Diversity-Based Pool of Models for Dealing with Recurring Concepts

Chun Wai Chiu, Leandro L. Minku
Department of Informatics, University of Leicester
University Road, Leicester, LE1 7RH, UK
Email: {cwc13,leandro.minku}@leicester.ac.uk

***Abstract*—Several data stream applications involve recurring concepts, i.e., concept drifts that change the underlying distribution of the data to a distribution previously seen in the data stream. Examples include electricity price prediction and tweet topic classification. In such scenario, it is useful to maintain a pool of old models that could be recovered if their knowledge matches the recurring concept well. A few existing online learning approaches maintain such pools. However, there has been a little investigation on what is the best strategy to maintain an online learning pool with a limited size. We propose to make use of diversity to decide which models to keep in the pool once the pool reaches the maximum size. The motivation behind is that a diverse pool is more likely to maintain a set of representative models with considerably different concepts, helping to handle recurring concepts. We perform experiments to investigate if, when and why maintaining a diverse pool is helpful. The results show that the use of diversity to maintain pools can indeed be helpful to handle recurring concepts. However, the relationship between diversity and accuracy in the presence of concept drift is not straightforward. In particular, an initially good accuracy obtained when using diversity can lead to a stronger subsequent drop in accuracy than other strategies.**

I. Introduction

With the growth of smart phones and tablet computers in this digital era, the amount of data in our world is growing faster than we could have imagined. With such large and ever-growing quantity of data, it is difficult to store and wait for knowledge extraction. Therefore, data have to be learnt in sequential order as a data stream [1]. Data stream learning [2] has been widely used in real-world applications, including spam filtering [3], software engineering [4], credit card fraud detection [5] and so on.

Data streams typically suffer changes in the underlying distribution of the data. We refer to a change in the joint probability distribution of the data as a *concept drift* [6], whereas a given joint probability distribution is referred to as a *concept*. Other terms have also been used in the literature. For example, some authors refer to a change in the joint probability distribution as a dataset shift, and a subcategory of dataset shift that affect the class boundaries as a concept drift [7].

Data stream learning algorithms must be able to cope with concept drift. In order to enable swift reaction to drifts, it is frequently desirable for these algorithms to be *online learning algorithms*. We define these as algorithms that operate in an online learning scenario where each training example arrives separately and is learnt as soon as it arrives [2][1]. Therefore, this paper focuses on online learning.

Recurring concepts refer to concepts that have been previously present in the data stream and then reappear. When recurring concepts occur, it is worth using approaches that maintain a pool of models representing different concepts seen in the data stream [1]. This is because these approaches can exploit the knowledge of stored models to handle recurring concepts. Two important points for such type of approach are to determine which model(s) in the pool to be used for learning and making predictions (*model selection strategy*), and which model(s) from the pool to eliminate once the maximum size of the pool is reached (*memory strategy*). In terms of the memory strategy, most work has used either a FIFO strategy where the oldest models are removed [1] or an elitist strategy where the least accurate models on the current concept are removed [8]. However, such strategies are not always adequate. For instance, the concept represented by old models may reoccur in the future. Similarly, a model that performs poorly on the current concept may be helpful for future concepts [9]. In these cases, it would be better to keep these old or inaccurate models in the pool.

With that in mind, we propose a memory strategy based on diversity[2]. Intuitively, diversity can be seen as a level of disagreement between models, even though there is no single agreed definition of diversity for classification problems. The idea of our approach is that diverse models represent various concepts. Thus, if we maintain a pool with diverse models, we reduce the risk of having only one or few concepts in the pool. In other words, we increase the chances of having more various concepts in the pool to handle recurring concepts.

Although there are papers studying different diversity measures and how they impact ensemble accuracy [9], [12], [13], no previous study has provided a detailed investigation of when and to what extent diversity can help as a memory strategy for deciding which models to keep in a pool to deal with non-stationary environments. To fill in this research gap, this paper aims to answer the following research questions:

- **RQ-MemStr**: Can a pool memory strategy based on diversity help to handle recurring concepts in comparison with other memory strategies? When and why?

[1]This example may or may not be discarded after being learnt.
[2]This strategy has been preliminary investigated in [10], [11].

- **RQ-Cmp**: How does our diversity-based proposed approach perform in comparison with other known approaches designed for dealing with concept drift?

To answer RQ-MemStr, we compare the diversity memory strategy against three other memory strategies through experiments on data streams containing several different types of recurring concepts, as well as data streams with no recurring concepts. These memory strategies are: First In First Out (FIFO), Least Recent Used (LRU), and Delete-Worst (Elitism). Using the same set of data streams, the proposed approach is then compared against five existing approaches to answer RQ-Cmp: Hoeffding Tree with Naïve Bayes at leaves (HTNB) [14], Drift Detection Method Classifier (DDM) [15], Recurrent Concept Drift (RCD) [1], Online Accuracy Update Ensemble (OAUE) [16], Dynamic Weighted Majority (DWM) [8]. The experiment results show the diversity memory strategy is generally helpful to handle recurring concepts. However, the relationship between diversity and accuracy in the presence of concept drift is not straightforward. In particular, an initially good accuracy obtained when using diversity can lead to a stronger subsequent drop in accuracy than other strategies.

The rest of this paper is organised as follows. Section II discusses related work on recurring concepts and other concept drift recovery strategies. Section III presents our proposed approach. Section IV presents the experimental study. Section V presents conclusions and discusses future work.

II. Related Work

Several approaches have been proposed to handle concept drift [2], [6]. Here, we focus on online learning approaches. Some existing work focuses on detecting concept drifts [15], [17], [18]. For example, the Drift Detection Method (DDM) [15] monitors the error rate of a model. It assumes that, with the increasing number of examples in a stationary data stream, the error rate of the model to the stream will decrease. If the error rate of the model starts increasing, the DDM then assumes a concept drift is very likely to be happening and triggers a warning. If the error rate continues to increase further, DDM triggers a drift detection. Approaches that use drift detection methods, such as [15], [17], frequently reset the predictive model upon drift detection. This means that they cannot make use of previous knowledge if a given concept reoccurs.

