Affine Transformation-Based Lightweight Random Vector Functional Link with Fuzzy Attention for Real-Time Concept Drift Handling

Aanand Upadhayay[1], Amit K. Shukla[2,1], Pranab K. Muhuri[1]
[1] Department of Computer Science and Engineering, South Asian University, New Delhi, India.
[2] School of Technology and Innovations, University of Vaasa, Vaasa, Finland.
aanand@students.sau.ac.in, amit.shukla@uwasa.fi, pranabmuhuri@cs.sau.ac.in

***Abstract*— Online stream data exhibit concept drift that make hard for models to continuously adapt to evolving data distributions. Although many approaches exist but most relay on costly updates, unbounded memory growth, and instability. Random Vector Functional-Link (RVFL) network fits well under this streaming scenario due to their computational efficiency through closed-form output-weight estimation. But their performance degrades in streaming classification because of activation saturation and ineffective drift-handling strategies. To confront these constraints, this paper proposes a lightweight RVFL framework with affine feature calibration to prevents activation saturation and a fuzzy-attention-based forgetting mechanism to adaptively handle drift. Affine calibration adopts a lightweight approach by taking the output of two randomly selected optimized hidden nodes rather than all of them. This incurs minimal overhead and sufficiently preserves feature diversity. Concurrently, the fuzzy attention module also uses a compact rule base with $\{Low, Mid, High\}$ linguistic terms to map drift signals and adaptively modulate forgetting setup. Experimental results on both real-world sensor datasets confirms that the proposed framework attains outstanding classification accuracy with competitive runtime. Overall, it indicates that the proposed framework consistently outperforms state-of-the-art drift-aware learning models in real-time adaptation, robustness, and stability under concept drift.**

Keywords—Drift concept, Drift handling, Sensor & Streaming data, RVFL, Affine Transformation, fuzzy attention, Rule-base.

I. INTRODUCTION

Concept drift data in many real-world applications such as IoT sensors, financial transactions, and social media are generated when their input-output pattern changes continuously as a stream. These data streams are characterized by their real-time nature, continuity and potentially unbounded volume [1]. These characteristics brings unexpected temporal shifts in data distribution over time [2]. For example, models trained on data obtained from sensor readings such as temperature, pressure, or vibration may experience distribution shifts over time. This occurs due to sensor drift, aging, and variations in operating or environmental conditions [3]. These shifts can reduce the predictive accuracy of deployed models and hinder their ability to respond to emerging patterns. As a result, models may miss emerging hazards in real time. This can increase the likelihood of overlooked risks, accidents, and losses [3], [4]. So, effective drift handling remains an impactful area to explore with substantial practical importance. Existing approaches mainly include explicit detection-based methods and implicit ensemble-based strategies.

Both explicit and implicit drift-handling strategies aim to cope with distribution shift, but they differ in how adaptation is triggered. Explicit detection-based methods include ADWIN [5], HDDM [6], and CADM+ [7] to monitor statistics and trigger updates whenever drift is confirmed. These strategies add-ons additional overhead of unpredictable retraining times induced by drift detection. Whereas, implicit ensemble-based approaches like ROALE-DI [8], DES-ICD [9] and ACDWM [10] often utilize ensemble learning to respond drift by updating, replacing, or re-weighting ensemble components without explicitly tracking drift. These implicit strategies are typically less computationally expensive as they avoid explicit drift triggers and instead adapt continuously via incremental updates. But this requires an efficient learning system for continuous, low-latency adaptation under distribution shifts without relying on drift detection, frequent retraining, or computationally expensive optimization.

Randomize Neural Networks (RNNs) presents itself as an effective learning choice for streaming data classification due to their fast training and minimal overhead [11]. Popular RNN architectures include the Extreme Learning Machine (ELM), the Random Vector Functional Link (RVFL) network, and others [12]. In these architectures, hidden-layer weights and biases are randomly initialized and then kept fixed, while the output weights connecting the hidden and output layers are solved analytically using Recursive Least Squares (RLS) [13]. Since, these RNNs offer fast convergence, strong generalization, and computational efficiency [13]. Recent studies have adapted ELM and RVFL networks for streaming drift adaptation [14]. Such as, DOSELM [15] and AOELM [16] incorporate forgetting mechanisms to reduce older sample influence by dynamically adjusting the forgetting factor based on real-time accuracy or distribution shifts. Similarly, lite-RVFL [17] handles drift through weighted objective functions and incremental updates.

However, fixed random feature mappings in these frameworks become misaligned when concept drift introduces linear changes to data distributions over time. These shifts push hidden layer activations into saturation regions which bring gradient vanishing issues [12]. As a result, the quality of random projections during streaming adaptation degrades and limits incremental updates. Additionally, forgetting mechanisms in these streaming frameworks assume uniform drift rates across all scenarios and it may over-emphasize potentially noisy recent samples which lead numerical instability from exponentially growing weight terms [20].

To address the vanishing gradient issue caused by hidden-layer saturation and the limitations of uniform forgetting for drift adaptation, we propose an efficient RVFL-based Lite Affine-Fuzzy Attention framework for stream data. For lightweight affine adaptation, the scale and shift parameters of affine transformation (AT) are first optimized using two randomly selected hidden neurons. Their average is then used as the final estimate instead of applying AT to all hidden neurons. This is sufficient to preserve learning diversity while maintaining lower computational cost. Similarly, drift handling is also achieved through a lightweight fuzzy rule base that adaptively modulates forgetting by assigning greater importance to recent samples via recency-normalized scores perturbed by Gaussian fuzziness. The main contributions of this work are as follows:

1) Proposed a computationally efficient stream-classification framework with selective affine adaptation for hidden-layer saturation and dynamic fuzzy attention forgetting for ineffective uniform forgetting.
2) Introduced a lightweight affine transformation to counter drift-induced saturation with low overhead and preserved diversity.
3) Introduces a lightweight fuzzy-modulated forgetting mechanism with a compact rule base that perturbs recency-normalized scores with Gaussian fuzziness to adaptively emphasize recent samples in stream learning.
4) Extensive experiments on sensor datasets that demonstrate superior accuracy and computational efficiency over competitive baselines.

