

Article

Online Adaptive TSK Fuzzy Classifier with Incremental–Decremental Update Scheme for Non-Stationary Data Streams

Mingqing Jiao, Changbin Shao , Hualong Yu , Chunlong Hu  and Bin Qin *

School of Computer, Jiangsu University of Science and Technology, Zhenjiang 212003, China

* Correspondence: jsjxy_qb@just.edu.cn

Abstract

Takagi–Sugeno–Kang (TSK) fuzzy classifiers have achieved notable success in pattern recognition and data mining due to their combination of approximation capability and model interpretability. However, existing TSK classifiers predominantly rely on batch training, which becomes impractical in streaming environments characterized by continuous sample arrival, limited memory, and evolving data distributions. To address these challenges, this paper proposes the Online Adaptive TSK (OA-TSK) fuzzy classifier with an incremental–decremental update protocol for non-stationary data streams. First, we develop the Decremental Adaptive TSK Fuzzy Classifier (DA-TSK), whose update procedure applies the Sherman–Morrison formula to a symmetric rank-one downdate, enabling precise sample removal while maintaining exact equivalence to batch least-squares retraining. Building upon DA-TSK and our prior Dynamic Incremental Adaptive TSK Fuzzy Classifier (DIA-TSK), we then propose OA-TSK, which integrates incremental and decremental updates within a sliding-window mechanism to achieve continuous adaptation with bounded memory. The key theoretical contribution is that, in exact arithmetic, OA-TSK produces at every time step the same ridge-regularized least-squares solution as batch retraining on the active window. To further handle severe shifts in the input distribution, which can cause fixed rule centers to become poorly aligned with newly arriving samples, we propose Online Adaptive TSK with Antecedent Reclustering (OA-A-TSK), which incorporates an adaptive antecedent reclustering mechanism. Comprehensive experiments on synthetic and real-world data streams demonstrate that OA-TSK achieves classification accuracy comparable to that of ridge-regularized batch retraining while substantially reducing per-sample computational complexity. An analysis of average ranks shows that OA-TSK and OA-A-TSK rank among the top-performing methods across diverse drift scenarios while preserving the intrinsic interpretability of TSK fuzzy systems.

Keywords: Takagi–Sugeno–Kang (TSK) fuzzy classifier; online learning; concept drift; incremental learning; decremental learning; Sherman–Morrison formula; data streams



Academic Editors: Hsien-Chung Wu, Ziye Zhang and Rongrong Yu

Received: 22 September 2026

Revised: 15 October 2026

Accepted: 18 October 2026

Published: 21 October 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

Takagi–Sugeno–Kang (TSK) fuzzy systems have been extensively applied to various domains including system control, pattern recognition, and data mining [1–3], owing to their compelling combination of universal approximation capability and inherent interpretability [4,5]. Among different types of TSK fuzzy systems, zero-order TSK fuzzy classifiers, in which the consequent part of each fuzzy rule is a constant, have attracted

considerable attention due to their simple weighted output aggregation and clear semantic meaning [6–9]. However, while TSK fuzzy classifiers have demonstrated their distinctive value in real-world applications where model comprehensibility is as important as classification accuracy [10], they still face several critical challenges when processing non-stationary data streams.

Traditional TSK fuzzy classifiers are predominantly trained in batch mode, which requires the complete training dataset to be available during model construction. This paradigm becomes problematic in modern applications where data arrives continuously as streams, storage capacity is limited, computational resources are constrained, and the underlying data distribution may evolve over time due to concept drift [11–15]. In such scenarios, retraining the classifier from scratch whenever new data arrives is computationally prohibitive, while simply ignoring new information leads to performance degradation as the model becomes increasingly outdated.

Representative evolving TSK learners include Angelov and Filev’s evolving Takagi–Sugeno (eTS) model, which updates its rule base online [16]; Angelov and Zhou’s evolving classifier (eClass), a single-pass fuzzy-rule-based classifier [17]; and the eClass and FLEXFIS-Class architectures studied by Angelov, Lughofer, and Zhou [18]. Ensemble-based evolving fuzzy systems that reweight base learners under drift include Pratama, Pedrycz, and Lughofer’s pENsemble [19] and the dynamic similarity weighted evolving fuzzy system (DSW-EFS) of Li and Zhao [20]. Building on this line, Chen, Sha, and collaborators proposed the Dynamic Incremental Adaptive TSK Fuzzy Classifier (DIA-TSK) [21]. Its online variant, O-DIA-TSK, uses Sherman–Morrison rank-one updates to incorporate individual samples, whereas its batch variant, B-DIA-TSK, uses the Woodbury matrix identity for batch updates. Yet DIA-TSK and related incremental methods still cannot remove outdated or irrelevant samples, leading to two core issues: (1) the influence of historical samples accumulates indefinitely; (2) in non-stationary data streams, retaining data from $p(\mathbf{x} | t_0)$ can skew the boundary for $p(\mathbf{x} | t)$ and degrade current accuracy.

Concept-drift handling in machine learning typically relies on window-based methods such as Adaptive Windowing (ADWIN) [15], ensemble member replacement as in the Streaming Ensemble Algorithm of Street and Kim [22] and Dynamic Weighted Majority (DWM) [14], or explicit drift detection and forgetting in evolving fuzzy systems [23,24]. These strategies offer practical performance but remain heuristic approximations without guarantees of maintaining global optimality after updates. Moreover, most prior work focuses on adding samples or detecting distribution changes, without providing a principled and reversible sample removal mechanism for TSK models, even though reversible update methods have been developed in other domains, such as the incremental and decremental support vector machine (SVM) learning method of Cauwenberghs and Poggio [25].

In addition, existing studies on online and incremental learning have primarily focused on addressing concept drift, without explicitly accounting for the fact that concept drift may also lead to changes in the relative numbers of samples from different classes. Such changes may introduce or exacerbate class imbalance, which can significantly affect classifier performance.

In order to fundamentally address these challenges, we propose the following novel contributions in this study:

1. We first develop the Decremental Adaptive TSK Fuzzy Classifier (DA-TSK) for precise sample removal from TSK fuzzy classifiers. By applying the Sherman–Morrison formula to a symmetric rank-one downdate, DA-TSK enables the removal of a selected sample from the active training set, provided that the reduced Gram matrix remains nonsingular. The mathematical rigor ensures that after each removal operation, the consequent parameters satisfy the global optimality condition.

2. Building upon DA-TSK and the incremental capabilities of DIA-TSK, we propose Online Adaptive TSK (OA-TSK), an online fuzzy classifier that integrates incremental and decremental updates to continuously adapt to non-stationary data streams. The proposed incremental–decremental update protocol has two phases: Phase I incorporates a new sample via a Sherman–Morrison rank-one update, and Phase II removes the oldest sample via a Sherman–Morrison rank-one downdate when the window capacity is exceeded. This protocol maintains bounded memory usage while preserving global optimality at every time step.
3. While OA-TSK effectively handles concept drift through its sliding-window mechanism, severe shifts in the input distribution may cause the fixed rule centers to become poorly aligned with current samples. To address this, we propose Online Adaptive TSK with Antecedent Reclustering (OA-A-TSK), which uses an adaptive antecedent reclustering mechanism to monitor the mean normalized firing strength of each rule and update the rule centers when these values are severely imbalanced.
4. To address this issue, our proposed framework periodically monitors imbalances among the mean normalized firing strengths of the fuzzy rules during online learning and updates the fuzzy rule antecedent parameters when the corresponding monitoring statistics exceed their predefined thresholds. These statistics may indirectly reflect changes in class proportions when different classes exhibit different rule activation patterns. This mechanism allows the classifier to maintain relatively stable rule antecedents when the monitoring criteria are not met, while adaptively adjusting them in response to substantial imbalances in rule activation. Furthermore, a regularization term is incorporated to improve model generalization and reduce sensitivity to data fluctuations. By considering the potential impact of class imbalance associated with concept drift, the proposed framework aims to achieve a better balance between model stability, generalization, and adaptability in non-stationary data streams.

By maintaining exact equivalence to the ridge-regularized least-squares solution obtained by batch retraining at each time step, OA-TSK provides a principled theoretical foundation for online TSK fuzzy classification that was previously unavailable. The equivalence guarantee—that OA-TSK produces exactly the same model as batch retraining on the active window—eliminates the approximation errors inherent in gradient-based online methods and provides rigorous performance guarantees. Our experimental results on synthetic and real-world data streams with various drift patterns demonstrate the effectiveness of OA-TSK and OA-A-TSK in terms of both classification performance and computational efficiency. The results confirm that OA-TSK achieves accuracy comparable to batch retraining while enabling genuine online learning capabilities for TSK fuzzy classifiers.

The rest of this article is organized as follows. Section 2 surveys related work, covering classical zero-order TSK fuzzy classifiers and DIA-TSK. Section 3 introduces DA-TSK and its theoretical foundation. Section 4 details the proposed online learning methods, including OA-TSK and OA-A-TSK. Section 5 reports experimental evaluations on synthetic and real-world data streams. Section 6 concludes the paper.

2. Related Work

2.1. Classical Zero-Order TSK Fuzzy Classifier

Since zero-order TSK fuzzy classifiers form the foundation of OA-TSK, we first present their formal definition and mathematical formulation. A zero-order TSK fuzzy classifier consists of K fuzzy rules. The k -th rule is expressed as [1]

$$\text{IF } x_1 \text{ is } A_1^k \wedge x_2 \text{ is } A_2^k \wedge \dots \wedge x_d \text{ is } A_d^k \text{ THEN } y_k = a_k, \quad k = 1, 2, \dots, K. \quad (1)$$

Here, $\mathbf{x} = (x_1, x_2, \dots, x_d)^T \in \mathbb{R}^d$ is the input vector, and x_j is its j -th component, $j = 1, \dots, d$. The fuzzy set A_j^k is defined on the j -th feature for rule k . The symbol y_k denotes the output of rule k , and $a_k \in \mathbb{R}$ is its constant consequent. The Gaussian membership function adopted in this study is

$$\phi_j^k(x_j) = \exp \left[-\frac{(x_j - c_j^k)^2}{2(\sigma_j^k)^2} \right], \quad (2)$$

where c_j^k and σ_j^k denote the center and width of the Gaussian membership function for the k -th rule on the j -th feature dimension. These parameters can be initialized through clustering algorithms such as k-means [26] or fuzzy c-means [27], with the cluster centers serving as rule centers [28]. The firing strength of rule k is calculated as:

$$\omega^k(\mathbf{x}) = \prod_{j=1}^d \phi_j^k(x_j), \quad (3)$$

and its normalized firing strength is

$$g^k(\mathbf{x}) = \frac{\omega^k(\mathbf{x})}{\sum_{r=1}^K \omega^r(\mathbf{x})}. \quad (4)$$

The output of the scalar zero-order TSK model is

$$y = \sum_{k=1}^K g^k(\mathbf{x}) \cdot a_k. \quad (5)$$

For classification with C classes, each rule has one consequent for each class, producing C output scores, and the class associated with the maximum score is selected. Given a training set $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$, the normalized firing-strength matrix $\mathbf{G} \in \mathbb{R}^{N \times K}$ is constructed using

$$G_{ik} = g^k(\mathbf{x}_i) = \frac{\omega^k(\mathbf{x}_i)}{\sum_{r=1}^K \omega^r(\mathbf{x}_i)}, i = 1, \dots, N, k = 1, \dots, K. \quad (6)$$

For multi-class classification, the one-hot label matrix $\mathbf{Y} \in \{0, 1\}^{N \times C}$ is defined by

$$Y_{ic} = \begin{cases} 1, & y_i = c \\ 0, & \text{Otherwise} \end{cases} \quad i = 1, \dots, N, c = 1, \dots, C. \quad (7)$$

Let $\mathbf{A} \in \mathbb{R}^{K \times C}$ denote the consequent parameter matrix, where A_{kc} is the consequent of rule k for class c . Let $\mathbf{y}_i = [Y_{i1}, Y_{i2}, \dots, Y_{iC}]$ denote the one-hot row vector associated with the scalar class label y_i . Subsequent sections represent an encoded labeled sample as $(\mathbf{x}_i, \mathbf{y}_i)$. The output-score matrix is

$$\hat{\mathbf{Y}} = \mathbf{GA}. \quad (8)$$

The consequent parameters are learned by ridge-regularized least squares:

$$\mathbf{A}^* = \arg \min_{\mathbf{A}} \left\{ \|\mathbf{GA} - \mathbf{Y}\|_F^2 + \lambda \|\mathbf{A}\|_F^2 \right\}, \lambda > 0, \quad (9)$$

where $\|\cdot\|_F$ denotes the Frobenius norm and λ is the ridge regularization parameter. Its closed-form solution is:

$$\mathbf{A}^* = \left(\mathbf{G}^T \mathbf{G} + \lambda \mathbf{I}_K \right)^{-1} \mathbf{G}^T \mathbf{Y}, \quad (10)$$

where $\mathbf{I}_K$ is the $K \times K$ identity matrix. Define the inverse of the regularized Gram matrix as

$$\mathbf{P} = \left(\mathbf{G}^T \mathbf{G} + \lambda \mathbf{I}_K \right)^{-1}. \quad (11)$$

The optimal consequent parameters can then be written as

$$\mathbf{A}^* = \mathbf{P} \mathbf{G}^T \mathbf{Y}. \quad (12)$$

This matrix formulation provides the foundation for the incremental and decremental updates developed in subsequent sections.

Although batch training provides a closed-form solution, forming and inverting the regularized Gram matrix require $O(NK^2 + K^3)$ operations. Moreover, incorporating new samples requires repeated batch retraining, while assigning equal weight to older and more recent samples may reduce performance under concept drift. These limitations motivate the incremental and decremental closed-form updates developed in the following sections.

2.2. Dynamic Incremental Adaptive TSK Fuzzy Classifier (DIA-TSK)

In our previous work [21], we proposed DIA-TSK to address the limitations of batch training in streaming environments. Related incremental and sequential learning methods include PANFIS [29], incremental linear discriminant analysis [30], the fuzzy model with online incremental SVM learning proposed in [31], and McFIS [32]. DIA-TSK represents a significant advancement in incremental learning for TSK fuzzy classifiers by maintaining global optimality through closed-form parameter updates. The core innovation of O-DIA-TSK lies in its application of the Sherman–Morrison formula to incrementally update the inverse Gram matrix. When a new training sample $(\mathbf{x}_t, \mathbf{y}_t)$ arrives at time t , its normalized firing-strength vector $\mathbf{g}_t \in \mathbb{R}^{1 \times K}$ is first computed using the established fuzzy rule antecedents. To review the original unregularized DIA-TSK formulation, this subsection sets $\lambda = 0$ and assumes that $\mathbf{G}_{t-1}$ has full column rank. Let $\mathbf{G}_0$ and $\mathbf{Y}_0$ be the normalized firing-strength and label matrices for the N_0 initial training samples. The augmented normalized firing-strength and label matrices become

$$\mathbf{G}_t = \begin{bmatrix} \mathbf{G}_{t-1} \\ \mathbf{g}_t \end{bmatrix} \in \mathbb{R}^{(N_0+t) \times K}, \quad \mathbf{Y}_t = \begin{bmatrix} \mathbf{Y}_{t-1} \\ \mathbf{y}_t \end{bmatrix} \in \mathbb{R}^{(N_0+t) \times C}. \quad (13)$$

The corresponding Gram matrix satisfies:

$$\mathbf{G}_t^T \mathbf{G}_t = \mathbf{G}_{t-1}^T \mathbf{G}_{t-1} + \mathbf{g}_t^T \mathbf{g}_t. \quad (14)$$

Rather than recomputing the matrix inverse from scratch, DIA-TSK applies the Sherman–Morrison formula to obtain the updated inverse Gram matrix efficiently:

$$\mathbf{P}_t = \left(\mathbf{G}_t^T \mathbf{G}_t \right)^{-1} = \mathbf{P}_{t-1} - \frac{\mathbf{P}_{t-1} \mathbf{g}_t^T \mathbf{g}_t \mathbf{P}_{t-1}}{1 + \mathbf{g}_t \mathbf{P}_{t-1} \mathbf{g}_t^T}, \quad (15)$$

where $\mathbf{P}_{t-1} = \left(\mathbf{G}_{t-1}^T \mathbf{G}_{t-1} \right)^{-1}$ is already available from the previous time step. This formulation reduces the computational complexity of updating the inverse Gram matrix from $O(K^3)$ for direct matrix inversion to $O(K^2)$ for the rank-one update. With the updated

inverse Gram matrix $\mathbf{P}_t$, the consequent parameters are adjusted to maintain optimality. The parameter update is computed as:

$$\Delta \mathbf{A}_t = \mathbf{P}_t \mathbf{g}_t^T (\mathbf{y}_t - \mathbf{g}_t \mathbf{A}_{t-1}). \quad (16)$$

$$\mathbf{A}_t = \mathbf{A}_{t-1} + \Delta \mathbf{A}_t. \quad (17)$$

This formulation ensures that

$$\mathbf{A}_t = \left(\mathbf{G}_t^T \mathbf{G}_t \right)^{-1} \mathbf{G}_t^T \mathbf{Y}_t, \quad (18)$$

which is the globally optimal least-squares solution for the augmented dataset.

