Int. J. Mach. Learn. & Cyber. (2019) 10:563–578
DOI 10.1007/s13042-017-0738-9

ORIGINAL ARTICLE

A two ensemble system to handle concept drifting data streams: recurring dynamic weighted majority

Parneeta Sidhu[1] · **M. P. S. Bhatia**[1]

Received: 4 August 2016 / Accepted: 26 October 2017 / Published online: 2 November 2017

Abstract We present an ensemble system, recurring dynamic weighted majority (RDWM) that maintains two ensembles of experts, so as to accurately handle drifting concepts mainly recurrent drifts. The primary online ensemble represents the present concepts and the secondary ensemble represents the old concepts since the beginning of learning. An effective pruning methodology helps to remove redundant and old classifiers, which may have otherwise caused interference in learning the new concepts. Experimental evaluation using datasets proves that RDWM achieves very high generalization accuracy, irrespective of the speed or severity of drift; or presence of noise in the dataset.

Keywords Concept drift · Ensemble · Recurrent · Data stream

1 Introduction

Mining large streams of data is an upcoming area of research in the machine learning community. Data stream mining is the process of understanding the underlying concepts in data and analyzing drifts [3, 6, 32], so as to accurately classify the new instances. A drift could be sudden, gradual, recurring, or incremental. Sudden change is observed when the concept changes from one class to another within a single time step. Gradual change occurs when the new concept emerges gradually over time. A change is said to be recurrent if an old concept reappears after some time. The drift is incremental if any two consecutive concepts are almost similar and the drift is felt only after a longer time period. Further, a drift can also measured by its severity and speed. Severity represents the amount of changes caused by a new concept and speed is the inverse of the total time taken for a new concept to completely replace the old concept. Various applications where drifts have been observed are Market-Basket analysis [12], computer security, medical diagnosis etc.

Online approaches [1, 4, 6, 12, 16, 18, 26, 37] process each instance only “once” on arrival without storing it for further processing. These can be categorized as: approaches that explicitly use a mechanism to handle drifts [1, 6, 18]; and that does not explicitly use a mechanism for drift detection [4, 12]. Online approaches may either be a single classifier; or a single ensemble; or an active classifier and a set of weighted classifier systems. None of the existing systems maintain more than one ensemble in its model. It has been studied that an ensemble of classifiers [5, 7, 11, 35, 38] provides higher generalization accuracy [3, 30, 36] as compared to a single classifier system. Hence, we have proposed Recurring Dynamic Weighted Majority system (RDWM) that maintains two ensembles: a primary online ensemble and a secondary ensemble, for more accurate handling of drifting concepts mainly recurrent drifts. The ensembles vary in the type of concept they represent and may perform differently depending on the speed or severity of drift. The primary ensemble represents the present concepts and is trained and updated as in Dynamic Weighted Majority (DWM) [12]; and the secondary ensemble consists of the most accurate experts being copied from the primary ensemble at times of drift. Experimental evaluation using various datasets proves that for drifts with high speed or low speed (independent of severity), the primary ensemble provides

✉ Parneeta Sidhu
parneeta07@gmail.com

M. P. S. Bhatia
bhatia.mps@gmail.com

[1] Division of CoE, Netaji Subhas Institute of Technology, Sec-3, Dwarka, New Delhi 110078, India

better accuracy than the secondary ensemble. For recurrent drifts, the secondary ensemble provides better accuracy as compared to the primary ensemble. Hence, RDWM performs better or at least similar as the existing systems for drift detection.

2 Research questions and paper organization

The paper aims at answering the following questions:

1. Why do we need to maintain two ensembles in RDWM? How does it ensure improved system performance while handling drifts?
2. Does our system always provide better accuracy as compared to the existing systems for drift detection?
3. How does severity impact the performance of RDWM in terms of prequential accuracy, kappa statistic, model cost, time and memory?
4. What is the impact of change in the base classifier on the performance of RDWM?
5. How does the presence of noise impact our systems' performance?

The answer to the first question is that RDWM needs to maintain a primary online ensemble and a secondary ensemble so as to achieve the best generalization accuracy while handling drifts. If the drift has high speed and low severity, the new concept is not the same but quite similar to the recent old concept. Thus, the updated primary ensemble has a very high possibility of providing good accuracy. If the drift has high speed and high severity, it results in big changes very suddenly. Hence, the re-initialized primary ensemble may provide good accuracy. For low speed drift (independent of severity), the new concept gradually replaces the old concept. Hence, immediately after the beginning of drift the new concept would be quite similar to the old concept. Thus, the primary ensemble has a very high possibility of achieving better accuracy. However, longer after the drift when the new concept would be quite different from the old concept, the re-initialized primary ensemble may provide better accuracy levels. For recurrent drifts, the secondary ensemble maintaining the old, most accurate experts provides better classification accuracy than the primary ensemble.

For answering the second question, we evaluated RDWM using various datasets with variation in the speed of drift such as Stagger concepts [26] (sudden drift), moving hyperplane dataset [9] (gradual drift), real-time datasets such as electricity pricing dataset [8], power supply stream [25], KDD CUP 1999 dataset [34] and static datasets e.g. breast cancer dataset [31]. The analysis identifies that while handling sudden drift, RDWM achieved the best accuracy among the single ensemble DWM [12], single classifier EDDM [1], naïve bayes (NB) [20] and Hoeffding Tree (HT) [30]. Further, RDWM performs slightly better than DWM [12] and EDDM while handling gradual drifts. Experimental evaluation of RDWM using hyperplane dataset shows that RDWM performs the best when severity is high and performs worse or similar as the other approaches when severity is low. High severity of drift results in big changes in concept. The re-initialized primary ensemble helps RDWM achieve better accuracy as compared to the updated DWM ensemble; re-build single classifier in EDDM and the standard implementation of NB with no drift handling capabilities. When severity is low, the new concept is quite similar to the old concept. Our systems' prediction is the global prediction by its updated primary ensemble, which performs almost similarly as the updated ensemble in DWM. Evaluation using various real time drifting datasets shows that RDWM performs better than DWM, EDDM, ADWIN [23], DDM [6], PL [22], NB and HT. For static datasets, our system performs almost similarly as NB, but slightly better than DWM and EDDM.

For answering the third question, we evaluated our system using hyperplane dataset with varying severity levels. It has been observed that as the severity increases, RDWM's accuracy drops and the system has reduced homogeneity among its experts. However, variation in severity does not affect the performance of RDWM in terms of the other metrics.

For answering the fourth question, we evaluated RDWM using NB and HT as its base classifiers. NB treated all the attributes as independent whereas HT assumed feature dependence. RDWM with NB as the base classifier (RDWM-NB) performed better as compared to RDWM with HT as its base classifier (RDWM-HT). HT itself as a classifier achieves better or similar accuracy as NB. Hence, we can state that the better accuracy of RDWM-NB as compared to RDWM-HT is only because of the methodology inherent in RDWM and nothing due to the choice of the base classifier used.

For answering the next question, we evaluated RDWM in a noisy domain. For datasets with gradual drifts and noise, RDWM shows high sensitivity to noise, resulting in reduced prequential accuracy and kappa statistics. However, noise does not impact our systems' performance in terms of memory, evaluation time and model cost.

The paper is further organized as follows. In Sect. 3, we give an overview of the various existing approaches for drift detection. Section 4, gives an understanding of our proposed system in detail. In Sect. 5, we describe the various datasets and also perform a detailed evaluation of RDWM. Section 5 also discusses the statistical analysis of the experimental results. In the end, we summarize our paper and discuss the scope for future research.

3 Related work

3.1 Online approaches for handling drifting concepts

Weighted majority (WM) [13] believes that all features are not necessary for making a prediction. drift detection method (DDM) [6] detects drift by monitoring the online error-rate whereas early drift detection method (EDDM) [1] monitors the distance between prediction errors. Adaptive windowing (ADWIN) [23] uses sliding windows with variable sizes. Paired learner (PL) [22] maintains a stable learner that predicts based on its learning since the last replacement and a reactive learner that predicts based on its most recent experience.

Adaptive classifier ensemble (ACE) [20] uses an online classifier, a set of batch classifiers and a drift detection mechanism for handling recurrent drifts. An enhanced version of ACE [18] uses a pruning strategy to remove the old redundant classifiers. DWM [12] dynamically creates new experts and removes an expert if its weight reaches a threshold value. In Diversified dynamic weighted majority (DDWM) [30], the classification result is the class with the maximum support considering both the low diversity and the high diversity ensembles. L-GEM [7] is a dynamic fusion method that estimates the local competence of base classifiers in multiple classifier systems. pool and accuracy based stream classification (PASC) [28] maintains a pool of classifiers to track recurring concepts. A novel Just-In-Time (JIT) classifier [27] deals with recurrent drifts by means of a practical formalization of the concept representation and the definition of a set of operators working on such representations. A context-aware data stream learning system [15] uses available context information to improve existing ensemble approaches for handling recurrent concepts.

3.2 Performance metrics

- *Prequential accuracy* (%) It is the average accuracy calculated online by classifying every instance to be learned prior to its learning. For evaluating RDWM, we have used sliding window (w) as the forgetting mechanism [29].
- *Kappa statistic* (%) It gives a score of homogeneity among the experts [30].
- *Model cost (RAM-Hours)* One RAM-Hour is equivalent to one GB of RAM being deployed for one hour.
- *Time* (*CPU-seconds*) It is the total runtime that involves training and testing of the experts.
- *Memory* (*bytes*) It measures the total memory used to store the running statistics and the online model.