The most common way to handle recurring concepts is to store past models representing different concepts in a pool, and then exploit them if recurring concepts are identified. The Just-In-Time classifier approach (JIT) [19], [20] is a notable approach that attempts to identify the concept to which examples belong and maintain models representing different concepts in a pool. It shows that exploiting information acquired in the past helps to handle recurring concepts effectively. However, this approach stores all the past concepts in the pool, which can consume a lot of memory over time.

In order to maintain the pool with a limited size, *memory strategies* can be used to decide which models to delete when the maximum size of the pool is exceeded. For example, the Recurrent Concept Drift (RCD) approach [1] uses a single model as the representative to learn and make predictions. It also uses a First In First Out (FIFO) memory strategy to hold models with past knowledge, i.e., it deletes the oldest model when the memory exceeds its maximum size. Each past model is associated with a batch of data representing the knowledge it holds. When a concept drift is detected, a statistical test is used to compare the current batch of data with the batches of data associated to each model. If a batch of data associated to a model in the pool is similar to the current batch, the corresponding model from the pool is selected as the representative. Despite having a strong foundation on statistical tests to detect recurring concepts, this approach suffers from a weakness in terms of its FIFO memory strategy. If a concept reoccurs after a long period, RCD cannot exploit the previous knowledge to swiftly recover its performance.

Ensemble learning is another kind of approach that could potentially handle recurring concepts. Although most ensemble learning algorithms were not explicitly intended to handle recurring concepts, they hold an ensemble of models which could potentially contain knowledge from different concepts. This can be beneficial when there are recurring concepts. Online accuracy update ensemble (OAUE) [16] and Dynamic Weighted Majority (DWM) [8] are two of such approaches. They both maintain a weighted majority ensemble for making predictions and their memory strategies also consist of deleting the worst-performing models on the current concept. For OAUE, the least accurate model in its ensemble is substituted with a new model at every p ($p > 0$) time steps. For DWM, a new model is added to its ensemble if the ensemble makes a mistake, while the weight of an ensemble member is reduced if it makes a mistake. DWM deletes models with weight less than a predefined threshold, i.e., models that have performed continuously poorly.

Diversity and Transfer based Ensemble Learning (DTEL) [21] is a very recent batch-based approach to handle recurring concepts. DTEL creates a new base model whenever a whole new batch of training examples is received. When the ensemble reaches its maximum size, the new base model will first be added to the ensemble and an old model will be removed from the ensemble based on a diversity measure. The weight of each ensemble member is calculated based on the error rate on the most recent batch. Even though DTEL also uses diversity as the memory strategy, their work did not investigate whether diversity is really helpful. Besides, this approach is not an online learning approach.

Overall, no existing work used diversity as an online learning memory management strategy, and no existing work has investigated whether diversity as a memory strategy can help to handle concept drift.

III. Proposed Approach

To answer the research questions posed in section I, we propose a new algorithm called Diversity Pool (DP) (Algorithm 1). Its general idea is as follows. A pool of models is maintained to handle recurring concepts. Predictions are

Algorithm 1 DP algorithm

Parameters: Buffer Size (b), Pool Size (p), Data Stream (S)
Variables: Representative Model (c_r), Actual Buffer (b_a), New Model (c_n), Model Pool (P)

1: $c_r \leftarrow Create(c_n)$; $P \leftarrow E \cup c_r$
2: **for** each $s \in S$ **do**
3: $drift\_level \leftarrow DDM(c_r, s)$
4: **switch** ($drift\_level$)
5: **case** $WARNING$**:**
6: $SaveFIFO(b_a, s)$
7: $P.updateModelsWeightwithExample(s)$
8: $train(c_n, s)$
9: **case** $DRIFT$**:**
10: **if** $|E| < p$ **then**
11: $P \leftarrow P \cup c_n$; $c_r \leftarrow c_n$; $Create(c_n)$
12: **else**
13: **if** $|b_a| \geq b$ **then**
14: $c_{best} \leftarrow getBestByEvaluation(P, b_a)$
15: $c_r \leftarrow decideRepresentative(c_{best}, c_n, b_a)$
16: **if** $c_r == c_n$ **then**
17: $P.removeByDiversity()$
18: $P \leftarrow P \cup c_n$; $Create(c_n)$
19: **end if**
20: **else**
21: $SaveFIFO(b_a, s)$
22: $P.updateModelsWeightwithExample(s)$
23: $train(c_n, s)$
24: **end if**
25: **end if**
26: **end switch**
27: $train(c_r, s)$
28: **end for**

made by selecting a single representative model from the pool. Alternatively, if the approach is deemed not confident enough to select a representative model, the pool is used as a weighted majority vote ensemble and predictions are made based on it. Besides, a method is used to detect concept drifts, i.e., DP is an *active* [2] approach. New models are added to the pool upon concept drift detection. Therefore, each model holds the knowledge of a concept from the environment. One of the key aspects of the algorithm is that this pool decides which models to keep based on diversity [12]. This prevents the pool from storing only similar models when it reaches its maximum size. This increases its chances of holding useful models for handling recurring concepts. Overall, the algorithm consists of three components (online learning with concept drift detection, model selection strategy and memory strategy), which are described in more detail in sections III-A to III-C.

A. Online Learning with Concept Drift Detection

The algorithm learns new training examples sequentially. When a new training example is received, it is first used to check whether concept drift has occurred, based on a concept drift detector such as DDM [15] or EDDM [17] (line 3). If the drift detection method triggers a "warning" alarm (line 5), the algorithm uses a FIFO buffer to store incoming examples (line 6). At the same time, a new model is created and starts learning new examples (line 8). If the drift detection method triggers a "drift alarm" (line 9), the algorithm will decide whether to use the new model or retrieve an old model as representative for learning and making predictions. The new model will always be used as the representative when the maximum size of the pool is not reached yet. The procedures followed upon drift detection are explained in sections III-B and III-C.

