mathematics

MDPI

Article

A Semisupervised Concept Drift Adaptation via Prototype-Based Manifold Regularization Approach with Knowledge Transfer

Muhammad Zafran Muhammad Zaly Shah [1,*], Anazida Zainal [1], Taiseer Abdalla Elfadil Eisa [2], Hashim Albasheer [3] and Fuad A. Ghaleb [1]

[1] Faculty of Computing, Universiti Teknologi Malaysia, Iskandar Puteri 81310, Malaysia
[2] Department of Information Systems-Girls Section, King Khalid University, Mahayil 62529, Saudi Arabia
[3] Department of Information Systems, College of Computer Science, King Khalid University, Abha 61421, Saudi Arabia
* Correspondence: muhammad.zafran@graduate.utm.my

Abstract: Data stream mining deals with processing large amounts of data in nonstationary environments, where the relationship between the data and the labels often changes. Such dynamic relationships make it difficult to design a computationally efficient data stream processing algorithm that is also adaptable to the nonstationarity of the environment. To make the algorithm adaptable to the nonstationarity of the environment, concept drift detectors are attached to detect the changes in the environment by monitoring the error rates and adapting to the environment's current state. Unfortunately, current approaches to adapt to environmental changes assume that the data stream is fully labeled. Assuming a fully labeled data stream is a flawed assumption as the labeling effort would be too impractical due to the rapid arrival and volume of the data. To address this issue, this study proposes to detect concept drift by anticipating a possible change in the true label in the high confidence prediction region. This study also proposes an ensemble-based concept drift adaptation approach that transfers reliable classifiers to the new concept. The significance of our proposed approach compared to the current baselines is that our approach does not use a performance measur as the drift signal or assume a change in data distribution when concept drift occurs. As a result, our proposed approach can detect concept drift when labeled data are scarce, even when the data distribution remains static. Based on the results, this proposed approach can detect concept drifts and fully supervised data stream mining approaches and performs well on mixed-severity concept drift datasets.

Keywords: machine learning; semisupervised learning; manifold regularization; sequential learning; internet of things; data stream mining; concept drift

MSC: 68W27; 68T07; 68T30; 68U35

Citation: Muhammad Zaly Shah, M.Z.; Zainal, A.; Elfadil Eisa, T.A.; Albasheer, H.; Ghaleb, F.A. A Semisupervised Concept Drift Adaptation via Prototype-Based Manifold Regularization Approach with Knowledge Transfer. *Mathematics* **2023**, *11*, 355. https://doi.org/10.3390/math11020355

Academic Editors: Weihua Xu, Jinhai Li and Xibei Yang

Received: 26 November 2022
Revised: 2 January 2023
Accepted: 4 January 2023
Published: 9 January 2023

1. Introduction

The era of big data has transformed many applications by giving insights and predicting future observations to make business processes more effective and efficient. For example, the Internet of Things (IoT) application has improved manufacturing efficiency by predicting the failure rates of machines through data gathered from sensors [1,2]. Big data also improved sales productivity by predicting customer churn to maximize sales [3,4]. Big data applications such as IoT operate where large amounts of data are constantly being generated, known as data streams [5]. As a result, the challenge of mining from big data streams is often described using the three principal challenges: *volume*, *velocity*, and *volatility*, or simply 3Vs [6]. Volume and velocity refer to large data being generated

constantly. These data are possibly infinite, making storing them in a storage device impossible, requiring the data to be processed sequentially or streamed. A data streaming algorithm is also designed to be trained online compared to batch learning algorithms that require the whole dataset to be stored for the model to be trained [7–9]. Finally, volatility refers to the dynamic environment of data streams, where changes can occur spontaneously. In the data stream literature, this situation is known as concept drift [10,11].

Various data stream algorithms have been proposed, but the most popular data stream algorithm is based on the decision tree classifier due to its simple structure and generalization capabilities when constructed as an ensemble [12]. However, recently, other algorithm types, such as the Extreme Learning Machine (ELM), are being investigated for their ability to perform on high-dimensional data [13]. Nevertheless, most data stream algorithms that had been proposed manage to address the computational challenges of processing large amounts of data by having a similar performance to batch learning algorithms with significantly fewer computational resources [14]. However, most existing data stream algorithms operate in nonstationary environments where the relationship between the data and the label may change with time, making the model obsolete and causing erroneous and unreliable predictions to be produced [15,16]. For example, a model trained to predict customer churn may become obsolete when customer behavior changes, requiring the model to be updated. Therefore, data stream algorithms should be able to detect and adapt to environmental changes to preserve a good prediction performance. Tackling concept drift is nontrivial as detecting its occurrence can be challenging as it can occur with various severity, from an abrupt change to a gradual change to a new concept [17]. In addition, the following challenges also exist when attempting to detect and adapt to concept drifts [18–20]:

1. Concept drifts need to be detected as soon as they occur.
2. Overlapping concepts can cause noise to the concept drift detector, causing concept drift detection to be uncertain.
3. Detecting concept drift needs to occur continuously. Therefore, processing the data and deciding to trigger the drift alarm and adapt to drifts need to be computationally efficient to maximize the model's throughput.

To detect concept drifts, the concept drift detector uses signals that may indicate the occurrence of concept drifts [21]. These signals are usually the error rates produced by the prediction model, which works by comparing the current error rate and the previous error rate. Concept drift is assumed to have occurred if the current error rate is significantly higher than the previous error rate. To measure the significance of the increase in the error rate, some approaches use the control chart approach [21,22], which signals a concept drift if the running mean of the error rate is n standard deviations above the normal rate. Another approach uses statistical learning theory to calculate the error bound the classifier should not surpass, assuming a stationary environment [23]. So, if the classifier surpasses the error bound, a concept drift is assumed to have occurred. A more straightforward approach to detecting concept drift is using statistical hypothesis testing to detect if the error rate is significantly higher than the mean error rate produced by the classifier [24].

The model adaptation component is responsible for updating the classifier to respond to environmental changes to adapt to concept drifts. A trivial approach is to reset the classifier to completely remove the previous concept from the model and retrain it from scratch [25] or keep updating the model as each sample arrives to compensate for gradual drifts [26]. However, retraining the classifier is only effective if the concept drift is severe; otherwise, the adaptation would be inefficient as the classifier needs to relearn the concepts unaffected by the concept drift [25]. On the other hand, constantly retraining the classifiers to adapt to gradual drifts assumes that concept drift constantly occurs, which is incorrect as concept drift can occur with different durations [26,27]. Therefore, a more efficient approach to adapting to concept drift is by only removing the classifiers affected by the concept drift. This can be achieved by ensemble-based approaches that prune the underperforming classifiers [28,29]. Because ensemble-based approaches are trained using the bagging

sampling method that increases the diversity of the model by training the classifiers on different subsets of data, some classifiers may survive if the data it is trained on is not affected by the drift [25]. However, for the concept drift detection and model adaptation components to function, it requires a fully labeled data stream to calculate the error rates of the classifier to detect the concept drift and identify which classifiers are underperforming during the adaptation process [30]. Unfortunately, in most cases, labeled data are difficult to obtain, making most concept drift detection approaches impractical. It is also mentioned that only 20% of works in data stream mining are directed towards semisupervised and unsupervised concept drift handling to make data stream mining more label efficient [10]. Therefore, a more practical approach to detect and adapt to concept drift is to design an approach that requires less labeled data to make data stream mining more practical.

Existing concept drift detection approaches, which were designed for environments with scarce labeled data, detect concept drifts based on the changes in the data distribution. For example, the study in [31] used the change in the mean of the distribution as a signal for detecting concept drift. That is, the concept drift was detected when the center of the data distribution changed. Some approaches also consider the change in the direction of the principal component of the distribution, which is suitable for high-dimensional data [32]. However, changes in the data distribution do not necessarily translate into changes in the decision boundary, which is the main cause of concept drift [33]. This makes detecting concept drift in an environment with a lack of labeled data difficult, as changes in decision boundaries can only be detected by monitoring the error rate of the classifier [34]. This would cause many false positives to be triggered if there was no decision boundary change occurring that directly caused concept drifts [33]. To address this issue, concept drift detectors should be more efficient in using labeled data to detect concept drifts. In summary, the identified weaknesses of the current approaches are listed below:

1. Current approaches require labeled data that are expensive to obtain to calculate the performance measure, e.g., the error rate of the model to be used as the drift signal.
2. Current unsupervised concept drift detection is blind to the change in the decision boundary by only monitoring the change in the data distribution.

In this study, we propose a concept drift detection approach that is more efficient in detecting concept drifts by requiring only a partially labeled data stream, i.e., a semisupervised approach. We achieved this by assuming that even a highly confident prediction would be incorrect if a concept drift occurs. Then, the rate of an incorrect prediction made by confident predictions was recorded. The proposed approach triggers a concept drift alarm if the rate of the incorrect confident prediction is higher than the usual rate. In addition, we also propose a metric to estimate the reliability of the classifiers in the ensemble to adapt to concept drifts. The proposed metric allows for the estimation of the reliability of the classifiers without measuring their error rate that uses labeled data which improves the overall practicality of data stream mining in environments where labeled data are scarce. Adapting to drifts by selecting the most reliable classifiers in the ensemble has been investigated in several works [28,29]. However, our proposed approach does not rely on labeled data to estimate the classification performance. This study aimed at improving the practicality of data stream mining by reducing the dependency on labeled data. The contributions of this study are listed as follows:

1. A concept detection approach was designed and developed to detect concept drifts by measuring the error made by confident predictions.
2. An evaluation metric has been proposed to measure the performance of the classifiers in the ensemble without requiring labeled data.

2. Literature Review

2.1. Data Stream Mining

Early works in data stream mining were primarily focused on addressing the computational challenges of data stream mining. For example, the decision tree classifier proposed by Domingos and Hulten [35] is the Very Fast Decision Tree (VFDT) classifier that uses

Hoeffding's bound to calculate the minimum number of samples to reduce the error to a certain rate. This way, the decision tree is only trained with minimum samples rather than the whole dataset to conserve memory and computing resources. To improve the generalization capability of the VFDT, Oza and Russell [36] proposed the online bagging algorithm that approximates the batch learning of bagging decision trees, thus guaranteeing its performance to be close to a batch learning algorithm despite being trained sequentially. A more recent implementation of the VFDT is the Extremely Fast Decision Tree (EFDT) [37], which only updates the decision tree if it improves its performance to reduce computational load further.

As the computational challenges of data stream mining are mostly addressed, research in data stream mining attempts to tackle the challenge of mining from nonstationary environments [16]. For decision-tree-based classifiers, adapting to concept drift involves either the voting weight update approach or the structure update approach. The voting weight update approach essentially assigns higher influence to the predictions made by the classifiers that are currently performing well. For example, the Accuracy Weighted Ensemble (AWE) [38] and Accuracy Updated Ensemble 1 (AUE1) [39] weigh their classifiers based on their current accuracy. However, when all classifiers perform poorly, e.g., when a severe, abrupt drift occurs, the weights of the classifiers would be equal, causing the predictions to be assigned randomly by the classifiers. To address this, the Accuracy Updated Ensemble 2 (AUE2) [40] was proposed by regularly adding new classifier members to the ensemble and treating it as the perfect classifier. All classifiers in AUE2 are assigned weights based on their accuracy except the newest classifier, which always has the highest priority.

Adapting to concept drift by updating the classifier weights can be slow when more severe concept drift is encountered. Therefore, the structure update approach exists as a more aggressive approach to adapt to a more severe concept drift by removing and replacing the underperforming classifiers. Because of the aggressive nature of the structure update approach, prediction models adopting this approach are often paired with a concept drift detector to only update its structure when concept drift occurs. This is to prevent a phenomenon known as catastrophic forgetting, which happens when the model forgets relevant information that had been trained [41,42]. For example, the ambiguous Continuous VFDT (aCVFDT) [43] attaches a concept drift detector to the VFDT classifier and begins to search for an underperforming node in the decision tree that should be replaced when a concept drift is detected. Another approach is Hoeffding's Windowing Tree—Adaptive Windowing (HWT-ADWIN) [44] decision tree, which is more aggressive by training a completely new decision tree to replace the current decision tree. However, HWT-ADWIN only replaces the current decision tree when it has evidence that the new decision tree is significantly better than the current decision tree.

For ensemble-based approaches, the Adaptive Random Forest (ARF) [45] attaches a concept drift detector to each classifier, prepares a new classifier when a warning is issued by the concept drift detector, and replaces the current member classifier when a concept drift is detected. This allows for a faster adaptation to concept drift as it has less delay to adapt to the new concept caused by the new classifier taking time to learn the new concept. To deal with mixed-severity concept drift, the Diversity Dealing Drift (DDD) [25] decision tree ensemble is proposed by managing the diversity of the ensemble. During high severity drifts, the ensemble will have low diversity to specialize in the new concept as few overlapping concepts can be carried forward after the concept drift. Conversely, during low severity drift, as there is more concept to carry forward, DDD constructs a high diversity ensemble to adapt to the new concept.

Due to the uncertainty of data streams resulting from noisy data, approaches based on evolving fuzzy systems (EFS) have also been investigated. For example, approaches based on density clustering [46], vector quantization [46], and rule-based approaches [47,48] were proposed to learn from evolving data streams. EFS has also been deployed as an ensemble trained using the online bagging approach [49]. These approaches rely on forgetting mechanisms such as a forgetting factor and rule pruning to remove old concepts

suitable for gradual drifts. To adapt to sudden or abrupt drifts, these approaches rely on detecting inconsistencies between the rule that relies on labeled data or expects a change in the data distribution when the abrupt drifts occur. This proposed approach attempts to address the challenge of detecting and adapting concept drifts when the data distribution remains static without requiring a significant amount of labeled data. However, most data stream approaches assume that all data are labeled, which is a flawed assumption in many applications. Due to the rapid arrival of data in data stream mining, labeling data individually is impractical and time consuming. Therefore, some works in data stream mining have been directed to learn in environments where labeled data are scarce, more commonly known as semisupervised learning (SSL) [50,51]. For data stream mining, the manifold regularization approach of SSL is applied by assuming that similar data tends to have similar labels [52,53]. An example of this approach is the Semisupervised Online Sequential—Extreme Learning Machine (SOS-ELM) [54], which uses the manifold regularization approach to learn incrementally from big datasets. Additionally, to address the challenge of concept drift, the Elastic SOS-ELM (SSOE-ELM) [55] was proposed by regulating the forgetting factor, which increases forgetting during high-severity drift and lowers the forgetting rate during gradual or stable periods. Another approach is the Self Organizing Incremental Neural Network (SOINN) [56] and the Enhanced SOINN [56,57], which incrementally constructs an approximation of the dataset and uses lazy learning algorithms, such as the K-Nearest Neighbors (KNN), to classify data.