4 Recurring dynamic weighted majority (RDWM) approach

RDWM maintains two ensembles: a primary online ensemble (EO) and a secondary ensemble (EB). An expert maintains an accuracy weight, a pruning weight and an accuracy value. The accuracy weight is used for class prediction and the pruning weight helps in determining the pruning order. The accuracy value measures the accuracy of the expert for the most recent W (window size) instances. The primary ensemble in RDWM is updated or pruned as in DWM [12]. The secondary ensemble is neither updated nor trained but only copies the best expert from the primary ensemble.

For every new instance arriving in the data stream, Algorithm 1 (Global Prediction) gives the global prediction by each ensemble. Algorithm 2 (Final Prediction) is used to predict the final class prediction. Algorithm 3 (Drift Handling) updates the system upon drift detection. Algorithm 4 (Recurring Dynamic Weighted Majority) outlines the main procedure followed by our approach.

4.1 Algorithm 1: global prediction

The ensemble EO maintains online experts, each having an initial accuracy weight of one, both pruning weight and accuracy value of zero. When the local prediction (LO) is incorrect (lines 4–5), the accuracy weight of an expert in EO is reduced by a multiplicative constant (β, $0 \le \beta < 1$) [12]. However, when the local prediction is correct, the accuracy value is increased by one (lines 7–8, 21–22), at each time step. After every W instance, the accuracy value of each expert is set to zero so as to have a comparative analysis of the experts in terms of their accuracy on the most recent W instances (lines 9, 23–24).

The pruning weights of all the experts in both EO and EB are reduced by one at each time step (lines 16 and 28), except for the expert having the highest accuracy value for the most recent W instances.

```
Algorithm 1. Global Prediction ( $\{X, Y\}_1^n$ , $\beta$, $W$, $\theta$)
$\{\mathbf{X_i}, Y_i\}$: training instance at time step $i$, with a feature vector ($\mathbf{X_i}$) and class label ($Y_i$), $1 \le i \le n$
$C \in \mathbb{N}$: number of classes
$\{eo, awo, pwo, ao\}_1^m$ : primary ensemble EO containing $m$ experts and their accuracy weights, pruning weights and accuracy value, respectively
$\{eb, awb, pwb, ab\}_1^m$: secondary ensemble EB containing $m$ experts and their accuracy weights, pruning weights and accuracy value, respectively
$GO, LO, GB, LB \in \{1,\ldots, C\}$: global and local predictions by primary and secondary ensemble, respectively
$G \in \{1,\ldots, C\}$: final class prediction
WO, WB $\in \mathbb{R}^C$: sum of weighted predictions for each class by primary and secondary ensemble, respectively

1:  WO ← 0; WB ← 0;
// for experts in primary ensemble
2:    for $kl = 1,\ldots m$
3:        $LO$= classify ($\mathbf{EO}_{kl}$, $\mathbf{X}_i$); //local prediction
4:        if ($LO \neq Y_i$ and $i$ mod $W = 0$)
5:          update accuracy weight by $\beta$
6:        $WO_{LO} \leftarrow WO_{LO} + awo_{kl}$;
7:        if (local prediction correct)
8:            increase accuracy value by 1
9:      if ($i$ mod $W = 0$) {set accuracy value to 0}
10:   end for
// update pruning weight
11:   for $kl = 1,\ldots m$
12:     if (pruning weight of most accurate expert $\geq 0$)
13:         increase pruning weight by 1
14:     else if (pruning weight of most accurate expert $< 0$)
15:         set pruning weight to 0
16:     else {decrease pruning weight of expert by 1}
17:     end if
18:   end for
// for experts in secondary ensemble
19: for $kh = 1,\ldots m$
20:       $LB$= classify ($\mathbf{EB}_{kh}$, $\mathbf{X}_i$); //local prediction
21:       if (local prediction correct)
22:           increase accuracy value by 1
23:       if ($i$ mod $W = 0$)
24:           set accuracy value to 0
25:       $WB_{LB} \leftarrow WB_{LB} + awb_{kh}$;
26:   end for
27:   for $kh = 1,\ldots m$
28:     update pruning weight as in EO
29:   end for
// global prediction by each ensemble
30:    $GO \leftarrow \operatorname{argmax}_{kl} WO_{kl}$;   $GB \leftarrow \operatorname{argmax}_{kh} WB_{kh}$;
31:  if ($i$ mod $W = 0$)
32:     Normalize accuracy weights of EO experts
33:     Remove EO expert if pruning weight $\leq \theta$
34:  end if
35:  Output EO, EB, $GO$, $GB$;
```

Further, for this expert if the pruning weight is less than zero, we set its weight to zero (lines 15) else increase it by one (lines 13) at each time step. An expert in EO is removed, if its pruning weight reaches the threshold value θ (line 33).

The update of accuracy weights and removal of experts in EO is controlled by W (lines 4 and 31). The global prediction by each ensemble is the weighted majority vote of the experts' predictions and is the class with the maximum support (line 30). After each update, the accuracy weight of all the experts is normalized so that after transformation the maximum value of weight is one (line 32). Algorithm 1 outputs the global class prediction by both the ensembles (line 35).

4.2 Algorithm 2: final prediction

The final prediction (G) is the class with the maximum support, involving the weighted majority vote of the experts' predictions from both EO and EB (lines 2–4). If the support for the class as predicted by EO is more than EB, the final prediction is the class as predicted by EO (line 2) else as predicted by EB (line 3). However, in the initial learning phase when EB is empty, the final prediction is the class as predicted by EO (line 5). For every new instance, the algorithm outputs the final class prediction G (line 7).

```
Algorithm 2. Final Prediction ($\{X, Y\}_1^n$, $GO$, $GB$)
1: if (num_experts (EB) $\neq 0$)
2:     if ($WO_{GO} > WB_{GB}$)  {$G \leftarrow GO$ ;}
3:     else                      {$G \leftarrow GB$ ;}
4:     end if
5: else       {$G \leftarrow GO$ ;}
6: end if
7: Output $G$;
```

4.3 Algorithm 3: drift handling

After the first $2W$ instances have arrived in the data stream (line 1), the best expert having the highest accuracy on the most recent W instances is copied from EO into EB (lines 2–4). If the final class prediction G is incorrect (line 5), the following three cases arise:

Case I drift is detected by both ensembles.

Case II drift is detected by primary ensemble only.

Case III drift is detected by secondary ensemble only.

For Case I or Case II, the best online expert from EO is copied into EB (line10). EO is re-initialized so as to learn the next concept from scratch (line 11). However, if EB already contains m experts, the expert with the minimum pruning weight is removed from EB (lines 7–8).

Similarly for Case III, the best online expert from EO is copied into EB (line 17). A new expert trained as per the new concept is added in EO (line 21). However, if EO is already full, we remove the expert with the minimum pruning weight (lines 18–20). The handling of drift by RDWM occurs only after the first $2W$ instances have arrived in the data stream (line 1).

```
Algorithm 3.
Drift Handling ($\{X, Y\}_1^n$, $W$, $GO$, $GB$, $G$)
1:  if ( $i \geq 2W$ )
2:      if (EB contains no experts)
3:          Copy best expert from EO into EB;
4:      end if
5:      if ($G$ is incorrect)
// Case I or Case II
6:  if ( ($GO \neq Y_i$ and $GB \neq Y_i$ ) || ($GO \neq Y_i$) )
7:      if (EB contains m experts)
8:          remove expert with minimum pruning weight
9:      end if
10:     Copy best expert from EO into EB;
11:     re-initialize EO;     // re-initialize primary
12: end if
// Case III
13: if ($GB \neq Y_i$)
14:     if (EB contains m experts)
15:         remove expert with minimum pruning weight
16:     end if
17:     Copy best expert from EO into EB;
18:     if (EO contains m experts)
19:         remove expert with minimum pruning weight
20:     end if
21:     create a new expert in EO
22: end if
23: end if
24: end if
25: Output EO, EB;
```

4.4 Algorithm 4: recurring dynamic weighted majority

RDWM develops a primary ensemble (EO) using modified version of online bagging [21], containing m experts each with an accuracy weight (awo_{jl}) of one, both pruning weight (pwo_{jl}) and accuracy value (ao_{jl}) of zero (lines 1–4). Input to the system is n instances, each consisting of a feature vector and its corresponding class label (line 5).