Besides being used to detect concept drift, each new example is also used to train the representative model of the pool (line 27). The strategy for selecting a representative model is explained in section III-B.

B. Model Selection Strategy

When a drift is detected, the algorithm uses the new model as representative and stores it into the pool if the maximum size of the pool is not reached yet (line 10-11). Otherwise, it uses the buffer to evaluate the models in the pool (line 14). The best-stored model in terms of accuracy on the buffer is compared with the accuracy of the new model on the buffer (line 15). If the accuracy of the new model is within a 95% confidence interval of the best-stored model's accuracy, the best-stored model is selected as the new representative. Otherwise, the new model is used as the representative model and stored while an old model is discarded from the pool based on diversity (line 16-18) as explained in section III-C.

The model selection strategy is conservative but enables us to focus our analysis done to answer RQ-MemStr on the effectiveness of the diversity memory strategy itself. In particular, if the diversity strategy leads to worse predictive performance than other strategies, we can be confident that this is due to a poor behaviour of the diversity memory strategy itself but not a poor behaviour of the model selection strategy.

We have performed extra experiments with a less conservative strategy which compares the time-decayed test accuracy of old models on all examples since the warning level against the time-decayed prequential accuracy of the new model. We use these experiments to complement our analysis and will explicitly mention when we do so.

It is worth noting that concept drifts may be abrupt, causing the warning period to be shorter than the size of the FIFO buffer (line 13). In this case, when a drift is detected, the algorithm uses the whole pool as a weighted majority vote ensemble with the weights obtained during warning period (line 7). The algorithm keeps storing the incoming examples (line 21), weighting the stored models with the incoming examples (line 22) and training the new model (line 23) until the buffer is full. Then, the approach is assumed to be confident enough to select a single model from the pool as representative, using the procedure described above.

C. Pool Memory Strategy Based on Diversity

The algorithm considers that the pool has a maximum size. This makes sense for applications with limited memory and

time constraints, or applications where the data stream has a high incoming rate. Thus, a mechanism is needed to decide which model to delete once the pool reaches the maximum size and a new model has to be added. We refer to this mechanism as "memory strategy". The memory strategy used in this algorithm simulates the possible diversity levels of the pool after removing each of its existing models. After that, the algorithm deletes the model that leads to the state with the highest diversity. The time complexity of the simulation depends on the size of the pool and the diversity measure used. Therefore, applications with very strict time constraints should use smaller pool sizes and/or diversity measures that are faster to compute. Multi-threading can also be used to enable the simulations run in parallel, in order to reduce the overall computational time.

Diversity is calculated based on the FIFO buffer explained in section III-B. This strategy could potentially be modified to avoid the need for storing this buffer by updating the diversity measures whenever a new example arrives. Even though the term diversity is popularly used, there is no single definition or measure of it for classification problems [22]. Thus, both Yule's Q-statistics and Entropy Measure [12] were investigated as potential diversity measures for our memory strategy. Other diversity measures could be investigated as future work.

Considering two classifiers D_i and D_k, Yule's Q-statistics (Q) is shown in Eq. 1 [12]:

$$Q_{i,k} = \frac{N^{11}N^{00} - N^{01}N^{10}}{N^{11}N^{00} + N^{01}N^{10}} \tag{1}$$

where $N^{a,b}$ is the number of examples where the classification by D_i is a and the classification by D_k is b, 1 represents a correct classification and 0 represents a misclassification. Q varies between 1 and -1. Models that tend to classify the same examples correctly will have positive values of Q, while those which recognise different examples incorrectly will have negative values of Q. For a pool containing L models, Eq. 2 shows the calculation of the averaged Q statistics (Q_{av}) over all pairs of models, which is used as a diversity measure:

$$Q_{av} = \frac{2}{L(L-1)} \sum_{i=1}^{L-1} \sum_{k=i+1}^{L} Q_{i,k} \tag{2}$$

Eq. 3 shows the diversity calculation based on the Entropy Measure (E), suggested by Kuntcheva and Whitaker [12]:

$$E = \frac{1}{N} \sum_{j=1}^{N} \frac{1}{(L - \lceil L/2 \rceil)} min\{l(z_j), L - l(z_j)\} \tag{3}$$

where N refers to the number of examples in the FIFO buffer explained in section III-B; L is the size of the pool; z_j is the j-th example in the FIFO buffer; and $l(z_j)$ is the number of models from the pool that correctly classify z_j. E varies between 0 and 1, where 0 indicates no diversity and 1 indicates the highest diversity [12].

TABLE I
DATA STREAMS

Data Stream	Concept Drift Sequence	Width of the Drifts	Pool Size
A1	f2→f5→f3→f6→f1→f4→f1→f4→f1→f4	20,000	4
A2	f1→f2→f1→f3→f1	20,000	3
A3	f2→f6→f3→f6→f2	20,000	3
A4	f4→f2→f1→f3→f4	20,000	3
A5	f9→f8→f10→f7→f8→f7→f8→f7	1	3
A6	f1→f3→f6→f5→f4	20,000	3
A7	f7→f8→f9→f10	1	3
A8	f2→f5→f4→f6	20,000	3
A9	f1→f8→f3→f10→f9	1	3
S1	f3→f1→f2→f4→f3	20,000	4
S2	f5→f3→f1→f2→f4	20,000	3
S3	f5→f1→f4→f3→f2	1	3

"Ai" and "Si" refer to i-th data stream using Agrawal and SEA as stream type respectively. Coloured rows (lime / light grey) represent that the stream contains recurring concepts. fn represents the n-th function of the stream type, i.e., f1 in A1 is referring to the first function of the Agrawal generator.