However, DIA-TSK has a fundamental limitation: it can only add new samples to the model but cannot remove outdated or irrelevant samples. Although DIA-TSK can be implemented using fixed-size sufficient statistics, it cannot remove the influence of outdated samples; hence its effective training history grows indefinitely. The proposed DA-TSK and OA-TSK classifiers address this limitation by applying the Sherman–Morrison formula to a symmetric rank-one downdate.

3. Decremental Adaptive TSK Fuzzy Classifier (DA-TSK)

DIA-TSK has demonstrated remarkable capabilities in incremental learning scenarios through its closed-form parameter updates that preserve the least-squares optimum [21]. As in the DIA-TSK review in Section 2.2, the following derivation first considers the unregularized case ($\lambda = 0$); its ridge-regularized extension is presented in Section 4.1. However, a remaining limitation of the incremental recursion is that it does not include an explicit downdate for sample removal. For non-stationary data streams affected by concept drift or distribution shift, retaining all historical information not only causes the influence of outdated samples to accumulate but also deteriorates classification performance as outdated patterns interfere with current decision boundaries [23].

Therefore, DA-TSK enables precise sample removal while preserving exact equivalence to batch least-squares retraining of the TSK fuzzy classifier. Consider an active training set

$$\mathcal{D}_t = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i \in \mathcal{I}_t}, \quad N_t = |\mathcal{D}_t|, \quad (19)$$

where $\mathcal{I}_t$ is the index set of the currently active samples. The current model state is denoted by $\Theta_t = (\mathbf{A}_t, \mathbf{P}_t, \mathbf{G}_t, \mathbf{Y}_t)$. When the sample $(\mathbf{x}_r, \mathbf{y}_r)$ is removed, the resulting dataset and model state are denoted by $\mathcal{D}_t^{(-r)} = \mathcal{D}_t \setminus \{(\mathbf{x}_r, \mathbf{y}_r)\}$, $\Theta_t^{(-r)} = (\mathbf{A}_t^{(-r)}, \mathbf{P}_t^{(-r)}, \mathbf{G}_t^{(-r)}, \mathbf{Y}_t^{(-r)})$. Throughout this section, both $\mathbf{G}_t$ and $\mathbf{G}_t^{(-r)}$ are assumed to have full column rank, ensuring the existence of the inverse Gram matrices used below. The decremental learning problem seeks to compute $\Theta_t^{(-r)}$ efficiently while ensuring exact equivalence to batch retraining on $\mathcal{D}_t^{(-r)}$.

The theoretical foundation of DA-TSK rests upon applying a symmetric rank-one downdate to the Gram matrix. Before removal, the normalized firing-strength matrix can be decomposed as:

$$\mathbf{G}_t = \begin{bmatrix} \mathbf{G}_{\text{pre}} \\ \mathbf{g}_r \\ \mathbf{G}_{\text{post}} \end{bmatrix} \in \mathbb{R}^{N_t \times K}, \quad (20)$$

where $\mathbf{G}_{\text{pre}}$ contains the rows preceding the removed sample, $\mathbf{g}_r \in \mathbb{R}^{1 \times K}$ is its normalized firing-strength vector, and $\mathbf{G}_{\text{post}}$ contains the subsequent rows. After removal,

$$\mathbf{G}_t^{(-r)} = \begin{bmatrix} \mathbf{G}_{\text{pre}} \\ \mathbf{G}_{\text{post}} \end{bmatrix} \in \mathbb{R}^{(N_t-1) \times K}. \quad (21)$$

Similarly, the label matrices satisfy

$$\mathbf{Y}_t = \begin{bmatrix} \mathbf{Y}_{\text{pre}} \\ \mathbf{y}_r \\ \mathbf{Y}_{\text{post}} \end{bmatrix}, \quad \mathbf{Y}_t^{(-r)} = \begin{bmatrix} \mathbf{Y}_{\text{pre}} \\ \mathbf{Y}_{\text{post}} \end{bmatrix} \in \mathbb{R}^{(N_t-1) \times C}. \quad (22)$$

Consequently, the Gram matrix after removal satisfies

$$\left(\mathbf{G}_t^{(-r)}\right)^T \mathbf{G}_t^{(-r)} = \mathbf{G}_t^T \mathbf{G}_t - \mathbf{g}_r^T \mathbf{g}_r. \quad (23)$$

For a nonsingular matrix $\mathbf{M}$ and a column vector $\mathbf{u}$, the symmetric rank-one downdate identity is

$$\left(\mathbf{M} - \mathbf{u}\mathbf{u}^T\right)^{-1} = \mathbf{M}^{-1} + \frac{\mathbf{M}^{-1}\mathbf{u}\mathbf{u}^T\mathbf{M}^{-1}}{1 - \mathbf{u}^T\mathbf{M}^{-1}\mathbf{u}}, \quad (24)$$

provided that $1 - \mathbf{u}^T\mathbf{M}^{-1}\mathbf{u} \neq 0$. Setting $\mathbf{M} = \mathbf{G}_t^T \mathbf{G}_t$, $\mathbf{u} = \mathbf{g}_r^T$, and $\mathbf{M}^{-1} = \mathbf{P}_t$ gives

$$\mathbf{P}_t^{(-r)} = \mathbf{P}_t + \frac{\mathbf{P}_t \mathbf{g}_r^T \mathbf{g}_r \mathbf{P}_t}{1 - \mathbf{g}_r \mathbf{P}_t \mathbf{g}_r^T}. \quad (25)$$

Let

$$h_r = \mathbf{g}_r \mathbf{P}_t \mathbf{g}_r^T, \quad \eta_r = 1 - h_r, \quad \mathbf{q}_r = \frac{\mathbf{P}_t \mathbf{g}_r^T}{\eta_r}. \quad (26)$$

In the unregularized setting of this section, define the hat matrix as $\mathbf{H}_t = \mathbf{G}_t \left(\mathbf{G}_t^T \mathbf{G}_t\right)^{-1} \mathbf{G}_t^T$. The leverage score $h_r = \mathbf{g}_r \mathbf{P}_t \mathbf{g}_r^T$ is the diagonal entry of $\mathbf{H}_t$ corresponding to sample r , evaluated before its removal. Here, r identifies the sample to be removed, rather than the number of removed samples. Equation (26) applies to a single-sample removal: $\eta_r = 1 - h_r$ is the scalar downdate denominator, and $\mathbf{q}_r \in \mathbb{R}^{K \times 1}$ is the downdate gain vector. For successive removals, the leverage score is recomputed using the current inverse Gram matrix before each removal. The downdate is well-defined only if the remaining Gram matrix is nonsingular. When $\mathbf{P}_t$ is positive definite and $\mathbf{g}_r$ is a row of $\mathbf{G}_t$, this condition is equivalent to $\eta_r > 0$. If $\eta_r = 0$, removing the sample makes the remaining Gram matrix singular. If η_r is positive but numerically close to zero, the downdate may be unstable and should instead be computed using a numerically stable batch factorization or an explicitly regularized formulation.

The removal of a sample also requires updating the consequent parameters. The first-order optimality condition for the unregularized least-squares problem is the normal equation

$$\left(\mathbf{G}_t^{(-r)}\right)^T \left(\mathbf{G}_t^{(-r)} \mathbf{A}_t^{(-r)} - \mathbf{Y}_t^{(-r)}\right) = \mathbf{0}. \quad (27)$$

Equivalently,

$$\mathbf{A}_t^{(-r)} = \mathbf{P}_t^{(-r)} \left(\mathbf{G}_t^{(-r)}\right)^T \mathbf{Y}_t^{(-r)} = \mathbf{P}_t^{(-r)} \left(\mathbf{G}_t^T \mathbf{Y}_t - \mathbf{g}_r^T \mathbf{y}_r\right). \quad (28)$$

Using $\mathbf{A}_t = \mathbf{P}_t \mathbf{G}_t^T \mathbf{Y}_t$ and the downdate identity in (25), the consequent parameter update can be written compactly as

$$\Delta \mathbf{A}_r = -\mathbf{q}_r (\mathbf{y}_r - \mathbf{g}_r \mathbf{A}_t) \quad (29)$$

and

$$\mathbf{A}_t^{(-r)} = \mathbf{A}_t + \Delta \mathbf{A}_r = \mathbf{A}_t - \frac{\mathbf{P}_t \mathbf{g}_r^T}{1 - \mathbf{g}_r \mathbf{P}_t \mathbf{g}_r^T} (\mathbf{y}_r - \mathbf{g}_r \mathbf{A}_t). \quad (30)$$

The dimensions in (29) are $\underbrace{\mathbf{q}_r}_{K \times 1} \underbrace{(\mathbf{y}_r - \mathbf{g}_r \mathbf{A}_t)}_{1 \times C} \in \mathbb{R}^{K \times C}$, which agree with the dimensions of $\mathbf{A}_t$ and $\mathbf{A}_t^{(-r)}$.

This formulation ensures that $\mathbf{A}_t^{(-r)}$ satisfies the normal equation in (27) and is exactly equal to the least-squares solution obtained by batch retraining on the reduced dataset, provided that the remaining Gram matrix is nonsingular. Algorithm 1 summarizes the resulting single-sample decremental update, including the numerical validity check and fallback recomputation.

Algorithm 1: Single-Sample Decremental Update

Input: Current model state, consisting of $\mathbf{A}_t$, $\mathbf{P}_t$, $\mathbf{G}_t$, and $\mathbf{Y}_t$; the index $r \in \mathcal{I}_t$ of the sample to be removed; numerical tolerance $\varepsilon > 0$.

Output: Updated model state, consisting of $\mathbf{A}_t^{(-r)}$, $\mathbf{P}_t^{(-r)}$, $\mathbf{G}_t^{(-r)}$, $\mathbf{Y}_t^{(-r)}$.

Procedure:

- Step 1 Let $\mathbf{g}_r$ and $\mathbf{y}_r$ denote the rows of $\mathbf{G}_t$ and $\mathbf{Y}_t$, respectively, corresponding to sample $(\mathbf{x}_r, \mathbf{y}_r)$.
 - Step 2 Compute $\mathbf{z}_r \leftarrow \mathbf{P}_t \mathbf{g}_r^T$, $h_r \leftarrow \mathbf{g}_r \mathbf{z}_r$, $\eta_r \leftarrow 1 - h_r$.
 - Step 3 If $0 < \eta_r \leq \varepsilon$, form $\mathbf{G}_t^{(-r)}$ and $\mathbf{Y}_t^{(-r)}$ by removing the rows corresponding to sample $(\mathbf{x}_r, \mathbf{y}_r)$ from $\mathbf{G}_t$ and $\mathbf{Y}_t$, respectively. Recompute $\mathbf{P}_t^{(-r)} = \left[\left(\mathbf{G}_t^{(-r)} \right)^T \mathbf{G}_t^{(-r)} \right]^{-1}$, $\mathbf{A}_t^{(-r)} = \mathbf{P}_t^{(-r)} \left(\mathbf{G}_t^{(-r)} \right)^T \mathbf{Y}_t^{(-r)}$ using a numerically stable factorization, and return the recomputed model state. If $\eta_r \leq 0$, terminate the Sherman–Morrison downdate because the reduced Gram matrix is singular or the current inverse Gram matrix is numerically invalid. Otherwise, proceed to Step 4.
 - Step 4 Compute the downdate vector: $\mathbf{q}_r \leftarrow \frac{\mathbf{z}_r}{\eta_r}$.
 - Step 5 Update the inverse Gram matrix: $\mathbf{P}_t^{(-r)} \leftarrow \mathbf{P}_t + \frac{\mathbf{z}_r \mathbf{z}_r^T}{\eta_r}$.
 - Step 6 Compute the removed sample residual and update the consequent parameters: $\mathbf{e}_r \leftarrow \mathbf{y}_r - \mathbf{g}_r \mathbf{A}_t$, $\mathbf{A}_t^{(-r)} \leftarrow \mathbf{A}_t - \mathbf{q}_r \mathbf{e}_r$.
 - Step 7 Form $\mathbf{G}_t^{(-r)}$ and $\mathbf{Y}_t^{(-r)}$ by removing the rows corresponding to sample $(\mathbf{x}_r, \mathbf{y}_r)$ from $\mathbf{G}_t$ and $\mathbf{Y}_t$, respectively.
 - Step 8 Return $\mathbf{A}_t^{(-r)}$, $\mathbf{P}_t^{(-r)}$, $\mathbf{G}_t^{(-r)}$, $\mathbf{Y}_t^{(-r)}$.
-

By reusing $\mathbf{z}_r$ and $\mathbf{q}_r$, DA-TSK requires $O(K^2)$ operations to update the inverse Gram matrix and $O(KC)$ operations to update the consequent parameters. Thus, the arithmetic complexity of a single-sample removal via the rank-one downdate is $O(K^2 + KC)$. The fallback recomputation in Step 3 incurs the cost of batch retraining. Under the convention that C is fixed or $C \ll K$, complete batch retraining on the remaining $N_t - 1$ samples requires approximately $O((N_t - 1)K^2 + K^3)$ operations. If the dependence on C is retained explicitly, the batch complexity also includes the terms $O(N_t KC + K^2 C)$. Therefore, in exact

arithmetic, DA-TSK produces a model identical to batch least-squares retraining on the reduced dataset, provided that the antecedent parameters remain fixed and the reduced Gram matrix is nonsingular.

4. Proposed Online Learning Methods

4.1. Online Adaptive TSK (OA-TSK)

Building upon the incremental capabilities of DIA-TSK and the decremental capabilities of DA-TSK, OA-TSK integrates incremental and decremental updates to continuously adapt to non-stationary data streams while maintaining exact equivalence to batch retraining in exact arithmetic. The incremental and decremental derivations in Sections 2.2 and 3 present the unregularized rank-one update and downdate identities, respectively. Because the fixed regularization term $\lambda \mathbf{I}_K$ is unaffected by sample addition or removal, the same identities apply to the regularized Gram matrix after replacing $\mathbf{G}^T \mathbf{G}$ with $\mathbf{G}^T \mathbf{G} + \lambda \mathbf{I}_K$.

To improve numerical conditioning and maintain consistency with the initialization procedure, we adopt a fixed ridge parameter $\lambda > 0$ throughout the online learning process. At each time step t , OA-TSK maintains $\mathbf{P}_t = \left(\mathbf{G}_t^T \mathbf{G}_t + \lambda \mathbf{I}_K \right)^{-1}$, and the consequent parameter matrix satisfies

$$\mathbf{A}_t = \arg \min_{\mathbf{A}} \left\{ \|\mathbf{G}_t \mathbf{A} - \mathbf{Y}_t\|_F^2 + \lambda \|\mathbf{A}\|_F^2 \right\} = \left(\mathbf{G}_t^T \mathbf{G}_t + \lambda \mathbf{I}_K \right)^{-1} \mathbf{G}_t^T \mathbf{Y}_t. \quad (31)$$

This equivalence holds for any sequence of valid incremental and decremental operations, provided that the antecedent parameters and the ridge parameter λ remain fixed. Thus, OA-TSK produces the same model as ridge-regularized batch retraining on the active window.