For every new instance, we call Algorithm 1 to get the global prediction by each of the ensembles (line 6). The final class prediction for the instance is the class as predicted by Algorithm 2 (line 7). Algorithm 3 (line 8) is called when a drift is detected by our system. Training of the experts in EO is a continuous process happening at each time step and one could use any base learner considering the various parameters of the base learner (lines 9–11).

```
Algorithm 4. Recurring Dynamic Weighted
Majority ($\{X, Y\}_1^n$, $\beta$, $m$, $W$, $\theta$)
1:  EO ←new ensemble;          // primary ensemble
2:  for $jl$ =1,…. , $m$
3:      $awo_{jl} \leftarrow 1$; $pwo_{jl} \leftarrow 0$; $ao_{jl} \leftarrow 0$;
4:  end for
5:  for  $i$ =1,… $n$
6:      Global Prediction ($\{X, Y\}_1^n$, $\beta$, $W$, $\theta$)
7:      Final Prediction ($\{X, Y\}_1^n$, $GO$, $GB$)
8:      Drift Handling ($\{X, Y\}_1^n$, $W$, $GO$, $GB$, $G$)
9:      for ($kl$ =1, …… , $m$)
10:         $eo_{kl} \leftarrow$ learn_experts ($eo_{kl}$, $\mathbf{X_i}$, $Y_i$);
11:     end for
12:     output $G$;
13: end for
```

5 Experimental evaluation

5.1 Concept drifting data streams

5.1.1 Artificial datasets

5.1.1.1 Stagger concepts A Stagger concept [26] has 3 features: *shape* ∈ {triangle, circle, rectangle}, *size* ∈ {small, medium, large} and *color* ∈ {blue, green, red}. It contains 240 instances, with a new instance at each time step. A learner is evaluated based on a pair of features only. It contains abrupt drifts and recurrent drifts.

5.1.1.2 Moving hyperplane dataset The instances [9] are uniformly distributed in multi-dimensional space $[0, 1]^{10}$ and are classified as positive if they satisfy the condition as in Eq. (1)

$$w_0 \leqslant \Sigma_{i=1}^{10} w_i x_i \tag{1}$$

For the various runs of the dataset, the weights $\{w_i\}$ are initialized to [− 1, 1] randomly and updated as $w_i \leftarrow w_i + ds_i$ at each time step, where $s_i \in \{-1, 1\}$ represents the direction of change and d represents the magnitude of change. At each time step, the threshold w_0 is calculated as given in Eq. (2).

$$w_0 = \frac{1}{2}\Sigma_{i=1}^{10} w_i \tag{2}$$

$\{s_i\}$ is reset randomly after every 1000 instances. The dataset has a total of 3000 instances with gradual drifts and noise. For evaluating RDWM, the value of d was set to 0.001.

5.1.2 Real-world datasets

As these represent a real-world phenomenon, we cannot predict the occurrence of drift.

5.1.2.1 Electricity pricing domain The dataset [8] was obtained from the electricity supplier TransGrid, New South Wales Australia. It contains 45,312 instances collected at 30-min intervals between 7 May, 1996 and 5 December, 1998. Each instance consists of five features and a class label of either up or down. The prediction task is to predict the price of electricity and is affected by demand and supply.

5.1.2.2 Power supply stream The dataset [25] records hourly power supply of an Italian electricity company, measuring the supply from the main grid and the power transformed from the other grids. It maintains 3 year power supply records from 1995 to 1998, with a total of 29,928 instances. Each instance maintains two attributes. The prediction task is to predict the hour (1 out of 24) to which the current power supply belongs to.

5.1.2.3 KDD cup 1999 dataset KDD Cup 1999 [34] is a network intrusion detection dataset. It consists of a large variety of intrusions simulated in a military network environment. The data set contains 494,020 instances, with each instance maintaining 41 attributes. The target class identifies whether the connection is an attack or a normal connection. For evaluating RDWM, only 15% of the dataset i.e. 74,103 instances were used.

5.1.2.4 Breast cancer dataset This static dataset was obtained from UCI repository [31]. It classifies an instance as either a recurrence-event or a no-recurrence-event. The dataset was provided by the Oncology Institute, and maintains a total of 286 instances. Each instance maintains nine attributes with either a linear or a nominal value.

5.2 Experimental objectives, design and measures analyzed

The main objective is to study the behavior of RDWM in different situations, answering the research questions discussed in Sect. 2. Experiments were done using Massive Online Analysis (MOA) [2] tool. It is based on a Unix/ Linux system with Java 6 SDK installed. To run MOA, two jar files were needed: *moa.jar* and *sizeofag.jar*. RDWM was evaluated using various datasets, to study the change in its behavior with variations in the speed or severity of drift or presence of noise. Numerically as well as empirically RDWM has been compared with the existing approaches, measuring its average performance over 50 runs of each dataset. The results were completely in favor of RDWM.

Table 1 lists the parametric values used by each learning system. To have a fair comparison, the number of experts in each system must be same. So, we used EDDM, DDM, Adwin and PL along-with OzaBag [14]. The ensemble size in DWM was set to double the size (m) in RDWM. The examples in the real time datasets were processed in the same temporal order as they appear in the dataset, with one example at each time step. The base learners used were NB (that assumes feature independence) and HT (having a dependent feature set). In MOA, the width of sliding window (w) was set to 1000.

RDWM was evaluated using Stagger concepts with $n = 240$ instances. The three concepts are (1) $size = \text{medium}$ or $size = \text{large}$, (2) $color = \text{green}$ or $shape = \text{circle}$, (3) $size = \text{small}$ and $color = \text{red}$. The target concept changes after every 80 instances in the order (1)–(2)–(3). For evaluation of RDWM while handling recurrent drifts, we used Stagger concepts with $n = 720$ instances. As per the three concepts, the target concept changes after every 80 instances in the order (1)–(2)–(3)–(1)–(2)–(3)–(1)–(2)–(3). We randomly generate 80 examples of the current target concept with one example at each time step.

RDWM was evaluated using hyperplane dataset with $n = 3000$ instances. Noise was randomly introduced by switching the class label of 5% of the instances, with a new example at each time step. Our system was evaluated using electricity pricing dataset containing $n = 45{,}312$ instances.

RDWM was evaluated using power supply stream dataset with $n = 29{,}928$ instances. EDDM, DDM and ADWIN were used along-with OzaBag [14], containing a maximum of 96 experts. PL was used along-with OzaBag, maintaining 48 experts in each of its stable and reactive learners. Another real-time drifting dataset used was KDD CUP 1999 dataset, with $n = 74{,}103$ instances. EDDM was used along-with OzaBag, maintaining 32 experts in its system.

For evaluating RDWM on a real time static dataset, we used the Breast cancer dataset with 286 instances. EDDM was used along-with OzaBag containing 10 experts and NB as its base classifier.

RDWM was evaluated using many varied static datasets from UCI repository [31]. 30% instances were randomly selected for testing and the remaining 70% were used for training. NB was used as the base classifier, with one example at each time step. We measured the performance of our system with two settings of window size (W): 10 and 50. The width of the sliding window (w) varies as per the dataset used.

We have compared our system with some of the existing approaches for drift detection. Our approach results in better performance as compared to these existing models. It has been studied that an ensemble of classifiers provides better generalization accuracy as compared to a single classifier model [5]. Hence, the combined prediction of two ensembles in RDWM would surely provide more accurate results as compared to the single classifier systems such as EDDM, DDM and ADWIN.

Either of the ensembles in RDWM may perform better than the other ensemble, depending on the type of drift. The final prediction is that by the ensemble with better predictive accuracy, resulting in RDWM's better performance as compared to a single ensemble DWM approach. Further, the primary ensemble in RDWM is either updated or re-initialized, resulting in its better performance as compared to a single ensemble being only updated in DWM.

In RDWM, the pruning of the experts considers both the age of the expert and the history of its predictions, which is not considered in DWM. PL [22] maintains a stable learner and a reactive learner that may differ in their predictive accuracy but the systems' prediction is always that of the stable learner, even if the reactive learner may give better performance. RDWM maintains a primary and a secondary ensemble and the systems' prediction is that

Table 1 The Parametric values used by learning systems

Dataset	Values for preliminary experiments					Final parametric values for various tested models
	α (EDDM)	β' (EDDM)	p (DWM)	m (RDWM)	W (RDWM)	
Stagger	{0.96, 0.97, 0.98, 0.99, 1.1, 1.2, 1.3}	{0.75, 0.85, 0.95}	{5,10,15, 20,25, 30,35,40}	{3,4,5,6, 10, 11,15, 16,48,49, 50}	{10,20, 30, 40, 800, 1000, 2000, 3000, 4000}	$\alpha = 0.95$, $\beta' = 0.90$, $p = 10$, $m = 3$, $W = 10$, m' (DWM) = 6, num_experts (EDDM) = 6, base classifier = NB
Stagger (recurrent)						$\alpha = 0.95$, $\beta' = 0.90$, $p = 10$, $m = 3$, $W = 10$, m' (DWM) = 6, num_experts (EDDM) = 6, base classifier = NB
Hyperplane						$\alpha = 0.95$, $\beta' = 0.90$, $p = 5$, $m = 3$, $W = 800$, m' (DWM) = 6, num_experts (EDDM) = 6, base classifier = NB, HT
Electricity pricing						$\alpha = 0.95$, $\beta' = 0.90$, $p = 30$, $m = 15$, $W = 30$, m' (DWM) = 30, num_experts (EDDM) = 30, base classifier = NB,HT
Power supply						$\alpha = 0.95$, $\beta' = 0.90$, $p = 40$, $m = 48$, $W = 4000$, m' (DWM) = 96, num_ experts (EDDM, DDM, Adwin, PL) = 96, base classifier = NB
KDD CUP 1999						$\alpha = 0.95$, $\beta' = 0.90$, $p = 25$, $m = 16$, $W = 3000$, m' (DWM) = 32, num_ experts (EDDM) = 32, base classifier = NB
Breast cancer						$\alpha = 0.95$, $\beta' = 0.90$, $p = 40$, $m = 5$, $W = 40$, m' (DWM) = 10, num_ experts (EDDM) = 10, base classifier = NB

PL uses window size and threshold value of 12 and 0.2, respectively. For DWM and RDWM, β and θ has been set to 0.5 and 0.01, respectively. For DDM, the minimum number of instances before detecting a change is set to 30. Pilot studies indicated that these were the near-optimal settings of the ensemble with better generalization accuracy. Our system may provide better accuracy as compared to NB and HT that have not been designed to handle drifts and learn from all the examples in the stream, regardless of changes in the target concept.