IV. EXPERIMENTS

This section presents the experiments performed to evaluate the proposed approach and answer the research questions posed in section I. Massive Online Analysis (MOA) framework [23] was chosen as the platform to perform experiments. The approach's source code is available at https://github.com/michaelchiucw/DiversityPool. Section IV-A presents the data streams. Section IV-B presents the experimental setup. Sections IV-C and IV-D present the experiment results.

A. Data Streams

We created several artificial data streams based on MOA's Agrawal and SEA generators [24], [25]. The use of artificial data streams allows us to specify not only the concept drift point and width but also the sequence of concepts in the data stream. This enables us to gain a more detailed understanding of when our approach helps or is detrimental. We will extend this analysis to real-world data streams as a future work.

The Agrawal data consist of nine numerical input attributes and one binary categorical output. The available concepts used to compose our data streams are the same as those used by [24]. The SEA data consist of three numerical input attributes (the last of them is an irrelevant attribute) and one binary categorical output. The 1st to 4th functions available to represent concepts in our data streams are the same as those used by [25], which use thresholds 8, 9, 7, and 9.5, respectively. We added a 5th function using threshold 4. Concept drifts were simulated by connecting two data streams with different concepts by using a sigmoid function to decide the probability of examples coming from the old and new concepts during the transitional period [23]. The details of all data streams are shown in table I.

B. Experimental Setup

The base learner used by the approaches in the experiments was Hoeffding Tree with Naïve Bayes at leaves (HTNB) [14]. To answer RQ-MemStr, several variations of the proposed approach using different memory strategies were compared:

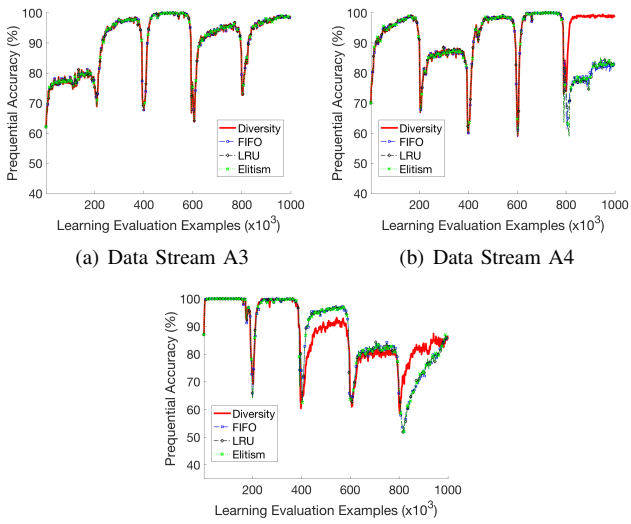
- *Diversity*: keep models that lead to a more diverse pool, as explained in section III.
- *First In First Out*: delete oldest model.
- *Least Recently Used*: delete model least recently used.
- *Elitism*: delete the model that performs worst over the FIFO buffer explained in section III-A.

To answer RQ-Cmp, the proposed approach using the diversity memory strategy was compared against five well known existing approaches. The approaches and the reasons for choosing these approaches are listed below:

- *HTNB* [14]: to analyse how the proposed approach compares to its base learner.
- *DDM* [15]: to see if the proposed approach improves over an approach that resets the system upon drift detection.
- *RCD* [1]: to see how the proposed approach compares to another online learning approach designed to handle recurring concepts.
- *OAUE* [16] and *DWM* [8]: to see how the proposed approach compares to ensemble approaches for non-stationary environments. Although these approaches were not explicitly intended to handle recurring concepts, ensembles can be seen as pools of models.

We performed Friedman tests with a level of significance of 0.05 to compare the prequential accuracies of different approaches on each data stream separately. This enables us to identify which specific data streams the proposed approach helped with or was detrimental. Accuracy was measured prequentially, i.e., each example is used to test the approach first and then to train the model [23]. A fading factor of 0.999 was used to make past examples less important to the current accuracy. Each group compared by the test corresponds to a different memory strategy or approach, and each observation within a group corresponds to the prequential accuracy sampled at every 1000th example of the data stream. Nemenyi post-hoc tests were performed to identify which specific pairs of groups are significantly different.

To choose parameters for the compared algorithms, we created two artificial data streams (one for Agrawal and one for SEA) containing a random sequence of concepts, where the concept drift points and widths were also randomly selected. For the approaches using drift detection method, DDM and EDDM were tested. For the proposed approach, diversity measure was tested with Q-statistics and Entropy. For all parameters which are related to example batch size or evaluation period, the values tested were 200, 400, 600, 800, and 1000. The values of k for the k-Nearest Neighbour used as the statistical test in RCD were 1, 3, 5, and 7, while the p-value was tested with 0.01 and 0.05. The parameters that lead to the highest average prequential accuracy for Agrawal and SEA were then chosen to be used with data streams A1-A9 and S1-S3, respectively. In particular, DDM and Entropy Measure were selected as the best-performing drift detection method and diversity measure respectively, for use with our proposed approach. The base learner, HTNB, and the drift detection methods were run with default MOA parameters [23].



(a) Data Stream A3 (b) Data Stream A4

(c) Data Stream A6

Fig. 1. Comparison of Memory Strategies

C. Comparison of Memory Strategies

This section presents the analysis done to answer RQ-MemStr, explained in section I. Table II shows the means, standard deviations, and results of the Friedman and Nemenyi tests to compare the prequential accuracy of the proposed approach using four different memory strategies on each data stream. This table shows that the diversity memory strategy obtained similar or better prequential accuracy in all cases, except for data stream A5 (in comparison to LRU). The elitism strategy was frequently worse than the diversity strategy, being the least recommended strategy.