The remainder of this paper is organized as follows: Section II covers the preliminaries, Section III presents the proposed model, Section IV discusses the experimental results, and Section V concludes the paper with future directions.

II. PRELIMINARIES

This section briefly reviews the key formulations of this work, such as the RVFL network, lightweight affine transformation mechanism for hidden unit feature calibration, and fuzzy-modulated attention for adaptive sample weighting under nonstationary conditions in stream data.

A. Random Vector Functional Link (RVFL)

We consider a streaming classification problem with sequentially arriving labeled samples $\{(x_t, y_t)\}_{t=1}^{\infty}$, where $x_t \in \mathbb{R}^d$ denotes the input feature vector and $y_t \in \{1,2,\dots,m\}$ is the class label. Let $\tilde{t}_t = \text{onehot}(y_t) \in \mathbb{R}^m$ denote the corresponding one-hot target vector. The RVFL hidden layer contains $N_h = N_e \times N_2$ hidden nodes arranged into N_e groups, each containing N_2 neurons. For the i^{th} group, the random weights $W_i = [w_{i,1}, \dots, w_{i,N_2}] \in \mathbb{R}^{d \times N_2}$ and biases $b_i = \left[b_{i,1}, \dots, b_{i,N_2}\right]^{\top} \in \mathbb{R}^{N_2}$ are initialized once and kept fixed thereafter. Given an input x_t, the pre-activation of the j^{th} neuron in the i^{th} group is $v_{t,i,j} = w_{i,j}^{\top} x_t + b_{i,j}$, and the corresponding group-wise pre-activation vector is $v_{t,i} = W_i{}^{\top} x_t + b_i \in \mathbb{R}^{N_2}$. By concatenating all group-wise vectors, the complete hidden pre-activation is obtained as:

$$H_{Net}(x_t) = [v_{t,1}^{\top}, v_{t,2}^{\top}, \dots., v_{t,N_e}^{\top}]^{\top} \in \mathbb{R}^{N_h} \quad (1)$$

Then, applying the activation function $f(\cdot)$ element-wise gives the conventional RVFL hidden representation as:

$$H(x_t) = f(H_{Net}(x_t)) \quad (2)$$

With the direct input-output skip connection, the extended representation is:

$$z_t = [\, x_t^{\top}, H(x_t)^{\top}]^{\top} \in \mathbb{R}^{d+N_h} \quad (3)$$

Then, stacking z_t and $\tilde{t}_t$ row-wise up to time t gives the design matrix as $Z_t = [z_1^{\top}, \dots, z_t^{\top}]^{\top} \in \mathbb{R}^{t \times (d+N_h)}$ and target matrix as $\tilde{T}_t = [\tilde{t}_1^{\top}, \dots, \tilde{t}_t^{\top}]^{\top} \in \mathbb{R}^{t \times m}$. Finally, the RVFL output weights matrix $\beta_t \in \mathbb{R}^{(d+N_h) \times m}$ is estimated via ridge regression:

$$\beta_t = \underset{\beta}{\text{argmin}}\{\lambda \|\beta\|_F^2 + \|Z_t\beta - \tilde{T}_t\|_F^2\} \quad (4)$$

Its closed-form solution is: $\beta_t = (\lambda I + Z_t^{\top} Z_t)^{-1} Z_t^{\top} \tilde{T}_t$, where $\lambda > 0$ ensures invertibility. For prediction at time t, the class-score vector is computed using the previously learned output weights as $\phi(x_t) = z_t^{\top} \beta_{t-1} \in \mathbb{R}^m$, and the predicted label is $\hat{y}_t = \underset{k}{\text{argmax}} \phi_k(x_t)$.

B. Lightweight Affine Transformation (AT)

An affine transformation adjusts its input as $A(v) = \mathfrak{s}v + \mathfrak{t}$, where $\mathfrak{s}$ and $\mathfrak{t}$ are the scale and shift parameters. In RVFL-based streaming learning, such parameters are used to reduce hidden-node activation saturation. However, estimating them separately for all hidden neurons is computationally expensive [12], [21]. Therefore, a lightweight shared affine transformation is adopted. For an initial batch of size N_0, the hidden pre-activation matrix is $V = \left[H_{\text{Net}}(x_1)^{\top}; \dots; H_{\text{Net}}(x_{N_0})^{\top}\right] \in \mathbb{R}^{N_0 \times N_h}$. Two distinct columns $v_1, v_2 \in \mathbb{R}^{N_0}$ are randomly selected from V. A finite calibration interval $R_c = [r_l, r_u]$ is defined in the activation output space and uniformly partitioned into $N_0 - 1$ subintervals to form the target vector u. After sorting and down-sampling, the reduced vectors $v_{1,d}$, $v_{2,d}$, and u_d are obtained with a common length N_d. For $k \in \{1,2\}$, the affine parameters $(\mathfrak{s}_k, \mathfrak{t}_k)$ are obtained by minimizing:

$$\xi_k(\mathfrak{s}_k, \mathfrak{t}_k) = \frac{1}{2}\left\|f\left(\mathfrak{s}_k v_{k,d} + \mathfrak{t}_k\right) - u_d\right\|_2^2,\ k \in \{1,2\} \quad (5)$$

Then, the final shared affine parameters are computed as:

$$(\mathfrak{s}_f, \mathfrak{t}_f) = (\,(\mathfrak{s}_1 + \mathfrak{s}_2)/2\,,\,(\mathfrak{t}_1 + \mathfrak{t}_2)/2) \quad (6)$$

Using $(\mathfrak{s}_f, \mathfrak{t}_f)$, the hidden pre-activation in Eq. (1) is calibrated as $\tilde{H}_{Net}(x_t) = \mathfrak{s}_f H_{Net}(x_t) + \mathfrak{t}_f \in \mathbb{R}^{N_h}$. Accordingly, the conventional hidden representation $H(x_t)$ in Eq. (2) is transformed into the affine-calibrated hidden representation as:

$$H_{AT}(x_t) = f(\tilde{H}_{Net}(x_t)) \quad (2a)$$

Then the extended representation in Eq. (3) becomes $\tilde{z}_t = [x_t^{\top}, H_{\text{AT}}(x_t)^{\top}]^{\top} \in \mathbb{R}^{d+N_h}$ and stacked as $\tilde{\text{Z}} = [\tilde{z}_1^{\top}, \dots, \tilde{z}_t^{\top}]$. This shared affine calibration stabilizes hidden activations and can help alleviate saturation under nonlinear data drift.