The OA-TSK classifier applies the incremental–decremental update protocol when a new sample $(\mathbf{x}_t, \mathbf{y}_t)$ arrives at time t , where $\mathbf{x}_t \in \mathbb{R}^d$ is the input vector and $\mathbf{y}_t \in \{0, 1\}^{1 \times C}$ is its one-hot label vector:

1. Phase I—Incremental Update: Temporarily augment the previous active window with the new sample using the Sherman–Morrison rank-one update employed in DIA-TSK.
2. Phase II—Decremental Update: If the temporary augmented window exceeds the window capacity W , remove its oldest sample using the Sherman–Morrison rank-one downdate derived in DA-TSK under a first-in-first-out (FIFO) policy.

For the incoming sample, the normalized firing-strength vector is

$$\mathbf{g}_t = \left[g_t^{(1)}, g_t^{(2)}, \dots, g_t^{(K)} \right] \in \mathbb{R}^{1 \times K}, \quad g_t^{(k)} = \frac{\omega^k(\mathbf{x}_t)}{\sum_{r=1}^K \omega^r(\mathbf{x}_t)}. \quad (32)$$

Thus, $\sum_{k=1}^K g_t^{(k)} = 1$. The label vector follows the one-hot encoding convention:

$$\mathbf{y}_t = [y_{t1}, y_{t2}, \dots, y_{tC}] \in \{0, 1\}^{1 \times C}, \quad \sum_{c=1}^C y_{tc} = 1. \quad (33)$$

The corresponding scalar class index can be recovered from the one-hot vector as $y_t = \arg \max_{m \in \{1, \dots, C\}} y_{tm}$. The augmented normalized firing-strength matrix is:

$$\mathbf{G}_{\text{tmp}} = \begin{bmatrix} \mathbf{G}_{t-1} \\ \mathbf{g}_t \end{bmatrix} \in \mathbb{R}^{(N_{t-1}+1) \times K}. \quad (34)$$

Its regularized Gram matrix satisfies:

$$\mathbf{G}_{\text{tmp}}^T \mathbf{G}_{\text{tmp}} + \lambda \mathbf{I}_K = \mathbf{G}_{t-1}^T \mathbf{G}_{t-1} + \lambda \mathbf{I}_K + \mathbf{g}_t^T \mathbf{g}_t. \quad (35)$$

Here, K is the number of fuzzy rules, and N_{t-1} is the number of active samples before the current update. The matrix $\mathbf{G}_{t-1} \in \mathbb{R}^{N_{t-1} \times K}$ contains their normalized firing-strength vectors, while $\mathbf{g}_t \in \mathbb{R}^{1 \times K}$ represents the incoming sample. The matrix $\mathbf{G}_{\text{tmp}} \in \mathbb{R}^{(N_{t-1}+1) \times K}$ is formed by appending $\mathbf{g}_t$ to $\mathbf{G}_{t-1}$. Throughout this subsection, the subscript “tmp” denotes the temporary state after sample addition and before any sample removal. The scalar $\lambda > 0$ is the fixed ridge regularization parameter, and $\mathbf{I}_K$ is the $K \times K$ identity matrix. Let

$$\mathbf{z}_t = \mathbf{P}_{t-1} \mathbf{g}_t^T, \quad \alpha_t = 1 + \mathbf{g}_t \mathbf{z}_t, \quad \mathbf{q}_t = \frac{\mathbf{z}_t}{\alpha_t}. \quad (36)$$

Applying the Sherman–Morrison formula gives

$$\mathbf{P}_{\text{tmp}} = \mathbf{P}_{t-1} - \frac{\mathbf{z}_t \mathbf{z}_t^T}{\alpha_t}. \quad (37)$$

The augmented label matrix is:

$$\mathbf{Y}_{\text{tmp}} = \begin{bmatrix} \mathbf{Y}_{t-1} \\ \mathbf{y}_t \end{bmatrix} \in \mathbb{R}^{(N_{t-1}+1) \times C}. \quad (38)$$

Here, C is the number of classes, $\mathbf{Y}_{t-1} \in \{0, 1\}^{N_{t-1} \times C}$ contains the one-hot labels of the previous active samples, and $\mathbf{y}_t \in \{0, 1\}^{1 \times C}$ is the one-hot label vector of the incoming sample. The augmented label matrix $\mathbf{Y}_{\text{tmp}}$ is obtained by appending $\mathbf{y}_t$ to $\mathbf{Y}_{t-1}$, with its rows aligned with those of $\mathbf{G}_{\text{tmp}}$. The consequent parameter matrix is updated using

$$\mathbf{A}_{\text{tmp}} = \mathbf{A}_{t-1} + \mathbf{q}_t (\mathbf{y}_t - \mathbf{g}_t \mathbf{A}_{t-1}). \quad (39)$$

Here, $\mathbf{A}_{t-1} \in \mathbb{R}^{K \times C}$ and $\mathbf{A}_{\text{tmp}} \in \mathbb{R}^{K \times C}$ are the consequent parameter matrices before and after incorporating the incoming sample, respectively. The vector $\mathbf{q}_t = \mathbf{z}_t / \alpha_t \in \mathbb{R}^{K \times 1}$, defined in (36), is the incremental gain vector. The residual $\mathbf{y}_t - \mathbf{g}_t \mathbf{A}_{t-1} \in \mathbb{R}^{1 \times C}$ is computed using the model before the update, so the resulting parameter increment has dimensions $K \times C$. Consequently, $\mathbf{A}_{\text{tmp}} = \left(\mathbf{G}_{\text{tmp}}^T \mathbf{G}_{\text{tmp}} + \lambda \mathbf{I}_K \right)^{-1} \mathbf{G}_{\text{tmp}}^T \mathbf{Y}_{\text{tmp}}$, which is the globally optimal ridge-regularized least-squares solution for the temporary augmented window.

The window capacity W is a positive integer selected before online processing and kept fixed throughout each run. In our experiments, it is selected by grid search in validation experiments, as described in Section 5.1. Before an update, the active window contains $N_{t-1} \leq W$ samples. Adding one sample gives $N_{\text{tmp}} = N_{t-1} + 1$, so the number of samples to remove is $\max(N_{\text{tmp}} - W, 0) \in \{0, 1\}$. If $N_{\text{tmp}} > W$, exactly one oldest sample is removed under the FIFO policy, restoring the window size to W . The capacity W itself remains unchanged.

If $N_{t-1} + 1 \leq W$, no sample is removed, and $(\mathbf{A}_t, \mathbf{P}_t, \mathbf{G}_t, \mathbf{Y}_t) = (\mathbf{A}_{\text{tmp}}, \mathbf{P}_{\text{tmp}}, \mathbf{G}_{\text{tmp}}, \mathbf{Y}_{\text{tmp}})$. If $N_{t-1} + 1 > W$, the temporary augmented window contains $W + 1$ samples. Under the FIFO policy, let s_t denote the index of the oldest sample in this temporary augmented window. The corresponding normalized firing-strength and label vectors are denoted by $\mathbf{g}_{s_t} \in \mathbb{R}^{1 \times K}$, $\mathbf{y}_{s_t} \in \mathbb{R}^{1 \times C}$. Because the samples are stored in chronological order, the temporary normalized firing-strength and label matrices can be partitioned as

$$\mathbf{G}_{\text{tmp}} = \begin{bmatrix} \mathbf{g}_{s_t} \\ \mathbf{G}_t \end{bmatrix}, \quad \mathbf{Y}_{\text{tmp}} = \begin{bmatrix} \mathbf{y}_{s_t} \\ \mathbf{Y}_t \end{bmatrix}, \quad (40)$$

where

$$\mathbf{G}_t \in \mathbb{R}^{W \times K}, \mathbf{Y}_t \in \mathbb{R}^{W \times C} \quad (41)$$

are the normalized firing-strength and label matrices retained after removing the oldest sample. Accordingly, the regularized Gram matrix after removal satisfies $\mathbf{G}_t^T \mathbf{G}_t + \lambda \mathbf{I}_K = \mathbf{G}_{\text{tmp}}^T \mathbf{G}_{\text{tmp}} + \lambda \mathbf{I}_K - \mathbf{g}_{s_t}^T \mathbf{g}_{s_t}$. Let

$$\mathbf{z}_{s_t} = \mathbf{P}_{\text{tmp}} \mathbf{g}_{s_t}^T, h_{s_t} = \mathbf{g}_{s_t}^T \mathbf{z}_{s_t}, \eta_{s_t} = 1 - h_{s_t}, \mathbf{q}_{s_t} = \frac{\mathbf{z}_{s_t}}{\eta_{s_t}}. \quad (42)$$

Here, s_t identifies the oldest sample in the temporary augmented window, and $\mathbf{g}_{s_t} \in \mathbb{R}^{1 \times K}$ is its normalized firing-strength vector. The matrix $\mathbf{P}_{\text{tmp}} = (\mathbf{G}_{\text{tmp}}^T \mathbf{G}_{\text{tmp}} + \lambda \mathbf{I}_K)^{-1} \in \mathbb{R}^{K \times K}$ is the inverse regularized Gram matrix before removal. The vector $\mathbf{z}_{s_t} = \mathbf{P}_{\text{tmp}} \mathbf{g}_{s_t}^T \in \mathbb{R}^{K \times 1}$ is an intermediate quantity. The scalar h_{s_t} is the sample's ridge leverage score, evaluated on the temporary augmented window; $\eta_{s_t} = 1 - h_{s_t}$ is the downdate denominator, and $\mathbf{q}_{s_t} = \mathbf{z}_{s_t} / \eta_{s_t} \in \mathbb{R}^{K \times 1}$ is the downdate gain vector. Applying the Sherman–Morrison rank-one downdate gives

$$\mathbf{P}_t = \mathbf{P}_{\text{tmp}} + \frac{\mathbf{z}_{s_t} \mathbf{z}_{s_t}^T}{\eta_{s_t}}. \quad (43)$$

Because $\lambda > 0$, the regularized Gram matrix remains positive definite in exact arithmetic. Nevertheless, when η_{s_t} is numerically close to zero, the rank-one downdate may become unstable. In this case, $\mathbf{P}_t$ and $\mathbf{A}_t$ should be recomputed on the updated active window using a numerically stable matrix factorization. The consequent parameter matrix is updated as

$$\mathbf{A}_t = \mathbf{A}_{\text{tmp}} - \mathbf{q}_{s_t} (\mathbf{y}_{s_t} - \mathbf{g}_{s_t} \mathbf{A}_{\text{tmp}}). \quad (44)$$

The dimensions of the correction term are $\underbrace{\mathbf{q}_{s_t}}_{K \times 1} \underbrace{(\mathbf{y}_{s_t} - \mathbf{g}_{s_t} \mathbf{A}_{\text{tmp}})}_{1 \times C} \in \mathbb{R}^{K \times C}$, which agree with the dimensions of $\mathbf{A}_t$. The updated consequent parameters satisfy

$$\mathbf{A}_t = (\mathbf{G}_t^T \mathbf{G}_t + \lambda \mathbf{I}_K)^{-1} \mathbf{G}_t^T \mathbf{Y}_t. \quad (45)$$

Therefore, in exact arithmetic, the decremental update produces the same solution as ridge-regularized batch retraining on the updated active window. Algorithm 2 presents the complete incremental–decremental update protocol, which integrates the two phases.

While Algorithm 2 provides the core incremental–decremental update protocol for processing individual samples, a complete online learning system requires proper initialization and continuous stream processing. Algorithm 3 presents the full OA-TSK framework, which integrates batch initialization on an initial training set with continuous online updates over the subsequent data stream.

For any selected set of active samples, the guarantee of equivalence to batch retraining remains valid provided that the antecedent parameters and the ridge parameter λ remain fixed and that the closed-form updates are numerically valid. Specifically, $\mathbf{A}_t = (\mathbf{G}_t^T \mathbf{G}_t + \lambda \mathbf{I}_K)^{-1} \mathbf{G}_t^T \mathbf{Y}_t$ holds after each valid incremental or decremental operation. Thus, the removal policy determines which samples are retained for optimization, whereas the incremental–decremental update protocol guarantees the globally optimal ridge-regularized solution for the retained sample set.

Algorithm 2: OA-TSK Incremental–Decremental Update Protocol

Input: Previous model state $\mathbf{A}_{t-1}$, $\mathbf{P}_{t-1}$, $\mathbf{G}_{t-1}$, and $\mathbf{Y}_{t-1}$; the precomputed normalized firing-strength vector $\mathbf{g}_t \in \mathbb{R}^{1 \times K}$; the corresponding one-hot label vector $\mathbf{y}_t \in \{0, 1\}^{1 \times C}$; window capacity W ; fixed ridge parameter $\lambda > 0$; and numerical tolerance $\varepsilon > 0$.

Output: Updated model state $\mathbf{A}_t$, $\mathbf{P}_t$, $\mathbf{G}_t$, and $\mathbf{Y}_t$.

Procedure:

Phase I: Incremental Update

Step 1 Use the precomputed normalized firing-strength vector

$$\mathbf{g}_t = [g_t^{(1)}, g_t^{(2)}, \dots, g_t^{(K)}] \in \mathbb{R}^{1 \times K}.$$

Step 2 Compute the incremental update quantities:

$$\mathbf{z}_t \leftarrow \mathbf{P}_{t-1} \mathbf{g}_t^T, \alpha_t \leftarrow 1 + \mathbf{g}_t \mathbf{z}_t, \mathbf{q}_t \leftarrow \frac{\mathbf{z}_t}{\alpha_t}.$$

Step 3 Update the inverse of the regularized Gram matrix and the consequent

$$\text{parameter matrix: } \mathbf{P}_{\text{tmp}} \leftarrow \mathbf{P}_{t-1} - \frac{\mathbf{z}_t \mathbf{z}_t^T}{\alpha_t}, \mathbf{A}_{\text{tmp}} \leftarrow \mathbf{A}_{t-1} + \mathbf{q}_t (\mathbf{y}_t - \mathbf{g}_t \mathbf{A}_{t-1}).$$

Step 4 Augment the normalized firing-strength and label matrices:

$$\mathbf{G}_{\text{tmp}} \leftarrow \begin{bmatrix} \mathbf{G}_{t-1} \\ \mathbf{g}_t \end{bmatrix}, \mathbf{Y}_{\text{tmp}} \leftarrow \begin{bmatrix} \mathbf{Y}_{t-1} \\ \mathbf{y}_t \end{bmatrix}. \text{ Set } N_{\text{tmp}} \text{ to the number of rows in } \mathbf{G}_{\text{tmp}}.$$

Phase II: Decremental Update

Step 5 Check whether the window capacity is exceeded. If $N_{\text{tmp}} \leq W$, set

$$(\mathbf{A}_t, \mathbf{P}_t, \mathbf{G}_t, \mathbf{Y}_t) \leftarrow (\mathbf{A}_{\text{tmp}}, \mathbf{P}_{\text{tmp}}, \mathbf{G}_{\text{tmp}}, \mathbf{Y}_{\text{tmp}}) \text{ and return the updated model state. Otherwise, proceed to Step 6.}$$

Step 6 Under the FIFO policy, let s_t denote the index of the oldest sample in the temporary augmented window. Partition the temporary matrices as

$$\mathbf{G}_{\text{tmp}} = \begin{bmatrix} \mathbf{g}_{s_t} \\ \mathbf{g}_t \end{bmatrix}, \mathbf{Y}_{\text{tmp}} = \begin{bmatrix} \mathbf{y}_{s_t} \\ \mathbf{y}_t \end{bmatrix}, \text{ where } \mathbf{g}_{s_t} \text{ and } \mathbf{y}_{s_t} \text{ correspond to the sample being removed, while } \mathbf{G}_t \text{ and } \mathbf{Y}_t \text{ contain the } W \text{ retained samples.}$$

Step 7 Compute: $\mathbf{z}_{s_t} \leftarrow \mathbf{P}_{\text{tmp}} \mathbf{g}_{s_t}^T$, $h_{s_t} \leftarrow \mathbf{g}_{s_t} \mathbf{z}_{s_t}$, $\eta_{s_t} \leftarrow 1 - h_{s_t}$. If $\eta_{s_t} \leq \varepsilon$, recompute

$$\mathbf{P}_t = (\mathbf{G}_t^T \mathbf{G}_t + \lambda \mathbf{I}_K)^{-1}, \mathbf{A}_t = \mathbf{P}_t \mathbf{G}_t^T \mathbf{Y}_t, \text{ using a numerically stable factorization, and then proceed directly to Step 10. Otherwise, compute } \mathbf{q}_{s_t} \leftarrow \frac{\mathbf{z}_{s_t}}{\eta_{s_t}}, \text{ and continue with Steps 8 and 9.}$$

Step 8 Update the inverse of the regularized Gram matrix: $\mathbf{P}_t \leftarrow \mathbf{P}_{\text{tmp}} + \frac{\mathbf{z}_{s_t} \mathbf{z}_{s_t}^T}{\eta_{s_t}}$.