Fig. 1 presents the prequential accuracy of the memory strategies over time for three data streams (A3, A4 and A6), as representative cases where the diversity strategy performed similarly most of the time, better and worse for some different periods of time w.r.t. other strategies. Plots for other data streams were omitted due to space constraints. In general, all memory strategies have the same performance at the early stage of all streams because the maximum memory size had not been reached yet for any of them. They just used a newly created model as the representative model after the concept drifts. Once the maximum memory size was reached, their performance started to have differences, depending on the data stream. Even though the diversity strategy did not typically present large accuracy improvements w.r.t. other strategies, it was the most consistent one among the best strategies in all data streams. This means that, in practice, diversity may be a safer strategy to adopt when the underlying characteristics of the data stream are not known in advance.

In order to provide a better understanding of the reasons behind these results, we discuss in more detail cases where diversity performed better, worse and similarly to the best among the other strategies.

a) Cases where the diversity strategy performed better: Fig. 1(b) shows that diversity was the most beneficial in data stream A4. In particular, diversity recovered its prequential

TABLE II
MEMORY STRATEGY COMPARISON – MEAN, STANDARD DEVIATIONS, FRIEDMAN TEST AND NEMENYI TEST RESULTS

Data Stream	Recurring Concepts?	Mean of Prequential Accuracy with Fading Factor 0.999 Across the Data Stream (Standard Deviation)				Friedman Test p-value	Nememyi Post Hoc Test p-value		
		Diversity	FIFO	LRU	Elitism		Diversity vs FIFO	Diversity vs LRU	Diversity vs Elitism
A1	Yes	84.359% (11.302%)	84.094% (11.322%)	84.094% (11.322%)	**84.362% (11.300%)**	7.522E-07	0.541	0.541	0.303
A2	Yes	**97.233% (6.548%)**	97.017% (7.555%)	97.017% (7.555%)	94.994% (8.367%)	<2.2E-16	0.44	0.44	5.5E-14
A3	Yes	90.581% (9.494%)	90.581% (9.494%)	90.581% (9.494%)	90.581% (9.494%)	1.00	-	-	-
A4	Yes	**93.468% (8.021%)**	89.637% (9.569%)	89.637% (9.569%)	89.689% (9.436%)	<2.2E-16	3.0E-14	3.5E-14	1.6E-13
A5	Yes	98.335% (1.768%)	98.501% (1.612%)	98.500% (1.609%)	**98.504% (1.613%)**	<2.2E-16	1.00	1.6E-09	0.82
A6	No	**87.912% (10.475%)**	87.792% (12.569%)	87.792% (12.569%)	87.794% (12.566%)	4.982E-06	0.0474	0.0474	0.0028
A7	No	98.320% (1.820%)	98.256% (1.780%)	**98.320% (1.813%)**	98.268% (1.783%)	<2.2E-16	1.5E-06	0.9788	0.0010
A8	No	**80.143% (8.694%)**	**80.143% (8.693%)**	**80.143% (8.693%)**	79.827% (8.539%)	<2.2E-16	1.00	1.00	4.4E-0.7
A9	No	98.951% (2.213%)	98.593% (2.217%)	**98.953% (2.214%)**	98.716% (2.443%)	<2.2E-16	0.731	0.077	2.2E-10
S1	Yes	98.297% (2.194%)	98.297% (2.194%)	98.297% (2.194%)	98.297% (2.194%)	<1.00	-	-	-
S2	No	**98.961% (1.129%)**	**98.961% (1.129%)**	**98.961% (1.129%)**	98.908% (1.185%)	3.683E-11	0.8861	0.8861	3.4E-05
S3	No	98.945% (1.204%)	99.027% (1.040%)	99.028% (1.056%)	**99.053% (1.003%)**	0.4422	-	-	-

"Ai" and "Si" refer to i-th data stream using Agrawal and SEA as stream type respectively. Cells with bold text represent the best performance in the data stream. Coloured Nemenyi p-value cells represent that there is significant difference (p-value ≤ 0.05) and the diversity strategy performed better (lime or light grey) / worse (orange or dark grey) based on the mean and standard deviation of its prequential accuracy. Nemenyi p-value cells in white represent no significant difference.

TABLE III
STATE OF DIVERSITY POOL P

(a) Data Stream A4

Example Period	$P[0]$	$P[1]$	$P[2]$	Actual Concept
0 - 199,810	0(f4)	-	-	f4
199,810 - 201,919	0(f4)	1(f2)	-	f2
201,919 - 394,594	0(f4)	1(f2)	2(f2)	f2
394,594 - 594,128	0(f4)	1(f2)	3(f1)	f1
594,128 - 594,386	0(f4)	3(f1)	4(f1)	f1
594,386 - 594,925	0(f4)	3(f1)	4(f1)	f1
594,925 - 600,859	0(f4)	3(f1)	5(f1)	f1
600,859 - 601,431	0(f4)	3(f1)	6(f3)	f3
601,431 - 601,823	0(f4)	3(f1)	6(f3)	f3
601,823 - 788,649	0(f4)	6(f3)	7(f3)	f3
788,649 - 799,364	0(f4)	6(f3)	8(f3)	f3
799,364 - 1,000,000	0(f4)	6(f3)	8(f3)	f4

(b) Data Stream A6

Example Period	$P[0]$	$P[1]$	$P[2]$	Actual Concept
0 - 171,261	0(f1)	-	-	f1
171,261 - 171,682	0(f1)	1(f1)	-	f1
171,682 - 175,383	0(f1)	1(f1)	2(f1)	f1
175,383 - 175,786	0(f1)	1(f1)	2(f1)	f1
175,786 - 190,502	0(f1)	2(f1)	3(f1)	f1
190,502 - 190,615	0(f1)	2(f1)	3(f1)	f1
190,615 - 193,250	2(f1)	3(f1)	4(f1)	f1
193,250 - 193,453	2(f1)	3(f1)	4(f1)	f1
193,453 - 197,003	2(f1)	4(f1)	5(f1)	f1
197,003 - 197,347	2(f1)	4(f1)	5(f1)	f1
197,347 - 201,681	2(f1)	4(f1)	5(f1)	f1
201,681 - 387,518	2(f1)	4(f3)	5(f1)	f3
387,518 - 394,373	4(f3)	5(f1)	6(f3)	f3
394,373 - 394,485	5(f1)	6(f3)	7(f3)	f3
394,485 - 395,046	5(f1)	6(f3)	7(f3)	f3
395,046 - 395,771	5(f1)	6(f3)	8(f3)	f3
395,771 - 396,264	5(f1)	6(f3)	8(f3)	f3
396,264 - 597,296	5(f1)	8(f3)	9(f6)	f6
597,296 - 798,344	5(f1)	9(f6)	10(f5)	f5
798,344 - 1,000,000	5(f1)	9(f6)	11(f4)	f4