C. Lightweight Fuzzy Recency Attention.

Recent samples in nonstationary data stream often carry more informative gain under concept drift. As in [17], the i^{th} sample assigned a recency weight $q_i = \theta^{2(i-1)}, (\theta > 1)$,so that newer observations receive greater emphasis. Using the affine-calibrated extended feature vector $\tilde{z}_t$, and incorporating the recency weights q_i into the ridge objective in Eq. (4), the recency-weighted problem becomes $\min_{\beta}\{\lambda\|\beta\|_F^2 + \sum_{i=1}^{t} q_i \|\tilde{z}_i^{\top}\beta - \tilde{t}_i^{\top}\|_2^2\}$ with closed-form solution as:

$$\beta_t^{\text{RW}} = (\lambda \text{I} + \tilde{\text{Z}}_t^{\top} \text{A}_t^{\top} \text{A}_t \tilde{\text{Z}}_t)^{-1} \tilde{\text{Z}}_t^{\top} \text{A}_t^{\top} \text{A}_t \tilde{\text{T}}_t \quad (7)$$

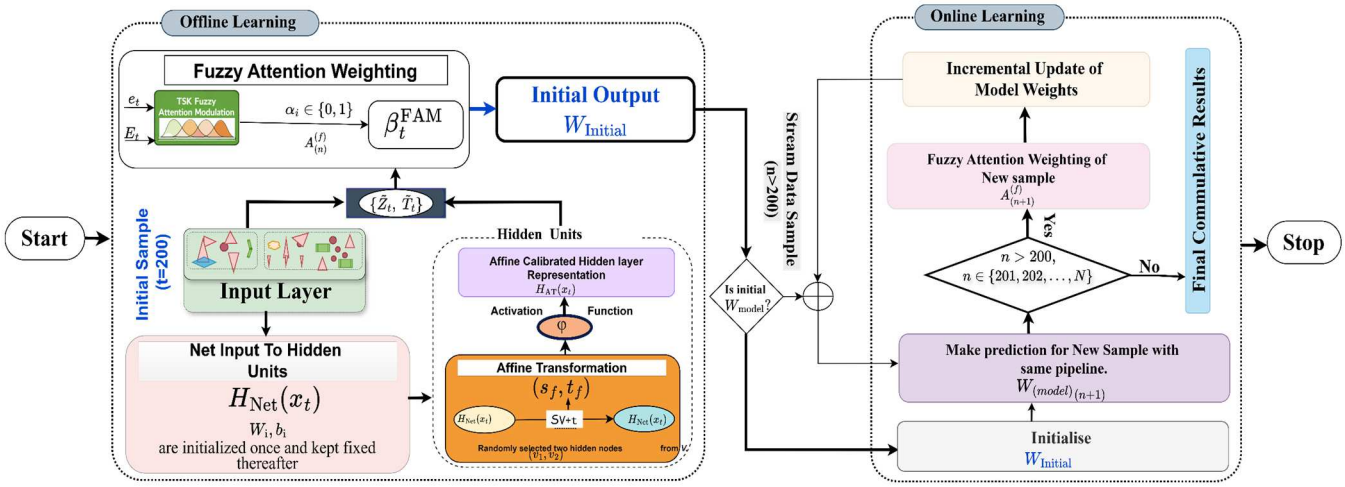


Fig. 1: Working mechanism of proposed Model.

where, $\tilde{Z}_t$ is affine-calibrated design matrix, and $A_t = \text{diag}(1, \theta, \ldots, \theta^{t-1})$ assigns geometrically increasing weights to more recent samples. However, a large θ may over-amplify less informative gains in the evolving data stream. So, to adaptively regulate this amplification, a TSK fuzzy attention modulation is from two lightweight indicators: instantaneous prediction error (e_t) and error persistence (E_t). For e_t, each input x_t is first mapped to the class-score vector $\phi(x_t) = \tilde{z}_t^{\top}\beta_{t-1}$. Then, the SoftMax layer converts $\phi(x_t)$ into the class-probability vector $p_t = softmax(\phi(x_t))$ [22]. Let $[p_t]_{y_t}$ denote the probability assigned to the true class y_t. The instantaneous misprediction severity is defined as:

$$e_t = 1 - [p_t]_{y_t} \tag{8}$$

Then, e_t is fuzzified into low, mid, and high levels using Gaussian membership functions as:

$$\mu_j(e_t) = \exp\left(-\frac{(e_t - c_j^{(e)})^2}{2(\sigma_j^{(e)})^2}\right), \quad j \in \{low, mid, high\} \tag{9}$$

Similarly, E_t tracks the consecutive error trends via an exponential moving average [23] using:

$$E_t = \rho_e E_{t-1} + (1 - \rho_e)e_t, where\ \rho_e \in (0,1), \mathrm{E}_0 = 0. \tag{10}$$

Then, E_t is fuzzified into the Gaussian membership degrees as $\mu_{\text{low}}(E_t), \mu_{\text{mid}}(E_t), \mu_{\text{high}}(E_t)$, using the same form as Eq. (9). Now, a zero-order TSK fuzzy system is used to infer the adaptive attention gain from the crisp antecedents (e_t, E_t). The r^{th} rule is formed as:

$$R_r: IF\ e_t\ is\ A_r\ AND\ E_t\ is\ B_r, THEN\ c_r, \tag{11}$$

where $c_r \in [0,1]$ is the crisp consequent of r^{th} rule. Its firing strength is computed using the product t-norm as: $\omega_r = \mu_{A_r}(e_t)\mu_{B_r}(E_t)$. The final fuzzy attention is obtained by normalized weighted averaging:

$$a_t = \frac{\sum_{r=1}^{R} \omega_r \cdot c_r}{\sum_{r=1}^{R} \omega_r} \in [0,1], \tag{12}$$