Step 9 Update the consequent parameter matrix: $\mathbf{A}_t \leftarrow \mathbf{A}_{\text{tmp}} - \mathbf{q}_{s_t} (\mathbf{y}_{s_t} - \mathbf{g}_{s_t} \mathbf{A}_{\text{tmp}})$.

Step 10 Retain $\mathbf{G}_t$ and $\mathbf{Y}_t$ from the matrix partition in Step 6, and return $\mathbf{A}_t$, $\mathbf{P}_t$, $\mathbf{G}_t$, $\mathbf{Y}_t$.

FIFO is adopted for its low sample-selection overhead and emphasis on recent observations. Under concept drift, selection criteria that ignore recency may discard newly arrived samples while retaining outdated observations. FIFO progressively replaces older samples, favoring adaptation when recent data better represent the current concept. Population-based or boundary-aware removal could account for sample informativeness but would require additional selection effort. Nevertheless, FIFO may discard rare or boundary-defining samples that remain relevant. Unlike gradient-based learners, which may retain historical influence in their weights, the active-window objective explicitly excludes discarded samples; gradient descent alone, however, does not guarantee preservation of critical information. Alternative removal policies are left for future work.

Algorithm 3: OA-TSK Online Learning Framework

Input: An initial training set $\mathcal{D}_0 = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{N_0}$, $K \leq N_0 \leq W$, where $\mathbf{x}_i \in \mathbb{R}^d$, $\mathbf{y}_i \in \{0, 1\}^{1 \times C}$, $\sum_{c=1}^C y_{ic} = 1$; the number of fuzzy rules K ; Gaussian width parameter $\delta > 0$; fixed ridge parameter $\lambda > 0$; window capacity W ; numerical tolerance $\varepsilon > 0$; and a potentially infinite data stream $\{(\mathbf{x}_t, \mathbf{y}_t)\}_{t=1}^{\infty}$, where i indexes the initial training samples and t denotes the online time step.

Output: Updated model state $\mathbf{A}_t, \mathbf{P}_t, \mathbf{G}_t, \mathbf{Y}_t$, together with the number of samples N_t in the active window at each time step t .

Procedure:

Step 1 Apply k-means with K clusters to obtain the cluster centers $\{\mathbf{c}^k\}_{k=1}^K$, which are used as the rule centers. Set c_j^k to the j -th component of the k -th rule center, and set $\sigma_j^k = \delta$, $j = 1, \dots, d, k = 1, \dots, K$.

Step 2 Construct the initial normalized firing-strength matrix $\mathbf{G}_0 \in \mathbb{R}^{N_0 \times K}$. For each sample $\mathbf{x}_i$ and rule k , compute

$$\omega^k(\mathbf{x}_i) = \prod_{j=1}^d \exp \left[-\frac{(x_{ij} - c_j^k)^2}{2\delta^2} \right], \quad G_{0,ik} = \frac{\omega^k(\mathbf{x}_i)}{\sum_{r=1}^K \omega^r(\mathbf{x}_i)}.$$

Step 3 Construct the initial label matrix by vertically stacking the one-hot label vectors:

$$\mathbf{Y}_0 = \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \vdots \\ \mathbf{y}_{N_0} \end{bmatrix} \in \{0, 1\}^{N_0 \times C}. \text{ Each row satisfies } \sum_{c=1}^C Y_{0,ic} = 1, \quad i = 1, \dots, N_0.$$

Step 4 Initialize the inverse of the regularized Gram matrix and the consequent parameter matrix: $\mathbf{P}_0 = (\mathbf{G}_0^T \mathbf{G}_0 + \lambda \mathbf{I}_K)^{-1}$, $\mathbf{A}_0 = \mathbf{P}_0 \mathbf{G}_0^T \mathbf{Y}_0$.

Step 5 For $t = 1, 2, \dots$, follow the test-then-train protocol below.

Step 5.1 Receive the incoming input vector $\mathbf{x}_t$. Compute its normalized firing-strength vector $\mathbf{g}_t = [g_t^{(1)}, g_t^{(2)}, \dots, g_t^{(K)}] \in \mathbb{R}^{1 \times K}$ according to (32).

Step 5.2 Using the previous model, compute the output-score vector $\mathbf{o}_t = \mathbf{g}_t \mathbf{A}_{t-1} \in \mathbb{R}^{1 \times C}$. The predicted class label is $\hat{y}_t = \underset{m \in \{1, \dots, C\}}{\operatorname{argmax}} o_{tm}$.

Step 5.3 After the prediction has been made, receive the one-hot label vector $\mathbf{y}_t$. Obtain the true scalar class label as $y_t = \underset{m \in \{1, \dots, C\}}{\operatorname{argmax}} y_{tm}$. A prediction is correct if and only if $\hat{y}_t = y_t$. Update the online evaluation metrics using the predicted and true scalar class labels $\hat{y}_t$ and y_t , respectively.

Step 5.4 After completing the prediction and evaluation, call Algorithm 2 using $(\mathbf{A}_{t-1}, \mathbf{P}_{t-1}, \mathbf{G}_{t-1}, \mathbf{Y}_{t-1}, \mathbf{g}_t, \mathbf{y}_t, W, \lambda, \varepsilon)$.

Step 5.5 Obtain the updated model state $(\mathbf{A}_t, \mathbf{P}_t, \mathbf{G}_t, \mathbf{Y}_t)$ and update the number of active samples as $N_t = \min(N_{t-1} + 1, W)$.

Step 6 Continue until the data stream terminates or a stopping criterion is met.

4.2. Online Adaptive TSK with Antecedent Reclustering (OA-A-TSK)

OA-TSK assumes that the antecedent fuzzy partition remains fixed after initialization. Specifically, the rule centers $\{\mathbf{c}^k\}_{k=1}^K$, where $\mathbf{c}^k = [c_1^k, \dots, c_d^k]^T$, are initialized by k-means clustering on the initial training set $\mathcal{D}_0$, whereas all Gaussian widths are set to the fixed value $\sigma_j^k = \delta$. Both the rule centers and Gaussian widths remain unchanged during the subsequent incremental and decremental updates. This assumption is computationally

efficient because the incremental and decremental procedures update the model state without modifying the antecedent parameters. OA-A-TSK additionally maintains a FIFO

feature buffer $\mathbf{X}_t = \begin{bmatrix} \mathbf{x}_{t_1}^T \\ \vdots \\ \mathbf{x}_{t_{N_t}}^T \end{bmatrix} \in \mathbb{R}^{N_t \times d}$, which is used only for antecedent reclustering. The

feature buffer $\mathbf{X}_t$ is updated according to the same FIFO policy as $\mathbf{G}_t$ and $\mathbf{Y}_t$, thereby preserving row-wise alignment among the three matrices. $\mathbf{X}_t$ is omitted from Figure 1 for visual clarity.

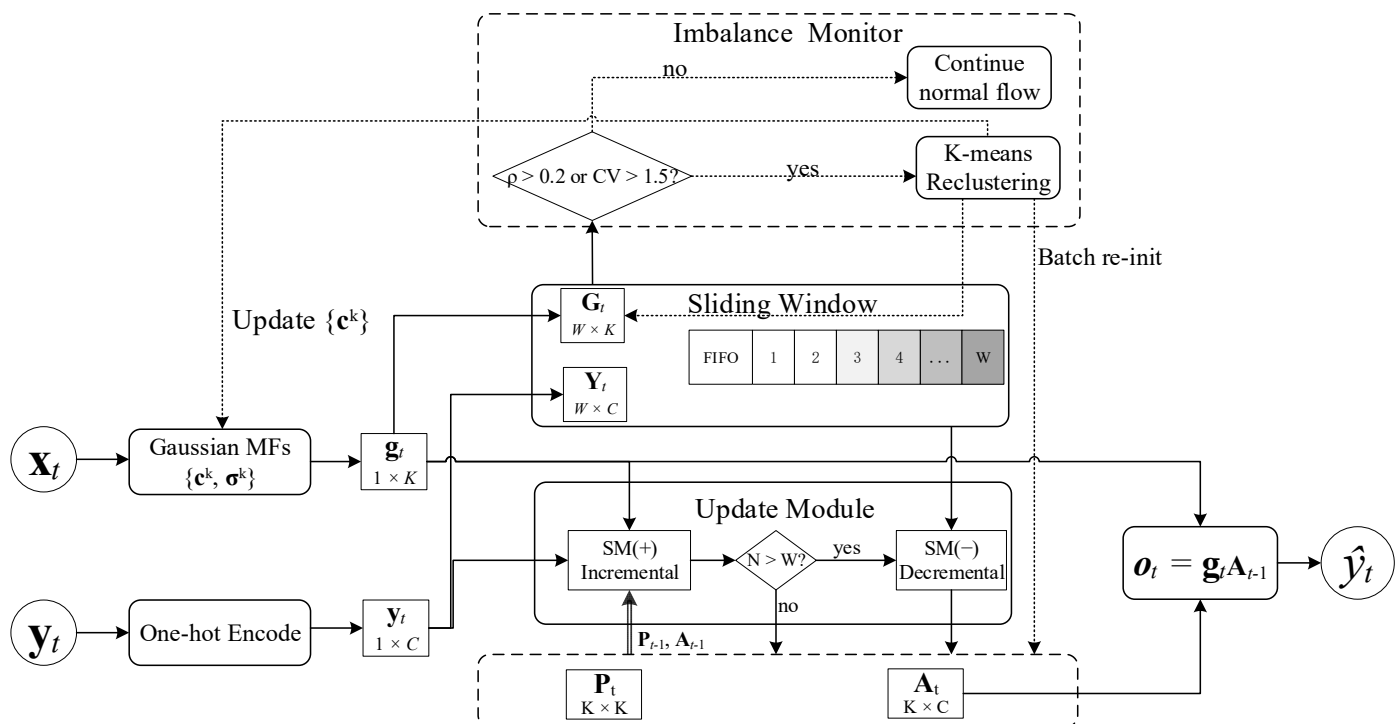


Figure 1. Architecture of the OA-A-TSK online learning framework. The Gaussian MFs receive the input feature vector $\mathbf{x}_t \in \mathbb{R}^d$. The One-hot encode block receives the true class label $\mathbf{y}_t \in \{1, \dots, C\}$, revealed after prediction, and converts it into the one-hot label vector $\mathbf{y}_t \in \{0, 1\}^{1 \times C}$. Single-line solid arrows indicate the main online update flow, the double-line solid arrow carrying $(\mathbf{P}_{t-1}, \mathbf{A}_{t-1})$ denotes temporal feedback from time $t-1$ to t , and dashed arrows represent the antecedent reclustering mechanism triggered by a severe imbalance among the mean normalized firing strengths of the rules.

However, when the input distribution shifts substantially, the original rule centers may no longer represent the current input distribution adequately. Consequently, some rules may exhibit consistently low mean normalized firing strengths, whereas a small subset may dominate the fuzzy representation. This imbalance among the mean normalized firing strengths of the rules may reduce classification performance. Changes in class proportions may also affect these activation statistics when different classes exhibit different distributions of rule firing strengths.

To address this issue, OA-A-TSK augments OA-TSK with an adaptive antecedent reclustering mechanism. Hereafter, antecedent reclustering refers specifically to re-estimating the rule centers by k-means while keeping the Gaussian widths fixed. The monitoring component, labeled the Imbalance Monitor in Figure 1, computes the mean normalized firing strength of each rule over the active window and triggers antecedent reclustering when these values are severely imbalanced. The monitoring criteria below are presented using the reference values $a = 0.1$, $b = 0.2$, and $c = 1.5$. More generally, the

inactive-rule threshold is a/K , and reclustering is triggered when $\rho_{\text{inactive}} > b$ or $CV > c$. Section 5.6 examines the local sensitivity to these parameters.

For each fuzzy rule k , its mean normalized firing strength over the active window is defined as:

$$\bar{g}_t^{(k)} = \frac{1}{N_t} \sum_{i=1}^{N_t} G_{t,ik}, \quad k = 1, 2, \dots, K, \quad (46)$$

where $G_{t,ik}$ is the (i, k) -th element of the current normalized firing-strength matrix $\mathbf{G}_t$. Because the normalized firing strengths of each sample sum to one, $\sum_{k=1}^K \bar{g}_t^{(k)} = 1$. Two statistics are used to quantify the imbalance among the mean normalized firing strengths of the rules. The first is the proportion of inactive rules, defined as

$$\rho_{\text{inactive}} = \frac{1}{K} \sum_{k=1}^K \mathbb{I}\{\bar{g}_t^{(k)} < 0.1K^{-1}\}, \quad (47)$$

where $\mathbb{I}\{\cdot\}$ is the indicator function, which returns 1 when the specified condition is satisfied and 0 otherwise. The threshold $0.1K^{-1}$ is 10% of the uniform baseline K^{-1} . The second statistic is the coefficient of variation (CV) of the mean normalized firing strengths:

$$CV = \frac{\sigma(\bar{g}_t^{(1)}, \dots, \bar{g}_t^{(K)})}{\mu(\bar{g}_t^{(1)}, \dots, \bar{g}_t^{(K)})}, \quad (48)$$

where $\mu(\cdot)$ and $\sigma(\cdot)$ denote the mean and standard deviation, respectively. Because $\sum_{k=1}^K \bar{g}_t^{(k)} = 1$, the arithmetic mean of $\bar{g}_t^{(1)}, \dots, \bar{g}_t^{(K)}$ is $1/K$. A large value of CV indicates that the current samples are represented predominantly by a small subset of fuzzy rules. The imbalance among the mean normalized firing strengths of the rules is considered severe when $\rho_{\text{inactive}} > 0.2$ or $CV > 1.5$. These conditions are used as heuristic indicators that the existing rule centers may be poorly aligned with the samples in the active window. Let L denote the monitoring interval. In practice, the two statistics are evaluated every L online samples using the updated active window. If reclustering is triggered, the updated rule centers and recomputed consequent parameters are used from the next time step.

When either imbalance condition is satisfied, OA-A-TSK applies k-means clustering to the input vectors in the active window $\mathcal{D}_t$. Assuming $N_t \geq K$, the clustering procedure obtains updated rule centers $\{\mathbf{c}_{\text{new}}^k\}_{k=1}^K$. The rule centers are then updated as $\mathbf{c}^k \leftarrow \mathbf{c}_{\text{new}}^k, k = 1, 2, \dots, K$. The Gaussian widths remain fixed at $\sigma_j^k = \delta$ after reclustering. Using the updated rule centers, the normalized firing-strength vectors of all N_t samples in

the active window are recomputed to construct the updated $\mathbf{G}_t = \begin{bmatrix} \mathbf{g}_{t,1} \\ \mathbf{g}_{t,2} \\ \vdots \\ \mathbf{g}_{t,N_t} \end{bmatrix} \in \mathbb{R}^{N_t \times K}$, where

$\mathbf{g}_{t,i} \in \mathbb{R}^{1 \times K}$ is the normalized firing-strength vector of the i -th sample in the active window evaluated using the updated rule centers.