5.3 Evaluation and results

5.3.1 Evaluation on Stagger concepts

RDWM-NB achieves similar prequential accuracy as DWM-NB (DWM with NB as base learner), EDDM-NB (EDDM with NB as base learner) and NB on the first target concept as seen in Fig. 1a. However, on the second and third target concepts, it achieves the highest accuracy among all the approaches. The graph of RDWM has a better slope and asymptote, showing its quick convergence to the new target concepts.

RDWM requires more evaluation time than DWM as shown in Fig. 1b. The rate of updates in RDWM is higher as compared to DWM, with RDWM having a higher slope. However, our system requires almost similar time as EDDM. This is because whenever the drift level is triggered in EDDM, the system is reset requiring more time to train the new model. It has been observed that all the graphs follow a similar stair case pattern, illustrating a sudden increase in evaluation time. Analysis of the results in Table 2 led us to state that the variation in ensemble size does not impact the performance of RDWM in terms of any of the performance metrics, on average.

From the analysis of the results in Table 3, we can state that RDWM performed very poorly in terms of accuracy and kappa statistic, when the performance was based on only the recent instance i.e. window size (*W*) of 1. RDWM with window size of 10 achieved higher accuracy as compared to system with window size of 100. Hence, RDWM should have an optimal window size. If *W* is too large, our window consists of instances belonging to different concepts and RDWM cannot accurately classify any given concept. If *W* is too small, then there are not enough instances to develop and train a highly accurate model. However, there is no impact of window size on time and memory requirements of the system.

The analysis of the results in Table 4 led us to state that RDWM provides the best average accuracy of 91.38%, far better than the other approaches while handling Stagger concepts. Our system also performed the best in terms of homogeneity among the experts as validated by its kappa statistics value. RDWM proves to be better than EDDM, achieving higher accuracy at lower model-cost.

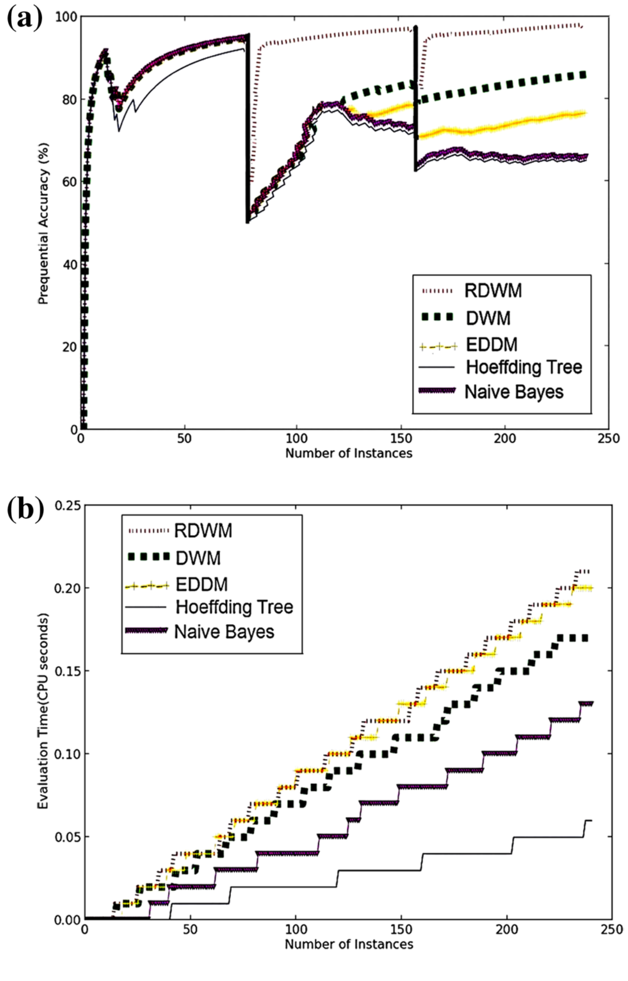


Fig. 1 Performance of RDWM-NB on Stagger concepts, **a** prequential accuracy, **b** evaluation time

Table 2 Results of evaluation of RDWM-NB on Stagger concepts with variation in ensemble size (*m*)

	$m=1$	$m=10$	$m=$ no limit
Accuracy	91.38	91.38	91.38
Kappa statistic	71.68	71.68	71.68
Model cost (*exp. − 9)	0.28	0.28	0.28
Time	0.10	0.10	0.10
Memory	0.01	0.01	0.01

Table 3 Results of evaluation of RDWM-NB on Stagger concepts with variation in window size (*W*)

	$W=10$	$W=1$	$W=100$
Accuracy	91.38	66.26	77.83
Kappa statistic	71.68	14.97	32.78
Model cost (*exp. − 9)	0.28	0.28	0.28
Time	0.10	0.10	0.10
Memory	0.01	0.01	0.01

Table 4 Average results of evaluation of RDWM-NB on Stagger concepts

	RDWM	EDDM	DWM	HT	NB
Accuracy	91.38	80.89	84.37	77.65	76.57
Kappa statistic	71.68	42.64	52.39	24.29	23.46
Cost (*exp. − 9)	0.28	0.32	0.25	0.00	0.00
Time	0.10	0.10	0.09	0.02	0.01
Memory	0.01	0.00	0.01	0.00	0.00

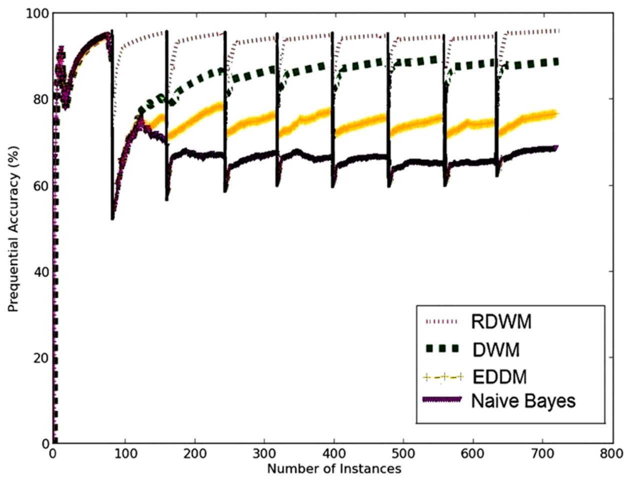


Fig. 2 Performance of RDWM-NB on Stagger concepts containing recurrent drifts

While handling Stagger concepts with recurrent drifts RDWM-NB performs similarly as DWM-NB, EDDM-NB and NB classifier on the first target concept as seen in Fig. 2. However, as learning progresses RDWM achieves the highest accuracy along-with quick convergence to the target concepts. The analysis led us to state that the better performance of RDWM as compared to DWM is because of better learning of its experts.

While handling Stagger concepts with recurrent drifts the time and cost required by RDWM is lower as compared to EDDM, proving it to be highly resource efficient as shown in Table 5. RDWM is better than DWM, achieving higher accuracy with a slight increase of time and cost. From the analysis of the results in Tables 4 and 5, we can state that RDWM performs with higher accuracy averaging nearly 93.05% when handling recurrent drifts as compared to 91.38% accuracy while handling Stagger concepts with no recurrent drifts. Moreover, a drop in the accuracy of EDDM and NB has been observed while handling recurrent concepts, proving RDWM to be the best system.

Table 5 Results on Stagger concepts with recurrent drifts

	RDWM	EDDM	DWM	NB
Accuracy	93.05	76.51	86.87	69.65
model cost (*exp. − 9)	0.78	0.84	0.72	0.14
Time	0.32	0.34	0.26	0.12

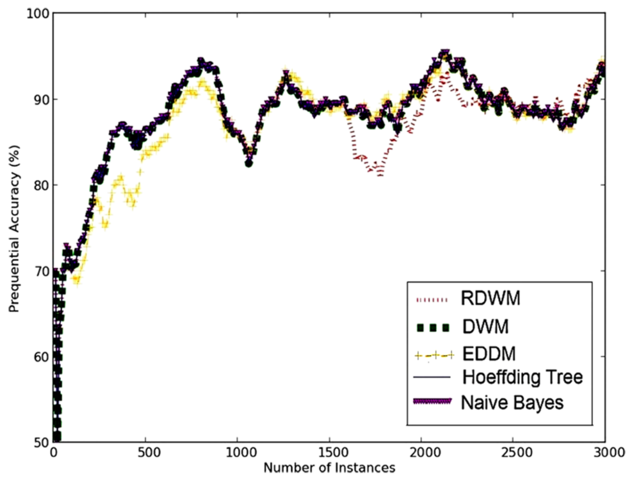


Fig. 3 Accuracy of RDWM-NB on hyperplane dataset

Table 6 Different severity levels in hyperplane dataset

Level of severity	Low	Medium	High
Magnitude of change (d)	0.0001	0.001	0.01

5.3.2 *Evaluation on moving hyperplane dataset*

As illustrated in Fig. 3, on the first 950 instances RDWM-NB performs with better accuracy than EDDM-NB and similarly as DWM-NB, NB and HT. In the period between time steps, 950 and 1600, RDWM and EDDM perform with almost similar accuracy. However, after 2600 time steps RDWM responds quickly to changes, achieving better accuracy than EDDM and DWM.