Starting from the recency-weighted solution in Eq. (7), the fuzzy attention gains $\{a_i\}_{i=1}^{t}$ modulate the geometric recency weights, giving the effective weight $\alpha_i = a_i q_i = a_i \theta^{2(i-1)}$. Hence, the fuzzy-attention-weighted ridge objective is $\min_{\beta}\{\lambda\|\beta\|_{\mathrm{F}}^2 + \sum_{i=1}^{t} \alpha_i \|\tilde{z}_i^{\top}\beta - \tilde{t}_i^{\top}\|_2^2\}$. Accordingly, the fuzzy attention weighted diagonal matrix is $A_t^{(f)} = \text{diag}(\alpha_1, \alpha_2, \ldots \alpha_t)$, and the resulting ridge solution is:

$$\beta_t^{\text{FAM}} = (\lambda \mathrm{I} + \tilde{\mathrm{Z}}_t^{\top}(A_t^{(f)})^{\top} A_t^{(f)} \tilde{\mathrm{Z}}_t)^{-1} \tilde{\mathrm{Z}}_t^{\top}(A_t^{(f)})^{\top} A_t^{(f)} \tilde{\mathrm{T}}_t \tag{13}$$

This realizes an error-aware and interpretable sample weighting scheme in the spirit of fuzzy weighted least squares and robust adaptive weighted least squares (WLS) methods.

III. Proposed Approach

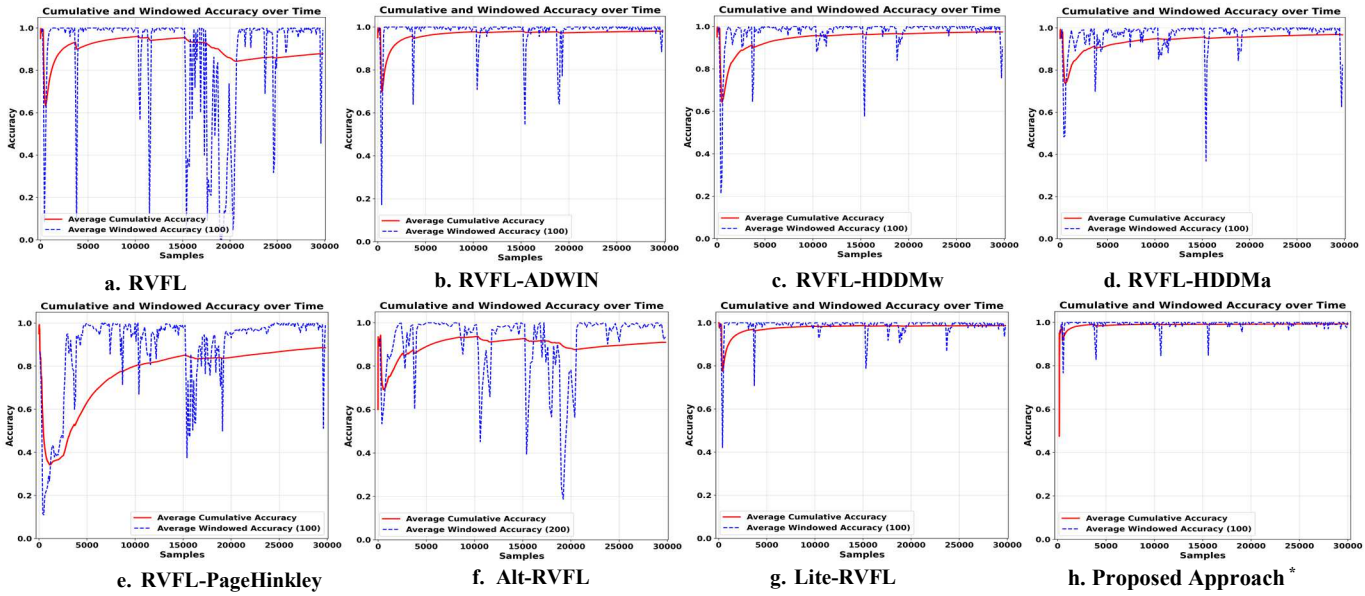
This section outlines the key stages of the proposed approach: offline initialization and online incremental learning. In the offline stage, an initial batch (e.g., $N_0 = 200$) is first processed through affine-transformed hidden representations and then passed to a fuzzy-attention module. In the online stage, these same weights are adaptively updated for incoming samples through the same pipeline. Fig. 1 illustrates the overall framework, and each stage is described below.

A. Stage 1: Offline Learning

In the offline learning stage, initial batch $\{(x_t, \tilde{t}_t)\}_{t=1}^{N_0}$ is prepared, where $\tilde{t}_t \in \mathbb{R}^m$ is the one-hot target vector corresponding to the class label y_t. The hidden weights W_i and biases b_i are randomly initialized once and kept fixed, so that only the output weights are learned during streaming. First, the $H_{Net}(x_t)$ is computed and stacked into $V \in \mathbb{R}^{N_0 \times N_h}$. Next, two distinct calibration interval columns v_1 and v_2 are randomly

Algorithm 1: Optimization of Affine Parameters $(\mathfrak{s}_f, \mathfrak{t}_f)$

