the model's classification accuracy as shown in Fig. 7. However, due to their insignificant impact, after removing those features, the overall accuracy is still 94.44%. On the other hand, among 20 features of the FinFET dataset, the top five important features are 13, 9, 8, 14, and 12 respectively as pictured in Fig. 8. It is also observed that none of the features of FinFET are impacting negatively. Thus, removing any features with even a very small weight is also impacting the LREC adversely and the classification rate is decreasing. Besides the IF-THEN linguistic rule-based expression, LREC's ability to analyze important features is making it a more explainable model.

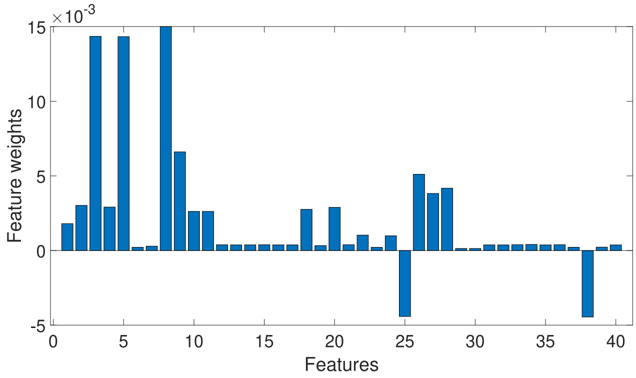


Fig. 7. Feature weights of various features in GAA-FET dataset.

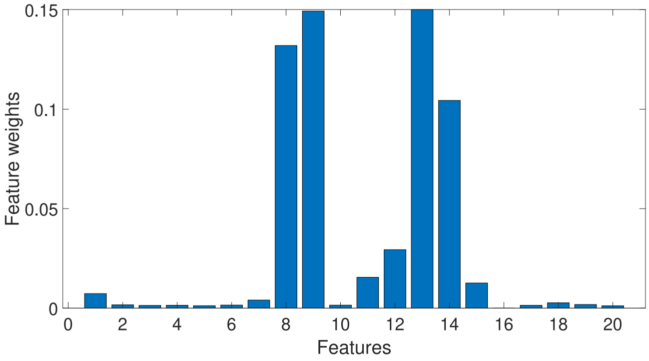


Fig. 8. Feature weights of various features in FinFET dataset.

7. Experiments under prequential test-then-train protocol

Prequential test-then-train protocol is another popular evaluation mechanism for data stream algorithms. Experiments of our proposed LREC under such settings have been discussed in this section.

7.1. Datasets for prequential classifier

The prequential LREC's performance has been evaluated with four popular data stream problems namely electricity-pricing, hyperplane, SEA and weather (https://github.com/ContinualAL/DEVFNN/tree/master/dataset). Electricity demand fluctuation in the state of New South Wales (NSW), Australia is described in the electricity pricing dataset by eight different input features. This dataset shows non-stationary behavior due to the dynamic market condition and economic affairs. It contains 45312 samples, which is partitioned into 91 mini batches. The hyperplane dataset is obtained from an open source software to analyze data streams known as Massive Online Analysis (MOA) [46]. It is a dataset containing four input features, 120 K samples and partitioned into 240 equal mini batches. SEA is a popular binary classification problem that contains abrupt drift component due to three dramatic changes in the class boundary. SEA dataset's modified version of Ditzler and Polikar [47] is utilized here, which exhibits class imbalance and recurring drift behavior varying from 5% to 25% of the minority class. It consists of 100 K artificially generated data samples that is divided into 200 equal batches. The weather dataset used here is a daily weather conditions records from Offutt air force base in Bellevue, Nebraska [47]. It is a binary classification problem to predict the chances of occurring rain on the next day using eight different input features. Weather data, utilized here, contains 18159 samples that is partitioned into 36 small batches. In all datasets, these mini batches are passed to the learning algorithms in a streaming fashion without changing the data order. The reported results for all these datasets are the average value across all mini batches.

Table 5
Performance comparison in prequential test-then-train setting.