Because the antecedent parameters and hence the entire normalized firing-strength matrix have changed, the Sherman–Morrison update cannot be applied directly across the reclustering event. Instead, the inverse of the regularized Gram matrix and the consequent parameter matrix are recomputed in batch form:

$$\mathbf{P}_t = \left(\mathbf{G}_t^T \mathbf{G}_t + \lambda \mathbf{I}_K \right)^{-1}, \quad (49)$$

$$\mathbf{A}_t = \mathbf{P}_t \mathbf{G}_t^T \mathbf{Y}_t. \quad (50)$$

Therefore, $\mathbf{A}_t = \arg \min_{\mathbf{A}} \left\{ \|\mathbf{G}_t \mathbf{A} - \mathbf{Y}_t\|_F^2 + \lambda \|\mathbf{A}\|_F^2 \right\}$. Thus, reclustering changes the antecedent mapping and the corresponding design matrix, while batch recomputation restores the globally optimal ridge-regularized solution for the updated fuzzy representation. After reclustering, the classifier resumes the standard OA-TSK incremental–decremental update protocol until the reclustering condition is satisfied again.

Figure 1 illustrates the overall architecture of OA-A-TSK. The main data flow consists of fuzzification, sliding-window maintenance, incremental and decremental state updates, and prediction. For each incoming input vector $\mathbf{x}_t$, the Gaussian membership functions characterized by $\{\mathbf{c}^k, \sigma^k\}$ map $\mathbf{x}_t$ to the normalized firing-strength vector $\mathbf{g}_t \in \mathbb{R}^{1 \times K}$, where $\mathbf{c}^k = [c_1^k, \dots, c_d^k]^T$ and $\sigma^k = [\sigma_1^k, \dots, \sigma_d^k]^T$. After prediction, the revealed class label is represented by the one-hot vector $\mathbf{y}_t \in \{0, 1\}^{1 \times C}$. The Sliding Window stores the normalized firing-strength matrix $\mathbf{G}_t \in \mathbb{R}^{W \times K}$ and the label matrix $\mathbf{Y}_t \in \mathbb{R}^{W \times C}$ when the window is full. The Update Module performs the incremental and decremental updates through the $SM(+)$ and $SM(-)$ blocks, respectively, where $SM(+)$ and $SM(-)$ denote the Sherman–Morrison update and dwnupdate operations. If $N_{tmp} > W$, the $SM(-)$ block removes the oldest sample according to the FIFO policy. The block containing $\mathbf{P}_t$ and $\mathbf{A}_t$ stores the inverse of the regularized Gram matrix and the consequent parameter matrix after each update. At time t , the double-line solid feedback arrow from this block to the $SM(+)$ block provides $(\mathbf{P}_{t-1}, \mathbf{A}_{t-1})$ for the Phase I incremental update. The Prediction Module computes the output-score vector and predicted class label as $\mathbf{o}_t = \mathbf{g}_t \mathbf{A}_{t-1}$, $\hat{y}_t = \arg \max_{m \in \{1, \dots, C\}} o_{tm}$.

At each monitoring event, which occurs every L online samples, the Imbalance Monitor evaluates $\rho_{inactive}$ and CV using the normalized firing strengths contained in $\mathbf{G}_t$. When $\rho_{inactive} > 0.2$ or $CV > 1.5$, the dashed branch triggers k-means reclustering. The rule centers $\{\mathbf{c}^k\}_{k=1}^K$ and $\mathbf{G}_t$ are updated, after which $\mathbf{P}_t$ and $\mathbf{A}_t$ are recomputed according to (49) and (50). If neither imbalance condition is satisfied, OA-A-TSK continues with the standard OA-TSK incremental–decremental update protocol.

4.3. Complexity Analysis

Let K , C , and d denote the numbers of fuzzy rules, output classes, and input features, respectively; let W denote the window capacity; and let $N_t \leq W$ denote the number of active samples at time t . For each incoming sample, Gaussian fuzzification requires $O(dK)$ operations. Updating the inverse of the regularized Gram matrix through the Sherman–Morrison formula requires $O(K^2)$ operations, while updating the consequent parameter matrix requires $O(KC)$ operations. When the window is full, the subsequent decremental update has the same asymptotic cost. Therefore, the per-sample time complexity of OA-TSK is $O(dK + K^2 + KC)$, or $O(K^2 + KC)$ after the normalized firing-strength vector has been computed. In comparison, ridge-regularized batch retraining on the active window requires $O(N_t K^2 + N_t KC + K^3 + K^2 C)$ operations to form the regularized Gram matrix $\mathbf{G}_t^T \mathbf{G}_t + \lambda \mathbf{I}_K$ and the cross-product matrix $\mathbf{G}_t^T \mathbf{Y}_t$, factorize the regularized Gram matrix, and obtain the consequent parameters. When C is treated as fixed, this reduces to approximately $O(N_t K^2 + K^3)$.

OA-TSK stores the inverse of the regularized Gram matrix $\mathbf{P}_t$, the consequent parameter matrix $\mathbf{A}_t$, $\mathbf{G}_t$ and $\mathbf{Y}_t$, and the antecedent parameters. Its space complexity is therefore $O(K^2 + KC + W(K + C) + dK)$. This storage is bounded by W and is independent of the total number of samples processed. OA-A-TSK additionally maintains the feature buffer $\mathbf{X}_t \in \mathbb{R}^{N_t \times d}$, introducing $O(Wd)$ additional storage.

For OA-A-TSK, direct evaluation of the monitoring statistics requires $O(WK)$ operations per monitoring event, corresponding to an amortized cost of $O(WK/L)$ per sample.

If reclustering is triggered, k-means with I iterations costs $O(IWKd)$, while recomputing $\mathbf{G}_t$, $\mathbf{P}_t$, and $\mathbf{A}_t$ costs $O(WKd + WK^2 + WKC + K^3 + K^2C)$. Hence, the total cost of a reclustering event is $O(IWKd + WK^2 + WKC + K^3 + K^2C)$. Because reclustering is performed only when a severe imbalance among the mean normalized firing strengths of the rules is detected, the additional reclustering cost is incurred intermittently rather than at every online update.

Finally, classifying one sample requires $O(dK)$ operations for fuzzification, $O(KC)$ operations for computing the output scores, and $O(C)$ comparisons for class selection. Thus, the prediction complexity is $O(dK + KC)$, which is identical to that of a batch-trained TSK classifier with the same rule structure.

5. Experimental Evaluation

This section evaluates the proposed OA-TSK and OA-A-TSK on synthetic and real-world data streams. The experiments are designed to assess classification accuracy, computational efficiency, numerical equivalence to batch retraining, robustness under different drift patterns, and local sensitivity to hyperparameter settings.

5.1. Experimental Setup

Unless otherwise stated, the settings in this subsection apply to the main comparative experiments; the configurations for the local sensitivity analysis are specified in Section 5.6. Six benchmark data streams are used, including three synthetic streams with controlled concept drift and three real-world streams. The synthetic streams are 100,000-sample variants produced by the SEA stream generator [22], the Sine generator [33], and the Hyperplane generator [34], with 10% class noise added in our experiments. Three drift intervals, denoted by N_d , are considered: $N_d \in \{1000, 10,000, 50,000\}$ samples. In the synthetic stream experiments, SEA uses abrupt concept drift induced by switching the SEA classification threshold, Sine uses gradual nonlinear drift, and Hyperplane uses incremental concept drift generated by a rotating hyperplane. The three real-world streams are constructed from the Electricity [35], Covertype [36], and GasDrift [37] datasets obtained from OpenML [38]. For Covertype, we used the binary OpenML version and retained the first 200,000 instances in their original order for stream evaluation.

The synthetic streams contain numerical input features and were selected to evaluate adaptation under controlled abrupt, gradual, and incremental drift, respectively. Electricity contains temporal and electricity-market attributes, including prices, demands, and scheduled transfers, and provides a real-world binary classification task. Covertype contains cartographic attributes, including elevation, slope, and wilderness-area and soil-type indicators, and was selected to evaluate performance on a larger stream with mixed numerical and binary features. GasDrift contains 128 features extracted from chemical sensor responses for six gas classes and was selected to evaluate adaptation to sensor drift in a high-dimensional multiclass setting. The real-world datasets are evaluated by presenting their records sequentially under the same online evaluation protocol. The characteristics of the six data streams are summarized in Table 1. We compare OA-TSK and OA-A-TSK with six representative baselines: forgetting parameters extreme learning machine (FP-ELM) [39], Hoeffding Adaptive Tree (HAT) [40], TSK-Batch, TSK-A-Batch, adaptive random forest (ARF) [41], and O-DIA-TSK [21]. TSK-Batch performs ridge-regularized batch retraining on the active window, whereas TSK-A-Batch additionally applies antecedent reclustering using the same monitoring criteria as OA-A-TSK. O-DIA-TSK provides an incremental-only reference without sliding-window sample removal, while ARF extends the comparison to an ensemble method for evolving data streams. OA-TSK, OA-A-TSK, TSK-Batch, and TSK-A-Batch use the same K , δ and W . For the regularization parameter λ , although grid

search is a widely adopted method, we found that, for Gaussian membership function values computed from normalized sample features, $\lambda = 10^{-6}$ already works well across all datasets. For these four methods, k-means was run with an independent random initialization for each method and each repetition. Therefore, OA-TSK and TSK-Batch were not constrained to share identical rule centers in these experiments.

Table 1. Data Stream Characteristics.

#	Data Stream	No. of Samples	No. of Features	No. of Classes
1	SEA	100,000	3	2
2	Sine	100,000	2	2
3	Hyperplane	100,000	10	2
4	Electricity	45,312	8	2
5	Covertime	200,000	54	2
6	GasDrift	13,910	128	6

The hyperparameter settings of FP-ELM, HAT, and ARF were adopted from the recommendations in their respective references [39–41] and kept fixed during evaluation. For the TSK-based methods, K , δ , and W were selected by grid search in validation experiments, with the selected configurations reported in Table 2. O-DIA-TSK used the same K , δ , and ridge parameter as OA-TSK, while W determined only its initialization size. For the TSK-based methods, the search spaces were $K \in \{15, 20, 25, 30, 35, 40\}$ and $\delta \in \{1.5, 2.0, 2.5, 3.0, 3.5\}$. The search space for W was $\{100, 200, 500, 600, 1000, 1500, 2000, 2500, 3000, 5000\}$. For each stream setting, the first W samples were used as a common initialization block for all methods. For the TSK-based methods, this block constituted the initial training set $\mathcal{D}_0$, so that $N_0 = W$. The samples in the initialization block were excluded from the reported prequential accuracies, and evaluation began with the next sample. All comparison models were initialized and trained on the data streams used in this study. FP-ELM, HAT, and ARF used the first W labeled samples for initialization and subsequently performed online updates after each prediction and label revelation. TSK-Batch and TSK-A-Batch obtained their initial consequent parameters by ridge-regularized least squares and recomputed them on the updated active window after each labeled sample. TSK-Batch retained fixed rule centers, whereas TSK-A-Batch additionally performed reclustering when the monitoring criteria were satisfied. For OA-A-TSK and TSK-A-Batch, the threshold on the mean normalized firing strength used to define an inactive rule, the threshold on the proportion of inactive rules, and the coefficient-of-variation threshold were fixed at $0.1/K$, 0.2 , and 1.5 , respectively. The monitoring intervals for SEA, Sine, and Hyperplane were 500/5000/10000, 500/2000/5000, and 1000/3000/8000 samples for drift intervals of 1K/10K/50K, respectively; those for Electricity, Covertime, and GasDrift were 5000, 10,000, and 2000 samples, respectively. For FP-ELM, a sigmoid activation function and a regularization coefficient of $\lambda_{FP} = 0.02$ were used. For SEA, Sine, and Hyperplane, the selected numbers of hidden nodes were 20, 25, and 30, respectively. The corresponding values of the forgetting parameter were 0.950/0.980/0.995, 0.960/0.987/0.997, and 0.970/0.990/0.9975, respectively, where the three values in each group correspond to drift intervals of 1K/10K/50K. For Electricity, Covertime, and GasDrift, the selected (number of hidden nodes, forgetting parameter) settings were (30,0.9975), (35,0.998), and (35,0.998), respectively.

For HAT, the selected grace period, split-confidence parameter, and tie-breaking threshold were 200, $\delta_{HAT} = 10^{-7}$, and $\tau = 0.05$, respectively. Adaptive Naive Bayes was used for leaf prediction, with no maximum-depth constraint. ARF was implemented using the River library with 10 trees and a resampling parameter of 6. At each split, the number

of randomly selected candidate features was set to approximately the square root of the total number of input features. ADWIN was used for drift and warning detection with confidence parameters of 0.001 and 0.01, respectively. The grace period was 50, the split-confidence parameter was 0.01, and the tie-breaking threshold was 0.05. Adaptive Naive Bayes was used for leaf prediction. This configuration was fixed across all stream settings. For comparison with OA-TSK, we used a ridge-regularized implementation of O-DIA-TSK with the same K and δ values listed in Table 2 and $\lambda = 10^{-6}$. Its rule centers were initialized by k-means on the first $N_0 = W$ samples and remained fixed thereafter. O-DIA-TSK performed only incremental updates, retaining the contributions of all observed samples without sample removal or antecedent reclustering. Here, W determines the initialization size only and does not impose a training-window limit on O-DIA-TSK.

Table 2. Selected Hyperparameter Configurations for the TSK-Based Methods.

#	Data Stream	Drift Interval	K	δ	W
1	SEA	1K/10K/50K	20	2.0	200/1500/3000
2	Sine	1K/10K/50K	25	2.5	200/500/1000
3	Hyperplane	1K/10K/50K	30	3.0	600/1500/2500
4	Electricity	-	30	2.0	1000
5	Covertypes	-	35	2.5	1500
6	GasDrift	-	35	3.0	1500

All methods were evaluated under the prequential test-then-train protocol [42]. After initialization, samples are processed one at a time, with prediction performed before the corresponding label is used for learning. For OA-TSK, OA-A-TSK, TSK-Batch, and TSK-A-Batch, the active training window contains the most recent W samples and advances by one sample at each online step. Thus, W specifies the training-window capacity rather than the interval between updates. The drift interval N_d controls the drift schedule of the synthetic generators, whereas the monitoring interval L specifies how often the reclustering criteria are evaluated. These quantities are measured in numbers of samples. The 500-prediction moving window is used only to evaluate recent classification accuracy and is separate from the training window. Feature standardization was determined from the initialization block and subsequently fixed for the remaining stream. Moving-window accuracy was computed over the 500 most recent predictions, whereas global accuracy was computed over all streaming samples evaluated after initialization. Macro-F1, defined as the unweighted mean of the class-wise F1 scores, was additionally reported to assess classification performance across classes. For each run, macro-F1 was computed from all post-initialization prequential predictions. Each experiment was independently repeated ten times. The results are reported as the mean $\pm$ standard deviation over the ten runs. All methods were evaluated on the same computing platform equipped with an Intel Core i5-10300H CPU and 16 GB RAM, using Python 3.8. The reported per-sample processing time includes prediction and online model updating, but excludes data loading, initialization, metric calculation, and file operations.

5.2. Classification Performance

Table 3 reports the classification accuracy of all compared methods. Each entry presents the final moving-window accuracy in the first row and the global accuracy in the second row, expressed as the mean $\pm$ standard deviation over ten runs. The best global accuracy for each stream setting is highlighted in bold. Because the rule centers were initialized independently, OA-TSK and TSK-Batch are not expected to produce identical accuracies in Table 3. Their numerical equivalence under an identical antecedent mapping is examined separately in Section 5.5.