Input	$V \in \mathbb{R}^{N_0 \times N_h}$: Initial Batch Pre-Activation, η: Learning rate, K: Max Iteration, ε = Tolerance, R_c: Calibration Interval.
Output	$(\mathfrak{s}_f, \mathfrak{t}_f)$
Step 1	Construct $u \in \mathbb{R}^{N_0}$ by uniformly partitioning R_c into $N_0 - 1$ subintervals.
Step 2	Randomly select two distinct columns $v_1, v_2 \in \mathbb{R}^{N_0}$ from V.
Step 3	Sort v_1 and v_2, then down-sample them together with u using the same sampling rate to obtain $(v_{(1,d)}, u_d)$ and $(v_{(2,d)}, u_d)$.
Step 4	For $k = 1, 2$:
Step 5	- Initialize $(\mathfrak{s}_k, \mathfrak{t}_k)$ randomly. - Set $\xi_k(\mathfrak{s}_k, \mathfrak{t}_k) = \frac{1}{2}\|f(\mathfrak{s}_k v_{k,d} + \mathfrak{t}_k) - u_d\|_2^2$, iter $\leftarrow 0$. - While $\xi_{curr} > \varepsilon$ and $iter < K$: • Compute: $\nabla_{\mathfrak{s}_k}, \nabla_{\mathfrak{t}_k} : \nabla\xi_k(\mathfrak{s}_k, \mathfrak{t}_k)$ • Update: $\mathfrak{s}_k \leftarrow \mathfrak{s}_k - \eta\nabla\mathfrak{s}_k$, $\mathfrak{t}_k \leftarrow \mathfrak{t}_k - \eta\nabla\mathfrak{t}_k$ • Recompute: ξ_{curr}. • Set $iter \leftarrow iter + 1$.
Step 6	Compute: $(\mathfrak{s}_f, \mathfrak{t}_f) = (\mathfrak{s}_1 + \mathfrak{s}_2)/2, (\mathfrak{t}_1 + \mathfrak{t}_2)/2)$ and Return $(\mathfrak{s}_f, \mathfrak{t}_f)$.



a. RVFL **b. RVFL-ADWIN** **c. RVFL-HDDMw** **d. RVFL-HDDMa**

e. RVFL-PageHinkley **f. Alt-RVFL** **g. Lite-RVFL** **h. Proposed Approach** *

Fig. 2. Cumulative accuracy and windowed accuracy for proposed and basslines on DSMS dataset.

selected from V, and the R_c is uniformly partitioned to construct the target vector u. After sorting and down-sampling the selected columns together with u, the pairs $(v_{(1,d)}, u_d)$ and $(v_{(2,d)}, u_d)$ are obtained. For $k \in \{1,2\}$, the affine parameters $(\mathfrak{s}_k, \mathfrak{t}_k)$ are estimated by minimizing the affine calibration loss in Eq. (5) using learning rate η until the error falls below ε or max iterations K is reached. The final shared $(\mathfrak{s}_f, \mathfrak{t}_f)$ is computed using Eq. (6), and the complete procedure for the same is detailed in Algorithm 1.

Using these parameters, the affine-calibrated hidden representation $H_{AT}(x_t)$ is obtained as in Eq. (2a). and the corresponding affine-calibrated extended feature vector $\tilde{\mathrm{z}}$ is formed accordingly. Since recent samples are more informative under concept drift, a large θ in q_i can amplify noise and outliers. So, a zero-order TSK fuzzy system modulates effective recency $a_i \in [0,1]$ from e_t and E_t. To obtain the fuzzy attention values on the initial batch, we first compute a provisional affine-calibrated recency-weighted model using Eq. (7). The provisional predictions are then used to evaluate e_t, update E_t, and infer a_t for $t = 1, \ldots, N_0$. Finally, initial fuzzy-attention-modulated output weights $\beta_{N_0}^{\mathrm{FAM}}$ are computed using Eq. (13).

Thus, the offline stage yields the fixed hidden parameters $\{W_i, b_i\}$, the shared affine parameters (s_f, t_f), the initialized fuzzy state, and the initial output weights $\beta_{N_0}^{\mathrm{FAM}}$ (Eq. (13)) for subsequent online adaptation.

B. Stage 2: Online Learning

In this phase, at first it initializes offline-trained parameters and states such as the fixed hidden weights W_i, and biases b_i, optimized affine parameters $(\mathfrak{s}_f, \mathfrak{t}_f)$, initial recency weights q_i, fuzzy attention states a_t and output weights β_t^{FAM}. When a new labeled sample (x_{n+1}, y_{n+1}) arrives at time $t = n+1$, we first compute its affine-transformed hidden representation $H_{\mathrm{AT}}(x_{n+1})$ and form the affine-calibrated extended RVFL feature vector $\tilde{z}_{n+1} = [x_{n+1}^{\top}, H_{\mathrm{AT}}(x_{n+1})^{\top}]^{\top}$. The design and target matrices are then expanded by appending the new row, i.e., $\tilde{\mathrm{Z}}_{n+1} = [\tilde{\mathrm{Z}}_n; \tilde{z}_{n+1}^{\top}]$ and $\tilde{T}_{n+1} = [\tilde{T}_n; \tilde{t}_{n+1}^{\top}]$, where $\tilde{t}_{n+1}$ is the one-hot encoding of y_{n+1}.

Before updating the model, prediction is performed using the previous weights β_n^{FAM} by computing $\phi(x_{n+1}) = \tilde{z}_{n+1}^{\top}\beta_n^{\mathrm{FAM}}$ and $p_{n+1} = \mathrm{softmax}(\phi(x_{n+1}))$, from which the instantaneous misprediction severity is obtained as $e_{n+1} = 1 - [p_{n+1}]_{y_{n+1}}$ and the persistence indicator is updated via $E_{n+1} = \rho_e E_n + (1-\rho_e)e_{n+1}$. These indicators yield the fuzzy attention gain $a_{n+1} \in [0,1]$through a lightweight zero-order TSK system. Following the geometric recency scheme, the base recency weight for the newest sample is $q_{n+1} = \theta^{2n}$ $(\theta > 1)$, and the effective weight becomes $\alpha_{n+1} = q_{n+1}a_{n+1} = a_{n+1}\theta^{2n}$, which assigns the highest priority to the latest sample while allowing fuzzy attention to suppress isolated noisy updates.