Datasets	Algorithms	CR	Precision	Recall	#Rules
Electricity Pricing	HAT[a]	57± 8.4	0.1	0.15	20
	DEN[a]	57± 8.6	0.08	0.16	16
	PNN[a]	59± 3.4	0.35	0.66	78
	DSSCN[a]	68± 13	0.72	0.62	3.58± 1.45
	SDEVFNN[a]	70± 13	0.56	**0.9**	24
	DEVFNN[a]	70± 13	0.75	0.78	4.24± 2.25
	LREC	**71.10**± 10.14	**0.80**	0.74	9.77±6.02
Hyperplane	HAT[a]	76.18± 7.82	0.69	0.96	12
	DEN[a]	91.14± 3.86	0.91	0.92	8
	PNN[a]	85.55± 5.83	0.11	0.10	42
	DSSCN[a]	91.25± 1.79	0.91	0.91	5.47± 3.23
	SDEVFNN[a]	55.52± 14.2	0.53	**0.99**	4
	DEVFNN[a]	91.57± 1.76	**0.92**	0.91	4.73± 1.36
	LREC	**92.14**±1.79	**0.92**	0.92	1.30±0.59
SEAa	HAT[a]	74.6± 10.1	0.79	0.86	10
	DEN[a]	62.9± 7.7	0.63	0.99	6
	PNN[a]	83.2± 6.3	0.85	0.73	33
	DSSCN[a]	91.5± 4.09	0.92	**0.95**	10.29± 4.09
	SDEVFNN[a]	91.1± 6.05	0.9	0.95	9
	DEVFNN[a]	91.7± 5	**0.95**	0.92	4.36± 1.08
	LREC	**92.88**±5.58	**0.95**	0.93	4.72±2.25
Weather	HAT[a]	67.36± 8.27	0.008	0.027	20
	DEN[a]	68.46± 5.16	0	0	10
	PNN[a]	68.7± 1.18	0.3	0.93	78
	DSSCN[a]	80± 2	0.71	0.59	5.08± 2.9
	SDEVFNN[a]	68± 1.23	0.51	**0.97**	24
	DEVFNN[a]	80± 3.75	**0.90**	0.83	4.7± 1.6
	LREC	**80.88**±4.23	**0.90**	0.85	2.58±1.02
SEAg	HAT	–	–	–	–
	DEN	–	–	–	–
	PNN	–	–	–	–
	DSSCN	87.96± 0.86	0.81	**0.88**	11.53± 4.40
	SDEVFNN	–	–	–	–
	DEVFNN	86.67± 0.77	0.81	0.84	5.18± 1.49
	LREC	**88.84**±0.98	**0.83**	**0.88**	1±0.00

[a] Reproduced from [21].

7.2. Evaluation metrics and baselines

The performance metrics used in this paper are as follows: overall classification rate (CR), precision, recall and number of fuzzy rules. Classification performance of our proposed LREC is compared with some prominent and recently developed continual learning algorithms such as HAT [48], DEN [49], PNN [50], DSSCN [20], static DEVFNN (SDEVFNN) [21] and DEVFNN [21]. Among these algorithms, HAT [48], DEN [49], PNN [50] are deep architecture equipped with self constructive hidden nodes. However, they have a static number of layers. Unlike them, DSSCN [20] and DEVFNN [21] have an adaptive network depth. However, due to utilization of multivariate Gaussian MF, they require to adapt the parameters associated with that MF.

7.3. Results and discussion

In this work, the results are listed and analyzed by comparing the proposed model with baselines. In all datasets, small batches are streamed to all the algorithms, where they were tested first to evaluate their generalization performance before being trained. The overall classification performance in terms of the mean and standard deviation of the classification rate (CR), number of fuzzy rules, mean of precision and recall are recorded in Table 5. Highlighted bold values are indicating the best results. Among all the

Table 6
Sensitivity analysis for LREC considering rule growing thresholds.