Table 3. Classification Accuracy Comparison.

Method	SEA	SEA	SEA	Sine	Sine	Sine	Hyperplane	Hyperplane	Hyperplane	Electricity	Coverttype	GasDrift
Drift Interval	1K	10K	50K	1K	10K	50K	1K	10K	50K	-	-	-
OA-TSK	0.870 ± 0.007	0.876 ± 0.015	0.880 ± 0.014	0.731 ± 0.025	0.849 ± 0.021	0.876 ± 0.010	0.816 ± 0.017	0.831 ± 0.013	0.832 ± 0.014	0.556 ± 0.000	0.274 ± 0.000	0.859 ± 0.023
	0.859 ± 0.002	0.876 ± 0.001	0.883 ± 0.001	0.716 ± 0.002	0.857 ± 0.001	0.873 ± 0.001	0.836 ± 0.008	0.849 ± 0.002	0.853 ± 0.003	0.575 ± 0.001	0.407 ± 0.000	0.764 ± 0.023
OA-A-TSK	0.883 ± 0.007	0.882 ± 0.007	0.876 ± 0.014	0.742 ± 0.019	0.705 ± 0.012	0.832 ± 0.007	0.865 ± 0.036	0.863 ± 0.037	0.862 ± 0.037	0.893 ± 0.013	0.736 ± 0.003	0.870 ± 0.020
	0.850 ± 0.001	0.870 ± 0.001	0.885 ± 0.001	0.747 ± 0.002	0.719 ± 0.001	0.844 ± 0.001	0.865 ± 0.020	0.864 ± 0.022	0.862 ± 0.016	0.756 ± 0.001	0.817 ± 0.002	0.824 ± 0.008
FP-ELM	0.796 ± 0.014	0.822 ± 0.028	0.848 ± 0.011	0.765 ± 0.015	0.804 ± 0.015	0.820 ± 0.012	0.734 ± 0.022	0.772 ± 0.011	0.807 ± 0.011	0.814 ± 0.024	0.747 ± 0.003	0.979 ± 0.006
	0.787 ± 0.005	0.826 ± 0.005	0.856 ± 0.005	0.764 ± 0.004	0.798 ± 0.009	0.825 ± 0.010	0.738 ± 0.020	0.778 ± 0.018	0.821 ± 0.008	0.733 ± 0.007	0.810 ± 0.004	0.969 ± 0.003
HAT	0.888 ± 0.012	0.882 ± 0.016	0.882 ± 0.011	0.756 ± 0.013	0.775 ± 0.030	0.777 ± 0.033	0.840 ± 0.014	0.837 ± 0.008	0.836 ± 0.009	0.849 ± 0.049	0.747 ± 0.019	0.240 ± 0.000
	0.850 ± 0.000	0.856 ± 0.001	0.876 ± 0.003	0.745 ± 0.002	0.772 ± 0.007	0.817 ± 0.006	0.848 ± 0.013	0.847 ± 0.014	0.847 ± 0.013	0.817 ± 0.005	0.840 ± 0.006	0.550 ± 0.002
TSK-Batch	0.872 ± 0.010	0.871 ± 0.011	0.868 ± 0.010	0.746 ± 0.012	0.841 ± 0.019	0.851 ± 0.014	0.828 ± 0.014	0.825 ± 0.013	0.828 ± 0.012	0.556 ± 0.000	0.499 ± 0.108	0.887 ± 0.020
	0.851 ± 0.001	0.876 ± 0.001	0.878 ± 0.002	0.746 ± 0.001	0.838 ± 0.001	0.850 ± 0.001	0.844 ± 0.004	0.844 ± 0.004	0.845 ± 0.004	0.572 ± 0.001	0.617 ± 0.016	0.801 ± 0.017
TSK-A-Batch	0.871 ± 0.010	0.869 ± 0.011	0.870 ± 0.010	0.750 ± 0.012	0.852 ± 0.019	0.849 ± 0.014	0.830 ± 0.014	0.821 ± 0.013	0.827 ± 0.012	0.883 ± 0.011	0.735 ± 0.005	0.993 ± 0.001
	0.853 ± 0.002	0.874 ± 0.001	0.880 ± 0.002	0.732 ± 0.001	0.840 ± 0.001	0.849 ± 0.001	0.856 ± 0.004	0.834 ± 0.004	0.850 ± 0.004	0.759 ± 0.003	0.814 ± 0.001	0.921 ± 0.003
ARF	0.883 ± 0.012	0.889 ± 0.014	0.890 ± 0.010	0.723 ± 0.018	0.800 ± 0.022	0.835 ± 0.020	0.830 ± 0.035	0.833 ± 0.032	0.825 ± 0.037	0.922 ± 0.009	0.793 ± 0.013	0.973 ± 0.008
	0.852 ± 0.000	0.875 ± 0.001	0.881 ± 0.001	0.725 ± 0.002	0.806 ± 0.004	0.850 ± 0.002	0.838 ± 0.032	0.838 ± 0.032	0.838 ± 0.031	0.782 ± 0.002	0.834 ± 0.001	0.957 ± 0.002
O-DIA-TSK	0.887 ± 0.010	0.888 ± 0.012	0.865 ± 0.016	0.750 ± 0.013	0.742 ± 0.021	0.742 ± 0.021	0.849 ± 0.026	0.845 ± 0.031	0.843 ± 0.032	0.638 ± 0.106	0.726 ± 0.002	0.457 ± 0.031
	0.853 ± 0.001	0.857 ± 0.001	0.870 ± 0.002	0.747 ± 0.001	0.747 ± 0.001	0.750 ± 0.001	0.831 ± 0.016	0.830 ± 0.017	0.849 ± 0.017	0.638 ± 0.057	0.494 ± 0.007	0.697 ± 0.007

Table 4. Comparison of Macro-F1 and Per-Sample Processing Time.

Method	SEA	SEA	SEA	Sine	Sine	Sine	Hyperplane	Hyperplane	Hyperplane	Electricity	Coverttype	GasDrift
Drift Interval	1K	10K	50K	1K	10K	50K	1K	10K	50K	-	-	-
OA-TSK	0.831 ± 0.001	0.862 ± 0.001	0.877 ± 0.001	0.698 ± 0.001	0.841 ± 0.002	0.850 ± 0.001	0.789 ± 0.068	0.787 ± 0.083	0.789 ± 0.087	0.530 ± 0.061	0.393 ± 0.001	0.726 ± 0.003
	0.276 ± 0.065	0.180 ± 0.042	0.284 ± 0.046	0.277 ± 0.046	0.275 ± 0.018	0.260 ± 0.031	0.216 ± 0.028	0.282 ± 0.049	0.261 ± 0.041	0.212 ± 0.058	0.202 ± 0.064	0.438 ± 0.070
OA-A-TSK	0.833 ± 0.002	0.863 ± 0.001	0.877 ± 0.001	0.698 ± 0.001	0.701 ± 0.001	0.840 ± 0.001	0.769 ± 0.066	0.778 ± 0.081	0.781 ± 0.087	0.671 ± 0.006	0.810 ± 0.001	0.810 ± 0.016
	0.335 ± 0.047	0.414 ± 0.009	0.305 ± 0.005	0.311 ± 0.020	0.292 ± 0.014	0.404 ± 0.018	1.590 ± 0.011	1.568 ± 0.024	1.576 ± 0.007	0.660 ± 0.006	2.984 ± 0.018	2.196 ± 0.005
FP-ELM	0.764 ± 0.005	0.806 ± 0.005	0.846 ± 0.005	0.764 ± 0.004	0.798 ± 0.009	0.825 ± 0.010	0.666 ± 0.060	0.701 ± 0.069	0.741 ± 0.087	0.569 ± 0.012	0.799 ± 0.003	0.966 ± 0.004
	0.203 ± 0.029	0.206 ± 0.036	0.261 ± 0.046	0.229 ± 0.030	0.225 ± 0.040	0.223 ± 0.040	0.281 ± 0.031	0.277 ± 0.039	0.275 ± 0.029	0.276 ± 0.057	0.314 ± 0.034	0.321 ± 0.088
HAT	0.833 ± 0.001	0.841 ± 0.001	0.868 ± 0.002	0.745 ± 0.002	0.763 ± 0.005	0.811 ± 0.003	0.779 ± 0.051	0.779 ± 0.050	0.779 ± 0.050	0.809 ± 0.006	0.828 ± 0.009	0.544 ± 0.002
	0.263 ± 0.029	0.226 ± 0.028	0.247 ± 0.040	0.235 ± 0.063	0.234 ± 0.031	0.219 ± 0.074	0.381 ± 0.078	0.378 ± 0.066	0.373 ± 0.052	0.313 ± 0.037	1.018 ± 0.239	5.979 ± 0.251
TSK-Batch	0.832 ± 0.001	0.862 ± 0.001	0.877 ± 0.002	0.738 ± 0.001	0.812 ± 0.002	0.850 ± 0.001	0.770 ± 0.067	0.777 ± 0.082	0.785 ± 0.078	0.561 ± 0.006	0.594 ± 0.000	0.790 ± 0.007
	2.217 ± 0.072	2.194 ± 0.144	2.176 ± 0.026	2.400 ± 0.021	2.371 ± 0.042	2.368 ± 0.047	4.456 ± 0.033	4.470 ± 0.063	4.456 ± 0.032	3.438 ± 0.112	14.222 ± 0.119	64.576 ± 2.813
TSK-A-Batch	0.833 ± 0.001	0.863 ± 0.001	0.878 ± 0.001	0.730 ± 0.002	0.840 ± 0.002	0.847 ± 0.001	0.762 ± 0.075	0.777 ± 0.085	0.779 ± 0.085	0.670 ± 0.006	0.810 ± 0.001	0.808 ± 0.011
	2.449 ± 0.058	2.431 ± 0.052	2.388 ± 0.080	2.365 ± 0.032	2.305 ± 0.058	2.423 ± 0.071	4.532 ± 0.309	4.906 ± 0.020	4.802 ± 0.078	4.741 ± 0.067	25.825 ± 2.878	114.604 ± 11.936
ARF	0.835 ± 0.000	0.859 ± 0.001	0.881 ± 0.001	0.725 ± 0.002	0.806 ± 0.004	0.850 ± 0.002	0.660 ± 0.122	0.660 ± 0.123	0.659 ± 0.124	0.778 ± 0.002	0.824 ± 0.002	0.956 ± 0.002
	3.545 ± 0.203	2.573 ± 0.057	2.759 ± 0.058	19.198 ± 2.010	3.641 ± 1.434	4.672 ± 0.188	3.378 ± 0.284	3.348 ± 0.352	3.221 ± 0.261	2.332 ± 0.046	3.374 ± 0.088	4.186 ± 0.085
O-DIA-TSK	0.836 ± 0.001	0.841 ± 0.001	0.860 ± 0.002	0.747 ± 0.001	0.747 ± 0.001	0.750 ± 0.001	0.700 ± 0.142	0.689 ± 0.155	0.686 ± 0.155	0.593 ± 0.081	0.424 ± 0.005	0.660 ± 0.009
	0.133 ± 0.001	0.134 ± 0.001	0.136 ± 0.006	0.137 ± 0.001	0.144 ± 0.007	0.153 ± 0.003	0.156 ± 0.001	0.156 ± 0.002	0.142 ± 0.001	0.162 ± 0.016	0.159 ± 0.004	0.173 ± 0.002

The first row of each entry reports macro-F1, and the second row reports the average processing time per sample in milliseconds. Values are presented as mean ± standard deviation over ten runs. The highest mean macro-F1 for each stream setting is highlighted in bold.

On the synthetic streams, OA-TSK achieves accuracy comparable to TSK-Batch while avoiding repeated batch retraining. On the real-world streams, fixed rule centers become less effective, especially for high-dimensional data streams. Compared with O-DIA-TSK, OA-TSK achieves higher global accuracy in eight of the nine synthetic stream settings, with Sine at $N_d = 1000$ being the exception. However, O-DIA-TSK achieves higher global accuracy on Electricity and Covertypes. HAT obtains the highest global accuracy and macro-F1 on Electricity and Covertypes. Table 4 provides complementary macro-F1 results. OA-A-TSK substantially improves over OA-TSK on Electricity and Covertypes by realigning the rule centers with the current input distribution. On GasDrift, TSK-A-Batch achieves the highest moving-window accuracy, whereas FP-ELM achieves the highest global accuracy. OA-A-TSK nevertheless achieves higher moving-window and global accuracies than HAT while preserving the interpretable TSK rule structure.

5.3. Computational Efficiency

The second row of each entry in Table 4 reports the average processing time per sample. O-DIA-TSK is the fastest method across all evaluated settings, with processing times of 0.133–0.173 ms per sample. OA-TSK requires 0.180–0.438 ms per sample and is faster than ARF and both batch TSK baselines in all reported settings. Its processing time is comparable to that of FP-ELM. The difference between O-DIA-TSK and OA-TSK reflects the additional work associated with maintaining and updating the active window.

OA-A-TSK incurs additional cost from monitoring the mean normalized firing strengths and performing k-means reclustering. This overhead is relatively small on SEA and Sine and becomes more pronounced on Hyperplane, Covertypes, and GasDrift. Nevertheless, reclustering is triggered only when a severe imbalance among the mean normalized firing strengths of the rules is detected, so the additional cost is amortized over multiple online updates. These results are consistent with the complexity analysis in Section 4.3. ARF requires approximately 3.53, 1.13, and 1.91 times the average per-sample processing time of OA-A-TSK on Electricity, Covertypes, and GasDrift, respectively.

5.4. Moving-Window Accuracy Evolution

Figures 2 and 3 show the moving-window accuracy curves on synthetic and real-world streams, respectively.

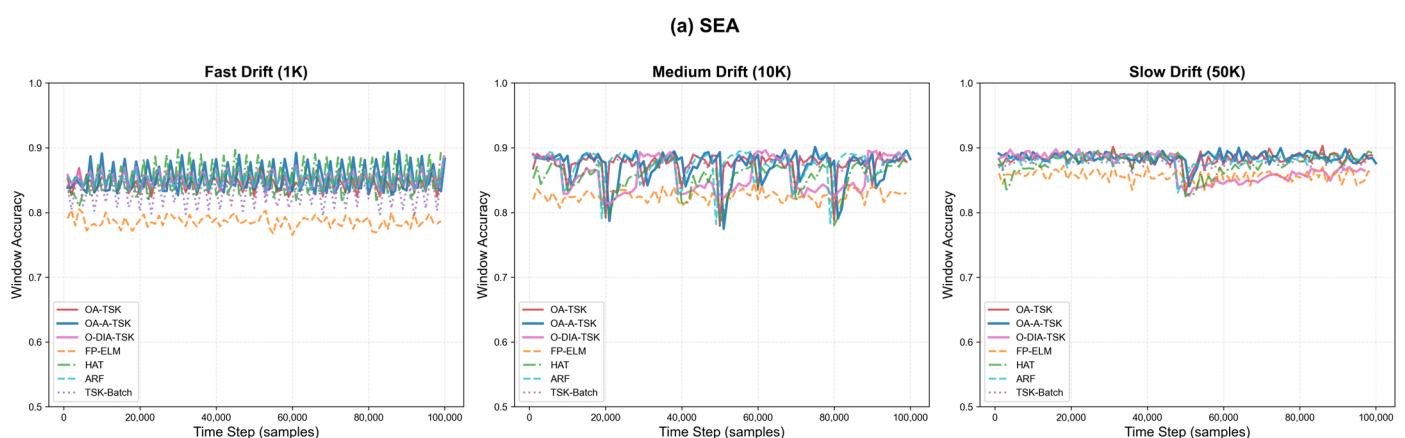


Figure 2. Cont.