Accordingly, the diagonal weighting matrix is expanded to $A_{n+1}^{(f)} = \mathrm{diag}(\alpha_1, \alpha_2, \ldots \alpha_{n+1})$. To update the ridge solution efficiently, we maintain the sufficient statistics $R_n = (\lambda I + \tilde{\mathrm{Z}}_n^{\top}(A_n^{(f)})^{\top}A_n^{(f)}\tilde{\mathrm{Z}}_n)^{-1}$ and $Q_n = \tilde{\mathrm{Z}}_n^{\top}(A_n^{(f)})^{\top}A_n^{(f)}\tilde{T}_n$, so that $\beta_n^{\mathrm{FAM}} = R_nQ_n$. At $t = n+1$, the sufficient statistic Q_n is updated incrementally as $Q_{n+1} = Q_n + \alpha_{n+1}\tilde{z}_{n+1}\tilde{t}_{n+1}^{\top}$. Defining $u_{n+1} = \sqrt{\alpha_{n+1}}\tilde{z}_{n+1}$, the Woodbury identity gives rank-one update as:

$$R_{n+1} = R_n - \frac{R_n u_{n+1} u_{n+1}^{\top} R_n}{1 + u_{n+1}^{\top} R_n u_{n+1}} \tag{14}$$

after which the updated output weights are obtained as $\beta_{n+1}^{\mathrm{FAM}} = R_{n+1}Q_{n+1}$. Finally, the same lightweight procedure is then repeated for each subsequent sample arrival at $t = n+2, n+3, \ldots$, ensuring efficient continual learning in streaming environments while adaptively regulating the effective contribution of recent samples through fuzzy attention.

IV. Experiments and Results Analysis

In this section, we present the experimental setup, dataset used to evaluate the performance of the proposed approach in streaming data under concept drift. The goal is to assess its

Table I. Experimental settings and hyperparameter configuration

Aspect	Details
Evaluation Metrics	Accuracy (primary), Runtime (secondary).
Comparative Methods	RVFL [18], RVFL-ADWIN [5], RVFL-HDDMw [6], RVFL-HDDMa [6], RVFL-PageHinkley [19], Alt-RVFL [17], Lite- RVFL [17].
Hyperparameter Configurations	For DSMS: $N_h = N_e \times N_2 : [5{:}\,5{:}\,20]_{NodeGroup} \times [5{:}\,5{:}\,20]_{NodesPerGroup}$ For GSAD: $N_h = N_e \times N_2 : 8_{NodeGroup} \times 8_{NodesPerGroup}$ Activation Function: sigmoid, relu, tanh, sinh, linear. $\lambda \in [10^{-4}\ 10^{-1}]$, $\theta \in \{1.001, 1.003, 1.005, \ldots, 1.030\}, (\theta > 1)$.
No of Runs:	20 Runs.

adaptability, accuracy, and efficiency when applied to real-time data streams.

A. Dataset and its description

The experiments were conducted on two benchmark datasets with distribution drift: the Deep-sea Manned Submersible (DSMS) dataset [24] and the Gas Sensor Array Drift (GSAD) dataset [25]. The DSMS dataset contains 30,000 time-ordered samples with 24 features and three safety-state labels, whereas the GSAD dataset contains 13,910 samples represented by 128-dimensional features and organized into 10 time-ordered batches exhibiting long-term drift.

B. Experimental Setup

The experiments use the first 200 samples from both the DSMS and GSAD datasets for offline training, with the remaining samples used for streaming evaluation. Model configurations are selected via grid search over the number of node groups, nodes per group, activation functions, and regularization parameters. Additional settings and hyperparameter combinations are listed in Table I.

C. Results and its Analysis

In this subsection, we present the effectiveness of the proposed approach through its average accuracy and runtime (mean ± standard deviation). The results were obtained using the experimental setting and configuration given in Table I. As shown in Table II, our proposed approach achieved the highest average accuracy of 99.47% ± 0.03% outperforming Lite-RVFL (98.48% ± 0.05%: rank 2) and other baselines on DSMS datasets over 20 runs. This superior performance stems from

Table II. Accuracy and runtime of proposed and baseline model on DSMS datasets.

Methods	RVFL	RVFL-ADWIN	RVFL-HDDMw	RVFL-HDDMa	RVFL-PageHinkley	Alt-RVFL	Lite-RVFL	Proposed Approach
Accuracy (mean ± std)	87.66% ±0.17%	97.32% ±0.23%	96.99% ±0.28%	97.14% ±0.36%	88.52% ±0.53%	91.64% ±0.17%	98.48% ±0.05%	**99.47% ±0.03%**
Rank	8	3	5	4	7	6	2	**1**
Time (s) (mean ± std)	**12.09s ±0.24s**	16.16s ±0.05s	15.26s ±0.09s	14.82s ±0.18s	15.17s ±0.13s	12.68s ±0.11s	12.51s ±0.34s	13.85s ±0.13s
Rank	**1**	8	7	5	6	3	2	4
Drifts Detected	**0**	22	5	5	2	--	--	--

Optimal configuration: N_e=10, N_2=10, λ=0.1, θ=1.001, Activation Function: Sigmoid

Table III. Accuracy and runtime of proposed and baseline model on GASD datasets.

Methods	RVFL	RVFL-ADWIN	RVFL-HDDMw	RVFL-HDDMa	RVFL-PageHinkley	Alt-RVFL	Lite-RVFL	Proposed Approach
Accuracy (mean ± std)	85.96% ±0.280%	96.45% ±0.034%	94.39% ± 0.54%	94.38% ±0.35%	89.96% ±1.07%	89.46% ±1.16%	97.24% ±0.18%	**98.73% ± 0.10%**
Rank	8	3	4	5	6	7	2	**1**
Time (s) (mean ± std)	**08.22s ± 0.06s**	11.04s ± 0.09s	10.24s ± 0.13s	10.84s ± 0.16s	10.15s ± 0.17s	9.61s ± 0.18s	9.72s ± 0.22s	9.17s ±0.14s
Rank	**1**	8	6	7	5	3	4	2
Drifts Detected	**0**	22	5	5	2	--	--	--

Optimal configuration: N_e=8, N_2=8, λ=0.1, θ=1.003, Activation Function: Sigmoid.

the affine transformation that aligns shifted data distributions under concept drift and fuzzy attention that intelligently modulates recency weights based on recent data context.