Parameters		CR	#Rules	Precision	Recall
b_1	b_2				
0.80	0.2	92.85	5.54	0.95	0.93
0.85	0.2	92.86	5.06	0.95	0.93
0.86	0.2	92.82	4.53	0.95	0.93
0.87	0.2	92.83	4.50	0.95	0.93
0.88	0.2	92.83	4.50	0.95	0.93
0.89	0.2	92.83	4.50	0.95	0.93
0.90	0.2	92.81	4.28	0.95	0.93
0.85	0.17	92.86	5.06	0.95	0.93
0.85	0.18	92.86	5.06	0.95	0.93
0.85	0.19	92.86	5.06	0.95	0.93
0.85	0.21	92.89	5.49	0.95	0.94
0.85	0.22	92.86	4.57	0.95	0.93
0.85	0.23	92.88	4.79	0.95	0.93

models, the better performance is witnessed from our proposed LREC in most of the cases.

The summary of the recorded results in Table 5 can be stated as follows: in an electricity-pricing dataset, worse results are obtained from HAT, DEN, and PNN. Though these algorithms can adapt their nodes, due to a static layer, they performed poorly in dealing with non-stationary streams of mini-batches. Though the CR is better in SDEVFNN, due to a static network, its performance deteriorated as witnessed by the low precision value of 0.56. Among six baselines, having fully adaptive nodes and layers, DSSCN and DEVFNN perform better than others. However, the best results are obtained from the proposed model. Besides having a fully evolving architecture, the requirement of less learning parameters in contrast with fully evolving baselines support it to achieve such performance. To be specific, we have used leaky-ReLU-based MF, which has no associated learning parameters, thus, computational complexity is reduced. Having a static structure, SDEVFNN fails to deal with the drift that occurs in the hyperplane dataset, thus, a lower CR and precision is witnessed. Having a partial and fully evolving structure, other baselines' performance is satisfactory in this dataset and the best result is also witnessed from LREC. Similar findings have been observed in other the two datasets. In short, to deal with this non-stationary and imbalanced binary classification problem, our proposed LREC performs better in terms of classification rate and precision with a comparable recall value in a prequential test-then-protocol. Requirement of less learning parameters along with its evolving architecture support it to attain better performance than the state-of-the-art baselines.

7.4. Robustness against parameter sensitivity

To evaluate the robustness of the proposed method against variation in the rule-growing threshold, the SEA dataset with abrupt drift has been considered. During experimentation, it has been observed that for both b_1 and b_2, a lower value adds more rules and vice-versa. To investigate this phenomenon, we analyzed the sensitivity of b_1 and b_2 using the SEA dataset with the same settings as described in Section 7.1. Initially, we varied from 0.80 to 0.90 by keeping the b_2 fixed at 0.2. Afterward, b_2 is varied between 0.17 to 0.23 with an increment of 0.01. In this case, the value of b_1 is maintained at 0.85. As a performance metric to evaluate the sensitivity, we utilized CR, the number of rules, precision and recall, which are recorded in Table 6 for the above-described values of b_1 and b_2. All values are presented as the mean of all the mini-batches. In all cases, the mean precision is fixed at 0.95 and the mean recall range is from 0.93 to 0.94. The mean values for the number of rules are varying from 4.28 to 5.54, whereas the overall classification rates mean values are varying insignificantly between 92.81 to 92.89. Insignificant variations in different metrics against different values of rule-growing thresholds are declaring LREC's robustness parameter sensitivity.

8. Conclusions

Traditionally deployed ML-based defect detectors operate in batch mode demanding a significant amount of labeled data and having a static structure. Such features impede their real-time deployment in the defect analysis of the nano-scaled transistors. In such nano-scaled devices, faults are mostly non-visible, which makes these faults very difficult for human experts to detect. To mitigate all these limitations of defect analysis techniques, a leaky-ReLU MF-based online ENFS called LREC is developed here, and compared with various MF/AF-based online ENFS and DNN to detect locations of the bridge defect in a FinFET and dislocation defect in a GAA-FET device. Some desirable features of LREC in detecting such defects are its online learning capability with evolving structure, simplicity in structure in terms of fewer learning parameters through the utilization of leaky-ReLU MF. These features support LREC to achieve a better classification rate than benchmark online ENFSs and DNNs with less data as observed experimentally in this work. Finally the proposed algorithm has been evaluated by the popular prequential test-then-train mechanism with four benchmark data stream problems. By maintaining the same settings, its performance has been compared with six popular data stream handling baselines. The best classification performance is also witnessed from the proposed LREC due to the above-mentioned benefits.