Graph-based approaches have also influenced SSL by exploiting its structured data representation characteristics [58,59]. Some examples of graph-based approaches for SSL is the label propagation (LP) approach [60–62] and the manifold regularization approach [63,64]. Due to the graph structure, manifold regularization approaches are more straightforward as the graph Laplacian can be easily computed using the readily available adjacency matrix. However, it also makes it less suitable for sequential data processing as inserting a new node in the adjacency matrix is more expensive than the SOINN approach that only requires the insertion of a new node into an array, which only has a time complexity of $O(N)$.

SSL in data streams also uses the clustering approach by maintaining a set of clusters and labeling each instance in the cluster based on the similarity between the labeled data to the cluster centroid [65–67]. The clustering approach can potentially adapt to concept drift, as the instances stored in the cluster are constantly updated along with its labels. However, due to the curse of dimensionality, clustering approaches are not suitable for modeling high-dimensional data streams [68]. Unfortunately, other approaches in SSL for data stream mining cannot adapt to concept drift due to the inability to detect concept drifts caused by the lack of labeled data.

2.2. Concept Drift Detection

Research in concept drift detection initially adopted techniques from the field of process control dedicated to detecting if a system behaves abnormally [30]. A stable system would have its measurement centered around the mean and out of control when the measurement is higher than n standard deviations above the mean. For concept drift detection, the measurement would be the error rate produced by the classifier. Hence, a concept drift is detected when the error is n standard deviations above the mean.

An example of a control chart approach adopted for concept drift detection is the Page–Hinkley Test (PHT) [22], a popular process control approach. However, the Exponential Weighted Moving Average (EWMA) [69] was proposed as a control chart approach that is specialized for detecting concept drifts. EWMA improves the PHT by giving higher weights to more recent observations to make it more sensitive to concept drift. EWMA for Concept Drift Detection (ECDD) [21] was also proposed to improve the PHT by not requiring the mean and standard deviation to be determined a priori to initialize the control chart.

A more theoretically robust approach to detecting concept drifts is by applying the Probably Approximately Correct (PAC) [42] assumption for a more accurate concept drift

detection, which assumes that the error classifier will decrease under a stationary distribution. Therefore, a classifier that violates this assumption is assumed to be the result of concept drift. Another assumption is Hoeffding's bound [70], which bounds the classifier's error based on the number of data that had been observed. Hence, if the error exceeds the bound, a concept drift is assumed to have occurred. The Drift Detection Method (DDM) [71] uses the PAC assumption, which models the error produced by the classifier as a Bernoulli distribution with a rate parameter that describes the rate of incorrect classification. The Early Drift Detection Method (EDDM) [71] was proposed as an improvement to the DDM to detect slower drifts (gradual drifts) by comparing the current error rate to a fixed landmark window.

The Hoeffding's bound, on the other hand, is applied by the Adaptive Windowing (ADWIN) [72] to detect drift by detecting if there exists a statistically significant split in the current window between the two subwindows. One of the state-of-the-art (SOTA) approaches that apply Hoeffding's bound is Hoeffding's Drift Detection Method (HDDM) [23]. HDDM achieves SOTA performance using Hoeffding's bound to bound the false positives. Therefore, it allows the HDDM approach to trigger the drift alarm confidently.

Another category of concept drift detection is adopting the changepoint detection approach, which is utilized for time-series analysis to detect points of change in the state of the time series [73]. The online changepoint detection approaches [74–76] can be applied for concept drift detection due to their sequential data processing characteristic using performance measures, e.g., accuracy as the signal. Concept drift is detected when a change is detected in the model's performance. However, all of the existing approaches require labeled data to detect drifts as they detect drifts based on the error produced by the classifier. This is a limitation for environments with scarcely labeled data. Therefore, several concept drift detectors have been proposed that do not require labeled data, i.e., that do not calculate the error produced by the classifier. In most proposed approaches, these concept drift detectors detect drifts by detecting if there exists any change in the data distribution. For example, one approach uses the Kullback–Leibler Divergence (KL-Divergence) to measure if there exists any significant difference between the data distribution of the current and previous data [31]. Changes in the data distribution can also be detected based on the change in the direction of the distribution's principal axis, obtained through the Principal Component Analysis (PCA) dimensionality reduction algorithm [32]. However, a distribution change may not necessarily indicate a change in the decision boundary, which is the main cause of concept drift [33]. As a result, false positives may cause an unnecessary model update. Therefore, a more efficient concept drift detection can detect concept drift when few labeled data are desired.

Detecting concept drift as a result of a change in the decision boundary when labeled data are scarce has been attempted by estimating the accuracy of the classifier using minimal labeled data obtained using the Active Learning (AL) method and combining it with the PHT control chart approach [77]. However, this approach uses the classifier accuracy that can be overestimated when too few labeled data are recorded, resulting in poor concept drift detection when few labeled samples are recorded through the AL method. This proposed approach differs by diverting from the accuracy estimation as the performance indicator but instead uses the conflict between the labeled data and confidence prediction as the performance indicator. Conflict-based indicators have also been investigated [78] by measuring the prediction disagreement among the ensemble classifiers. However, this approach relies on the ensemble to be diverse to ensure that predictions between the classifiers tend to disagree when concept drift occurs [16]. In comparison, the proposed approach in this study does not use the disagreement between the classifiers in the ensemble. Instead, the proposed approach uses the disagreement between the label and the prediction provided by high-confidence prediction as the concept drift signal. In addition, our proposed approach does not impose a strict requirement of a fully labeled dataset to detect concept drift.

3. Preliminaries

This section will discuss the proposed approach's building blocks, which consist of the Enhance Self Organizing Incremental Neural Network (ESOINN) [57] and Semisupervised Online Sequential—Extreme Learning Machine (SOS-ELM) [54]. To improve readability, a list of symbols is provided in Table 1 to describe each symbol and variable used in this paper. The ESOINN is required to estimate the prediction confidence of the SOS-ELM, which will be used to detect concept drift, as will be explained in Section 4.

Table 1. List of symbols for the ESOINN and SOS-ELM.

Symbol	Description
$\mathcal{D}$	Dataset
$\mathbf{x}_i$	Datapoint i, $\mathbf{x}_i \in \mathcal{D}$
S	Set of nodes
N_i	Nearest node to the ith winner node
$WT(a)$	Number of times node a has been selected as the winner
$\mathcal{E}$	Set of edges in the SOINN
$e\left(a_i, a_j\right)$	Edge between nodes a_i and a_j, $e\left(a_i, a_j\right) \in \mathcal{E}$
T_i	Distance threshold for node insertion
$age\left(a_i, a_j\right)$	Age of the edge between nodes a_i and a_j
$\mathcal{N}_a$	Set of neighbors for node a
SD_1	Number of standard deviations above the mean for node deletion when anode a has only one neighbor.
SD_2	Number of standard deviations above the mean for node deletion when node a has two neighbors.
L	Graph Laplacian matrix
D	Degree matrix
W	Similarity matrix
T	Prediction matrix
β	Trainable parameters for SOS-ELM
H	Input-hidden output matrix for SOS-ELM
J	Regularization term matrix
$G(.)$	Activation function
P_k	Kalman gain
α	Labeled–unlabeled importance tradeoff
λ	Radial Basis Function (RBF) kernel width parameter

3.1. Self-Organizing Incremental Neural Network (SOINN)

The Enhanced Self-Organizing Incremental Neural Network (ESOINN) is based on the Self-Organizing Incremental Neural Network (SOINN) that learns the structure of the dataset by storing samples of the data as nodes in the form of a network. The ESOINN or SOINN undergoes a process of node insertion, node merging, and node deletion to form the structure of the network. Due to its network structure, the ESOINN could be used to discover and analyze the similarity between data clusters.

The node insertion to add a new node in the network is determined by using the node addition criterion that determines the informativeness of the new node candidate $\mathbf{x}$. By comparing the similarity of the new data and the whole node in the library, the most similar node referred to as the first winner N_1 and the second most similar data referred to as the second winner N_2 is selected. Then, using Equation (1), two thresholds T_i are calculated for each winner N_1 and N_2, where $\mathcal{N}_i$ is the neighbors set connected to winner i via an edge. The new node is considered novel and informative if the distance between the new node candidate $\mathbf{x}$ and either winner N_i is larger than the threshold T_i, as in line 8 of Algorithm 1.

Algorithm 1: Enhanced Self-Organizing Incremental Neural Network (ESOINN)
Input: Dataset $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^{N}$, maximum egde age age_{max}, noise remove interval λ
Output: Set of prototypes S

function train_esoinn($\mathcal{D}, age_{max}, \lambda$):
foreach $\mathbf{x} \in \mathcal{D}$:
if $|S| < 2$:
  $S = S \cup \mathbf{x}$
  continue
$N_1 = \min_{a \in S} ||a - \mathbf{x}||$ //find 1st winner
$N_2 = \min_{a \in S \setminus N_1} ||a - \mathbf{x}||$ //find 2nd winner
if $||\mathbf{x} - N_1|| > T_1$ or $||\mathbf{x} - N_2|| > T_2$:
  $S = S \cup \mathbf{x}$
else:
  $N_1 = N_1 - \frac{(\mathbf{x} - N_1)}{WT(N_1)}$ //update 1st winner
  $a = a - \frac{(\mathbf{x} - N_1)}{100 \cdot WT(N_1)}$, $a \in \mathcal{N}_1$ //update 1st winner neighbors
  $WT(N_1) = WT(N_1) + 1$
  if $e(N_1, N_1) \notin \mathcal{E}$:
    $\mathcal{E} = \mathcal{E} + e(N_1, N_2)$ //add a connection between 1st winner and 2nd winner
  else:
    $age(N_1, N_2) = 0$
  $age(N_1, a) = age(N_1, a) + 1, \ a \in \mathcal{N}_1$
  remove all edges if age > age_{max}
  if number of data is multiple of λ:
    foreach λ:
      if $|\mathcal{N}_a| = 0$:
        $S = S \setminus a$ //remove node a
      else if $|\mathcal{N}_a| = 1$ **and** $WT(a) < SD_1 \cdot WT_{mean}$:
        $S = S \setminus a$ //remove node a
      else if $|\mathcal{N}_a| = 2$ **and** $WT(a) < SD_2 \cdot WT_{mean}$:
end function

$$T_i = \begin{cases} \min_{a \in \mathcal{N}_i} ||N_i - a||, \ \mathcal{N}_i \neq \varnothing \\ \max_{a \in \mathcal{S}/N_i} ||N_i - a||, \ \mathcal{N}_i = \varnothing \end{cases} \tag{1}$$

If the node novelty criterion is not satisfied, it suggests that the new node is not novel and does not add new information to the SOINN model. However, it also suggests that the first and second winners are similar and likely belong to the same cluster. Therefore, the first and second winner undergoes a merging process by connecting them with an edge if no edge currently exists. If an edge already exists, the edge age is reset to zero and the first winner N_1 is updated using Equation (2), where $WT(N_1)$ is the winning times of the winner node N_1. The first winner N_1 neighbors are also updated using Equation (3) and its edge that is connected to the first winner N_1 is incremented by one:

$$N_1 = N_1 - \frac{1}{WT(N_1)}(\mathbf{x} - N_1) \tag{2}$$

$$a = a - \frac{1}{100WT(N_1)}(\mathbf{x} - a), \ a \in \mathcal{N}_1 \tag{3}$$

Finally, when an interval defined by λ is reached, the SOINN will undergo a node deletion process to remove possible noisy nodes in the SOINN by removing isolated nodes, which are nodes that are not connected by an edge. However, the ESOINN extends this by removing nodes with fewer than two edges and has less than the average winning time compared to the whole network.

3.2. Semisupervised Online Sequential—Extreme Learning Machine

An Extreme Learning Machine (ELM) is a neural network structure consisting of only one hidden layer. The hidden layer size usually consists of 1000 hidden nodes, as it is shown to work well for a large variety of datasets [79]. The output $\hat{T}$ of the ELM is given by Equation (4), where H is the input-hidden layer matrix and β is the hidden-output weight matrix. The input-hidden layer matrix H given by Equation (5) is obtained by passing the input $\mathbf{x}$ through a $D \times L$ matrix where D is the input data dimension and L is the hidden-layer size. Each entry of the matrix H is the function of an activation function $G(a_i, b_i, \mathbf{x}_i)$, e.g., ReLU or Sigmoid function parameterized by the weight a_i and bias b_i that is assigned randomly.

$$N_1 = N_1 - \frac{1}{WT(N_1)}(\mathbf{x} - N_1) \tag{4}$$

$$a = a - \frac{1}{100WT(N_1)}(\mathbf{x} - a),\ a \in \mathcal{N}_1 \tag{5}$$

Given a dataset $\mathcal{D} = \{(\mathbf{x}_i, \mathrm{t}_i)\}_{i=0}^{N}$, the ELM is trained by minimizing the objective function in Equation (6) with respect to the hidden-output matrix β. Via the Least Square approximation, Equation (6) is minimized using Equation (7), where $H^{\dagger}$ is the Moore–Penrose pseudoinverse of matrix H.

$$\min_{\beta} ||H\beta - T|| \tag{6}$$

$$\beta = H^{\dagger}T \tag{7}$$

The Semisupervised ELM (SS-ELM) [80] is a derivation of ELM for learning when labeled data are scarcely available by exploiting the smoothness assumption. The smoothness assumption is exploited using the manifold regularization term that forces the model to predict the same labels for similar data. The manifold regularization term L is obtained in Equation (8), where $W = [w_{ij}]$ is the similarity matrix with each entry given by $w_{ij} = \exp(-\lambda||\mathbf{x}_i - \mathbf{x}_j||^2)$, which is the output of the RBF kernel between data $\mathbf{x}_i$ and $\mathbf{x}_j$ and D is the degree matrix with elements $D_{ii} = \sum_{j=0}^{N} w_{ij}$. λ is the width parameter to control the radius of influence of the RBF kernel. The matrix W can be a sparse matrix to reduce computational consumption by only considering the k nearest neighbors for each data.

$$L = D - W \tag{8}$$

The objective function in Equation (9) is the new objective function with the manifold regularization term L incorporated, where $J = diag(C_i, \ldots, C_l, 0, \ldots, 0)$ with its first l entries, which corresponds to the labeled data, equals C for regularization against overfitting, and α is the labeled–unlabeled data importance tradeoff. The SS-ELM is trained in closed form by minimizing Equation (9) via Equation (10).

$$\min_{\beta} \frac{1}{2}||\beta||^2 + ||J\hat{T} - T||^2 + \alpha\hat{T}^T L\hat{T} \tag{9}$$

$$\beta^* = \left(I + H^T H + \alpha\hat{T}^T L\hat{T}\right)^{-1} H^T JH \tag{10}$$

To make the SS-ELM more practical, the Semisupervised Online Sequential—Extreme Learning Machine was proposed to allow the SS-ELM to be trained sequentially as the pseudoinverse matrix $H^{\dagger}$ could not be computed for large H due to the cubic computational

complexity. Similar to the study in [54], the Recursive Least Squares (RLS) algorithm is applied to train the SOS-ELM, given by Equations (11) and (12).

$$\beta^{(k+1)} = \beta^{(k)} + P_k H_{k+1}^T \left(J_{k+1} T_{k+1} - (J_{k+1} + \alpha L_{k+1}) H_{k+1} \beta^{(k)} \right) \tag{11}$$

$$P_{k+1} = P_k - P_k H_{k+1}^T \left(I + (J_{k+1} + \alpha L_{k+1}) H_{k+1} P_k H_{k+1}^T \right)^{-1} . (J_{k+1} + \alpha L_{k+1}) H_{k+1} P_k \tag{12}$$

The pseudocode to train the SOS-ELM is described below:

Initialization Phase:

Obtain an initial dataset $\mathcal{D}_o = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{N_o}, y_{N_o})\}$ with true labels

1. Randomly assign input weights a_i and bias b_i for $i = 1, \ldots, L$
2. Calculate the hidden layer output matrix:
3. Calculate the initial hidden-output weights:

$$\beta^o = P_o H_o^T Y_o \tag{13}$$

where,

$$P_o = \left(H_o^T H_o \right)^{-1}, Y_o = (y_1, \ldots, y_{N_o})^T \tag{14}$$

4. Set $t = 0$

Sequential Learning phase:

Obtain a chunk of data $\mathcal{D}_t = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l), (\mathbf{x}_{l+1}), \ldots, (\mathbf{x}_u)\}_{i=1}^{l+u}$ labeled by which consists of l labeled data and u unlabeled data.