To obtain different levels of severity of drift, we vary the magnitude of change as shown in Table 6. When the drift has low severity, RDWM performed with almost similar accuracy as the other learners during the first 1600 time steps as seen in Fig. 4a. However, its performance dropped considerably in the period surrounding time step 1600, owing to large number of misclassifications. But after extensive updates, around 2800 time steps RDWM outperforms all the other learners. RDWM is highly robust to change as is evident around time step 2600, where RDWM's accuracy varies slightly by 2% on average.

While handling medium severity drift our approach achieved better accuracy than EDDM in the first 900 time steps, as seen in Fig. 4b. However, in the period between time steps, 900 and 1250, all the approaches performed similarly. Around time step 1600 RDWM suffered a performance drop of nearly 6%, which is very low as compared to its 14% drop while handling low severity drifts (Fig. 4a). Hence, RDWM proves to be more robust to medium severity drift. After 2300 time steps, our system outperforms all the other learners achieving nearly 95% accuracy.

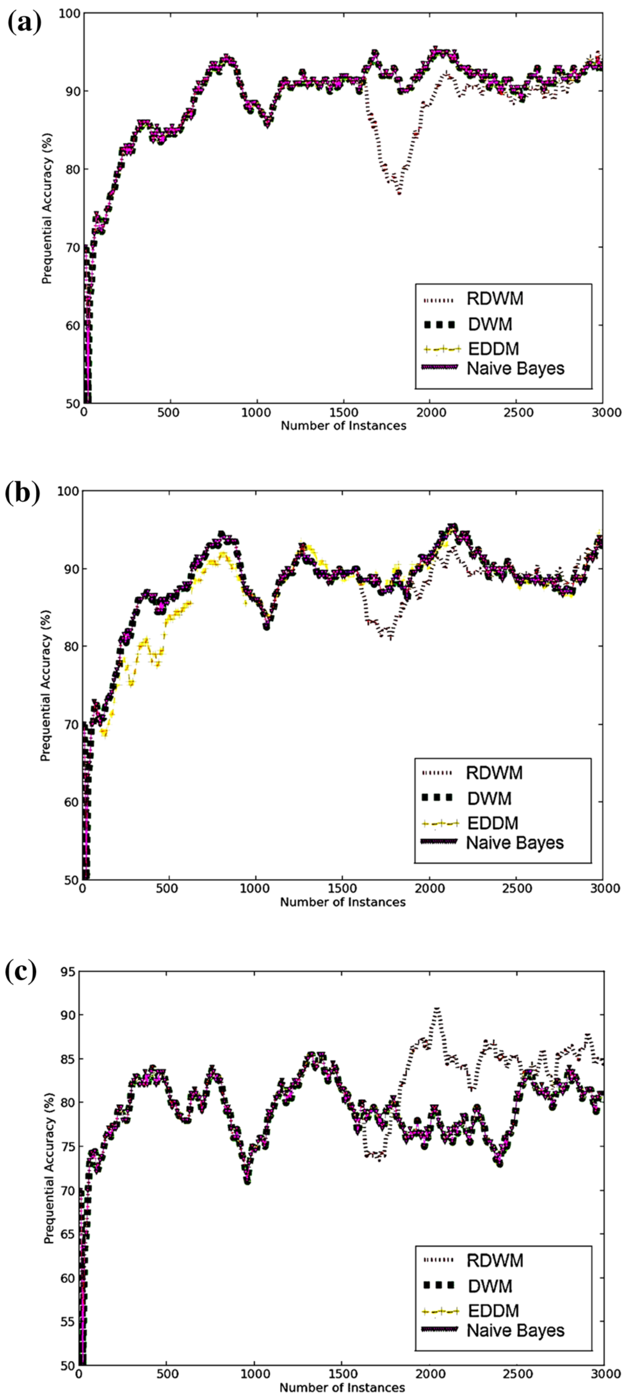


Fig. 4 Performance of RDWM-NB on hyperplane dataset with varying severity levels **a** low, **b** medium, **c** high

As observed from Fig. 4a–c, we can state that in the first 1600 time steps, RDWM achieved similar accuracy as DWM and NB, irrespective of severity levels. However, after 1600 time steps, the drop in accuracy of RDWM in high severity drift is similar to its drop in medium severity, as seen in Fig. 4c, b, respectively. As shown in Fig. 4c, after 1800 time steps RDWM converges very quickly to the target concepts, achieving the highest accuracy among EDDM, DWM and NB. Analysis of the results in Table 7 states that RDWM performs the best in terms of accuracy and kappa statistics, when the severity of drift is low. However, a change in severity does not impact the performance of RDWM in terms of time and memory.

When RDWM is evaluated using hyperplane dataset without noise, it achieves better accuracy as compared to its accuracy in a noisy domain as seen in Fig. 5. Around 900 time steps, our approach made more misclassifications and resulted in sudden drop of roughly 13% accuracy in the noisy domain as compared to a steady drop of only 4% in a non-noisy domain. As learning progresses the differential in the performance of RDWM is reduced, with a differential of only 3% around 2500 time steps as compared to a huge differential of 10% around 1000 time steps.

While handling gradual drifting dataset, RDWM achieves the highest accuracy averaging 88.44% (independent of the base classifier), as shown in Table 8. However, the use of HT as base-classifier resulted in a considerable increase in its time and memory requirements. Hence, for handling gradual drifts the best base classifier for RDWM is NB. Our system also performed the best in terms of homogeneity among its experts. RDWM provides a better system than EDDM, achieving higher accuracy in lesser time. Our approach also outperforms DWM, achieving higher accuracy levels.

5.3.3 *Evaluation on electricity pricing domain*

As seen in Fig. 6a, RDWM-NB performs the best, achieving the highest accuracy levels. Overall, NB averaged 73.40% accuracy, EDDM has accuracy of 84.82%, DWM has 84.18% accuracy whereas RDWM-NB achieved the highest

Table 7 Performance of RDWM on moving hyperplane dataset with different levels of severity of drift

	Low	Med	High
Accuracy (%)	88.74	88.44	82.44
Kappa statistic (%)	77.38	76.42	64.72
Time (CPU-s)	0.20	0.20	0.20
Memory (bytes)	0.01	0.01	0.01

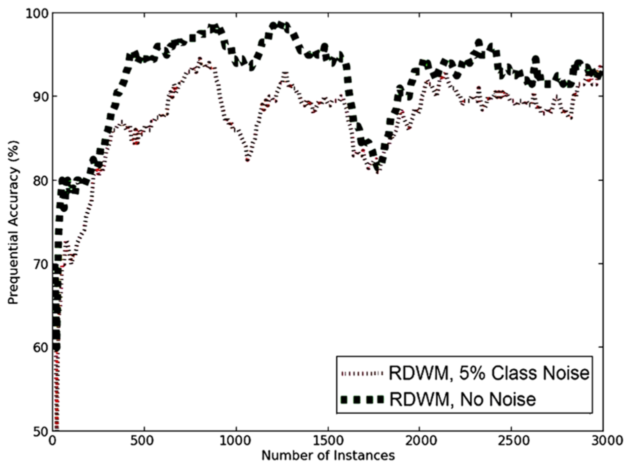


Fig. 5 Accuracy of RDWM on hyperplane dataset with 5% noise

accuracy of 85.80%. Our system has high adaptability to the new concepts as RDWM shows an increase in its accuracy levels whereas DWM, EDDM and NB observe a gradual drop around time step 23,000. Further, between time steps, 22,000 and 24,500, RDWM provides a highly stable system as it shows a slight variation of 2% in its accuracy levels as compared to a huge variation of 7.5% for EDDM.

The graphs in Fig. 6b led us to conclude that RDWM-NB performs better than RDWM-HT, achieving higher accuracy along-with quick convergence to the target concepts around time step 40,000. RDWM-HT is more robust to change as compared to RDWM-NB. This is illustrative in the period surrounding time step 34,000, where RDWM-NB shows a sudden drop of 11% in its accuracy whereas RDWM-HT shows a gradual slight decrease in its accuracy levels.

Table 9 shows that a variation in the threshold value or the multiplicative factor does not impact the performance of RDWM in terms of any of the performance metrics apart from evaluation time. Further, the most appropriate value of θ is 0.001 and β is 0.9, resulting in a highly resource efficient system.

The results in Table 10 led us to state that RDWM-NB performs with almost similar average accuracy and kappa statistics as RDWM-HT, with reduced memory and evaluation time. Hence, our system performs the best when NB is used as the base classifier. RDWM-NB achieves better accuracy than DWM-NB and EDDM-NB, with slightly higher evaluation time.

5.3.4 Evaluation on power supply stream

As illustrated in Fig. 7, RDWM-NB achieves the highest average accuracy among all the approaches. As seen around 18,000 time steps, our system converges very quickly to the target concepts. RDWM-NB observes an increase in its accuracy levels whereas ADWIN, PL, EDDM and DDM show a gradual drop. Most illustrative is a sudden rise of 18% in accuracy of RDWM as compared to only 11% in case of DWM, between time steps, 24,000 and 24,500. RDWM provides a highly stable system as seen between time steps, 21,000 and 23,000, with a slight variation of nearly 1.4% accuracy as compared to a huge variation of 6% in accuracy of DDM, 8% for ADWIN, 3.8% for PL and 4% for DWM.