CRediT authorship contribution statement

Md Meftahul Ferdaus: Methodology, Software, Validation, Formal analysis, Data curation, Writing – original draft, Visualization. **Bangjian Zhou:** Validation, Data curation, Writing – review & editing. **Ji Wei Yoon:** Formal analysis, Visualization, Writing – review & editing. **Kain Lu Low:** Investigation, Writing – review & editing. **Jieming Pan:** Investigation, Writing – review & editing. **Joydeep Ghosh:** Visualization, Writing – review & editing. **Min Wu:** Conceptualization, Resources, Writing – review & editing. **Xiaoli Li:** Conceptualization, Resources, Writing – review & editing, Funding acquisition. **Aaron Voon-Yew Thean:** Resources, Writing – review & editing, Project administration, Funding acquisition. **J. Senthilnath:** Conceptualization, Supervision, Writing – original draft, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This study is supported by the Accelerated Materials Development for Manufacturing Program at A*STAR via the AME Programmatic Fund by the Agency for Science, Technology and Research, Singapore under Grant No. A1898b0043.

References

[1] Y. Wang, Z. Wei, J. Yang, Feature trend extraction and adaptive density peaks search for intelligent fault diagnosis of machines, IEEE Trans. Ind. Inf. 15 (1) (2018) 105–115.

[2] J. Long, S. Zhang, C. Li, Evolving deep echo state networks for intelligent fault diagnosis, IEEE Trans. Ind. Inf. 16 (7) (2019) 4928–4937.

[3] N. Loubet, T. Hook, P. Montanini, C.-W. Yeung, S. Kanakasabapathy, M. Guillom, T. Yamashita, J. Zhang, X. Miao, J. Wang, et al., Stacked nanosheet gate-all-around transistor to enable scaling beyond FinFET, in: 2017 Symposium on VLSI Technology, IEEE, 2017, pp. T230–T231.

[4] J. Pan, K.L. Low, J. Ghosh, S. Jayavelu, M.M. Ferdaus, S.Y. Lim, E. Zamburg, Y. Li, B. Tang, X. Wang, et al., Transfer learning-based artificial intelligence-integrated physical modeling to enable failure analysis for 3 nanometer and smaller silicon-based CMOS transistors, ACS Appl. Nano Mater. 4 (7) (2021) 6903–6915.

[5] C.-W. Teo, K.L. Low, V. Narang, A.V.-Y. Thean, TCAD-Enabled machine learning defect prediction to accelerate advanced semiconductor device failure analysis, in: 2019 International Conference on Simulation of Semiconductor Processes and Devices, SISPAD, IEEE, 2019, pp. 1–4.

[6] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, G. Zhang, Learning under concept drift: A review, IEEE Trans. Knowl. Data Eng. 31 (12) (2018) 2346–2363.

[7] Y. Song, G. Zhang, H. Lu, J. Lu, A self-adaptive fuzzy network for prediction in non-stationary environments, in: 2018 IEEE International Conference on Fuzzy Systems, FUZZ-IEEE, IEEE, 2018, pp. 1–8.

[8] N. Anh, S. Suresh, M. Pratama, N. Srikanth, Interval prediction of wave energy characteristics using meta-cognitive interval type-2 fuzzy inference system, Knowl.-Based Syst. 169 (2019) 28–38.

[9] C. Za'in, M. Pratama, E. Pardede, Evolving large-scale data stream analytics based on scalable PANFIS, Knowl.-Based Syst. 166 (2019) 186–197.

[10] T.T. Ademujimi, M.P. Brundage, V.V. Prabhu, A review of current machine learning techniques used in manufacturing diagnosis, in: IFIP International Conference on Advances in Production Management Systems, Springer, 2017, pp. 407–415.

[11] B. Li, T. Han, F. Kang, Fault diagnosis expert system of semiconductor manufacturing equipment using a Bayesian network, Int. J. Comput. Integr. Manuf. 26 (12) (2013) 1161–1171.

[12] C. Wang, F.R. Lizana, Z. Li, A.V. Peterchev, S.M. Goetz, Submodule short-circuit fault diagnosis based on wavelet transform and support vector machines for modular multilevel converter with series and parallel connectivity, in: IECON 2017-43rd Annual Conference of the IEEE Industrial Electronics Society, IEEE, 2017, pp. 3239–3244.