Format: $\{number_i\}$(f$\{number_j\}$), where i refers to the index of the models created and j refers to the concept that was active when they were created. The coloured cells (lime or light grey) indicate the representative model in the time period analysed. When all models from the pool P are highlighted, this means that a weighted majority ensemble was used, rather than a single representative model.

accuracy a lot better than all other strategies after the concept drift around 800k examples. The concepts of 2nd and 4th Agrawal functions are by definition very similar to each other but quite distinct from 1st and 3rd functions. According to concept change sequence of stream A4 shown in table I, the knowledge of the 4th function at the beginning of the stream should have been forgotten by the strategy of FIFO, LRU and elitism by the time the concept of the 4th function starts to recur. The knowledge of the 2nd function was also forgotten due to false alarms given by the drift detection method and the memory strategies themselves. In contrast, the diversity strategy, as shown in table III(a), still holds the knowledge of the 4th function despite the false alarm. Hence, it can exploit the previous knowledge of the function to make predictions when the concept of the 4th function reoccurred. Therefore, it has a better performance recovery w.r.t. other strategies, confirming that the original reason for adopting the diversity memory strategy can indeed help to handle concept drift.

b) Cases where the diversity strategy performed worse: Fig. 1(c) shows that diversity strategy has a significant under-performance for a long period (around 400k examples to 600k examples) in data stream A6. According to table I, stream A6 has a drift point at 400k examples, with a transitional period of 10k examples before and after. The concepts surrounding that drift point are 3rd and 6th Agrawal functions which are quite distinct by definition. A closer look reveals that a concept drift detection was performed at 396,264 examples when using the diversity strategy while the concept drift detection for the other strategies was at 400,928 examples. All strategies created a new model as representative after that drift detection. The earlier drift detection suffered by the diversity strategy means that this new model was trained with around 3.7k examples more likely to belong to the old concept, resulting in a lower performance on the new concept (400k examples to 600k examples). Even though there is no direct link between the diversity strategy itself and the under-performance, given that the diversity strategy is the only difference between these approaches, we further investigate why the diversity strategy indirectly led to worse accuracy.

Through a closer look at the memory state of the diversity strategy throughout the data stream A6 (table III(b)) and Fig. 1(c), we can observe that the diversity strategy successfully made use of old models around the first drift, achieving higher and more stable accuracy in some short periods of time during the second concept. The more stable accuracy made

the concept drift detector more likely to detect drifts during the second concept. This led to some false alarms during the second concept (200k examples to 400k examples), and to the early drift detection corresponding to the third concept (400k examples to 600k examples). This earlier drift detection, in turn, resulted in worse accuracy during the third concept, as explained in the previous paragraph. This shows that, even though diversity can help as expected and demonstrated in stream A4, its increased accuracy can sometimes lead to a subsequent worse accuracy.

c) Cases where the diversity strategy performed similarly: Table II shows the performance of the diversity strategy does not have a significant difference w.r.t. FIFO and LRU in most cases. A representative example is shown in Fig. 1(a). In several cases, the difference in prequential accuracies between diversity and elitism is not very large either. This is an expected behaviour for half of the streams, which present no recurring concepts. For the other half, a closer look shows that this frequently happens because the approach creates a new model as the representative for the new concept, rather than retrieving a model from the pool. Therefore, all memory strategies behave similarly despite the memory content being different. This is expected, given that the model selection strategy used by the proposed approach is conservative. As explained in section III-A, this enables us to focus our analysis on the behaviour of the memory strategies themselves, rather than biasing it to the model selection strategy.

We have also performed extra experiments with a less conservative model selection strategy as explained in section III-A. However, it did not lead to improvements in the prequential accuracy of the proposed approach. A potential reason behind is that the less conservative strategy retrieves models which are not similar enough to the new concept, hindering the approach's accuracy. As we can see, the model selection strategy can highly influence the usefulness of the diversity memory strategy. As future work, we will investigate different model selection strategies to identify which of them is the most adequate for use with diverse pools.

D. Comparison against Existing Approaches

This section presents the analysis to answer RQ-Cmp, posed in section I. Table IV presents the means, standard deviations, and results of the Friedman and Nemenyi tests to compare the prequential accuracies of DP against existing approaches. It shows that DP performed better than its base learner, HTNB. HTNB assumes that the data stream does not present concept drift [14]. Therefore, it is not surprising that it performed worse than DP, which uses a drift detection method.

As shown on table IV, there is no significant difference between the performance of DP and DDM in most data streams presenting no recurring concept. This is an expected behaviour because, in the absence of recurring concepts, DP will always use the newly created model as the representative learner after concept drifts. For the data streams presenting recurring concepts with several distinct concepts before the recurrence (stream A2-A4), DP performed better than DDM. This is consistent with the fact that DP maintains a pool of models to handle recurring concepts. DP performed worse than DDM in two data streams only (A5 and S1).