Both mechanisms contribute to overall performance improvement but add minimal overhead with runtime dropping slightly to 13.85s (rank 4). This trade-off remains justified for sensor safety datasets as the 1% accuracy gain over Lite-RVFL significantly enhances reliability in concept drift scenarios while maintaining real-time feasibility. Similarly, in Table III, for the GASD datasets with 10 batches, the proposed approach secured rank 1 achieving 98.73% ± 0.10% mean accuracy when averaging across all batches from 20 independent runs outperforming Lite-RVFL (rank 2) and baselines. It also achieved competitive rank 2 runtime of 9.17s ± 0.14s, since the batched structure GASD required fewer affine calibrations. Fig. 2 shows cumulative and windowed accuracy for proposed and baseline model of the DSMS datasets. The cumulative accuracy shows long-term learning stability and the windowed accuracy (window size = 100) highlights short-term adaptability to concept drift. We can see that Drift-detector-based variants

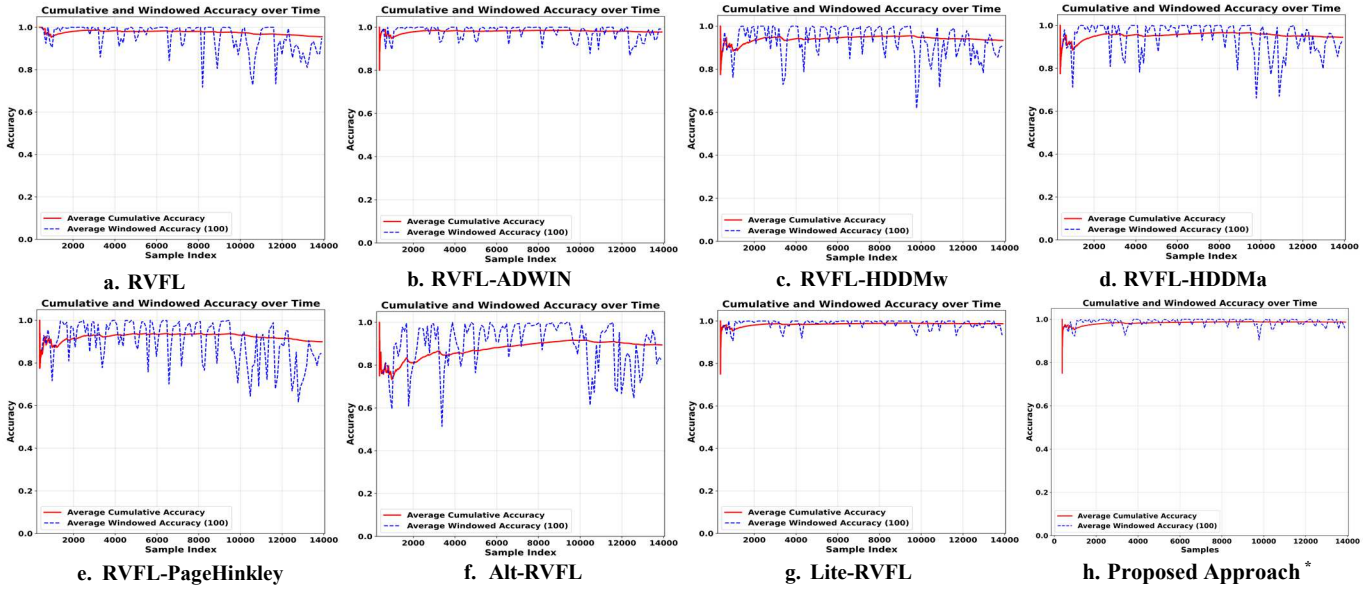


Fig. 3. Cumulative accuracy and windowed accuracy for proposed and basslines on DSMS dataset.

(ADWIN (Fig: 2.b), HDDM (Fig: 2.c & d), Page-Hinkley (Fig: 2.e)) show frequent strident drops in windowed accuracy due to delayed or over-sensitive drift responses resulting in poor cumulative performance. Alt-RVFL supports incremental updates but shows non-smooth cumulative accuracy (Fig. 2.f) due to polynomial weighting's historical inertia causing sharp windowed-accuracy drops and slow drift recovery. Lite-RVFL (Fig. 2.g) improves this with effective recent-sample influence ensuring smoother performance but still shows minor cumulative smoothness issues during drift due to data distribution shifts from outliers or noise. The proposed approach (Fig. 2.h) addresses this through affine transformation for stable hidden representations and fuzzy attention for data-adaptive recency weighting which results in smooth cumulative and windowed accuracy. Similarly, in GASD datasets drift-detector-based variants (ADWIN Fig. 3.b, HDDM Fig. 3.c-d, Page-Hinkley Fig. 3.e) show windowed accuracy drops due to delayed or over-sensitive drift responses resulting in poor cumulative performance. Alt-RVFL (Fig. 3.f) lacks smoothness while Lite-RVFL (Fig. 3.g) achieves smooth curves but final cumulative accuracy drops to 97.24%. The proposed approach (Fig. 3.h) improves this to 98.73% with both cumulative and windowed accuracy plots showing excellent smoothness.

V. Conclusion

This paper proposes a lightweight Affine and Fuzzy Attention enhanced Random Vector Functional Link model for adaptive learning under concept drift in streaming data. The proposed framework integrates two key modules: a lightweight affine transformation layer that calibrates hidden representations to mitigate nonlinear distortions in the activation space, and a zero-order *TSK* fuzzy attention mechanism that dynamically adjusts recency weights to emphasize influential samples while suppressing transient noise during drift periods. Through these module, closed-form output weights are efficiently updated without imposing additional computational overhead. The benchmark evaluations on DSMS and GSAD datasets confirm that the proposed approach consistently achieves higher cumulative accuracy, smoother adaptation, and stronger robustness compared to Lite-RVFL and drift detector–based variants. The results highlight the model's scalability, interpretability, and adaptability in non-stationary environments. Future work will explore lightweight ensemble extensions and more adaptive and uncertainty-aware mechanisms to further enhance stability across varying noise intensities and distribution shifts.