[13] K.B. Lee, S. Cheon, C.O. Kim, A convolutional neural network for fault classification and diagnosis in semiconductor manufacturing processes, IEEE Trans. Semicond. Manuf. 30 (2) (2017) 135–142.

[14] B. Das, J.V. Reddy, Fuzzy-logic-based fault classification scheme for digital distance protection, IEEE Trans. Power Deliv. 20 (2) (2005) 609–616.

[15] T. Dam, M.M. Ferdaus, S.G. Anavatti, S. Jayavelu, H.A. Abbass, Does adversarial oversampling help us? in: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, 2021, pp. 2970–2973.

[16] B. Su, H. Chen, Z. Zhou, BAF-Detector: An efficient CNN-based detector for photovoltaic cell defect detection, IEEE Trans. Ind. Electron. (2021).

[17] K. Su, J. Liu, H. Xiong, Hierarchical diagnosis of bearing faults using branch convolutional neural network considering noise interference and variable working conditions, Knowl.-Based Syst. 230 (2021) 107386.

[18] Z. Pan, Y. Wang, X. Yuan, C. Yang, W. Gui, A classification-driven neuron-grouped SAE for feature representation and its application to fault classification in chemical processes, Knowl.-Based Syst. 230 (2021) 107350.

[19] M. Pratama, W. Pedrycz, E. Lughofer, Evolving ensemble fuzzy classifier, IEEE Trans. Fuzzy Syst. 26 (5) (2018) 2552–2567.

[20] M. Pratama, D. Wang, Deep stacked stochastic configuration networks for lifelong learning of non-stationary data streams, Inform. Sci. 495 (2019) 150–174.

[21] M. Pratama, W. Pedrycz, G.I. Webb, An incremental construction of deep neuro fuzzy system for continual learning of nonstationary data streams, IEEE Trans. Fuzzy Syst. 28 (7) (2019) 1315–1328.

[22] M. Pratama, J. Lu, S. Anavatti, E. Lughofer, C.-P. Lim, An incremental meta-cognitive-based scaffolding fuzzy neural network, Neurocomputing 171 (2016) 89–105.

[23] P. Angelov, D. Filev, Simpl_ets: A simplified method for learning evolving takagi-sugeno fuzzy models, in: The 14th IEEE International Conference on Fuzzy Systems, 2005. FUZZ'05, IEEE, 2005, pp. 1068–1073.

[24] P.P. Angelov, D.P. Filev, An approach to online identification of Takagi-Sugeno fuzzy models, IEEE Trans. Syst. Man Cybern. B 34 (1) (2004) 484–498.

[25] M. Pratama, S.G. Anavatti, E. Lughofer, GENEFIS: Toward an effective localist network, IEEE Trans. Fuzzy Syst. 22 (3) (2014) 547–562.

[26] C.-H. Tu, C. Li, Multitarget prediction using an aim-object-based asymmetric neuro-fuzzy system: A novel approach, Neurocomputing 389 (2020) 155–169.

[27] M. Pratama, S.G. Anavatti, P.P. Angelov, E. Lughofer, PANFIS: A Novel incremental learning machine, IEEE Trans. Neural Netw. Learn. Syst. 25 (1) (2013) 55–68.

[28] H. He, et al., Applications of Reference Cycle Building and K-Shape Clustering for Anomaly Detection in the Semiconductor Manufacturing Process (Ph.D. thesis), Massachusetts Institute of Technology, 2018.

[29] O. Makhlouk, Time Series Data Analytics: Clustering-Based Anomaly Detection Techniques for Quality Control in Semiconductor Manufacturing (Ph.D. thesis), Massachusetts Institute of Technology, 2018.

[30] T. Chen, et al., Anomaly Detection in Semiconductor Manufacturing Through Time Series Forecasting Using Neural Networks (Ph.D. thesis), Massachusetts Institute of Technology, 2018.

[31] J. Kim, Y. Nam, M.-C. Kang, K. Kim, J. Hong, S. Lee, D.-N. Kim, Adversarial defect detection in semiconductor manufacturing process, IEEE Trans. Semicond. Manuf. 34 (3) (2021) 365–371.