Results in Table 11 shows that RDWM requires the least evaluation time as it updates only the primary ensemble. On the contrary, PL requires maximum time followed by EDDM, DDM and ADWIN. This is because PL involves the maximum number of updates. The number of updates in EDDM, DDM and ADWIN are almost equal but higher as compared to RDWM. Hence, RDWM proves to be the best system for handling power supply stream, achieving highest accuracy of 16.32% in least time.

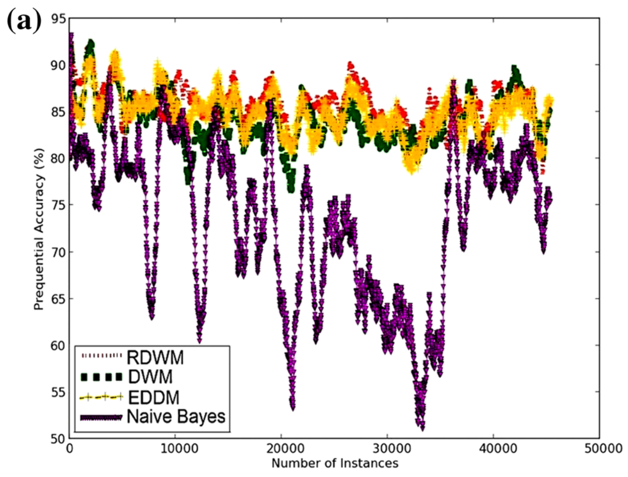
5.3.5 Evaluation on KDD CUP 1999 dataset

As seen in Fig. 8a, at each time step RDWM-NB performs with almost similar accuracy as DWM-NB. For the first 40,000 time steps, RDWM performs with similar accuracy as NB but as learning progresses, it adapts very quickly to the new target concepts achieving very high accuracy levels whereas NB observes a gradual drop, around 42,500 time steps.

In the initial learning phase, the evaluation time of RDWM is same as DWM as seen in Fig. 8b. However, around 59,000 time steps the differential between the two time graphs increases, with RDWM taking less time. Also the rate of updates in RDWM is less as compared to DWM, with its graph having a lower slope. The analysis of the results in Table 12 led us to state that RDWM-NB achieves better accuracy as compared to EDDM, with lower time and

Table 8 Experimental results for RDWM-NB on hyperplane dataset, average over 50 runs of the dataset

	RDWM-NB	EDDM-NB	DWM-NB	NB	HT	RDWM-HT
Accuracy	88.44	87.06	87.95	87.95	87.95	88.40
Kappa statistic	76.42	73.96	75.77	75.77	75.77	76.29
Model cost (*exp. – 9)	0.74	0.71	0.72	0.12	0.12	1.04
Time	0.20	0.25	0.17	0.10	0.10	0.31
Memory	0.01	0.00	0.01	0.01	0.01	0.02



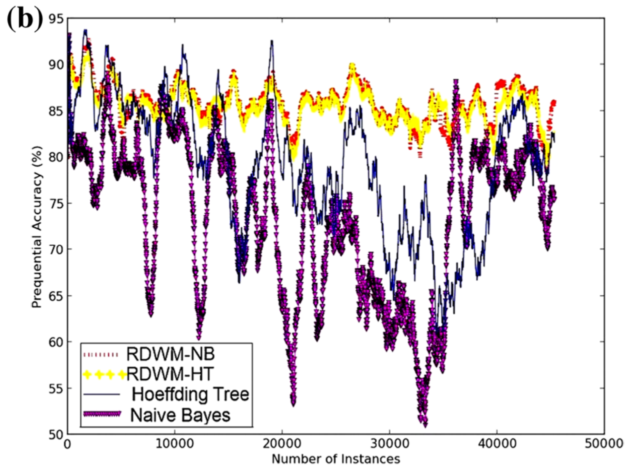


Fig. 6 Accuracy of RDWM on electricity pricing domain **a** using NB as base classifier, **b** using NB as compared to using HT as base classifiers

Table 9 Results for RDWM–NB on electricity pricing domain, varying threshold value (θ) and multiplicative factor (β)

	$\Theta=0.01$ $\beta=0.1$	$\Theta=0.01$ $\beta=0.9$	$\Theta=0.1,$ $\beta=0.5$	$\Theta=0.01$ $\beta=0.5$	$\Theta=0.001$ $\beta=0.5$
Accuracy	85.80	85.80	85.80	85.80	85.80
Kappa statistic	70.80	70.80	70.80	70.80	70.80
Time	4.39	4.26	4.43	4.72	4.40
Memory	0.01	0.01	0.01	0.01	0.01

memory requirements. Our system achieves similar accuracy as DWM in less evaluation time, making it the best system for handling drifts.

5.3.6 Evaluation on breast cancer dataset

With the progress in learning, RDWM-NB achieves better accuracy than EDDM-NB and DWM-NB, as seen in Fig. 9a. This is evident after 200 time steps, where RDWM converges very quickly to the target concepts achieving an accuracy of nearly 74%. However, DWM shows a gradual drop with nearly 68% accuracy and EDDM with further lower accuracy of 67%. RDWM provides a more stable system than DWM. This can be seen in the period between time steps 160 and 230, where accuracy of RDWM remains nearly constant (2% variation) while DWM illustrates a variation of almost 6%.

As seen in Fig. 9b, RDWM require least evaluation time as compared to EDDM and DWM. As learning progresses, the differential between the various time graphs increases. Overall, the rate of updates is the lowest in RDWM, as evident by the number of transitions in its time graph. Hence our approach provides the best system for drift detection.

5.3.7 Evaluation on static concepts

In RDWM no new functionality has been explicitly introduced for making it beneficial for handling static concepts. However, the empirical evaluation of RDWM on static datasets from UCI repository [31], led us to state that RDWM performs no worse as compared to a single classifier NB system. From the analysis of the results in Table 13, we can state that RDWM-NB outperformed NB or behaved similarly as NB on most of the tasks (9 out of 16), with an accuracy difference of nearly 1%. On the other tasks where NB performed better, the average difference was also within this range. Overall, the average accuracy difference is +0.41% in RDWM's favor.

To summarize, while handling Stagger concepts (abrupt drift with recurrence or non-recurrence) RDWM provides the best accuracy among all the approaches. Our system proves to be more resource efficient as compared to EDDM. Variation in ensemble size does not impact the performance of RDWM in terms of any of the performance

Table 10 Average results for RDWM on electricity pricing domain, with NB and HT as the base learners

	RDWM-HT	DWM-HT	HT	NB	RDWM-NB	DWM-NB	EDDM-NB
Accuracy (%)	85.51	87.54	79.23	73.40	85.80	84.18	84.82
Kappa statistic (%)	70.04	74.16	56.24	39.95	70.80	66.99	68.38
Time (CPU-s)	6.51	8.83	12.46	1.66	4.72	4.44	3.30
Memory (bytes)	0.02	0.03	0.07	0.01	0.01	0.01	0.00

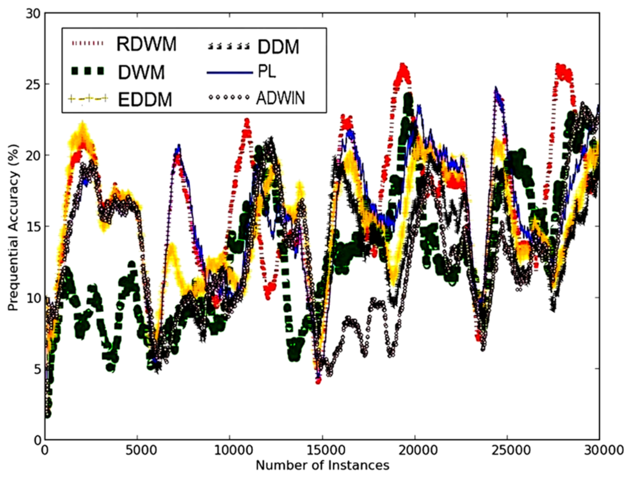


Fig. 7 Performance of RDWM-NB on power supply stream

metrics. Further, RDWM provides better prequential accuracy while handling recurrent concepts as compared to non-recurrent drifts.

While handling hyperplane dataset containing slow gradual drifts, RDWM-NB achieves the best accuracy and proves to be highly resource efficient as compared to the other learning models. It performs the best when the dataset contains low levels of severity of drift in a non-noisy domain.

Evaluation on Electricity pricing domain shows that RDWM-NB performs the best, achieving the highest accuracy along-with high adaptability to the new concepts. RDWM-NB provides a more resource efficient system than RDWM-HT. Variation in threshold value and multiplicative factor does not impact the performance of RDWM in terms of any of the performance metrics apart from evaluation time.

While handling power supply stream dataset, our system provides a highly stable, most time efficient and highly accurate system. Further, RDWM proves to be the best system for handling KDD CUP 1999 dataset. RDWM achieves almost similar accuracy as DWM and EDDM in lesser time and memory.

Evaluation on Breast Cancer dataset proves RDWM to be the best system, achieving highest accuracy in least time as compared to EDDM and DWM. RDWM performs no worse than single classifier NB system while handling static datasets.