Table IV also shows that DP performed better than RCD in most data streams. This is a very positive result for DP because both DP and RCD were designed to handle recurring concepts. DP lost to RCD in only one data stream (A6) which presents no recurring concept. DP also has better average prequential accuracies and lower standard deviations than RCD, especially in data streams presenting recurring concepts. A possible reason behind can be the difference in memory strategy (RCD uses FIFO while DP uses diversity). The benefits of using diversity over FIFO have been discussed in section IV-C. Another possible reason is that RCD's model selection strategy may be retrieving less adequate models than DP's conservative strategy. It is also worth mentioning that RCD used a different detection method (EDDM) from our approach (DDM). However, we have performed several extra experiments on RCD using DDM. They show that using DDM in RCD leads to worse prequential accuracy than using EDDM. Thus, the reason behind the worse performance obtained by RCD is not due to the difference in drift detection method.

Table IV shows that DP performed worse than OAUE and DWM in data streams A1 and A8. It also shows that there is no significant difference between the performance of DP and DDM in these data streams, suggesting that the model selection strategy may not have retrieved models from the pool. The pool and its diversity memory strategy, therefore, could not help with handling recurring concepts. This can be the potential reason for DP's under-performance in these two data streams. Meanwhile, OAUE and DWM use ensembles to make predictions, which may help them to achieve better performance than DP in these two data streams.

V. Conclusion

Recurring concepts are common in most real-world applications. Successfully dealing with them will facilitate machine learning algorithms to adapt to such changes by exploiting previous knowledge. A possible way to handle recurring concepts is by maintaining a pool of past models. This paper contributes to the advancement of this area by investigating a novel diversity strategy to decide which models to keep in the pool once the pool reaches its maximum size.

We show that maintaining a pool of diverse models can improve accuracy when there are recurring concepts (RQ-MemStr). However, the relationship between a diverse pool and the accuracy of the system is much more complex than one may initially have thought. In particular, the higher accuracy achieved through the diversity strategy can sometimes trigger a subsequent poorer accuracy. We also show that our proposed approach, which uses diversity as memory strategy, performed particularly well against existing approaches on data streams with recurring concepts (RQ-Cmp). In most cases, it performed better than RCD and DWM, which are approaches that directly or indirectly handle recurring concepts. It also performed similar to or better than DDM, which does not

TABLE IV
COMPARISON AGAINST EXISTING APPROACHES – MEAN, STANDARD DEVIATIONS, FRIEDMAN TEST AND NEMENYI TEST RESULTS

Data Stream	Recurring Concepts?	Mean of Prequential Accuracy with Fading Factor 0.999 Across the Data Stream (Standard Deviation)						Friedman Test p-value	Nememyi Post Hoc Test p-value				
		DP	HTNB	DDM	RCD	OAUE	DWM		DP vs HTNB	DP vs DDM	DP vs RCD	DP vs OAUE	DP vs DWM
A1	Yes	84.359% (11.301%)	69.170% (8.612%)	**84.581% (11.211%)**	83.906% (14.391%)	92.988% (8.926%)	89.871% (11.374%)	<2.2E-16	<2.2E-16	0.126	9.3E-08	<2.2E-16	<2.2E-16
A2	Yes	**97.233% (6.548%)**	82.134% (13.602%)	96.383% (7.611%)	91.629% (13.248%)	95.418% (9.243%)	92.302% (12.123%)	<2.2E-16	<2.2E-16	6.6E-09	<2.2E-16	0.653	5.2E-14
A3	Yes	90.581% (9.494%)	75.255% (6.792%)	88.331% (10.008%)	82.172% (14.324%)	**93.895% (7.939%)**	88.200% (9.192%)	<2.2E-16	<2.2E-16	0.0098	3.1E-14	<2.2E-16	0.9847
A4	Yes	**93.468% (8.021%)**	79.019% (12.432%)	91.423% (8.605%)	76.172% (10.693%)	93.700% (8.069%)	87.204% (10.802%)	<2.2E-16	<2.2E-16	6.1E-14	<2.2E-16	4.0E-13	<2.2E-16
A5	Yes	98.335% (1.768%)	86.611% (8.165%)	**98.484% (1.620%)**	97.210% (3.438%)	98.087% (2.421%)	97.165% (3.526%)	<2.2E-16	<2.2E-16	3.7E-08	<2.2E-16	<2.2E-16	<2.2E-16
A6	No	87.912% (10.475%)	72.865% (19.424%)	85.498% (12.290%)	89.844% (10.831%)	**93.283% (9.083%)**	87.358% (13.210%)	<2.2E-16	<2.2E-16	8.2E-06	7.0E-14	<2.2E-16	7.4E-11
A7	No	**98.320% (1.820%)**	88.097% (8.254%)	98.273% (1.795%)	96.180% (3.669%)	97.971% (2.888%)	97.130% (4.737%)	<2.2E-16	<2.2E-16	0.52	5.1E-14	5.2E-14	6.0E-14
A8	No	80.143% (8.694%)	72.137% (6.883%)	80.143% (8.694%)	75.968% (11.808%)	**87.771% (10.676%)**	83.560% (12.151%)	<2.2E-16	<2.2E-16	1.00	<2.2E-16	<2.2E-16	5.9E-14
A9	No	**98.951% (2.213%)**	91.103% (10.224%)	98.971% (2.095%)	98.114% (4.091%)	98.704% (2.848%)	97.156% (6.811%)	<2.2E-16	<2.2E-16	1.00	<2.2E-16	6.6E-12	4.8E-14
S1	Yes	98.297% (2.194%)	95.782% (4.120%)	98.819% (1.338%)	98.067% (2.133%)	98.705% (2.103%)	**98.823% (1.275%)**	<2.2E-16	<2.2E-16	1.3E-11	7.3E-14	2.9E-14	1.1E-13
S2	No	**98.961% (1.129%)**	97.353% (2.952%)	98.931% (1.188%)	98.579% (1.327%)	98.529% (2.050%)	98.651% (1.411%)	<2.2E-16	<2.2E-16	0.30756	<2.2E-16	5.5E-14	<2.2E-16
S3	No	**98.945% (1.204%)**	95.826% (4.390%)	98.942% (1.166%)	98.593% (1.347%)	98.248% (2.317%)	98.615% (1.622%)	<2.2E-16	<2.2E-16	1.00	5.4E-14	<2.2E-16	5.0E-11

"Ai" and "Si" refer to i-th data stream using Agrawal and SEA as stream type respectively. Cells with bold text represent the best performance in data stream. Coloured Nemenyi p-value cells represent that there is significant difference (p-value $\leq$ 0.05) and the proposed approach performed better (lime or light grey) / worse (orange or dark grey) based on the mean and standard deviation of its prequential accuracy. Nemenyi p-value cells in white represent no significant difference.

have mechanisms to handle recurring concepts. It performed sometimes better and sometimes worse than OAUE.