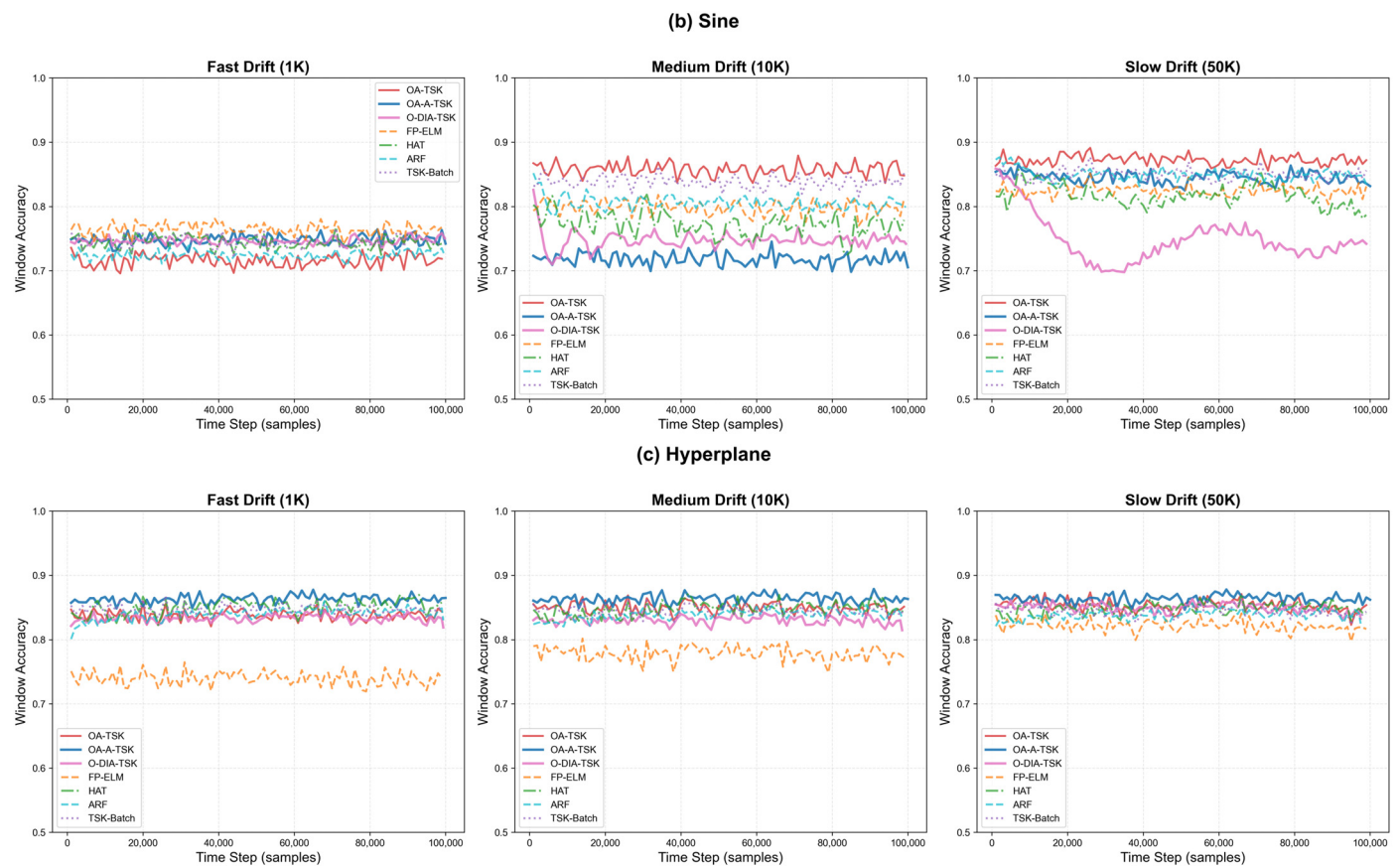


Figure 2. Moving-window accuracy evolution on synthetic data streams under varying drift intervals ($N_d \in \{1K, 10K, 50K\}$): (a) SEA, (b) Sine, and (c) Hyperplane.

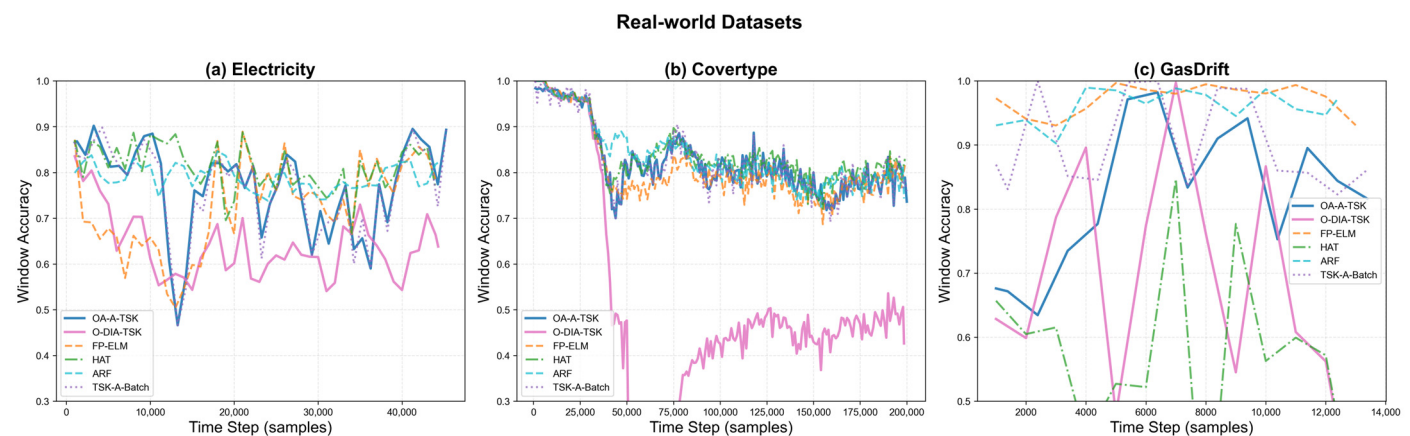


Figure 3. Moving-window accuracy evolution on real-world data streams: (a) Electricity, (b) Covertype, and (c) GasDrift. The compared methods are OA-A-TSK, O-DIA-TSK, FP-ELM, HAT, ARF, and TSK-A-Batch.

On the synthetic streams, OA-TSK and OA-A-TSK show competitive moving-window accuracy under the evaluated drift settings. On SEA, OA-TSK closely follows TSK-Batch, while OA-A-TSK maintains an advantage over OA-TSK on Hyperplane. On the real-world streams, OA-A-TSK tracks TSK-A-Batch closely on Electricity and Covertype and maintains higher moving-window accuracy than HAT on GasDrift. Although ARF generally exhibits higher moving-window accuracy and smoother trajectories than OA-A-TSK on the real-world streams, OA-A-TSK requires less processing time per sample on all three streams (Table 4) while retaining an explicit TSK rule structure.

5.5. Numerical Equivalence and Ranking Analysis

To verify the theoretical equivalence between OA-TSK and ridge-regularized batch retraining, we conduct a numerical validation on the SEA stream. Unlike the general performance comparison in Table 3, this numerical validation used exactly the same initialization block, feature standardization parameters, rule centers, Gaussian widths, active windows, and ridge parameter for OA-TSK and batch retraining. Consequently, both methods operated on identical normalized firing-strength matrices. For the SEA stream setting with $N_d = 1000$ and $W = 200$, comparisons are made at stream indices $t \in \{1000, 5000, 10,000, 50,000, 100,000\}$. Here, t denotes the absolute sample index in the original stream, and the first W samples constitute the initialization block. Therefore, an absolute stream index t corresponds to online time step $t - W$ in Algorithm 3. At each checkpoint, the matrices $\mathbf{P}_t$ and $\mathbf{A}_t$ maintained by OA-TSK are compared with the corresponding batch quantities recomputed on the same active window: $\mathbf{P}_t^{\text{batch}} = (\mathbf{G}_t^T \mathbf{G}_t + \lambda \mathbf{I}_K)^{-1}$, $\mathbf{A}_t^{\text{batch}} = \mathbf{P}_t^{\text{batch}} \mathbf{G}_t^T \mathbf{Y}_t$. Define $\text{RE}_A(t) = \frac{\|\mathbf{A}_t - \mathbf{A}_t^{\text{batch}}\|_F}{\|\mathbf{A}_t^{\text{batch}}\|_F}$, $\text{RE}_P(t) = \frac{\|\mathbf{P}_t - \mathbf{P}_t^{\text{batch}}\|_F}{\|\mathbf{P}_t^{\text{batch}}\|_F}$. The comparison includes the absolute and relative Frobenius norms of the errors in $\mathbf{A}_t$ and $\mathbf{P}_t$, the mean squared error (MSE) between the output-score matrices, and the prediction agreement (PA) on the active window.

As shown in Table 5, the relative errors of $\mathbf{A}_t$ range from 1.00×10^{-10} to 1.27×10^{-8} , while those of $\mathbf{P}_t$ range from 3.70×10^{-11} to 9.26×10^{-11} . Prediction agreement is 100% at all five checkpoints. These results support close numerical agreement between OA-TSK and ridge-regularized batch retraining under the tested shared-antecedent setting.

Table 5. Numerical Equivalence Between OA-TSK and Ridge-Regularized Batch Retraining on SEA.

t	$\ \mathbf{A}_t - \mathbf{A}_t^{\text{batch}}\ _F$	RE_A	$\ \mathbf{P}_t - \mathbf{P}_t^{\text{batch}}\ _F$	RE_P	MSE	PA
1000	5.41×10^{-8}	1.00×10^{-10}	9.98×10^{-6}	9.26×10^{-11}	2.20×10^{-20}	100%
5000	2.61×10^{-7}	5.88×10^{-10}	3.81×10^{-6}	5.06×10^{-11}	1.33×10^{-19}	100%
10,000	3.92×10^{-7}	9.78×10^{-10}	5.72×10^{-6}	6.38×10^{-11}	1.20×10^{-18}	100%
50,000	2.51×10^{-6}	9.29×10^{-9}	2.87×10^{-6}	3.70×10^{-11}	3.75×10^{-18}	100%
100,000	3.08×10^{-6}	1.27×10^{-8}	5.77×10^{-6}	7.32×10^{-11}	1.62×10^{-17}	100%

We further perform a ranking analysis of all eight methods over the nine synthetic stream settings formed by three synthetic streams and three drift intervals. Following the ranking procedure in [43], methods are ranked by mean global accuracy within each setting, with rank 1 assigned to the highest accuracy and average ranks assigned to ties. The average ranks over the nine settings are reported in Table 6. The average ranks over the nine settings are reported in Table 6 where the lowest (best) average rank is highlighted in bold.

Table 6. Average Ranks Over Nine Synthetic Stream Settings.

Method	OA-TSK	OA-A-TSK	TSK-Batch	TSK-A-Batch	HAT	FP-ELM	ARF	O-DIA-TSK
Rank	2.72	3.44	3.89	3.72	5.39	6.67	4.50	5.67

OA-TSK obtains the best average rank, followed by OA-A-TSK in second place and TSK-A-Batch in third place, while TSK-Batch ranks fourth. Thus, the two proposed methods occupy the top two positions over the nine synthetic stream settings.

5.6. Local Hyperparameter Sensitivity

We assess the local sensitivity of OA-A-TSK using independently generated SEA, Sine, and Hyperplane validation streams, each containing 20,000 samples. This length corresponds to 20% of the respective 100,000-sample stream used in the main experiments; the validation streams are generated separately rather than extracted from those streams. The drift intervals are 1000, 10,000, and 10,000 samples, respectively. Six parameters are varied one at a time: window capacity W , monitoring interval L , inactive-rule coefficient a , inactive-proportion threshold b , coefficient-of-variation threshold c , and ridge regularization parameter λ . A supplementary assessment uses the complete GasDrift stream, containing 13,910 samples and 128 features, to examine a , b , c , and λ . This assessment is used for sensitivity analysis and does not retune the reference configuration.

The reference window capacities for SEA, Sine, Hyperplane, and GasDrift are 200, 500, 1500, and 1500 samples, respectively; the corresponding monitoring intervals are 500, 2000, 3000, and 2000 samples. The reference values are $a = 0.1$, $b = 0.2$, $c = 1.5$, and $\lambda = 10^{-6}$, consistent with the main experiments. Here, a determines the inactive-rule threshold a/K , where K is the number of rules. The denominator tolerance is fixed at $\varepsilon = 10^{-8}$ throughout this analysis. The window capacity and monitoring interval are varied over $W \in \{0.5W_{\text{ref}}, W_{\text{ref}}, 2W_{\text{ref}}\}$ and $L \in \{0.5L_{\text{ref}}, L_{\text{ref}}, 2L_{\text{ref}}\}$, respectively, where W_{ref} and L_{ref} denote the reference values for each stream. The remaining parameters are varied over $a \in \{0.05, 0.1, 0.2\}$, $b \in \{0.1, 0.2, 0.3\}$, $c \in \{1.0, 1.5, 2.0\}$, and $\lambda \in \{10^{-7}, 10^{-6}, 10^{-5}\}$. The values of K and δ are retained from Table 2 for each corresponding stream setting. All other settings retain their reference values unless otherwise specified in this subsection.

Each configuration is evaluated over ten runs, with random seeds matched across parameter levels. For each synthetic stream, the same generated sample sequence is used across levels within a run. GasDrift retains its original sample order, with seeds controlling stochastic model initialization. To compare window capacities over a common evaluation interval, window scans use a shared initialization block of $N_0 = 0.5W_{\text{ref}}$ samples and begin scoring after sample $2W_{\text{ref}}$. Online updates continue between initialization and the start of scoring. Other scans use $N_0 = W_{\text{ref}}$, with scoring beginning immediately after initialization. These initialization and scoring choices apply specifically to this local analysis. Feature standardization is estimated from the corresponding initialization block and remains fixed thereafter. Each run is scored by the proportion of correct prequential predictions over its entire scoring interval. Figure 4 reports the mean and sample standard deviation across the ten runs. The experiments use the same computing platform as the main evaluation.

The monitoring thresholds define criteria for antecedent adaptation, although input dimensionality can still influence rule activations. Within each configuration, the positive ridge coefficient remains fixed, while the consequent weights continue to change as labeled samples arrive and the active window is updated. Thus, varying λ compares different regularization strengths without changing it during an individual run.

Figure 4 shows that the mean prequential accuracy varies modestly across most tested parameter levels, although the response depends on the data stream. More noticeable changes are observed for the window capacity on SEA, the inactive-proportion threshold on GasDrift, and the ridge coefficient on Sine. Within the tested ranges, increasing W improves the mean accuracy on SEA, increasing b improves that on GasDrift, and increasing λ reduces that on Sine. These observations characterize local sensitivity under one-at-a-time parameter changes.