5. Calculate the hidden layer output matrix:

$$H_{t+1} = \begin{bmatrix} G(a_1, b_1, \mathbf{x}_1) & \cdots & G(a_L, b_L, \mathbf{x}) \\ \vdots & \ddots & \vdots \\ G(a_1, b_1, \mathbf{x}_N) & \cdots & G(a_L, b_L, \mathbf{x}_N) \end{bmatrix}_{N \times L} \tag{15}$$

6. Create the Laplacian matrix $L = D - W$ and penalty matrix J
7. Calculate the hidden output weights

$$\beta^{(k+1)} = \beta^{(k)} + P_k H_{k+1}^T \left(J_{k+1} T_{k+1} - (J_{k+1} + \alpha L_{k+1}) H_{k+1} \beta^{(k)} \right) \tag{16}$$

8. Caculate

$$P_{k+1} = P_k - P_k H_{k+1}^T \left(I + (J_{k+1} + \alpha L_{k+1}) H_{k+1} P_k H_{k+1}^T \right)^{-1} . (J_{k+1} + \alpha L_{k+1}) H_{k+1} P_k \tag{17}$$

9. Set $t = t + 1$

4. The Proposed Approach

4.1. High Confidence Prediction Conflict-Based Concept Drift Detection

As discussed in the introduction, detecting concept drift via error rate monitoring is challenging as it requires significant labeled data to detect drifts. To reduce the reliance on labeled data to detect drifts, this approach proposes to detect drifts by detecting if there is label switching in the high confidence region by assuming that during concept drift, especially if it is severe, even areas with high prediction confidence would produce erroneous prediction due to label switching. This assumption is supported by Webb, Hyde [27], who define severe concept drift as the situation where the full domain space

of the data $Dom(x)$ changes some of its label y, which is formally defined in Equation (18) where t and u are different points in time.

$$\forall_{x \in Dom(x)} \exists_{y \in Dom(y)} P_t(Y \mid X) \neq P_u(Y \mid X) \tag{18}$$

To utilize this assumption that changes in labels in high confidence region signal drift occurs, the prediction confidence of the model has to be estimated. Unfortunately, using the model's class probability prediction is unsatisfactory as it may overestimate or underestimate the confidence of the prediction since it does not consider the region around the data. Using the traditional Bayesian approach to estimate the prediction uncertainty would introduce computational intractability as it requires approximation techniques to estimate the posterior distribution.

To efficiently estimate the prediction confidence of each data, we propose to take the average confidence of the region around the data to estimate its prediction confidence. This can be achieved by obtaining the k most similar prototypes around the data from the ESOINN discussed in Section 3.1 and taking the average of its prediction confidence.

By utilizing the confidence of the prediction obtained by averaging the prototypes of the ESOINN around the data, the prediction of the data is compared with the true label of the data. Then, if the prediction conflicts with the true label in a high-confidence region, concept drift is assumed to have occurred. Note that this approach assumes that some data are labeled.

However, triggering the drift alarm even if only one conflict occurs in the high confidence region may produce false positives if the label itself is erroneous due to label noise. Therefore, this approach is combined with a control chart method, e.g., the Page–Hinkley Test (PHT), to only trigger the drift alarm if the label conflict in the high confidence region occurs at a higher rate than usual. The overall algorithm is described in Algorithm 2. Note that in Step 5, the labeled data are obtained from the partially labeled data chunk $\mathcal{D}_t$ with l being the number of labeled samples. The labeled data could be obtained via the Active Learning (AL) method by requesting l samples of labeled data per chunk from a domain expert. Therefore, the size of l should be small compared to the size of the data chunk, e.g., one or two samples, to not overwhelm the expert on the labeling task.

Algorithm 2: Confidence Region Prediction Conflict Concept Drift Detection
Input: Set of prototypes S, confidence threshold τ, control chart method e.g., Page–Hinkley Test (PHT), partially labeled data chunk $\mathcal{D}_t = \{(\mathbf{x}_0, y_0), \ldots, (\mathbf{x}_l, y_l), \mathrm{x}_{l+1}, \ldots, \mathrm{x}_u\}_{i=0}^{N=l+u}$
Output: None

1: **function** drift_detect():
2: $\hat{y}_{0,\ldots,l} \leftarrow predict(\mathbf{x}_0, \ldots, \mathbf{x}_l)$
3: error = 0
4: for i in $\{0, \ldots, l\}$:
5: if $\hat{y}_i \neq y$:
6: $K \leftarrow nearest\_prototypes\ (S, \mathbf{x}_i)$
7: avg_confidence $\leftarrow$ calculate_confidence via #EQ
8: if avg_confidence $\geq \tau$:
9: error $+= 1$
10: control_chart.update(error)
11: **end function**

Another issue remains despite tackling the computational consumption challenge: labeled data scarcity and label noise issues. Because this proposed approach triggers a drift based on a specific region in the dataset where label-prediction conflict occurs, it tends to overestimate the severity of the drift as it assumes that the whole feature space has flipped its label. Resetting the model even though the drift only occurs in a specific data region is

wasteful and makes drift adaptation slower. Therefore, the method to deal with this issue is discussed in the next section.

4.2. Concept Drift Adaptation with Knowledge Transfer

Resetting classifiers to adapt to concept drift can be inefficient if the drift severity is low, as the classifier needs to relearn the entire concept from scratch. When a classifier encounters a low severity drift, it is best to preserve the overlapping concept to be reused for the new concept. This study proposes an ensemble approach that prunes the classifier members based on their estimated trustworthiness to produce accurate predictions.

The proposed metric to estimate the classifier's trustworthiness is shown in Equation (19), which this study refers to as the trustworthiness metric T_c. The trustworthiness metric T_c is the inverse of the sum of the classifier's age and the logarithm of the classifier's error rate. The proposed metric prioritizes removing old classifiers but also considers the estimated error rate. For example, an old classifier with a high error rate will be deleted sooner compared to a new classifier with a high error rate. The intuition is that an old classifier with a high error rate is more likely to be obsolete than a new classifier with a high error rate, which might be due to it still learning the new concept.

$$T_c = \frac{1}{Age_n + \log(\hat{E})} \tag{19}$$

However, the proposed metric requires the error rate to be estimated due to the lack of labeled data. This study refers to the works in an unsupervised error estimation approach by measuring the prediction disagreement of the classifier with the majority vote in the ensemble [81]. The error estimation algorithm and the calculation of the trustworthiness metric T_c are presented in Algorithm 3.

Algorithm 3: Trustworthiness of Classifiers in the Ensemble
Input: Data $x_1, \ldots, x_N$, Set of classifiers in the ensemble $f_1, \ldots, f_C$, Age of classifiers $Age_1, \ldots, Age_C$
Output: Trustworthiness of classifiers $T_1, \ldots, T_C$

function calculate_trustworthiness():
  $\hat{y}_1, \ldots, \hat{y}_N \leftarrow$ *predict_via_majority_vote*$(\mathbf{x}_1, \ldots, \mathbf{x}_N)$
  for c in $\{0, \ldots, C\}$:
    $\mathcal{A} \leftarrow \{\mathbf{x}_n | f_c(\mathbf{x_n}) \neq \hat{y}_n\}$ # set of data that the classifier f_c disagrees with the majority
    $\hat{E} \leftarrow |\mathcal{A}|/N$ # estimated error as the proportion of disagreement in the data
    $T_c \leftarrow$ use Equation (19) to calculate the trustworthiness of the classifier f_c
  return $T_1, \ldots, T_C$
end function

To create a diverse ensemble to improve performance, this study trains the ensemble via the online bagging approach [36] described in Algorithm 4. The λ parameter controls the diversity of the ensemble with a higher value creating a more diverse ensemble.

The full algorithm for adaptive semisupervised learning for manifold regularization is presented in Algorithm 5. The ensemble is updated at each data arrival using the online bagging approach (Algorithm 4). When a concept drift is detected, the 'drift' mode indicates that the model is currently adapting to the concept drift. During the 'drift' mode, the trustworthiness of the classifiers in the ensemble is updated, and the classifier with the lowest trustworthiness T_c will be removed. The 'drift' mode ends when the minimum ensemble size is reached.

Algorithm 4: Online Bagging
Input: Data $\mathbf{x}_1, \ldots, \mathbf{x}_N$, Classifier f_c, diversity parameter λ, training method to update classifier $training\_function(.)$
Output: Updated classifier f_c

function train_via_online_bagging():
  foreach $\mathbf{x}_1, \ldots, \mathbf{x}_N$:
    $k \leftarrow poisson(\lambda)$ #sample k from Poisson distribution λ
    repeat k times:
      $f_c \leftarrow training\_function(f_c, \mathbf{x}_n)$
  return f_c
end function

Algorithm 5: An Adaptive Prototype-Based Manifold Regularization Approach for Data Stream Mining
Input: Data Stream $\mathcal{D} = \{\mathcal{D}_t\}_{t=0}^{T=\infty}$, control method e.g., Page–Hinkley Test *control_chart*, ensemble set $f = \{f_1, \ldots, f_C\}$, minimum ensemble size *min_ensemble_size*

while (True)
  //Sequential learning
  Obtain partially labeled data $\mathcal{D}_t = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l), (\mathbf{x}_{l+1}), \ldots, (\mathbf{x}_u)\}_{i=1}^{N=l+u}$
  $S \leftarrow$ Process $\mathcal{D}_t$ using Algorithm 4.2 to create or update prototype set S
  $f_1, \ldots, f_c \leftarrow$ Update SOS-ELMs via manifold regularization method
  control_chart $\leftarrow$ update control chart statistics via Algorithm 5.1
  if $control\_chart.drift\_detected()$: // if concept drift is detected
    mode = 'drift'
    $f \cup \{f_1, \ldots, f_M\}$ #add M new classifiers and add it to the ensemble
  if mode == '**drift**':
    $T_1, \ldots, T_C \leftarrow$ calculate the trustworthiness of the classifiers using Equation (19)
    remove the classifier with the lowest T_c value
    if $C \leq min\_ensemble\_size$:
      mode = '**normal**'
end loop

5. Experimental Setup

Several benchmark algorithms were selected to evaluate and compare our proposed approach. The selected benchmark algorithms and its description are listed in Table 2. Some of the benchmark algorithms in Table 2 are fully supervised algorithms that require fully labeled data to function, which is suitable to investigate whether this approach can perform as well as the fully supervised approaches in adapting to concept drift. Comparing the performance of this proposed approach to the fully supervised approaches will provide evidence of whether this proposed approach can be a more practical alternative to the fully supervised approaches.

To evaluate the performance of concept drift datasets, we used several artificial datasets with defined concept drift locations or decision boundaries that can be manipulated to better evaluate concept drift compared to real-world datasets with unknown concept drift locations. Below is the list of the artificial datasets that will be used with their drift characteristics:

1. STAGGER concepts. STAGGER is an abrupt drift dataset that switches between three concepts by switching between three labeling rules. STAGGER has three boolean features, i.e., either 0 or 1.
2. Sine. Sine is also an abrupt drift dataset but has four different concepts. Its decision boundary resembles a sine wave function, making this dataset suitable to be evaluated on nonlinear decision boundaries.

In addition to these artificial datasets, we also used several mixed-severity concept drift datasets to evaluate the performance of our proposed approach on mixed-severity

concept drifts. These datasets are described in Table 3, consisting of four datasets with variables that can be manipulated to introduce concept drifts. The severity of the drifts can be manipulated based on the value of the variables before and after the concept drift.

Table 2. Benchmark Algorithms Used in this Study.

Algorithm	Reference	Description
Semisupervised Online Sequential—Extreme Learning Machine (SOS-ELM)	[54]	A version of Semisupervised—Extreme Learning Machine (SS-ELM) that learns sequentially.
Incremental Laplacian Regularized—Extreme Learning Machine (ILR-ELM)	[82]	An improvement in the SOS-ELM that could be updated without the requirement of each chunk having labeled data.
Semisupervised Online Elastic—Extreme Learning Machine (SSOE-ELM)	[83]	An adaptable version of the SOS-ELM. that can adapt to drift by adding a forgetting factor to remove old concepts from the model.
OzaBagAdwin	[14]	A derivation of the Online Bagging and Boosting Classifier that can handle nonstationary data streams by attaching the ADWIN classifier to the model.
Adaptive Random Forest (ARF)	[45]	Introduces diversity to the decision tree ensemble by selecting only a subset of features for each decision tree and assigning a drift detector per decision tree for adaptability to concept drifts.
Leveraging Bagging Classifier (LVB)	[84]	An ensemble version of the OzaBag classifier that also adds the ADWIN concept drift detector to the model for each decision tree.

Table 3. Artificial datasets on variable drift magnitude experiment.