References

1. Hoi, Steven CH, Doyen Sahoo, Jing Lu, and Peilin Zhao. "Online learning: A comprehensive survey." *Neurocomputing* 459 (2021): 249-289.
2. Shi, Qiushi, Rakesh Katuwal, Ponnuthurai N. Suganthan, and Muhammad Tanveer. "Random vector functional link neural network-based ensemble deep learning." *Pattern Recognition* 117 (2021): 107978.
3. Liu, Zeyi, Yi Zhang, Zhongjun Ding, and Xiao He. "An online active broad learning approach for real-time safety assessment of dynamic systems in nonstationary environments." *IEEE Tran. on Neural Networks and Learning Systems* 34, no. 10 (2022): 6714-6724.
4. Liu, Zeyi, Songqiao Hu, and Xiao He. "Real-time safety assessment of dynamic systems in non-stationary environments: A review of methods and techniques." In *2023 CAA Symposium on Fault Detection, Supervision and Safety for Technical Processes*, pp. 1-6. IEEE, 2023.
5. Bifet, Albert, and Ricard Gavalda. "Learning from time-changing data with adaptive windowing." In *Proceedings of the 2007 SIAM international conference on data mining*, pp. 443-448. Society for Industrial and Applied Mathematics, 2007.
6. Frias-Blanco, Isvani, José del Campo-Ávila, Gonzalo Ramos-Jimenez, Rafael Morales-Bueno, Agustin Ortiz-Diaz, and Yailé Caballero-Mota. "Online and non-parametric drift detection methods based on Hoeffding's bounds." *IEEE Tran. on Knowledge and Data Engineering* 27, no. 3 (2014): 810-823.
7. Hu, Songqiao, Zeyi Liu, Minyue Li, and Xiao He. "CADM+: Confusion-based learning framework with drift detection and adaptation for real-time safety assessment." *IEEE Tran. on Neural Networks and Learning Systems* 36, no. 3 (2024): 5126-5139.
8. Zhang, Hang, Weike Liu, and Qingbao Liu. "Reinforcement online active learning ensemble for drifting imbalanced data streams." *IEEE Tran. on Knowledge and Data Engineering* 34, no. 8 (2020): 3971-3983.
9. Jiao, Botao, Yinan Guo, Dunwei Gong, and Qiuju Chen. "Dynamic ensemble selection for imbalanced data streams with concept drift." *IEEE tran. on neural networks and learning systems* 35, no. 1 (2022): 1278-1291.
10. Lu, Yang, Yiu-Ming Cheung, and Yuan Yan Tang. "Adaptive chunk-based dynamic weighted majority for imbalanced data streams with concept drift." *IEEE Tran. on Neural Networks and Learning Systems* 31, no. 8 (2019): 2764-2778.
11. Wang, Junda, Minghui Hu, Ning Li, Abdulaziz Al-Ali, and Ponnuthurai Nagaratnam Suganthan. "Incremental online learning of randomized neural network with forward regularization." *IEEE Tran. on Pattern Analysis and Machine Intelligence* (2026).
12. Srivastav, Shubham, Sandeep Kumar, and Pranab K. Muhuri. "Enhancing robustness and time efficiency of random vector functional link with optimized affine parameters in activation functions and orthogonalization." *Applied Soft Computing* 167 (2024): 112184.
13. Vásquez-Coronel, José A., Marco Mora, and Karina Vilches. "A Review of multilayer extreme learning machine neural networks." *AI Review* 56, no. 11 (2023): 13691-13742.
14. Scardapane, Simone, and Dianhui Wang. "Randomness in neural networks: an overview." *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 7, no. 2 (2017): e1200.
15. Cao, Weipeng, Zhong Ming, Zhiwu Xu, Jiyong Zhang, and Qiang Wang. "Online sequential extreme learning machine with dynamic forgetting factor." *IEEE access* 7 (2019): 179746-179757.
16. Yu, Hualong, and Geoffrey I. Webb. "Adaptive online extreme learning machine by regulating forgetting factor by concept drift map." *Neurocomputing* 343 (2019): 141-153.
17. Hu, Songqiao, Zeyi Liu, and Xiao He. "Lite-RVFL: A Lightweight Random Vector Functional-Link Neural Network for Learning Under Concept Drift." *arXiv preprint arXiv:2506.08063* (2025).
18. Pao, Yoh-Han, Gwang-Hoon Park, and Dejan J. Sobajic. "Learning and generalization characteristics of the random vector functional-link net." *Neurocomputing* 6, no. 2 (1994): 163-180.
19. Page, Ewan S. "Continuous inspection schemes." *Biometrika* 41, no. 1/2 (1954): 100-115.
20. Aguiar, Gabriel Jonas, and Alberto Cano. "Class Informed Concept Drift Detection in Multi-Class Data Streams." *International Journal of Data Science and Analytics* 21, no. 1 (2026): 42.
21. Upadhayay, Aanand, Shubham Srivastav, Amit K. Shukla, and Pranab K. Muhuri. "An Enhanced Neuro-Fuzzy Random Vector Functional Link with Affine Transformations." In *2025 IEEE International Conference on Fuzzy Systems)*, pp. 1-6. IEEE, 2025.
22. Lokshin, Anatole, and Vladik Kreinovich. "Why Softmax? Because It Is the Only Consistent Approach to Probability-Based Classification." (2023).
23. Grebenkov, Denis S., and Jeremy Serror. "Following a trend with an exponential moving average: Analytical results for a Gaussian model." *Physica A: Statistical Mechanics and its Applications* 394 (2014): 288-303.
24. https://github.com/THUFDD/JiaolongDSMS_datasets.
25. https://archive.ics.uci.edu/dataset/224/gas+sensor+array+drift+dataset