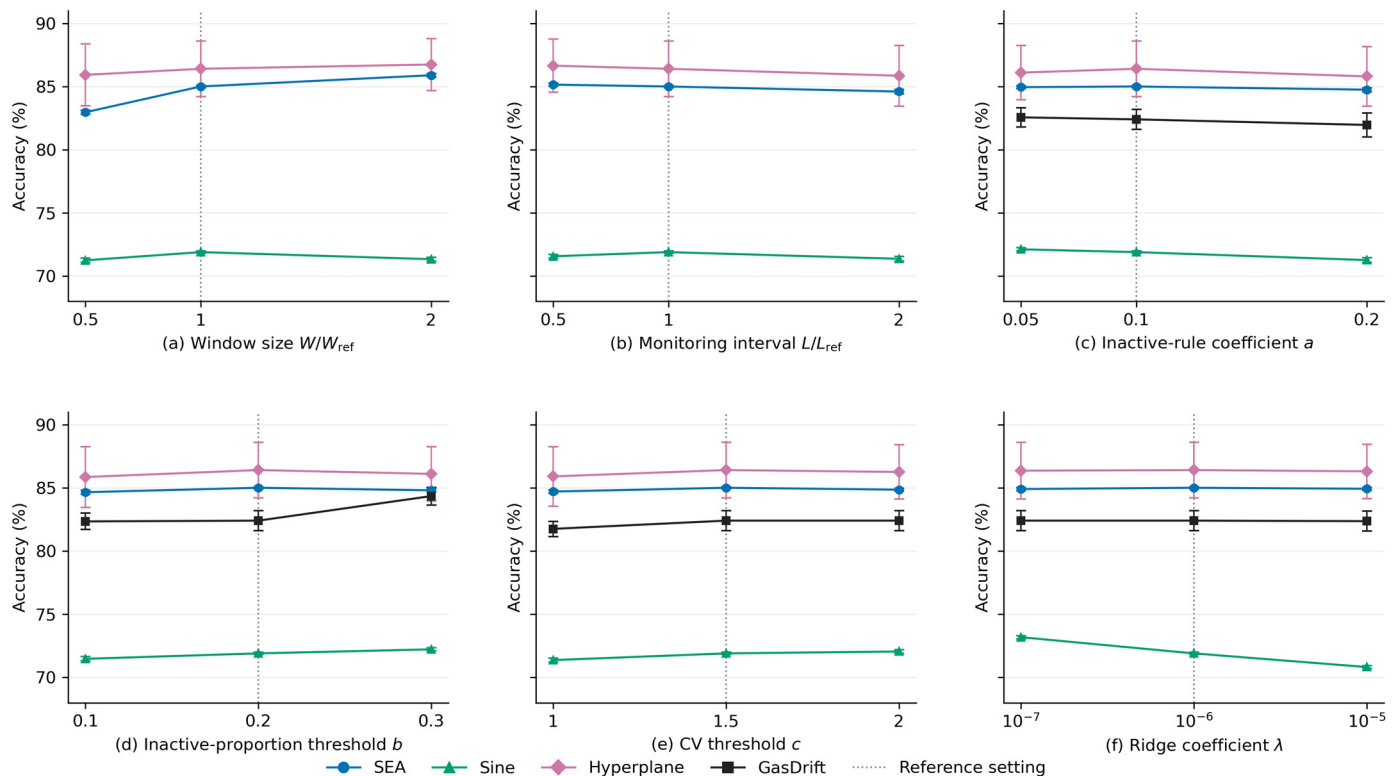


Figure 4. Local hyperparameter sensitivity of OA-A-TSK: (a) window capacity W ; (b) monitoring interval L ; (c) inactive-rule coefficient a ; (d) inactive-proportion threshold b ; (e) coefficient-of-variation threshold c ; and (f) ridge coefficient λ . All six parameters are examined on independently generated 20,000-sample SEA, Sine, and Hyperplane validation streams; panels (c–f) additionally include the complete GasDrift stream. Points and error bars represent the mean prequential accuracy and sample standard deviation over ten runs, respectively. Dotted vertical lines indicate the reference settings. Window-capacity scans use a different initialization size and scoring interval from the other scans.

6. Conclusions

This paper proposes OA-TSK, an online adaptive TSK fuzzy classifier for non-stationary data streams. OA-TSK integrates incremental and decremental closed-form updates within a sliding-window mechanism, enabling efficient sample insertion and removal while maintaining, in exact arithmetic, equivalence to ridge-regularized batch retraining on the active window. The decremental update applies the Sherman–Morrison formula to sample removal, and the resulting incremental–decremental update protocol reduces the per-sample update cost to $O(K^2 + KC)$. To address severe shifts in the input distribution, we further propose OA-A-TSK with adaptive antecedent reclustering. By monitoring the mean normalized firing strength of each rule and re-estimating the rule centers from the samples in the active window when these values are severely imbalanced, OA-A-TSK realigns the rule centers with the evolving input distribution while preserving the interpretable rule-based structure of TSK fuzzy systems. Experiments on synthetic and real-world data streams demonstrate that OA-TSK achieves classification accuracy comparable to ridge-regularized batch retraining at a substantially lower computational cost. Future work will investigate more adaptive sample-removal policies beyond FIFO and extend the proposed incremental–decremental update protocol to higher-order TSK fuzzy systems.

Author Contributions: Conceptualization, B.Q. and C.H.; methodology, M.J. and B.Q.; software, M.J.; validation, M.J. and C.S.; formal analysis, M.J. and H.Y.; investigation, M.J.; resources, B.Q.; data

curation, M.J.; writing—original draft preparation, M.J.; writing—review and editing, M.J., C.S., H.Y. and B.Q.; visualization, M.J.; supervision, B.Q.; project administration, B.Q.; funding acquisition, B.Q. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Natural Science Foundation of China under Grant 62376109 and Postgraduate Research and Practice Innovation Program of Jiangsu Province of China under Grant 26CXJH7370.

Data Availability Statement: The real-world datasets analyzed in this study are publicly available from OpenML, as cited in the manuscript. The synthetic data streams were generated using the SEA, Sine, and Hyperplane generators with the settings described in Section 5.1.

Acknowledgments: ChatGPT 5.5 was used to translate and polish the original Chinese manuscript into English, and was subsequently employed for revision and proofreading. The authors have reviewed and edited the output and will take full responsibility for the content of this publication.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Takagi, T.; Sugeno, M. Fuzzy identification of systems and its applications to modeling and control. *IEEE Trans. Syst. Man Cybern.* **1985**, *SMC-15*, 116–132. [\[CrossRef\]](#)
2. Sugeno, M.; Kang, G.T. Structure identification of fuzzy model. *Fuzzy Sets Syst.* **1988**, *28*, 15–33. [\[CrossRef\]](#)
3. Gu, X.; Han, J.; Shen, Q.; Angelov, P.P. Autonomous learning for fuzzy systems: A review. *Artif. Intell. Rev.* **2023**, *56*, 7549–7595. [\[CrossRef\]](#)
4. Wang, L.-X. Fuzzy systems are universal approximators. In Proceedings of the IEEE International Conference on Fuzzy Systems, San Diego, CA, USA, 8–12 March 1992; pp. 1163–1170. [\[CrossRef\]](#)
5. Wang, L.-X.; Mendel, J.M. Fuzzy basis functions, universal approximation, and orthogonal least-squares learning. *IEEE Trans. Neural Netw.* **1992**, *3*, 807–814. [\[CrossRef\]](#) [\[PubMed\]](#)
6. Ishibuchi, H.; Nakashima, T.; Morisawa, T. Voting in fuzzy rule-based systems for pattern classification problems. *Fuzzy Sets Syst.* **1999**, *103*, 223–238. [\[CrossRef\]](#)
7. Ishibuchi, H.; Nakashima, T.; Murata, T. Performance evaluation of fuzzy classifier systems for multidimensional pattern classification problems. *IEEE Trans. Syst. Man Cybern. B Cybern.* **1999**, *29*, 601–618. [\[CrossRef\]](#) [\[PubMed\]](#)
8. Wang, L.; Yang, Y.; Zhou, Y. Hierarchical TSK fuzzy classification through positive intervention for teaching evaluation. *Symmetry* **2026**, *18*, 1137. [\[CrossRef\]](#)
9. Ishibuchi, H.; Nozaki, K.; Yamamoto, N.; Tanaka, H. Selecting fuzzy if-then rules for classification problems using genetic algorithms. *IEEE Trans. Fuzzy Syst.* **1995**, *3*, 260–270. [\[CrossRef\]](#)
10. Shi, L.; Li, W.; Wang, H.; Yang, M.; Zhou, T. A doubly cross-interaction deep TSK fuzzy classifier with residual-directed dynamic collaboration for interpretable dependencies. *Symmetry* **2026**, *18*, 1118. [\[CrossRef\]](#)
11. Gama, J.; Žliobaitė, I.; Bifet, A.; Pechenizkiy, M.; Bouchachia, A. A survey on concept drift adaptation. *ACM Comput. Surv.* **2014**, *46*, 44. [\[CrossRef\]](#) [\[PubMed\]](#)
12. Lu, J.; Liu, A.; Dong, F.; Gu, F.; Gama, J.; Zhang, G. Learning under concept drift: A review. *IEEE Trans. Knowl. Data Eng.* **2019**, *31*, 2346–2363. [\[CrossRef\]](#)
13. Kabir, M.A.; Begum, S.; Ahmed, M.U.; Rehman, A.U. CODE: A moving-window-based framework for detecting concept drift in software defect prediction. *Symmetry* **2022**, *14*, 2508. [\[CrossRef\]](#)
14. Kolter, J.Z.; Maloof, M.A. Dynamic weighted majority: An ensemble method for drifting concepts. *J. Mach. Learn. Res.* **2007**, *8*, 2755–2790.
15. Bifet, A.; Gavalda, R. Learning from time-changing data with adaptive windowing. In Proceedings of the 2007 SIAM International Conference on Data Mining (SDM 2007), Minneapolis, MN, USA, 26–28 April 2007; pp. 443–448. [\[CrossRef\]](#)
16. Angelov, P.P.; Filev, D.P. An approach to online identification of Takagi–Sugeno fuzzy models. *IEEE Trans. Syst. Man Cybern. B Cybern.* **2004**, *34*, 484–498. [\[CrossRef\]](#) [\[PubMed\]](#)
17. Angelov, P.P.; Zhou, X. Evolving fuzzy-rule-based classifiers from data streams. *IEEE Trans. Fuzzy Syst.* **2008**, *16*, 1462–1475. [\[CrossRef\]](#)
18. Angelov, P.; Lughofer, E.; Zhou, X. Evolving fuzzy classifiers using different model architectures. *Fuzzy Sets Syst.* **2008**, *159*, 3160–3182. [\[CrossRef\]](#)
19. Pratama, M.; Pedrycz, W.; Lughofer, E. Evolving ensemble fuzzy classifier. *IEEE Trans. Fuzzy Syst.* **2018**, *26*, 2552–2567. [\[CrossRef\]](#)

20. Li, H.; Zhao, T. A dynamic similarity weighted evolving fuzzy system for concept drift of data streams. *Inf. Sci.* **2024**, *659*, 120062. [\[CrossRef\]](#)
21. Chen, H.; Sha, C.; Jiao, M.; Shao, C.; Gao, S.; Yu, H.; Qin, B. DIA-TSK: A dynamic incremental adaptive Takagi–Sugeno–Kang fuzzy classifier. *Mathematics* **2025**, *13*, 1054. [\[CrossRef\]](#)
22. Street, W.N.; Kim, Y. A streaming ensemble algorithm (SEA) for large-scale classification. In Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '01), San Francisco, CA, USA, 26–29 August 2001; pp. 377–382. [\[CrossRef\]](#)
23. Lughofer, E.; Angelov, P. Handling drifts and shifts in on-line data streams with evolving fuzzy systems. *Appl. Soft Comput.* **2011**, *11*, 2057–2068. [\[CrossRef\]](#)
24. Zhao, T.; Li, H. HWEFIS: A hybrid weighted evolving fuzzy inference system for nonstationary data streams. *IEEE Trans. Artif. Intell.* **2025**, *6*, 1679–1694. [\[CrossRef\]](#)
25. Cauwenberghs, G.; Poggio, T. Incremental and decremental support vector machine learning. In *Advances in Neural Information Processing Systems 13*; Leen, T.K., Dietterich, T.G., Tresp, V., Eds.; MIT Press: Cambridge, MA, USA, 2001; pp. 409–415.
26. MacQueen, J. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*; Le Cam, L.M., Neyman, J., Eds.; University of California Press: Berkeley, CA, USA, 1967; Volume 1, pp. 281–297.
27. Bezdek, J.C. *Pattern Recognition with Fuzzy Objective Function Algorithms*; Plenum Press: New York, NY, USA, 1981. [\[CrossRef\]](#)
28. Blažič, S.; Škrjanc, I. Incremental fuzzy C-regression clustering from streaming data for local-model-network identification. *IEEE Trans. Fuzzy Syst.* **2020**, *28*, 758–767. [\[CrossRef\]](#)
29. Pratama, M.; Anavatti, S.G.; Angelov, P.P.; Lughofer, E. PANFIS: A novel incremental learning machine. *IEEE Trans. Neural Netw. Learn. Syst.* **2014**, *25*, 55–68. [\[CrossRef\]](#) [\[PubMed\]](#)
30. Pang, S.; Ozawa, S.; Kasabov, N. Incremental linear discriminant analysis for classification of data streams. *IEEE Trans. Syst. Man Cybern. B Cybern.* **2005**, *35*, 905–914. [\[CrossRef\]](#) [\[PubMed\]](#)
31. Cheng, W.-Y.; Juang, C.-F. A fuzzy model with online incremental SVM and margin-selective gradient descent learning for classification problems. *IEEE Trans. Fuzzy Syst.* **2014**, *22*, 324–337. [\[CrossRef\]](#)
32. Subramanian, K.; Suresh, S.; Sundararajan, N. A metacognitive neuro-fuzzy inference system (McFIS) for sequential classification problems. *IEEE Trans. Fuzzy Syst.* **2013**, *21*, 1080–1095. [\[CrossRef\]](#)
33. Gama, J.; Medas, P.; Castillo, G.; Rodrigues, P.P. Learning with drift detection. In *Advances in Artificial Intelligence—SBIA 2004*; Bazzan, A.L.C., Labidi, S., Eds.; Springer: Berlin/Heidelberg, Germany, 2004; Volume 3171, pp. 286–295. [\[CrossRef\]](#)
34. Hulten, G.; Spencer, L.; Domingos, P. Mining time-changing data streams. In Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '01), San Francisco, CA, USA, 26–29 August 2001; pp. 97–106. [\[CrossRef\]](#)
35. Harries, M. *Splice-2 Comparative Evaluation: Electricity Pricing*; Technical Report UNSW-CSE-TR-9905; School of Computer Science and Engineering, University of New South Wales: Sydney, Australia, 1999.
36. Blackard, J.A.; Dean, D.J. Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables. *Comput. Electron. Agric.* **1999**, *24*, 131–151. [\[CrossRef\]](#)
37. Fonollosa, J.; Rodríguez-Luján, I.; Huerta, R. Chemical gas sensor array dataset. *Data Brief* **2015**, *3*, 85–89. [\[CrossRef\]](#) [\[PubMed\]](#)
38. van Rijn, J.N.; Bischl, B.; Torgo, L.; Gao, B.; Umaashankar, V.; Fischer, S.; Winter, P.; Wiswedel, B.; Berthold, M.R.; Vanschoren, J. OpenML: A collaborative science platform. In *Machine Learning and Knowledge Discovery in Databases*; Blockeel, H., Kersting, K., Nijssen, S., Železný, F., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; Volume 8190, pp. 645–649. [\[CrossRef\]](#)
39. Liu, D.; Wu, Y.; Jiang, H. FP-ELM: An online sequential learning algorithm for dealing with concept drift. *Neurocomputing* **2016**, *207*, 322–334. [\[CrossRef\]](#)
40. Bifet, A.; Gavaldà, R. Adaptive learning from evolving data streams. In *Advances in Intelligent Data Analysis VIII*; Adams, N.M., Robardet, C., Siebes, A., Boulicaut, J.-F., Eds.; Springer: Berlin/Heidelberg, Germany, 2009; Volume 5772, pp. 249–260. [\[CrossRef\]](#)
41. Gomes, H.M.; Bifet, A.; Read, J.; Barddal, J.P.; Enembreck, F.; Pfahringer, B.; Holmes, G.; Abdesslem, T. Adaptive random forests for evolving data stream classification. *Mach. Learn.* **2017**, *106*, 1469–1495, Correction in *Mach. Learn.* **2019**, *108*, 1877–1878. [\[CrossRef\]](#)
42. Gama, J.; Sebastião, R.; Rodrigues, P.P. On evaluating stream learning algorithms. *Mach. Learn.* **2013**, *90*, 317–346. [\[CrossRef\]](#)
43. Demšar, J. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.* **2006**, *7*, 1–30.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.