5.4 Statistical analysis of the experimental results

Results in Table 14 prove RDWM to be the best system, achieving the highest accuracy among all the approaches while handling Stagger concepts and Stagger concepts with recurrent drifts, power supply stream and KDD Cup 1999 dataset. RDWM with NB as its base classifier has a smaller value of standard deviation as compared to other systems. A smaller deviation meant RDWM to be highly stable, with slight variation in accuracy from the mean and high adaptability to the new concepts.

Results on hyperplane dataset show that RDWM-HT provides a more stable system than RDWM-NB. Results on electricity pricing dataset prove that RDWM-NB performs with slightly better accuracy than RDWM-HT, with smaller standard deviation resulting in a more stable system.

To validate the performance of RDWM, some statistical tests were applied on the experimental results. These tests have been widely applied in the machine learning domain [10, 17, 19, 24, 33]. In this study, we applied Friedman and Kruskal Wallis tests [33] to show significant difference between the performance of RDWM and other systems being compared. After rejection of null hypothesis, a Post-hoc test called Holm's method was also applied for the same. The value of level of confidence (α) was set to 1. The results of statistical tests are shown in Tables 15, 16 and 17. Table 15 shows the average ranking of the various systems that have been used for comparison. From the results, it can be seen that RDWM obtains the first rank whereas NB obtained the fourth rank. Statistical and p-values of both tests are mentioned in Table 16 and the null hypothesis is rejected. Further, a Post-hoc test called Holm's method was carried out to show that RDWM performs better than all the other approaches being compared. Table 17 shows the results of post-hoc test on the confidence level 1. It was also observed that the hypothesis was rejected; and based on the above observations we can state that RDWM provides better results experimentally as well as statistically.

Table 11 Average results of evaluation of RDWM-NB on power supply stream

	RDWM	DWM	DDM	EDDM	ADWIN	PL	NB
Accuracy	16.32	12.92	13.79	15.02	12.74	15.68	15.85
Kappa statistic	12.69	9.15	10.05	11.33	8.95	12.02	12.20
Time	2.64	19.29	112.77	124.12	108.18	209.74	1.16
Memory	0.01	0.01	0.30	0.30	0.30	0.15	0.01

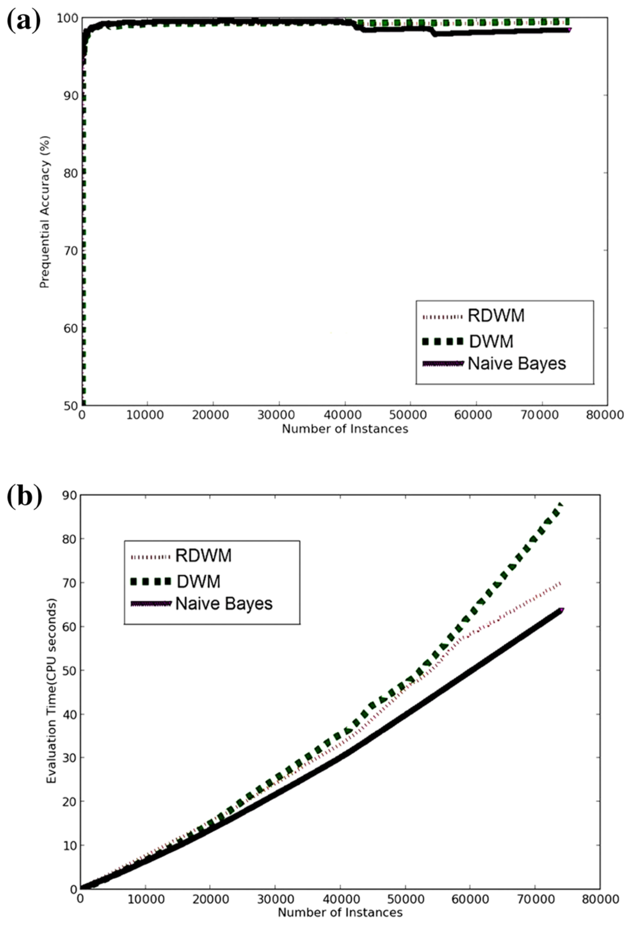


Fig. 8 Performance on KDD CUP **a** accuracy, **b** time

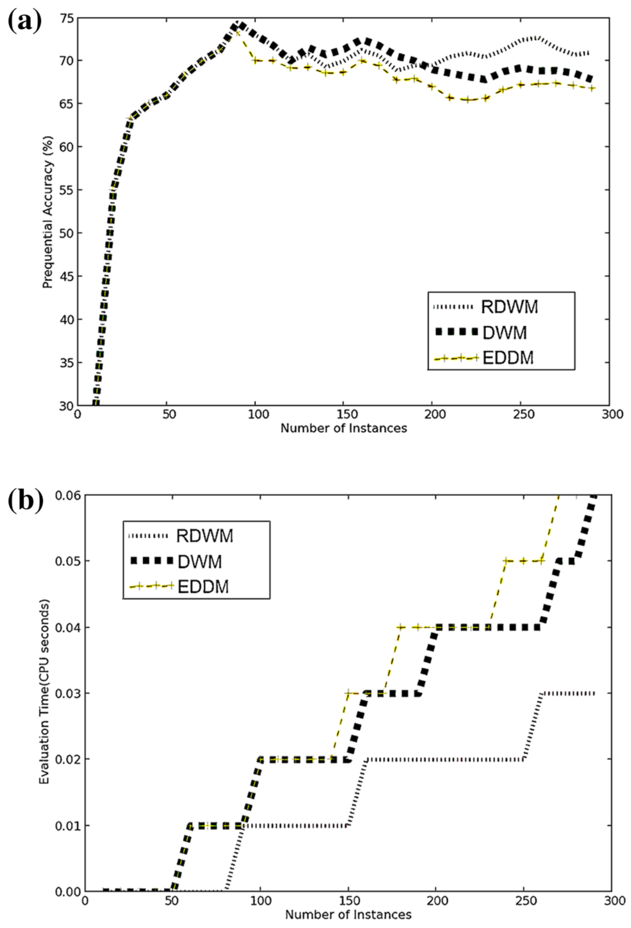


Fig. 9 Performance on breast cancer dataset **a** prequential accuracy, **b** evaluation time

Table 12 Results for RDWM-NB on KDD CUP 99 dataset

	RDWM	DWM	EDDM	NB
Accuracy (%)	99.26	99.28	99.15	98.88
Time (CPU-s)	33.17	35.93	671.98	29.01
Memory (bytes)	0.05	0.05	0.79	0.07

6 Conclusions

In our paper, we provide empirical as well as statistical evaluation of RDWM for handling drifting concepts, mainly recurrent drifts. Analysis of the results using Stagger concepts states that RDWM provides the best average prequential accuracy, using an optimal window size for handling sudden and recurrent drifts. Our approach responds quickly to gradual changes. For low severity drifts, RDWM performs the best in terms of accuracy and kappa statistics. It has been observed that a change in the severity level does not impact the performance of our system in terms of time and memory. The evaluation of RDWM using real-time drifting datasets proves RDWM to be the best system, performing very accurately even in a resource constrained environment. Experiments done on various static datasets conclude that our approach performs no worse than NB. Hence, RDWM could be used for classification of any dataset varying from static to highly dynamic datasets. The statistical analysis of the experimental results proves our system to be the best performing system among all the approaches.

For future work, we plan to improvise our approach for handling drifting datasets with weights assigned to instances. We would also like to enhance our approach for handling novel class detection in data streams.

Table 13 Static datasets from UCI repository with the various parameters

Dataset	Number of instances (n)	Ensemble size (m)	Sliding window (w)	NB	RDWM-NB		Δ%
					W = 10	W = 50	
Colic	368	3	1000	75.04 ± 0.24	73.99 ± 0.14	**75.59 ± 1.26**	**+0.55**
Glass	214	3	1000	46.73 ± 1.25	40.42 ± 1.17	**48.86 ± 0.21**	**+2.13**
Heart-c	920	3	1000	**82.04 ± 0.47**	67.37 ± 2.60	80.18 ± 0.36	**− 1.86**
Hepatitis	155	7	10	82.07 ± 0.55	**83.17 ± 0.76**	**83.17 ± 0.74**	**+1.10**
Hypo-thyroid	3772	4	1000	95.57 ± 0.06	94.10 ± 0.05	**95.66 ± 0.02**	**+0.09**
Ionosphere	351	6	10	**88.58 ± 0.41**	85.67 ± 0.71	87.84 ± 0.63	**− 0.74**
Labor	57	4	10	**89.67 ± 0.61**	88.37 ± 0.94	89.33 ± 1.00	**− 0.34**
Lung-cancer	32	4	10	**59.37 ± 3.13**	57.20 ± 0.95	**59.37 ± 3.13**	**0.00**
Lymph	148	3	100	**77.45 ± 1.77**	76.54 ± 0.75	76.54 ± 0.85	**− 0.91**
Mushroom	8124	3	1000	**98.20 ± 0.03**	97.90 ± 0.30	97.49 ± 0.43	**− 0.30**
Segment	2310	5	1000	**79.54 ± 0.16**	48.92 ± 3.48	78.55 ± 0.65	**− 0.99**
Sick	3772	4	1000	92.06 ± 1.24	93.23 ± 0.27	**93.34 ± 1.86**	**+1.28**
Soybean	683	6	1000	**91.87 ± 0.16**	89.81 ± 0.28	89.99 ± 1.15	**− 1.88**
Splice	3190	6	1000	94.92 ± 0.22	**98.90 ± 0.70**	97.73 ± 0.97	**+3.98**
Vehicle	946	3	1000	45.97 ± 1.64	44.73 ± 0.44	**47.20 ± 1.41**	**+1.23**
Zoo	101	7	1000	**88.06 ± 0.07**	84.09 ± 0.08	**88.06 ± 0.07**	**0.00**