Dataset	Equation	Fixed Variables	Before/After Variables	Drift Severity	Change Percent (%)
Circle	$(x-a)^2 + (y-b)^2 \leq r^2$	$a = 0.5$, $b = 0.5$	$r = 0.2 \rightarrow 0.3$	Low	16%
			$r = 0.2 \rightarrow 0.4$	Medium	38%
			$r = 0.2 \rightarrow .05$	High	66%
SineV	$y \leq a\sin(bx + c) + d$	$a = 1$, $b = 1$, $c = 0$	$d = -2 \rightarrow 1$	Low	15%
			$d = -5 \rightarrow 4$	Medium	45%
			$d = -8 \rightarrow 7$	High	75%
SineH	$y \leq a\sin(bx + c) + d$	$a = 5$, $d = 5$, $b = 1$	$c = 0 \rightarrow -\pi/4$	Low	36%
			$c = 0 \rightarrow -\pi/2$	Medium	57%
			$c = 0 \rightarrow -\pi$	High	80%
Plane	$y \leq -a_0 + a_1x_1 + a_2x_2$	$a_1 = 0.1$, $a_2 = 0.1$,	$a_0 = -2 \rightarrow -2.7$	Low	14%
			$a_0 = -1 \rightarrow -3.2$	Medium	44%
			$a_0 = -.7 \rightarrow -4.4$	High	74%

In addition to the artificial datasets, real-world datasets will also be used to evaluate this approach's performance in real-world environments. Table 4 lists and describes the selected real-world datasets used in this study. The selected real-world datasets were obtained from sensors or IoT devices, which is the target application of this proposed approach. These datasets also contain common challenges that applications such as IoT might encounter, such as noisy data and class imbalance. The final column of Table 4 describes the severity of the class imbalance as the imbalance ratio between the minority

and the majority class. For datasets with nonbinary classes, we summed the classes with the fewest samples and divided them by the sum of the other classes to obtain the imbalance ratio. As described in Table 4, most datasets have some class imbalance except for the Sensorless Drive dataset, where all classes have the same amount of samples.

Table 4. Real-world datasets were used in this study.

Dataset	#Samples	#Features	#Classes	#Training Samples	#Test Samples	#Training Steps	#Labeled Samples	Data Characteristics	Imbalance Ratio
Sensorless Drive	58,509	49	11	46,807	11,701	469	2340	High Dimensional	-
Magic Gamma	19,020	11	11	15,216	3804	152	760	Noisy	46:54
Human Activity Recognition (HAR)	10,299	561	6	5881	4418	58	290	Noisy/High Dimensional	1:99
Crop Mapping	325,834	175	7	60,000	65,167	600	3000	High Dimensional	1:99
KDDCup 1999	494,021	42	12	50,000	98,805	500	2500	Noisy/High Dimensional	1:99
Physical Activity Monitoring (PAMAP2)	1,942,872	52	12	50,000	388,575	500	2500	Noisy/High Dimensional	8:92

The following experiments and evaluations were carried out using the datasets and baselines that were specified:

1. Abrupt drift adaptation evaluation. This experiment evaluates the performance of this approach against other baselines on abrupt drift (via the STAGGER and Sine) datasets, where the concept changes rapidly. This is to investigate whether this approach can detect changes to the concept and adapt to the changes quickly.
2. Mixed magnitude drifts. This experiment is more challenging than the abrupt drift evaluation as it has a mixture of high- and low-severity drifts. The mixed severity of the drift makes it important to tailor the adaptation method based on its severity. Therefore, it allows this experiment to evaluate the performance of the proposed transfer learning adaptation approach.
3. Overall performance ranking evaluation. This evaluation aggregates all the experiments' performances and analyzes their overall ranking. This evaluation aims to analyze whether there is any significant improvement in the comparable manifold regularization approaches and if it can provide a reasonable alternative to the fully supervised approaches. This evaluation expects that this approach significantly improves the performance of the manifold regularization approaches while no significant difference in performance is observed compared to the supervised approaches.

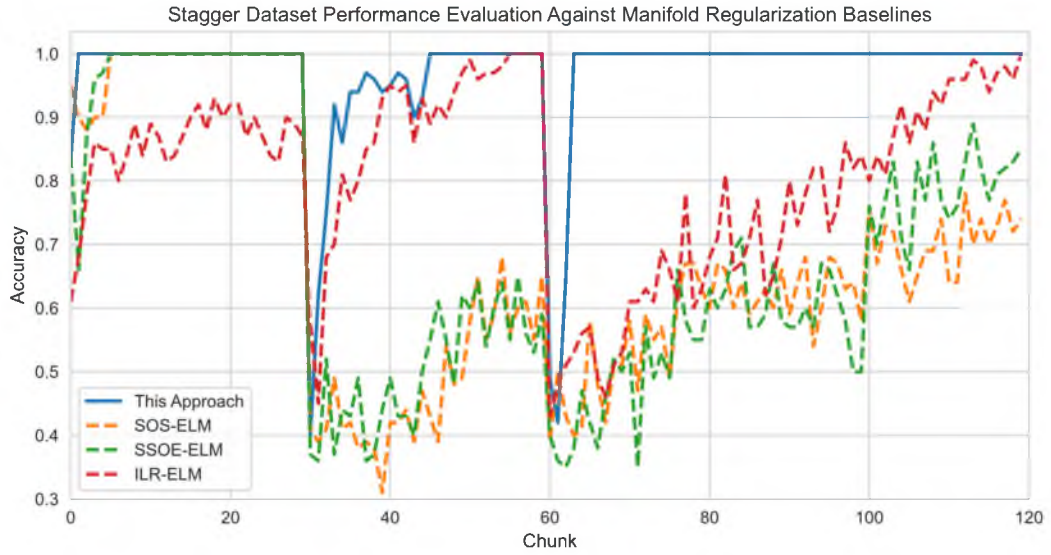
The experiments were conducted on a platform configured with the Intel i5 processor clocked at 2.40 GHz with 12 GB of RAM using the Python programming language. The OzaBagAdwin, ARF and LVB classifiers were implemented using the Scikit-Multiflow Python library [85]. All of the experiments were executed on the Jupyter Notebook software.

6. Results

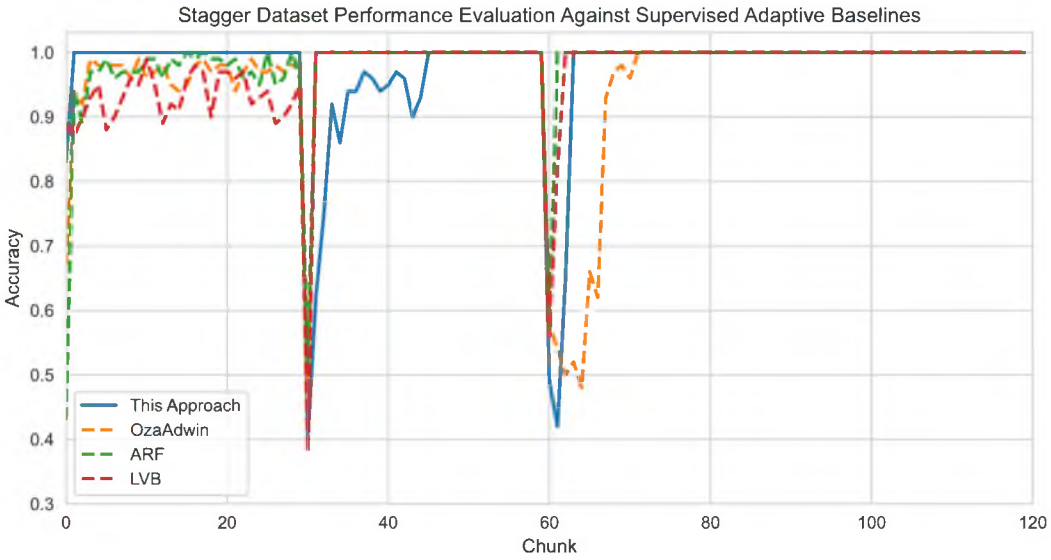
6.1. Abrupt Drift Evaluation

We first evaluated the performance of this proposed approach on abrupt drift challenges using the Stagger dataset. The Stagger dataset contains three concept drifts triggered every 30th data chunk interval with 100 samples in each chunk. This experiment's results are presented in Figure 1a,b. Figure 1a shows the comparison of this approach against the

manifold regularization approaches, while Figure 1b compares this approach against the supervised approaches.



(**a**)



(**b**)

Figure 1. Performance evaluation on Stagger dataset. (**a**) The performance of this proposed approach against other manifold regularization approaches. (**b**) The performance of this proposed approach against other fully supervised approaches.

Figure 1a clearly shows that this proposed approach performed well and was more adaptable than the current manifold regularization approaches by detecting and adapting to concept drifts. During the first concept drift, the proposed approach managed to detect the concept drift, adapt the model, and regain optimum classification to 100% accuracy faster than the other approaches. On the other hand, approaches such as SOS-ELM and SSOE-ELM managed to recover only to 60% accuracy before the next concept drift occurred. During the second concept drift, this proposed approach also managed to detect and adapt to concept drift and regained the maximum performance of 100% accuracy, whereas other approaches were slower to recover from the concept drifts.

The results presented in Figure 1a were expected as the manifold regularization approaches require significantly more time to adapt to drifts as it needs to overwhelm the

previous concept in the model before the new concept can be learnt by not being able to detect drifts and reset its classifier. As a result, current manifold regularization approaches are less adaptable to this proposed approach.

Comparing this proposed approach with the fully supervised approaches, this approach was as adaptable to the fully supervised approaches as shown in Figure 1b. This was indicated by this proposed approach having almost simultaneous concept drift detection points and adapting almost as well as the fully supervised approaches. As shown in Figure 1b, during the first concept drift, the proposed approach detected the concept drift simultaneously with the other fully supervised approaches, indicating that this approach managed to detect concept drift as well as the fully supervised approaches. Despite this, there were some performance deficits, particularly after the first concept drift occurred, which might have been due to the weakness of SSL, which can be slow to learn due to the lack of labeled data. During the second concept drift, the proposed approach detected the concept drift slightly later than the fully supervised approaches at the 61st chunk compared to the 60th chunk, where the concept drift was set to occur. However, this was an acceptable performance as the concept drift was still detected close to the true time that the concept drift occurred.

Next, the performance on abrupt drift performance was evaluated on the Sine dataset; it was more challenging to provide more evidence of this approach's ability to detect and adapt to concept drifts. The Sine dataset evaluation results are presented in Figure 2a,b. Like the Stagger dataset experiment, 30 chunks of data with 100 samples were used to generate each concept.

Figure 2a,b show a similar outcome to the Stagger dataset experiment, in which this approach was more adaptable than the baseline manifold regularization approaches and was as adaptable to the fully supervised approaches. These results show that this proposed approach could perform well by detecting and adapting to drifts even with complex decision boundaries. Comparing this proposed approach with the semisupervised approaches, Figure 2a shows that this proposed approach managed to detect the concept drift close to where it occurred, allowing it to adapt and regain its maximum performance of around 90%. On the other hand, the semisupervised approaches only achieved 60% accuracy when the next concept drift occurred. However, because the semisupervised approach did not finish learning before the second concept drift occurred, it still had some plasticity in the model, allowing it to learn the new concept easier, giving it an even performance to our proposed approach. However, once the third concept drift occurred, the supervised approach lost its accuracy and adapted slower than the proposed approach, only attaining 50% accuracy compared to 83% accuracy for the proposed approach.

Despite the good performance against the semisupervised approaches, this proposed approach performed slightly poorer than the supervised approaches on the last two concepts shown in Figure 2b. This might have been due to the smoothness assumption being violated by the complex decision boundary of the Sine dataset. This was most obvious during the second and third concepts as this proposed approach only attained around 83% accuracy, compared to the 96% accuracy that the fully supervised approach obtained. Nevertheless, we achieved the objective of our study to detect concept drift as well as fully supervised approaches by detecting the concept drift close to its point of occurrence.

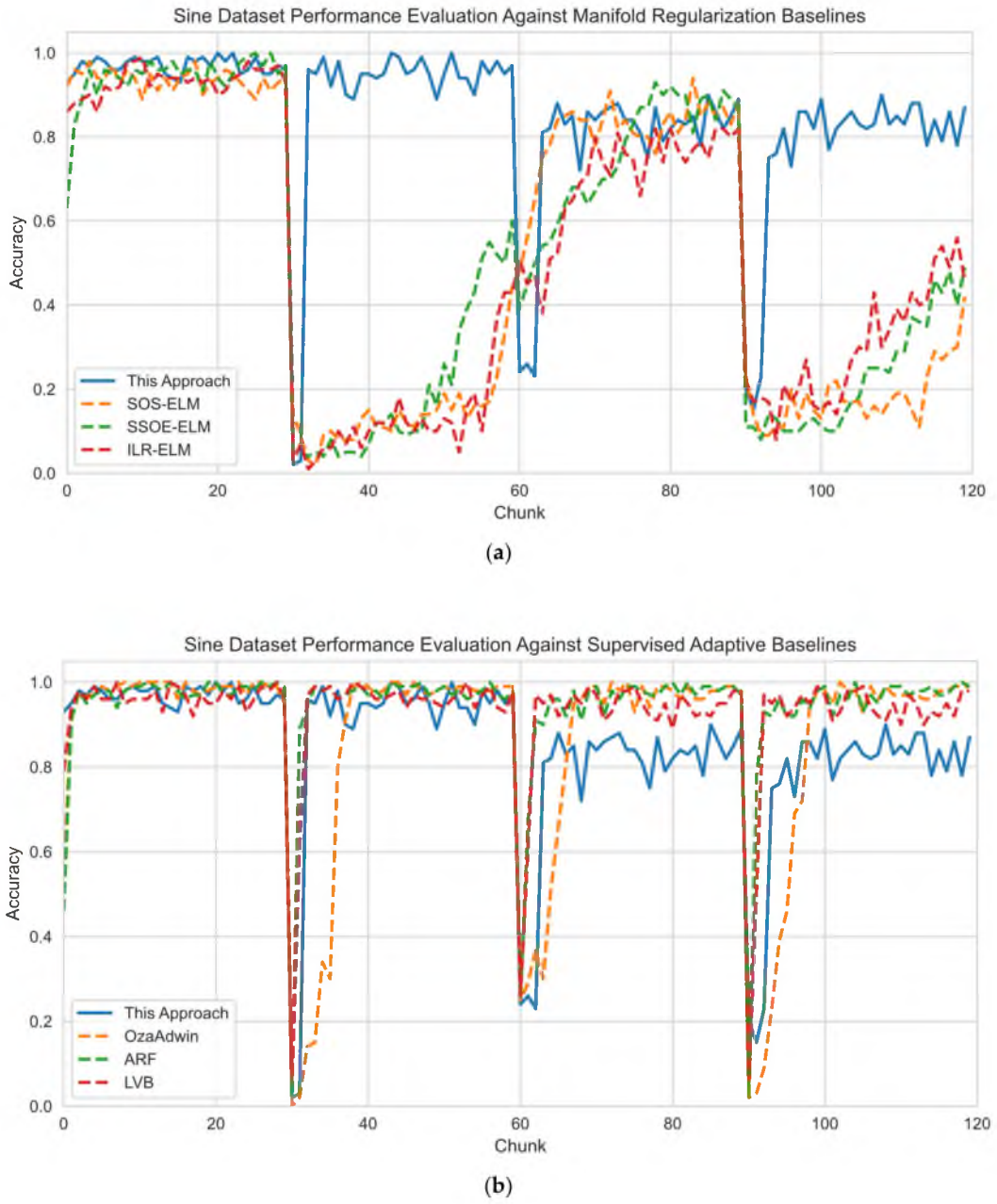


Figure 2. Sine dataset performance evaluation. (**a**) The performance of this proposed approach against other manifold regularization approaches. (**b**) The performance of this proposed approach against other fully supervised approaches.