Future work includes further investigation on how the diversity memory strategy interacts with other model selection strategies and diversity measures, and how it performs on real-world data streams. Improvements by integrating the proposed approach with other strategies to cope with false alarms and beneficial mechanisms from OAUE can also be investigated.

REFERENCES

[1] P. M. G. Jr and R. S. M. de Barros, "RCD: A Recurring Concept Drift Framework," *Pattern Recognit. Lett.*, vol. 34, no. 9, pp. 1018–1025, 2013.

[2] G. Ditzler, M. Roveri, C. Alippi, and R. Polikar, "Learning in Nonstationary Environments: A Survey," *IEEE CIM*, vol. 10, no. 4, pp. 12–25, 2015.

[3] S. J. Delany, P. Cunningham, A. Tsymbal, and L. Coyle, "A Case-based Technique for Tracking Concept Drift in Spam Filtering," *Know.-Based Syst.*, vol. 18, no. 4-5, pp. 187–195, 2005.

[4] L. L. Minku and S. Hou, "Clustering Dycom: An Online Cross-Company Software Effort Estimation Study," in *PROMISE*, 2017, pp. 12–21.

[5] A. D. Pozzolo, G. Boracchi, O. Caelen, C. Alippi, and G. Bontempi, "Credit Card Fraud Detection: A Realistic Modeling and a Novel Learning Strategy," *IEEE TNNLS*, pp. 1–14, 2018.

[6] B. Krawczyk, L. L. Minku, J. Gama, J. Stefanowski, and M. Woniak, "Ensemble Learning for Data Stream Analysis: A Survey," *Inf Fusion*, vol. 37, pp. 132–156, 2017.

[7] J. G. Moreno-Torres, T. Raeder, R. Alaiz-Rodrguez, N. V. Chawla, and F. Herrera, "A Unifying View on Dataset Shift in Classification," *Pattern Recognition*, vol. 45, no. 1, pp. 521–530, 2012.

[8] J. Z. Kolter and M. A. Maloof, "Dynamic Weighted Majority: A New Ensemble Method for Tracking Concept Drift," in *IEEE ICDM*, 2003, pp. 123–130.

[9] L. L. Minku and X. Yao, "DDD: A New Ensemble Approach for Dealing with Concept Drift," *IEEE TKDE*, vol. 24, no. 4, pp. 619–633, 2012.

[10] W. Zheng, "Tracking Recurring Concept Drift with Classifier Pruning Strategy," Master's thesis, School of Computer Science, University of Birmingham, 2011, supervisor: Leandro Minku.

[11] C. W. Chiu, "Machine Learning for Applications with Recurring Changes," Bachelor's Thesis, Department of Informatics, University of Leicester, 2017, supervisor: Leandro Minku.

[12] L. I. Kuncheva and C. J. Whitaker, "Measures of Diversity in Classifier Ensembles and Their Relationship with the Ensemble Accuracy," *Machine Learning*, vol. 51, no. 2, pp. 181–207, 2003.

[13] L. L. Minku, A. P. White, and X. Yao, "The Impact of Diversity on Online Ensemble Learning in the Presence of Concept Drift," *IEEE TKDE*, vol. 22, no. 5, pp. 730–742, 2010.

[14] P. Domingos and G. Hulten, "Mining High-speed Data Streams," in *KDD*, 2000, pp. 71–80.

[15] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with Drift Detection," in *Adv in A.I.*, A. L. C. Bazzan and S. Labidi, Eds., 2004, pp. 286–295.

[16] D. Brzezinski and J. Stefanowski, "Combining Block-based and Online Methods in Learning Ensembles from Concept Drifting Data Streams," *Information Sciences*, vol. 265, pp. 50–67, 2014.

[17] M. Baena-García, J. Del Campo-Ávila, R. Fidalgo, and A. Bifet, "Early Drift Detection Method," in *IWKDDS*, 2006, pp. 77–86.

[18] C. Alippi and M. Roveri, "Just-in-Time Adaptive Classifiers - Part I: Detecting Nonstationary Changes," *IEEE TNN*, vol. 19, no. 7, pp. 1145–1153, 2008.

[19] C. Alippi, G. Boracchi, and M. Roveri, "Just-in-time Ensemble of Classifiers," in *IJCNN*, 2012, pp. 1–8.

[20] C. Alippi and G. Boracchi and M. Roveri, "Just-In-Time Classifiers for Recurrent Concepts," *IEEE TNNLS*, vol. 24, no. 4, pp. 620–634, 2013.

[21] Y. Sun, K. Tang, Z. Zhu, and X. Yao, "Concept Drift Adaptation by Exploiting Historical Knowledge," *CoRR*, 2017.

[22] E. K. Tang, P. N. Suganthan, and X. Yao, "An Analysis of Diversity Measures," *Machine Learning*, vol. 65, no. 1, pp. 247–271, 2006.

[23] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer, "MOA: Massive Online Analysis," *JMLR*, vol. 11, pp. 1601–1604, 2010.

[24] R. Agrawal, T. Imielinski, and A. Swami, "Database Mining: A Performance Perspective," *IEEE TKDE*, vol. 5, no. 6, pp. 914–925, 1993.

[25] W. N. Street and Y. Kim, "A Streaming Ensemble Algorithm (SEA) for Large-scale Classification," in *KDD*, 2001, pp. 377–382.