Measures are prequential accuracy of RDWM-NB, with window size of 10 and 50, averaged over 50 runs of each dataset. Maximum accuracies are typeset in boldface. The last column is the % difference between NB and the better performing RDWM system

Table 14 Results for RDWM on various datasets, average over 50 runs of each dataset

Dataset	Approaches		
Stagger concepts	**RDWM-NB: 91.38 ± 2.2**; HT: 77.65 ± 9.7	EDDM-NB: 80.89 ± 2.56; NB: 76.57 ± 10.98	DWM-NB: 84.37 ± 2.30
Stagger concepts with recurrent drift	**RDWM-NB: 93.05 ± 1.26**; NB: 69.65 ± 3.68	EDDM-NB: 76.51 ± 1.51	DWM-NB: 86.87 ± 2.71
Hyperplane dataset	**RDWM-NB: 88.44 ± 2.59; RDWM-HT: 88.40 ± 1.84**	EDDM-NB: 87.06 ± 3.4; NB: 87.95 ± 2.48	DWM-NB: 87.95 ± 4.62; HT: 87.95 ± 1.87
Power supply stream	**RDWM-NB: 16.32 ± 2.55;** EDDM: 15.02 ± 4.23; NB: 15.85 ± 7.60	DWM-NB: 12.92 ± 8.13; ADWIN: 12.74 ± 7.59; PL: 15.68 ± 8.34	DDM: 13.79 ± 4.41
Electricity pricing	**RDWM-HT: 85.51 ± 1.86;** DWM-NB: 84.18 ± 1.77; NB: 73.40 ± 6.06	**RDWM-NB: 85.80 ± 1.07**; EDDM-NB: 84.82 ± 1.26	DWM-HT: 87.54 ± 2.65; HT: 79.23 ± 6.43
KDD Cup 1999	**RDWM-NB: 99.26 ± 0.09**; EDDM: 99.15 ± 0.11		**DWM-NB: 99.28 ± 0.12;** NB: 98.88 ± 0.51
Breast cancer	**RDWM-NB: 68.25 ± 2.73**; NB: 68.42 ± 3.41		DWM-NB: 67.66 ± 0.17; EDDM-NB: 66.28 ± 0.6

Measures are prequential accuracies along-with the corresponding standard deviations. Maximum and more stable accuracies are typeset in boldface

Table 15 Average ranking based on Friedman test using accuracy parameter

RDWM	EDDM	DWM	NB
1	3	2.7	3.3

Table 16 Statistics of Friedman and Kruskal Wallis tests on accuracy parameter

Method	Statistical value	p value	Hypothesis
Friedman test	9.7347	0.0263	Rejected
Kruskal–Wallis test	11.7286	0.0189	Rejected

Table 17 Results of Post-hoc test after rejection of null hypothesis using accuracy parameter

Algorithm	z values	p values	α/i, α = 1	Hypothesis
NB	1.8779	0.0604	0.25	Rejected
EDDM	1.633	0.1025	0.5	Rejected
DWM	1.388	0.1651	1	Rejected

References

1. Baena-Garcia M, Campo-Avila JD, Fidalgo R, Bifet A (2006) Early drift detection method. In: Proc. 4th ECML PKDD Int'l Workshop Knowled. Discovery from Data Streams, pp 77–86
2. Bifet A, Holmes G, Kirkby R, Pfahringer B (2010) MOA: massive online analysis, a framework for stream classification and clustering, JMLR: workshop and conference proceedings, vol 11, p 44
3. Gama J, Žliobaitė I, Bifet A, Pechenizkiy M, Bouchachia A (2014) A survey on concept drift adaptation. ACM Comput Surv 46(4):1–37
4. Dawid A, Vovk V (1999) Prequential probability: principles and proper ties. Bernoulli 5(1):125–162
5. Dietterich TG (1997) Machine learning research: four current directions. Artif Intell 18(4):97–136
6. Gama J, Medas P, Castillo G, Rodrigues P (2004) Learning with drift detection, SBIA'04, pp 286–295
7. Patrick PK, Yeung DS, Ng WWY et al (2012) Dynamic fusion method using localized generalization error model. Inf Sci 217:1–20
8. Harries M (1999) Splice-2 comparative evaluation: electricity pricing, Technical report. University of New South Wales, Australia, July 1999
9. Hulten G, Spencer L, Domingos P (2001) Mining time-changing data streams. KDD, San Francisco, pp 97–106
10. Garcı´a S, Ferna´ndez A, Luengo J et al (2010) Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power. Inf Sci 180(10):2044–2064
11. Ramamurthy S, Bhatnagar R (2007) Tracking recurrent concept drift in streaming data Using ensemble classifiers. ICMLA'07, pp 404–409
12. Kolter JZ, Maloof MA (2007) Dynamic weighted majority: an ensemble method for drifting concepts. JMLR 8:2755–2790
13. Littlestone N, Warmuth M (1994) The weighted majority algorithm. Inform Comput 108:212–261
14. Oza N, Russell S (2001) Online bagging and boosting. In: Artificial intelligence and statistics 2001". Morgan Kaufmann, pp 105–112
15. Gomes J, Menasalvas E, Sousa P (2011) Learning recurring concepts from data streams with a context-aware ensemble. In: ACM Symp. on Applied Computing, pp 994–999
16. Minku LL, Yao X (2012) DDD: a new ensemble approach for dealing with concept drift. ITKDE 24(4):619
17. Demsar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30
18. Nishida K, Yamauchi K (2007) Adaptive classifiers-ensemble system for tracking concept drift. In: ICMLC'07, pp 3607–3612
19. Kumar Y, Sahoo G (2015) Hybridization of magnetic charge system search and particle swarm optimization for efficient data clustering using neighborhood search strategy. Soft Comput 19(12):3621–3645
20. Nishida K, Yamauchi K, Omori T (2005) ACE: Adaptive classifiers-ensemble system for concept-drifting environments. In: 6th Int'l Workshop on Multiple Classifier Systems, ser. LNCS, vol 3541, pp 176–185
21. Oza NC, Russell S (2001) Experimental comparisons of online and batch versions of bagging and boosting. In: Proc. of Seventh ACM SIGKDD'01. ACM, NY, pp 359–364
22. Bach SH, Maloof MA (2008). Paired learners for concept drift. ICDM'08, Los Alamitos, pp 23–32
23. Bifet A, Gavaldà R (2007) Learning from time-changing data with adaptive windowing. In: SDM'07. SIAM, Florida, pp 443–448
24. Kumar Y, Sahoo G (2015) A two-step artificial bee colony algorithm for clustering. NCAA, pp 1–15
25. Zhu X (2010) Stream data mining repository. http://www.cse.fau.edu/~xqzhu/stream.html. Accessed 13 Mar 2016
26. Schlimmer JC, Granger RH (1986) Incremental learning from noisy data. Mach Learn 1(3):317–354
27. Alippi C, Boracchi G, Roveri M (2013) Just-in-time classifiers for recurrent concepts. IEEE Trans Neural Netw Learn Syst 24(4):620–634
28. Hosseini M, Ahmadi Z, Beigy H (2013) Using a classifier pool in accuracy based tracking of recurring concepts in data stream classification. ES 4:43–60
29. Gama J, Sebastiao R, Rodrigues PP (2009) Issues in evaluation of stream learning algorithms. In: ACM SIGKDD'09, pp 329–338
30. Sidhu P, Bhatia MPS (2015) A novel online ensemble approach to handle concept drifting data streams: diversified dynamic weighted majority, IJMLC. Springer, Berlin Heidelberg
31. Asuncion A, Newman DJ (2007) UCI machine learning repository. University of California, Irvine
32. Gama J, Sebastião R, Rodrigues PP (2009) Issues in evaluation of stream learning algorithms. In: KDD'09, pp 329–338
33. Daniel, Wayne W (1990). Friedman two-way analysis of variance by ranks. Applied nonparametric statistics, 2nd edn. PWS-Kent, Boston, pp 262–274. ISBN 0-534-91976-6
34. The UCI KDD (1999) Archive. http://mlr.cs.umass.edu/ml/databases /kddcup99/kddcup99.html. Accessed 10 May 2016
35. Wang X-z, Xing H-J, Li Y et al (2015) A study on relationship between generalization abilities and fuzziness of base classifiers in ensemble learning. IEEE Trans Fuzzy Syst 23(5):1638–1654
36. Wang X, Rana A, Ai-Min F (2015) Fuzziness based sample categorization for classifier performance improvement. JIFS 29:1185–1196
37. Ashfaq RAR, Wang XZ, Huang JZ, Abbas H, He YL (2017) Fuzziness based semi-supervised learning approach for intrusion detection system., Inf Sci 378:484–497
38. Khan I, Huang JZ, Ivanov K (2016) Incremental density-based ensemble clustering over evolving data streams. Neurocomputing 191:34–43. doi:https://doi.org/10.1016/j.neucom.2016.01.009