6.2. Mixed Drift Magnitude Evaluation

Based on the datasets described in Table 3, the results for the variable drift magnitude are presented in this section. This proposed approach will be compared with other manifold regularization approaches by resetting the classifier to adapt to concept drifts. Each dataset in Table 3 triggers the concept drifts at the 20th chunk, with 100 samples for each chunk.

Figure 3 shows the result for the Circle dataset, which shows that this proposed approach managed to recover from concept drifts quickly. In Figure 3a, when the drift severity was low, preserving the classifiers from the previous concept and pruning the outdated classifiers allowed knowledge that can be reused in the new concept to be used to adapt to the new concept to minimize the drop in performance. Figure 3a shows that this proposed approach adapted better when concept drift occurred than the baseline approaches. For example, during the 23rd chunk, this proposed approach achieved 95%

accuracy compared to other approaches that only achieved less than 85% accuracy. The superiority of this proposed approach can also be seen clearly between the 30th and the 35th chunk, where this proposed approach consistently achieved higher performance than other approaches by consistently achieving more than 95% accuracy.

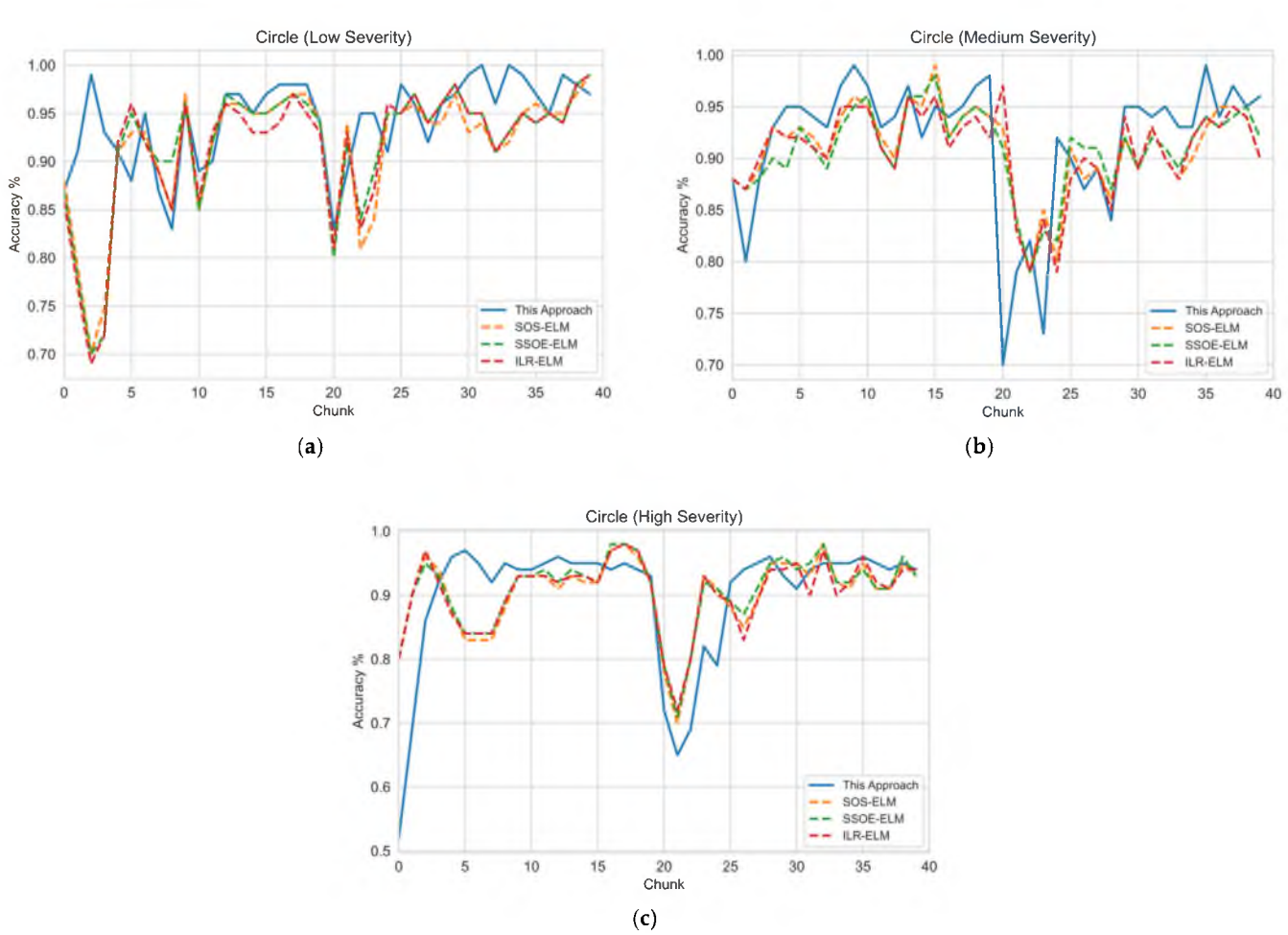


Figure 3. Circle dataset experiment on variable concept drift severity. (**a**) Performance on low severity dataset. (**b**) Performance on medium severity dataset. (**c**) Performance on high severity dataset.

For the medium severity experiment of the Circle dataset, Figure 3b shows that this proposed approach had a quite severe drop in performance during concept drift. This was caused by this proposed approach removing old concepts from the ensemble, causing a significant drop in performance. However, this turned out beneficial for the model in the long term, as, after the 29th chunk, this proposed approach consistently achieved more than 90% accuracy compared to the other approaches.

For the high severity experiment in Figure 3c, due to the high severity of the concept drift, this proposed approach was less adaptable than other baseline approaches as it could not prune the outdated classifiers quickly enough to adapt to the drift. However, despite not overtaking other semisupervised approaches, employing an ensemble-based approach in this proposed approach allowed the performance to be more stable compared to the baseline approaches by consistently achieving more than 90% accuracy.

In Figure 4, the results for the SineV dataset are presented, which shows that this proposed approach maintained a good accuracy when a low severity drift occurred. In Figure 4a, the result of the low severity drift indicated that this proposed approach managed to avoid a significant drop in accuracy between the 20th and 23rd data chunks, maintaining around 93% accuracy after the drift occurred. For the medium severity drift in Figure 4b,

there was a drop in the performance of this proposed approach. However, as discussed in the results for Figure 3b, the drop in performance was temporary as this proposed approach achieved a superior performance between the 25th and 30th chunks. Beyond the 30th chunk, this proposed approach had a more stable performance due to the ensemble approach, by having more than 95% accuracy consistently. However, as in the Circle dataset, when the drift was severe, this proposed approach failed to adapt faster than other baseline approaches as it could not remove the classifiers quickly compared to other baseline approaches. Despite this, this proposed approach eventually overtook all baseline approaches except ILR-ELM by achieving more than 80% accuracy.

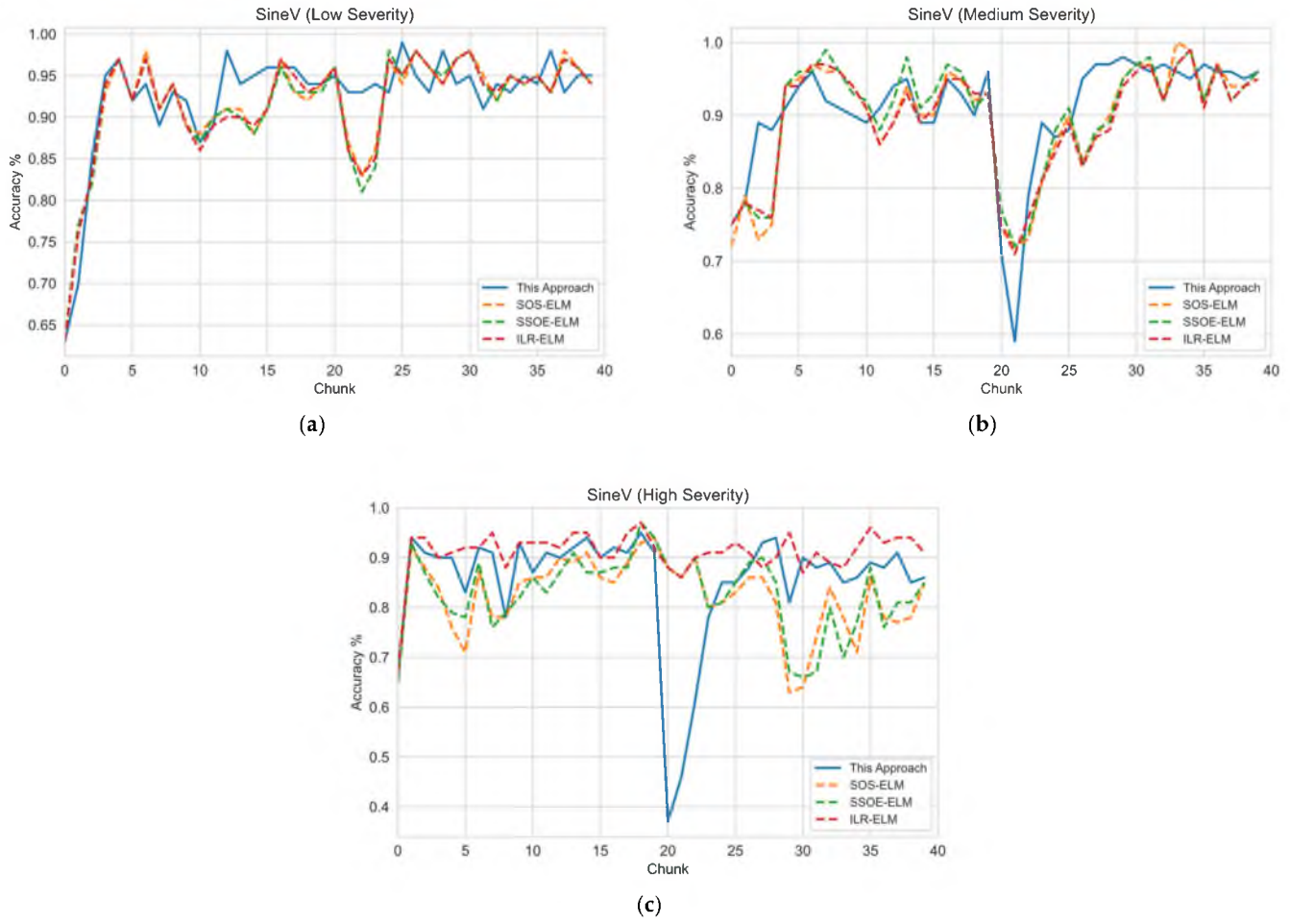


Figure 4. SineV dataset experiment on variable concept drift severity. (**a**) Performance on low severity dataset. (**b**) Performance on medium severity dataset. (**c**) Performance on high severity dataset.

In Figure 5, the results from the SineH dataset showed a more promising result by being more adaptable to all severity levels than the baseline approaches. Figure 5a shows that this proposed approach had a less severe drop than the baseline approaches and recovered more quickly after concept drift occurred than the other baseline approaches. As seen between the 20th and 26th data chunks, this proposed approach achieved more than 92% accuracy. Beyond the 26th chunk, all approaches performed similarly. When the concept drift had medium severity, Figure 5b shows that this proposed approach adapted faster than the other approaches immediately after the concept drift occurred at the 20th chunk. For high-severity concept drift, contrary to other datasets, this proposed approach adapted faster than other baseline approaches and consistently outperformed other baseline approaches until the 25th data chunk. After the 25th data chunk, all approaches performed similarly.

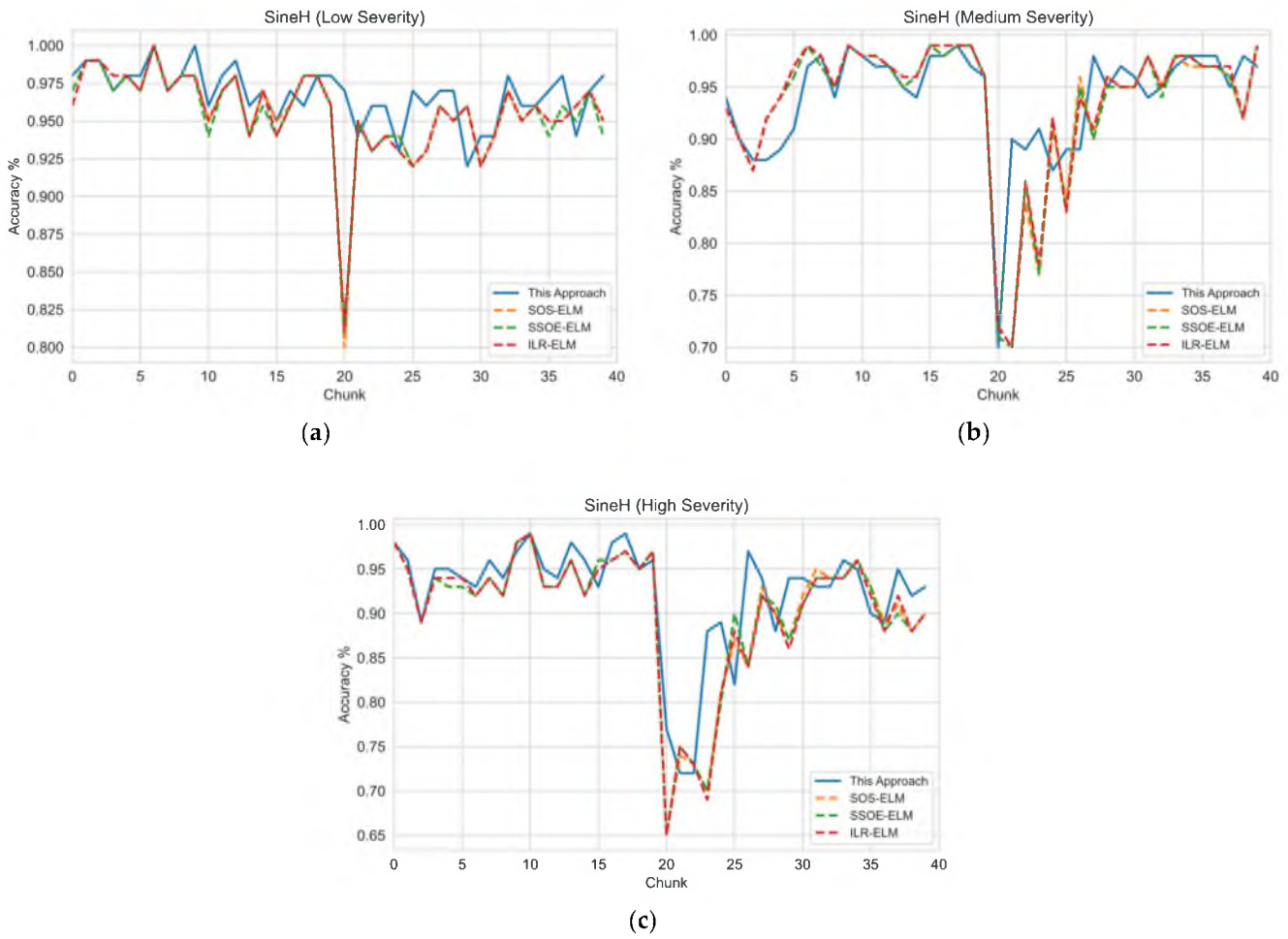


Figure 5. SineH dataset experiment on variable concept drift severity. (**a**) Performance on low severity dataset. (**b**) Performance on medium severity dataset. (**c**) The performance on high severity dataset.