[32] M.M. Ferdaus, M. Pratama, S.G. Anavatti, M.A. Garratt, PALM: AN incremental construction of hyperplanes for data stream regression, IEEE Trans. Fuzzy Syst. 27 (11) (2019) 2115–2129.

[33] E. Lughofer, J.-L. Bouchot, A. Shaker, On-line elimination of local redundancies in evolving fuzzy systems, Evol. Syst. 2 (3) (2011) 165–187.

[34] J. Senthilnath, A. Kumar, A. Jain, K. Harikumar, M. Thapa, S. Suresh, G. Anand, J.A. Benediktsson, BS-McL: Bilevel segmentation framework with metacognitive learning for detection of the power lines in UAV imagery, IEEE Trans. Geosci. Remote Sens. 60 (2022) 1–12.

[35] W. Zou, C. Li, N. Zhang, A TS fuzzy model identification approach based on a modified inter type-2 FRCM algorithm, IEEE Trans. Fuzzy Syst. 26 (3) (2017) 1104–1113.

[36] M. Pratama, S.G. Anavatti, M. Joo, E.D. Lughofer, pClass: an effective classifier for streaming examples, IEEE Trans. Fuzzy Syst. 23 (2) (2014) 369–386.

[37] R.-F. Xu, S.-J. Lee, Dimensionality reduction by feature clustering for regression problems, Inform. Sci. 299 (2015) 42–57.

[38] J.-Y. Jiang, R.-J. Liou, S.-J. Lee, A fuzzy self-constructing feature clustering algorithm for text classification, IEEE Trans. Knowl. Data Eng. 23 (3) (2011) 335–349.

[39] M. Pratama, P.P. Angelov, E. Lughofer, M.J. Er, Parsimonious random vector functional link network for data streams, Inform. Sci. 430 (2018) 519–537.

[40] P. Mitra, C. Murthy, S.K. Pal, Unsupervised feature selection using feature similarity, IEEE Trans. Pattern Anal. Mach. Intell. 24 (3) (2002) 301–312.

[41] C.-H. Kim, M.-S. Kim, Incremental hyperplane-based fuzzy clustering for system modeling, in: 33rd Annual Conference of the IEEE Industrial Electronics Society, 2007. IECON 2007, IEEE, 2007, pp. 614–619.

[42] D.J. MacKay, BayesIan interpolation, Neural Comput. 4 (3) (1992) 415–447.

[43] C. Blake, UCI Repository of Machine Learning Databases, University of California, Department of Information and Computer Science, 1998, http://www.ics.uci.edu/~{}mlearn/mlrepository.html.

[44] T.C. Wei, V. Narang, A. Thean, Electrical characterization of FEOL bridge defects in advanced nanoscale devices using TCAD simulations, in: 2018 IEEE International Symposium on the Physical and Failure Analysis of Integrated Circuits, IPFA, IEEE, 2018, pp. 1–4.

[45] M. Robnik-Šikonja, I. Kononenko, Theoretical and empirical analysis of relieff and rrelieff, Mach. Learn. 53 (1–2) (2003) 23–69.

[46] A. Bifet, G. Holmes, B. Pfahringer, P. Kranen, H. Kremer, T. Jansen, T. Seidl, Moa: Massive online analysis, a framework for stream classification and clustering, in: Proceedings of the First Workshop on Applications of Pattern Analysis, PMLR, 2010, pp. 44–50.

[47] G. Ditzler, R. Polikar, Incremental learning of concept drift from streaming imbalanced data, IEEE Trans. Knowl. Data Eng. 25 (10) (2012) 2283–2301.

[48] J. Serra, D. Suris, M. Miron, A. Karatzoglou, Overcoming catastrophic forgetting with hard attention to the task, in: International Conference on Machine Learning, PMLR, 2018, pp. 4548–4557.

[49] J. Yoon, E. Yang, J. Lee, S.J. Hwang, Lifelong learning with dynamically expandable networks, 2017, arXiv preprint arXiv:1708.01547.

[50] A.A. Rusu, N.C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, R. Hadsell, Progressive neural networks, 2016, arXiv preprint arXiv:1606.04671.