Similar to the SineH dataset, the Plane dataset also showed that this proposed approach performed well on all severity levels by being more adaptable than the baseline approaches. The results from Figure 6a–c show that this proposed approach maintained its accuracy when the concept drift occurred and recovered better than the baseline approaches after the concept drift occurred. This was observed in Figure 6a for low severity drift, which showed that this proposed approach had a less severe drop in accuracy compared to the other approaches that dropped below 60% accuracy. After the concept drift, Figure 6a shows that this proposed approach consistently achieved more than an 85% accuracy. In the medium severity drift shown in Figure 6b, this proposed approach had a higher average performance compared to other approaches by having an average classification performance at around 82% accuracy, whereas other approaches had drops in performance as low as 65% accuracy. Finally, when the concept drift had high severity, Figure 6c shows that this proposed approach had a less severe drop compared to other baseline approaches that dropped as low as 70% accuracy. The proposed approach also learned from the new concept faster than the other approaches, which was obvious between the 22nd and the 25th chunk as this approach consistently increased its accuracy compared to other baseline approaches.

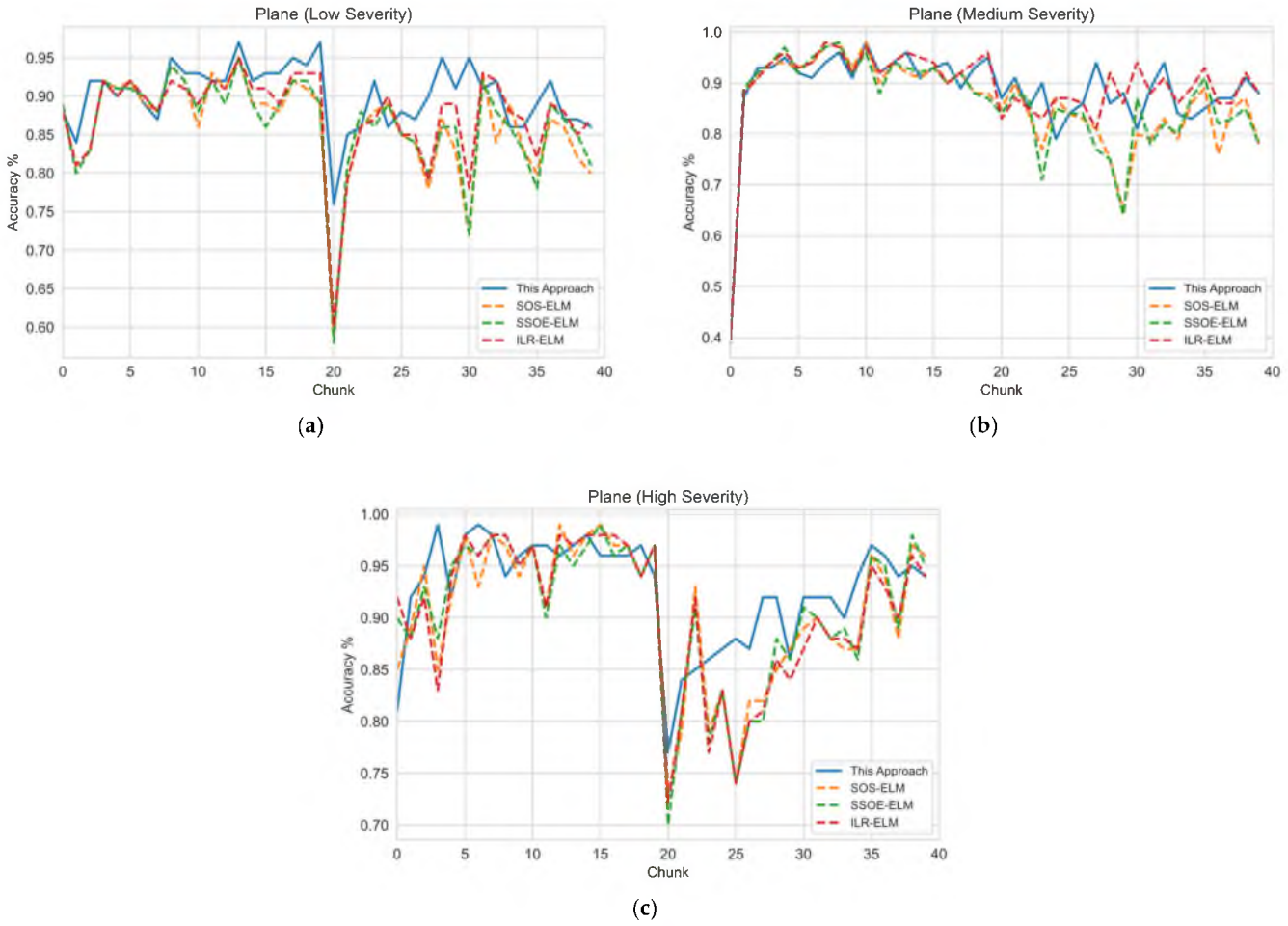


Figure 6. Plane dataset experiment on variable concept drift severity. (**a**) Performance on low severity dataset. (**b**) Performance on medium severity dataset. (**c**) Performance on high severity dataset.

The results in Figures 3–6 show that this proposed approach performed well in low and medium severity drifts by preserving the knowledge from the previous concept. Despite this, the proposed approach may perform poorly when the concept drift has a very high severity. However, this study argues that this proposed approach is a tradeoff between prioritizing performance in low or medium drift severity versus high drift severity. This is because this proposed approach is the midpoint between completely using previous classifiers prior to drift, which is more suitable for low and medium drift severity and resetting the classifier that is more suitable for high severity drifts. Therefore, this proposed approach could be considered in an environment where low and medium severity drift is expected.

6.3. Real-World Dataset Performance Evaluation

This section compares this proposed approach with the baseline approaches based on their performance in each dataset. Both artificial and real-world datasets will be used for this evaluation to analyze their ability to adapt to drifts and tackle challenges in real-world datasets, such as noise and class imbalance. As the real-world datasets contain class imbalance, the F1 measure will be used to take class imbalance into account in the performance measure. The mean F1 measure throughout the stream was obtained for each dataset and is tabulated in Table 5. The entries that are bolded are the top semisupervised approaches based on the experiments.

Table 5. Performance comparison of this approach compared with other related semisupervised learning techniques and adaptive supervised learning approaches.

Approach	Label Size	This Approach	SOS-ELM	SSOE-ELM	ILR-ELM	OzaBag Adwin	ARF	LVB
Stagger	1	**0.8556**	0.6195	0.6937	0.735	0.9569	0.9828	0.9737
	5	**0.9653**	0.6756	0.6818	0.809	0.9569	0.9828	0.9737
	10	**0.9566**	0.6991	0.6822	0.8298	0.9569	0.9828	0.9737
Average		0.93	0.66	0.69	0.79	0.96	0.98	0.97
Average Rank		3.67	6.67	6.33	5.00	3.33	1.00	2.00
Sine	1	0.428	0.5394	**0.553**	0.5135	0.8827	0.9433	0.9309
	5	**0.7905**	0.5318	0.5168	0.5293	0.8827	0.9433	0.9309
	10	**0.8653**	0.5192	0.5207	0.526	0.8827	0.9433	0.9309
Average		0.69	0.53	0.53	0.52	0.88	0.94	0.93
Average Rank		5.00	5.67	5.67	5.67	3.00	1.00	2.00
$Hyperplane_{0.001}$	1	0.5845	**0.6809**	0.6121	0.6657	0.7649	0.776	0.6628
	5	**0.783**	0.753	0.6071	0.7128	0.7649	0.776	0.6628
	10	**0.7875**	0.753	0.6071	0.7128	0.7649	0.776	0.6628
Average		0.72	0.73	0.61	0.70	0.76	0.78	0.66
Average Rank		3.00	3.67	6.67	4.67	2.67	1.67	5.67
$Hyperplane_{0.01}$	1	0.6781	**0.7005**	**0.7005**	0.6667	0.7692	0.7901	0.6656
	5	**0.7799**	0.763	0.763	0.7088	0.7692	0.7901	0.6655
	10	**0.7792**	0.7747	0.7747	0.731	0.7692	0.7901	0.6657
Average		0.75	0.75	0.75	0.70	0.77	0.79	0.67
Average Rank		3.00	3.83	3.83	6.00	3.33	1.00	7.00
HAR	1	0.7303	**0.7506**	0.7381	0.7428	0.8675	0.7091	0.8023
	5	**0.8621**	0.8437	0.8376	0.8596	0.8675	0.6862	0.8023
	10	0.8621	0.8437	**0.8683**	0.8597	0.8675	0.6862	0.8023
Average		0.82	0.81	0.81	0.82	0.87	0.69	0.80
Average Rank		3.67	4.00	3.67	3.67	1.33	7.00	4.67
Magic Gamma	1	0.7389	**0.7393**	0.7261	0.6892	0.2452	0.8363	0.7375
	5	**0.7928**	0.79	0.7022	0.7458	0.2452	0.8363	0.7375
	10	**0.8085**	0.7948	0.6715	0.7531	0.2452	0.8363	0.7375
Average		0.78	0.77	0.70	0.73	0.25	0.84	0.74
Average Rank		2.33	2.67	5.67	4.67	7.00	1.00	4.67
PAMAP2	1	**0.5643**	0.4682	0.4568	0.4633	0.7027	0.8226	0.5817
	5	**0.7216**	0.727	0.6733	0.7037	0.7027	0.8226	0.5817
	10	**0.798**	0.7936	0.7789	0.777	0.7027	0.8226	0.5817
Average		0.69	0.66	0.64	0.65	0.70	0.82	0.58
Average Rank		3.00	3.33	5.67	5.00	4.33	1.00	5.67
Sensorless Drive	1	0.6618	0.6687	0.5054	**0.6928**	0.8536	0.8905	0.6928
	5	0.78	0.7814	0.7193	**0.8092**	0.8536	0.8905	0.6928
	10	0.8225	0.8185	0.8195	**0.8481**	0.8536	0.8905	0.6928
Average		0.75	0.76	0.68	0.78	0.85	0.89	0.69
Average Rank		5.00	5.00	6.00	3.17	2.00	1.00	5.83
KDDCup1999	1	**0.9926**	0.9924	0.5054	0.9923	0.9881	0.9955	0.985
	5	0.9926	0.9924	0.7193	0.9923	0.9881	0.9955	0.985
	10	0.9957	0.9833	0.8195	0.9848	0.9881	0.9955	0.985
Average		0.99	0.99	0.68	0.99	0.99	1.00	0.99
Average Rank		1.67	4.00	7.00	4.33	4.33	1.33	5.33
Overall Average Accuracy		0.7911	0.74	0.6767	0.7422	0.7811	0.8589	0.7811
Overall Average Ranking		3.37	4.31	5.61	4.69	3.48	1.78	4.76

As expected, this approach ranked lower on average than the fully supervised approaches. This was because of the advantage of the fully labeled dataset that the supervised approaches had access to, which made them perform better than this approach, which only had a partially labeled dataset. However, this is acceptable as long as the fully supervised approach does not significantly outrank this approach. The ranking significance can be ana-

lyzed via the Friedman–Nemenyi post hoc statistical test ($\alpha > 0.05$) to determine if the fully supervised approaches significantly outranked this proposed approach. The results are presented as the Critical Distance (CD) diagram shown in Figures 7 and 8 for the artificial and real-world datasets, respectively. The CD diagram helps to visualize the significance of the ranking between the approaches. Each approach is plotted on the average ranking obtained. Approaches that did not have significant differences in the ranking are connected on the line.

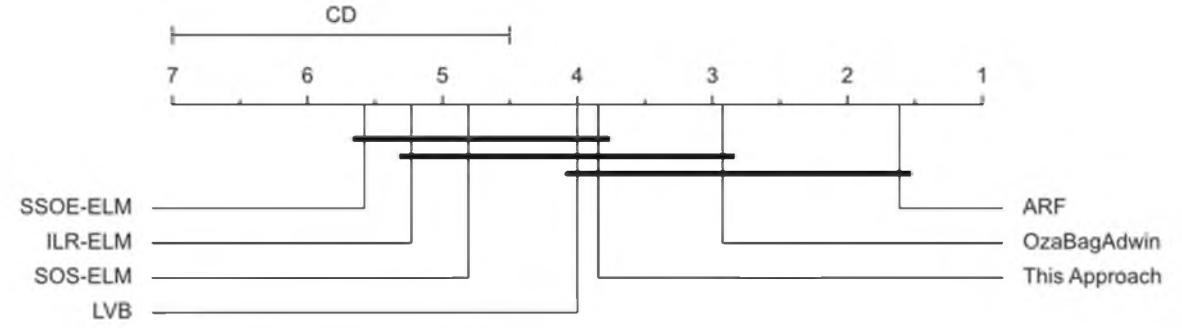


Figure 7. Critical Distance diagram of performance ranking comparison in artificial concept drift dataset.

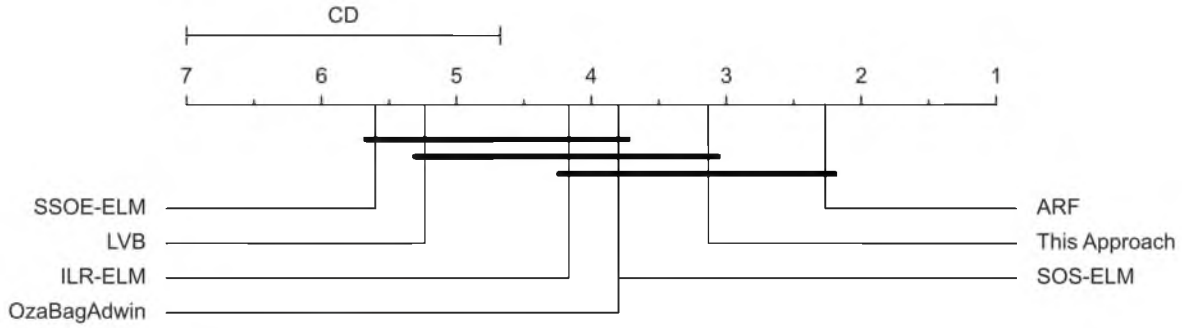


Figure 8. Critical Distance diagram of performance ranking comparison in real-world dataset.

Figure 7 confirms that this approach ranked below the fully supervised approaches except for LVB. However, according to the CD diagram in Figure 7, the fully supervised approaches did not significantly outrank this proposed approach, which suggests that this approach performed on par with the fully supervised approaches. Nevertheless, this approach did perform significantly better than the baseline manifold regularization approaches, which was the goal of this study.

Figure 8 shows that this approach ranked higher on average for the real-world datasets' ranking analysis than all the supervised approaches except ARF. By ranking higher than the OzaBagAdwin approach, this approach gained a place in the overall ranking compared to the artificial datasets' performance ranking. A possible explanation for this approach's increased ranking in real-world datasets is that this approach performs better on real-world datasets as the fully supervised approaches only use simple decision tree classifiers that perform poorly on high-dimensional datasets. In addition, this approach was not significantly outranked by the ARF approach but managed to outrank the SSOE-ELM semisupervised and LVB approaches significantly.

The key findings from both statistical tests are that despite the fully supervised approaches generally performing better, it did not perform significantly better than this approach. This suggests that this approach can provide a reasonable alternative to the fully supervised approaches that are more practical in terms of labeled data.

6.4. Execution Time Analysis

Because this proposed approach is targeted for data stream mining applications, it is important to have insight into the computational cost of this proposed approach compared with other approaches. In this section, the computational costs for each approach will be analyzed by measuring the time in seconds (s) to execute its model update. The results are presented in Figure 9a–h for every dataset, indicating the average time to execute the model update for each approach.

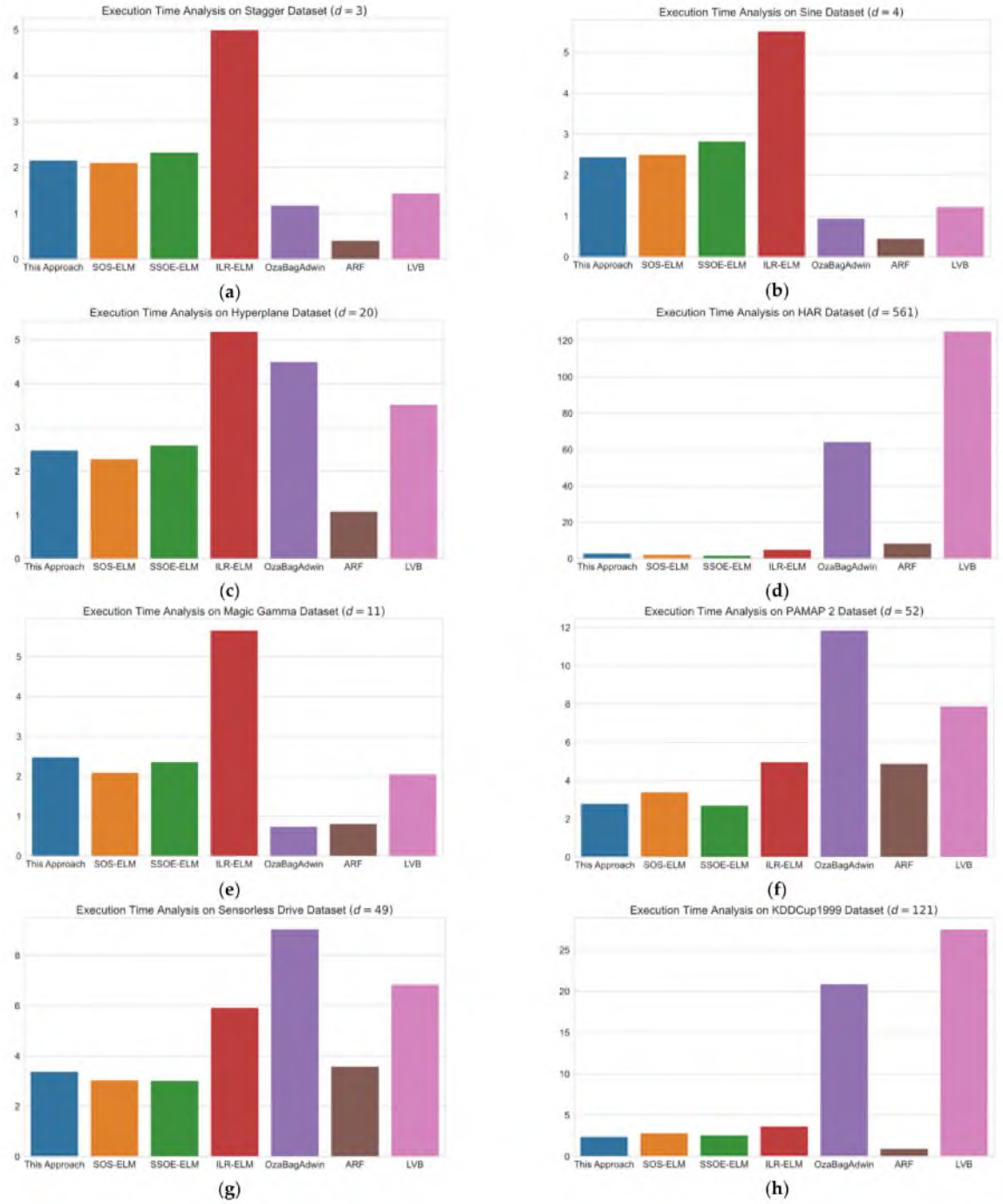


Figure 9. Comparison of execution time of this approach and different baselines. (**a**–**h**) The results of the execution time from the dataset listed in Tables 3 and 4, where d is the dimensionality of the dataset.

For lower-dimensional datasets such as Stagger and Sine, this approach and other manifold regularization approaches had a longer execution time than the fully supervised approaches. However, this changed as the dimension of the data increased. This approach executed faster than fully supervised approaches in higher dimensional datasets. This is shown in Figure 7a–c, which were lower dimensional datasets, indicating that this approach required a longer time to update its parameters. However, in Figure 9d–h, the ordering was reversed as the dimension of the data was higher by this approach, and other manifold regularization approaches had a shorter execution time than the supervised baselines.

Based on the execution time analysis, this proposed approach is more scalable than the supervised approaches for higher-dimensional datasets. This adds another point of practicality in addition to the reduced labeling requirement by executing in a reasonable amount of time for high-dimensional datasets.

7. Discussion

The experiments carried out were performed to evaluate and analyze the ability of this approach to adapt to drifts and handle the challenges of real-world datasets and their computational consumption compared with the baseline approaches. In summary, based on these experiments, this proposed approach outperformed baseline manifold regularization approaches in adapting to concept drifts but also performed on par with the supervised approaches. At the same time, this proposed approach performed well on real-world datasets and could be updated efficiently within a reasonable execution time.

Using artificial datasets, two types of concept drift situations, abrupt drift and irregular drift, were simulated. The findings in Sections 6.1 and 6.2 show that by being able to detect abrupt drifts and react to the drifts, this approach managed to adapt to drifts better than the manifold regularization approaches, which needs to adapt by overwhelming the old concept in the model with the new concept. Meanwhile, this proposed approach also performed on par despite a slight delay in detecting the drifts due to the lack of labeled data to which this approach has access to, making it slightly less sensitive to drifts. Nevertheless, this approach managed to recover from drifts quickly once the drift was detected.

The performance of this proposed approach was investigated further by analyzing the overall ranking of this proposed approach compared with the baseline approaches. Based on the ranking analysis conducted, this proposed approach ranked lower on average than most of the supervised approaches on concept drift, which was expected. However, the difference in ranking was not significant, suggesting that this proposed approach can provide a strong alternative that is more practical than the supervised approaches.

The computational cost was investigated by measuring the time it took to update the model to investigate this proposed approach's practicality. The findings in Section 6.4 show that this proposed approach initially had a longer execution time than the supervised approach on lower-dimensional datasets. However, as the dimension of the data increased, this approach had a lower execution time compared with the supervised approaches. This experiment showed that this approach is scalable and practical for high-dimensional datasets.

8. Conclusions

This study proposes a prototype-based concept drift detection and concept drift adaptation that attempts to address the difficulty of detecting and adapting to concept drift when there is a lack of labeled data. The proposed approach was evaluated on simulated datasets with concept drift and real-world datasets. Based on the results, the proposed concept drift detection managed to detect the presence and other fully supervised concept drift detection despite only having partially labeled data in each chunk. For concept drift adaptation, the proposed approach adapted well to low- and medium-severity concept drifts. However, in some datasets, the adaptation to high-severity drifts can be poor due to this approach retaining some classifiers after concept drift occurs. In future works, we

aim to address this issue by making this approach able to adapt to high-severity drifts successfully by minimizing the drop in accuracy and making the adaptation faster.

Author Contributions: Conceptualization, M.Z.M.Z.S. and A.Z.; methodology, M.Z.M.Z.S., A.Z. and F.A.G.; validation, M.Z.M.Z.S. and A.Z.; formal analysis, M.Z.M.Z.S. and A.Z.; writing—original draft preparation, M.Z.M.Z.S., A.Z. and F.A.G.; writing—review and editing, F.A.G., A.Z., H.A. and T.A.E.E.; visualization, M.Z.M.Z.S., A.Z., H.A., T.A.E.E. and F.A.G.; supervision, A.Z., F.A.G. and H.A.; funding acquisition, T.A.E.E. and H.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research has been supported by the funding from the Deanship of Scientific Research at King Khalid University, Large Groups. (Project under grant number (RGP.2/49/43)).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: For more information on the datasets used in the experiments, please visit:

(1) Sensorless Drive Diagnosis data set, 2015. UCI Machine Learning Repository: Data set for Sensorless Drive Diagnosis Data Set;
(2) MAGIC Gamma Telescope data set, 2007. UCI Machine Learning Repository: MAGIC Gamma Telescope Data Set;
(3) Human Activity Recognition (HAR) data set, 2012. UCI Machine Learning Repository: Human Activity Recognition Using Smartphones' Data Set;
(4) Crop Mapping Using Fused Optical Radar data set, 2020. UCI Machine Learning Repository: Crop mapping using fused optical-radar data set;
(5) Knowledge Discovery and Data Mining Competition (KDDCup) data set, 1999. KDD Cup 1999 Data (uci.edu);
(6) Physical Activity Monitoring (PAMAP2) data set, 2012. UCI Machine Learning Repository: PAMAP2 Physical Activity Monitoring Data Set;

Acknowledgments: The authors would like to acknowledge the Young Academic Training Scheme Scholarship (SLAM) for sponsoring the first author of this study. In addition, the authors extend their appreciation to the Deanship of Scientific Research at King Khalid University for funding this work through Large Groups. (Project under grant number (RGP.2/49/43)).

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

References

1. Aljaaf, A.J.; Al-Jumeily, D.; Hussain, A.J.; Dawson, T.; Fergus, P.; Al-Jumaily, M. Predicting the likelihood of heart failure with a multi level risk assessment using decision tree. In Proceedings of the 2015 Third International Conference on Technological Advances in Electrical, Electronics and Computer Engineering (TAEECE), Beirut, Lebanon, 29 April–1 May 2015.
2. Li, J.; Stones, R.J.; Wang, G.; Liu, X.; Li, Z.; Xu, M. Hard drive failure prediction using Decision Trees. *Reliab. Eng. Syst. Saf.* **2017**, *164*, 55–65. [CrossRef]
3. Ko, Y.H.; Hsu, P.Y.; Cheng, M.S.; Jheng, Y.R.; Luo, Z.C. Customer Retention Prediction with CNN. In *Data Mining and Big Data*; Springer Singapore: Singapore, 2019.
4. De Caigny, A.; Coussement, K.; De Bock, K.W.; Lessmann, S. Incorporating textual information in customer churn prediction models based on a convolutional neural network. *Int. J. Forecast.* **2020**, *36*, 1563–1578. [CrossRef]
5. De Francisci Morales, G.; Bifet, A.; Khan, L.; Gama, J.; Fan, W. Iot big data stream mining. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, 14–18 August 2016.
6. Krempl, G.; Žliobaite, I.; Brzeziński, D.; Hüllermeier, E.; Last, M.; Lemaire, V.; Noack, T.; Shaker, A.; Sievi, S.; Spiliopoulou, M.; et al. Open challenges for data stream mining research. *ACM SIGKDD Explor. Newsl.* **2014**, *16*, 1–10. [CrossRef]
7. Mala, A.; Dhanaseelan, F.R. Data stream mining algorithms: A review of issues and existing approaches. *Int. J. Comput. Sci. Eng.* **2011**, *3*, 2726–2732.
8. Homayoun, S.; Ahmadzadeh, M. A review on data stream classification approaches. *J. Adv. Comput. Sci. Technol.* **2016**, *5*, 8–13. [CrossRef]
9. Alothali, E.; Alashwal, H.; Harous, S. Data stream mining techniques: A review. *Telkomnika* **2019**, *17*, 728–737. [CrossRef]
10. Iwashita, A.S.; Papa, J. An Overview on Concept Drift Learning. *IEEE Access* **2019**, *7*, 1532–1547. [CrossRef]
11. Agrahari, S.; Singh, A. Concept drift detection in data stream mining: A literature review. *J. King Saud Univ. Comput. Inf. Sci.* **2021**, *34*, 9523–9540. [CrossRef]
12. Gaber, M.M.; Zaslavsky, A.; Krishnaswamy, S. Mining data streams: A review. *ACM Sigmod Rec.* **2005**, *34*, 18–26. [CrossRef]

13. Huang, G.B.; Liang, N.Y.; Rong, H.J.; Saratchandran, P.; Sundararajan, N. On-Line Sequential Extreme Learning Machine. *Comput. Intell.* **2005**, *2005*, 232–237.
14. Oza, N.C. Online bagging and boosting. In Proceedings of the 2005 IEEE International Conference on Systems, Man and Cybernetics, Waikoloa, HI, USA, 12 October 2005.
15. Lu, J.; Liu, A.; Dong, F.; Gu, F.; Gama, J.; Zhang, G. Learning under concept drift: A review. *IEEE Trans. Knowl. Data Eng.* **2018**, *31*, 2346–2363. [CrossRef]
16. Khamassi, I.; Sayed-Mouchaweh, M.; Hammami, M.; Ghédira, K. Discussion and review on evolving data streams and concept drift adapting. *Evol. Syst.* **2018**, *9*, 1–23. [CrossRef]
17. Barros, R.S.M.; Santos, S.G.T.C. A large-scale comparison of concept drift detectors. *Inf. Sci.* **2018**, *451*, 348–370. [CrossRef]
18. Gama, J.; Sebastiao, R.; Rodrigues, P. On evaluating stream learning algorithms. *Mach. Learn.* **2013**, *90*, 317–346. [CrossRef]
19. Gama, J.; Žliobaitė, I.; Bifet, A.; Pechenizkiy, M.; Bouchachia, A. A survey on concept drift adaptation. *ACM Comput. Surv. (CSUR)* **2014**, *46*, 1–37. [CrossRef]
20. Wares, S.; Isaacs, J.; Elyan, E. Data stream mining: Methods and challenges for handling concept drift. *SN Appl. Sci.* **2019**, *1*, 1412. [CrossRef]
21. Ross, G.J.; Adams, N.M.; Tasoulis, D.K.; Hand, D.J. Exponentially weighted moving average charts for detecting concept drift. *Pattern Recognit. Lett.* **2012**, *33*, 191–198. [CrossRef]
22. Page, E.S. Continuous inspection schemes. *Biometrika* **1954**, *41*, 100–115. [CrossRef]
23. Frias-Blanco, I.; del Campo-Ávila, J.; Ramos-Jimenez, G.; Morales-Bueno, R.; Ortiz-Diaz, A.; Caballero-Mota, Y. Online and non-parametric drift detection methods based on Hoeffding's bounds. *IEEE Trans. Knowl. Data Eng.* **2014**, *27*, 810–823. [CrossRef]
24. Nishida, K.; Yamauchi, K. Detecting concept drift using statistical testing. In Proceedings of the International Conference on Discovery Science, Sendai, Japan, 1–4 October 2007; Springer: Berlin/Heidelberg, Germany, 2007.
25. Minku, L.L.; Yao, X. DDD: A new ensemble approach for dealing with concept drift. *IEEE Trans. Knowl. Data Eng.* **2011**, *24*, 619–633. [CrossRef]
26. Liu, A.; Zhang, G.; Lu, J. Fuzzy time windowing for gradual concept drift adaptation. In Proceedings of the 2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), Naples, Italy, 9–12 July 2017.
27. Webb, G.I.; Hyde, R.; Cao, H.; Nguyen, H.L.; Petitjean, F. Characterizing concept drift. *Data Min. Knowl. Discov.* **2016**, *30*, 964–994. [CrossRef]
28. Shen, Y.; Du, J.; Tong, J.; Dou, Q.; Jing, L. A parallel and reverse Learn++. NSE classification algorithm. *IEEE Access* **2020**, *8*, 64157–64168. [CrossRef]
29. Chen, Y.; Zhu, Y.; Chen, H.; Shen, Y.; Xu, Z. A Pruning Optimized Fast Learn++ NSE Algorithm. *IEEE Access* **2021**, *9*, 150733–150743. [CrossRef]
30. Hu, H.; Kantardzic, M.; Sethi, T.S. No Free Lunch Theorem for concept drift detection in streaming data classification: A review. *WIREs Data Min. Knowl. Discov.* **2020**, *10*, e1327. [CrossRef]
31. Dasu, T.; Krishnan, S.; Venkatasubramanian, S.; Yi, K. An information-theoretic approach to detecting changes in multi-dimensional data streams. In Proceedings of the Symposium on the Interface of Statistics, Computing Science, and Applications, Pasadena, CA, USA, 24–27 May 2006.
32. Kuncheva, L.I.; Faithfull, W.J. PCA feature extraction for change detection in multidimensional unlabeled data. *IEEE Trans. Neural Netw. Learn. Syst.* **2013**, *25*, 69–80. [CrossRef] [PubMed]
33. Moreno-Torres, J.G.; Raeder, T.; Alaiz-Rodríguez, R.; Chawla, N.V.; Herrera, F. A unifying view on dataset shift in classification. *Pattern Recognit.* **2012**, *45*, 521–530. [CrossRef]
34. Gemaque, R.N.; Costa, A.F.J.; Giusti, R.; Dos Santos, E.M. An overview of unsupervised drift detection methods. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2020**, *10*, e1381. [CrossRef]
35. Domingos, P.; Hulten, G. Mining high-speed data streams. In Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Boston, MA, USA, 20–23 August 2000.
36. Oza, N.C.; Russell, S. *Online Ensemble Learning*; University of California: Berkeley, CA, USA, 2001.
37. Bifet, A.; Zhang, J.; Fan, W.; He, C.; Zhang, J.; Qian, J.; Holmes, G.; Pfahringer, B. Extremely fast decision tree mining for evolving data streams. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, 13–17 August 2017.
38. Wang, H.; Fan, W.; Yu, P.S.; Han, J. Mining concept-drifting data streams using ensemble classifiers. In Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, 24–27 August 2003; Association for Computing Machinery: Washington, DC, USA, 2003; pp. 226–235.
39. Brzeziński, D.; Stefanowski, J. Accuracy updated ensemble for data streams with concept drift. In Proceedings of the International Conference on Hybrid Artificial Intelligence Systems, Wroclaw, Poland, 23–25 May 2011; Springer: Berlin/Heidelberg, Germany, 2011.
40. Brzezinski, D.; Stefanowski, J. Reacting to different types of concept drift: The accuracy updated ensemble algorithm. *IEEE Trans. Neural Netw. Learn. Syst.* **2013**, *25*, 81–94. [CrossRef] [PubMed]
41. McCloskey, M.; Cohen, N.J. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of Learning and Motivation*; Elsevier: Amsterdam, The Netherlands, 1989; pp. 109–165.
42. French, R.M. Catastrophic forgetting in connectionist networks. *Trends Cogn. Sci.* **1999**, *3*, 128–135. [CrossRef] [PubMed]

43. Liu, J.; Li, X.; Zhong, W. Ambiguous decision trees for mining concept-drifting data streams. *Pattern Recognit. Lett.* **2009**, *30*, 1347–1355. [CrossRef]
44. Bifet, A.; Gavaldà, R. Adaptive learning from evolving data streams. In Proceedings of the International Symposium on Intelligent Data Analysis, Lyon, France, 31 August–2 September 2009; Springer: Berlin/Heidelberg, Germany, 2009.
45. Gomes, H.M.; Bifet, A.; Read, J.; Barddal, J.P.; Enembreck, F.; Pfharinger, B.; Holmes, G.; Abdessalem, T. Adaptive random forests for evolving data stream classification. *Mach. Learn.* **2017**, *106*, 1469–1495. [CrossRef]
46. 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]
47. Lughofer, E.; Pratama, M.; Skrjanc, I. Incremental rule splitting in generalized evolving fuzzy systems for autonomous drift compensation. *IEEE Trans. Fuzzy Syst.* **2017**, *26*, 1854–1865. [CrossRef]
48. Pratama, M.; Lu, J.; Lughofer, E.; Zhang, G.; Er, M.J. An incremental learning of concept drifts using evolving type-2 recurrent fuzzy neural networks. *IEEE Trans. Fuzzy Syst.* **2016**, *25*, 1175–1192. [CrossRef]
49. Lughofer, E.; Pratama, M.; Škrjanc, I. Online bagging of evolving fuzzy systems. *Inf. Sci.* **2021**, *570*, 16–33. [CrossRef]
50. Zhu, X.J. *Semi-Supervised Learning Literature Survey*; University of Wisconsin: Madison, WI, USA, 2005.
51. Chapelle, O.; Scholkopf, B.; Zien, A. Semi-supervised learning. *IEEE Trans. Neural Netw.* **2009**, *20*, 542. [CrossRef]
52. Belkin, M.; Niyogi, P.; Sindhwani, V. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *J. Mach. Learn. Res.* **2006**, *7*, 2399–2434.
53. Moh, Y.; Buhmann, J.M. Manifold regularization for semi-supervised sequential learning. In Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing, Taipei, Taiwan, 19–24 April 2009.
54. Jia, X.; Wang, R.; Liu, J.; Powers, D.M. A semi-supervised online sequential extreme learning machine method. *Neurocomputing* **2016**, *174*, 168–178. [CrossRef]
55. Da Silva, C.A.; Krohling, R.A. Semi-Supervised Online Elastic Extreme Learning Machine for Data Classification. In Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 8–13 July 2018.
56. Kamiya, Y.; Ishii, T.; Furao, S.; Hasegawa, O. An online semi-supervised clustering algorithm based on a self-organizing incremental neural network. In Proceedings of the 2007 International Joint Conference on Neural Networks, Orlando, FL, USA, 12–17 August 2007.
57. Furao, S.; Ogura, T.; Hasegawa, O. An enhanced self-organizing incremental neural network for online unsupervised learning. *Neural Netw.* **2007**, *20*, 893–903. [CrossRef]
58. Chong, Y.; Ding, Y.; Yan, Q.; Pan, S. Graph-based semi-supervised learning: A review. *Neurocomputing* **2020**, *408*, 216–230. [CrossRef]
59. Song, Z.; Yang, X.; Xu, Z.; King, I. Graph-based semi-supervised learning: A comprehensive review. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *in press*. [CrossRef] [PubMed]
60. Zhou, K.; Martin, A.; Pan, Q.; Liu, Z. SELP: Semi-supervised evidential label propagation algorithm for graph data clustering. *Int. J. Approx. Reason.* **2018**, *92*, 139–154. [CrossRef]
61. Wada, Y.; Su, S.; Kumagai, W.; Kanamori, T. Robust Label Prediction via Label Propagation and Geodesic k-Nearest Neighbor in Online Semi-Supervised Learning. *IEICE Trans. Inf. Syst.* **2019**, *102*, 1537–1545. [CrossRef]
62. Iscen, A.; Tolias, G.; Avrithis, Y.; Chum, O. Label propagation for deep semi-supervised learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019.
63. Kejani, M.T.; Dornaika, F.; Talebi, H. Graph Convolution Networks with manifold regularization for semi-supervised learning. *Neural Netw.* **2020**, *127*, 160–167. [CrossRef]
64. Liu, W.; Fu, S.; Zhou, Y.; Zha, Z.J.; Nie, L. Human activity recognition by manifold regularization based dynamic graph convolutional networks. *Neurocomputing* **2021**, *444*, 217–225. [CrossRef]
65. Din, S.U.; Shao, J.; Kumar, J.; Ali, W.; Liu, J.; Ye, Y. Online reliable semi-supervised learning on evolving data streams. *Inf. Sci.* **2020**, *525*, 153–171. [CrossRef]
66. Casalino, G.; Castellano, G.; Mencar, C. Data stream classification by dynamic incremental semi-supervised fuzzy clustering. *Int. J. Artif. Intell. Tools* **2019**, *28*, 1960009. [CrossRef]
67. Murilo Gomes, H.; Grzenda, M.; Mello, R.; Read, J.; Huong Le Nguyen, M.; Bifet, A. A survey on semi-supervised learning for delayed partially labelled data streams. *ACM Comput. Surv. (CSUR)* **2022**, *55*, 75.
68. Casalino, G.; Castellano, G.; Mencar, C. Incremental adaptive semi-supervised fuzzy clustering for data stream classification. In Proceedings of the 2018 IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS), Rhodes, Greece, 25–27 May 2018.
69. Roberts, S. Control chart tests based on geometric moving averages. *Technometrics* **2000**, *42*, 97–101. [CrossRef]
70. Hoeffding, W. Probability Inequalities for Sums of Bounded Random Variables. *J. Am. Stat. Assoc.* **1963**, *58*, 13–30. [CrossRef]
71. Baena-Garcıa, M.; del Campo-Ávila, J.; Fidalgo, R.; Bifet, A.; Gavalda, R.; Morales-Bueno, R. Early drift detection method. In Proceedings of the Fourth International Workshop on KNOWLEDGE discovery from Data Streams, Philadelphia, PA, USA, 20 August 2006.
72. Bifet, A.; Gavalda, R. Learning from time-changing data with adaptive windowing. In Proceedings of the 2007 SIAM International Conference on Data Mining, Minneapolis, MN, USA; SIAM: Philadelphia, PA, USA, 2007.
73. Aminikhanghahi, S.; Cook, D.J. A survey of methods for time series change point detection. *Knowl. Inf. Syst.* **2017**, *51*, 339–367. [CrossRef] [PubMed]

74. Hao, C. Sequential change-point detection based on nearest neighbors. *Ann. Stat.* **2019**, *47*, 1381–1407.
75. Fearnhead, P.; Rigaill, G. Changepoint Detection in the Presence of Outliers. *J. Am. Stat. Assoc.* **2019**, *114*, 169–183. [CrossRef]
76. Ferrari, A.; Richard, C.; Bourrier, A.; Bouchikhi, I. Online change-point detection with kernels. Pattern Recognition. *Pattern Recognit.* **2023**, *133*, 109022. [CrossRef]
77. Lughofer, E.; Weigl, E.; Heidl, W.; Eitzinger, C.; Radauer, T. Recognizing input space and target concept drifts in data streams with scarcely labeled and unlabelled instances. *Inf. Sci.* **2016**, *355*, 127–151. [CrossRef]
78. Nikzad-Langerodi, R.; Lughofer, E.; Cernuda, C.; Reischer, T.; Kantner, W.; Pawliczek, M.; Brandstetter, M. Calibration model maintenance in melamine resin production: Integrating drift detection, smart sample selection and model adaptation. *Anal. Chim. Acta* **2018**, *1013*, 1–12. [CrossRef]
79. Huang, G.-B.; Zhu, Q.-Y.; Siew, C.-K. Extreme learning machine: Theory and applications. *Neurocomputing* **2006**, *70*, 489–501. [CrossRef]
80. Huang, G.; Song, S.; Gupta, J.N.; Wu, C. Semi-supervised and unsupervised extreme learning machines. *IEEE Trans. Cybern.* **2014**, *44*, 2405–2417. [CrossRef]
81. Platanios, E.A.; Blum, A.; Mitchell, T.M. Estimating Accuracy from Unlabeled Data. *UAI* **2014**, *14*, 10.
82. Yang, L.; Yang, S.; Li, S.; Liu, Z.; Jiao, L. Incremental laplacian regularization extreme learning machine for online learning. *Appl. Soft Comput.* **2017**, *59*, 546–555. [CrossRef]
83. Da Silva, C.A.; Krohling, R.A. Semi-Supervised Online Elastic Extreme Learning Machine with Forgetting Parameter to deal with concept drift in data streams. In Proceedings of the 2019 International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary, 14–19 July 2019.
84. Gomes, H.M.; Read, J.; Bifet, A. Streaming Random Patches for Evolving Data Stream Classification. In Proceedings of the 2019 IEEE International Conference on Data Mining (ICDM), Beijing, China, 8–11 November 2019.
85. Montiel, J.; Read, J.; Bifet, A.; Abdessalem, T. Scikit-multiflow: A multi-output streaming framework. *J. Mach. Learn. Res.* **2018**, *19*, 1–5.